information

MDPI

Systematic Review

Agile Software Requirements Engineering Challenges-Solutions—A Conceptual Framework from Systematic Literature Review

Zoe Hoy and **Mark Xu** *

Portsmouth Business School, University of Portsmouth, Portsmouth PO1 3DE, UK; zoe.hoy@port.ac.uk
* Correspondence: mark.xu@port.ac.uk

Abstract: Agile software requirements engineering processes enable quick responses to reflect changes in the client's software requirements. However, there are challenges associated with agile requirements engineering processes, which hinder fast, sustainable software development. Research addressing the challenges with available solutions is patchy, diverse and inclusive. In this study, we use a systematic literature review coupled with thematic classification and gap mapping analysis to examine extant solutions against challenges; the typologies/classifications of challenges faced with agile software development in general and specifically in requirements engineering and how the solutions address the challenges. Our study covers the period from 2009 to 2023. Scopus—the largest database for credible academic publications was searched. Using the exclusion criteria to filter the articles, a total of 78 valid papers were selected and reviewed. Following our investigation, we develop a framework that takes a three-dimensional view of agile requirements engineering solutions and suggest an orchestrated approach balancing the focus between the business context, project management and agile techniques. This study contributes to the theoretical frontier of agile software requirement engineering approaches and guidelines for practice.

Keywords: requirements engineering; agile software development; software testing; project management

check for updates

Citation: Hoy, Z.; Xu, M. Agile Software Requirements Engineering Challenges-Solutions—A Conceptual Framework from Systematic Literature Review. *Information* **2023**, *14*, 322. https://doi.org/10.3390/info14060322

Academic Editor: Luigi Benedicenti

Received: 25 April 2023
Revised: 28 May 2023
Accepted: 31 May 2023
Published: 6 June 2023

1. Introduction

Agile methodology, since its introduction about 20 years ago, has been widely implemented for software development [1]. Agile approaches help organisations to act with speed [2] and flexibility to quickly adapt and react to changing situations [3]. There are many agile variants; Scrum, for example, is one of the most used approaches in industry [4], as it can have short software development cycles of typically 2–4 weeks, namely "sprints" [5], and identify user requirements just in time [6]. It is reported that agile projects are twice as likely to succeed when compared to traditional projects [7]. However, there are contradictory reports that show 8% of agile projects failed and 50% were challenged for being over-budget, over the time estimate, or with smaller scope than initially specified [7].

Using agile is not without challenges, and the challenges tend to vary along the software development process. For example, Rehman [8] asserts that there is no proper separate phase of testing in agile methodology, and taking out time for comprehensive testing is difficult, thus keeping hold of the testing documents, test data, test cases, and scenarios during agile rapid development is a challenge that is to be worked upon. Agile is a popular methodology used for requirements engineering (RE)—a vital process to gather requirements for software development [9,10]. Agile for RE also faces challenges due to multiple stakeholders and changing dynamics of the business context. The process encompasses eliciting the requirements, analysing, specifying and validating, which requires interaction with stakeholders, prioritising the requirements and managing conflicting requirements [11]. RE produces specifications including both functional (FR) and non-functional requirements (NFR). Since customer requirements are continuously changing, agile teams are more focused on delivering the product rather than focusing on the documentation [8] which can weaken functional and non-functional design and testing. Some of the most frequently quoted ARE (agile requirement engineering) challenges concern non-functional requirements (NFRs) [12]. Weak requirements engineering is known to be a main reason for project failures [13].

The above challenges tend to be specific to the agile approach. Research also reveals challenges rooted from wide organisational settings. For example, Neto [14] noted that many challenges in the application of large-scale agile development relate to either RE or software testing, and in particular to the alignment between these areas. Alignment refers to coordination of practices, artifacts, and roles. The specific context here is the large organisation's agile project, and the challenges all point to dynamic changes and misalignment in documentation, understanding, and coordination for actions. Along similar lines, Luong et al. [1] suggest that anxiety, motivation, mutual trust, and communication competence were found to be significantly correlated with human-related challenges in agile teams. Solutions to these challenges require interaction with stakeholders, prioritising the requirements and managing conflicting requirements, using negotiation, collaboration and communication skills [11].

Some systematic literature up to 2020 on agile systems challenges shows consistent interest in describing the challenges, but lack of focus on ascertaining the problems in order to find solutions. In particular, it is not clear whether the key challenges are rooted in the agile methodology along, or in non-agile methodological issues such as the context of the business, the nature of the project, the fit/alignment between the chosen agile method, and the specific context of the project, and also the execution and management of the agile projects. An early study [15] found agile challenges are disproportionate and significantly mismatched when compared to the solutions. This raises two main research questions:

RQ1: What are the typologies/classifications of challenges faced with agile software development in general and specifically in requirements engineering?

RQ2: How do the solutions address the challenges?

Answering the questions can contribute to the literature debate on agile software development challenges with the insight drawn from the latest literature from 2020–2023, and for practice, it will help refocus on finding suitable solutions in order to cope with the challenges.

To achieve the aim as laid down in the research questions, we need to place a few boundary conditions on the scope of this paper. Firstly, given the vast and diverse literature on agile software development, we restrict the enquiry to agile requirements engineering. As aforementioned, RE is the most vital process in the software development cycle. Secondly, we assume other stages, e.g., agile software testing, face the same organisational and project conditions; hence, the non-agile methodology challenges are posited to be the same as that of RE. Thirdly, we include both RE challenges and solutions discovered from our own systematic literature review covering the period from 2009 to 2023.

The rest of the paper is structured as follows: Section 2 discusses the research method by explicating the research strategy and data extraction and analysis processes. Section 3 presents the results and discussions. Section 4 presents a typology/classification of the key challenges-solutions framework. Section 5 concludes the study with discussions of research contributions.

2. Methodology

This study took a two-step approach to address the research questions. First, a systematic literature review (SLR) was conducted to identify and classify all the challenges and solutions related to agile requirement engineering using the guidelines of Webster and Watson [16]; second, a three-dimensional framework addressing agile development challenges was conceptualised and then validated by using the latest empirical study data by Pertti Karhapää et al. [17].

2.1. Literature Search Strategy

Following the SLR approach suggested by Webster and Watson [16], we conducted a wider search for studies published in the last decade to gain a holistic view of agile requirements engineering challenges and solutions. Studies were selected with the following inclusion criteria: peer-reviewed journal articles published in English between 2009 and 2023. Studies that did not discuss agile requirements engineering challenges and studies that only mentioned agile methodology as an example without further discussion in the main article were excluded. Scopus—the largest database for credible academic publications was used primarily for this research. We also extended the search to other databases including Wiley Online Library, ProQuest, IEEE Xplore, Emerald, EBSCOHost, and Web of Science. The majority of the articles found in these other databases were also contained in the Scopus results, except one paper from EBSCOHost which was included in the final corpus. Duplicated papers from these multiple sources were removed. The final corpus was made up of 78 valid papers from 38 journals (see Appendix A Table A1).

The search string used for Scopus was (TITLE-ABS-KEY(“agile methodology” or “agile method” or “agile approach” or “agile development” or “agile practices” or “agile requirements engineering practices” or “agile way” or “agile software development”)) AND (TITLE-ABS-KEY(“requirements engineering” or “user story” or “user stories” or “feature” or “task” or “requirement”)) AND (TITLE-ABS-KEY(“challenge” or “problem” or “issue” or “obstacle” or “limitation”)). Search Terms commonly used to describe agile, requirements engineering, and challenges were adopted from previous studies. As a single reviewer collected the data, a protocol was not needed to ensure consistency among reviewers. Figure 1 shows the selection process followed, where Dybå and Dingsøyr’s [18] quality assessment questions were used to assess the eligibility of the included papers. The distribution of 78 journal articles by year of publication is shown in Appendix A Figure A1 and the distribution of research methods used in the included studies is shown in Appendix A Figure A2.

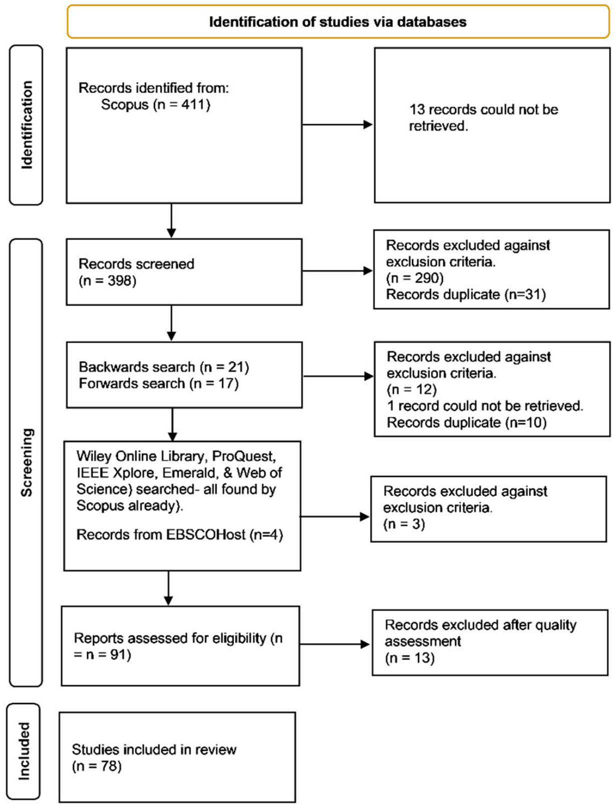
Identification of studies via databases
Identification
Records identified from:
Scopus (n = 411)
13 records could not be retrieved.
Screening
Records screened
(n = 398)
Records excluded against exclusion criteria.
(n = 290)
Records duplicate (n=31)
Backwards search (n = 21)
Forwards search (n = 17)
Records excluded against exclusion criteria.
(n = 12)
1 record could not be retrieved.
Records duplicate (n=10)
Wiley Online Library, ProQuest, IEEE Xplore, Emerald, & Web of Science) searched- all found by Scopus already).
Records from EBSCOHost (n=4)
Records excluded against exclusion criteria.
(n = 3)
Reports assessed for eligibility (n = n = 91)
Records excluded after quality assessment
(n = 13)
Included
Studies included in review
(n = 78)

Figure 1. PRISMA flow diagram, adapted from [19].

2.2. Data Analysis

Webster and Watson's [16] concept-centric approach was used to tabulate the challenges and solutions for data analysis. Concept A consisted of the agile requirements engineering challenges identified in the included studies, and concept B represented the solutions identified for the agile requirements engineering challenges in the included studies. According to Webster and Watson [16], a logical approach for analysing systematic literature is to group them and present them. Hence, a word cloud was created to visualize the reported challenges so as to identify the most prevalent terms used. The challenges associated with the most prevalent terms were organised into 11 subgroups, and likewise, the associated solutions were visualized and inductively grouped into nine subgroups. During the analysis phase, seven additional papers were found through backwards snowballing, to further evaluate some solutions cited in the included studies; these were additionally quality-assessed before use.

Next, we classify the 11 challenges and the nine solutions into three dimensions according to their thematical relevance to (1) organisational setting/business challenges, (2) project management challenges, and (3) agile methodology challenges. This classification is based on the themes that emerged with reference to previous research where one or more of these dimensions are mentioned mostly in a haphazard manner, (e.g., [20–24]).

As a systematic review, we are able to calculate the frequencies of the themes addressed in each study. Although the frequency does not necessarily mean the importance, it shows the commonality or prevalence of the issue in agile development and application, also the concentration of research attention, despite the variations and diversities in agile projects with different organisational contexts.

3. Results from SLR

3.1. Agile Requirements Engineering Challenges

From the 78 papers reviewed, we identified 11 challenges which are briefly discussed below. The sources for these 11 challenges are listed in Table A2.

Challenge 1 (C1)—Quality Requirements (QR) are neglected. The focus remains mainly on FRs [25]. QRs are ignored during the early sprints [26]. Late integration testing to verify requirements could reveal QR defects [27]. Late testing and late QRs can cause rework [27,28]. Changes in Functional Requirements (FRs) affect QRs [29] and QRs are rarely implemented in a single piece of code [27]. Requirements from different stakeholders could conflict [30] that lead to poor QRs.

Challenge 2 (C2)—Minimal documentation. Agile is less documentation-oriented, and documentation tends to be ad hoc, inaccurate, incomplete or non-existing [31]. Agile relies on sharing tacit knowledge [6], which could lose meaning over time [32,33] and through personnel turnover [21,34]. Problems arise when there is a communication breakdown [23], e.g., user stories are difficult to validate and maintaining documentation can be a daunting task [21].

Challenge 3 (C3)—Inappropriate prioritisation method. Ordinal scales and ratings to measure priority are difficult to make objective [35]. There are scalability problems with large numbers of requirements [35] that make it difficult to select the optimal requirements [30], and hard to maintain a requirement priority list [36]. Functional requirements (FRs) are often prioritised over non-functional requirements (NFRs) [37]. 'Business value' overweighs other aspects leading to ignoring NFRs—for example, maintainability or security [21]. There is a reported case where customers were unwilling to prioritise nor accurately represent their requirements [34].

Challenge 4 (C4)—Managing change. Business needs evolve [20] that require rapidly changing requirements and continuous reprioritisation [38], which causes an instability issue [39]. The impact and risk of requirement changes needs to be considered [40]. Requirement changes cause existing documents to be updated [41] and can impact on the cost and schedule of the project [42]. When requirement changes are not constrained to the start of a sprint, disruption or even cancelling a sprint can happen [43].

Challenge 5 (C5)—Poorly written requirements. Requirements can be inadequate [44], poorly written [28] with quality defects [21], ambiguous [45,46], underspecified, incomplete, inconsistent, and unfeasible [38], too granular or too high-level causing differing interpretations [47]. User stories focus on describing functionality rather than quality requirements [37].

Challenge 6 (C6)—Inaccurate effort estimation. Estimates can be changed substantially by a requirement change in subsequent iterations [20,48]. Unreliable estimation happens due to the absence of historical data and experts [21,49,50], particularly for large user stories [51] and for agile teams in distributed geographical locations [52].

Challenge 7 (C7)—Customer unavailable or low availability. Customer involvement and interactions with the development team are challenging [24,27,53], particularly with offshore development where the customer is located on a different site [21]. The challenge also refers to customers lacking authority to make decisions, surrogate customers not conveying real requirements [54], a weak relationship [38], and the low availability of customers for negotiation, clarification, and feedback [27,40]. Requirements ownership is a problem when many levels exist between the real customer and the development team [55].

Challenge 8 (C8)—Customer knowledge. When customers are lacking knowledge about agile or do not believe in the agile approach, it is hard to obtain quality feedback from them [54]. The challenge also encompasses customers' inability [24,46], lack of technical competence [54], incomplete domain knowledge [38,56], insufficient knowledge of requirements [21], and of agile software development methods [47].

Challenge 9 (C9)—Inappropriate architecture. Architecture decisions made in the early cycles can become redundant when new requirements arise [27,46,57]. There is then a cost to refactor the software to meet the customer's needs [46].

Challenge 10 (C10)—Communication methods. It can be difficult to communicate efficiently with stakeholders [40]. There is a reported case where communication methods were too costly, requiring a technical background to use, and were not designed for agile. Face-to-face meetings are the best communication method and instant messaging is the worst communication method [58].

Challenge 11 (C11)—Maintaining a software requirements specification. This can be difficult for an overloaded team who worked on several projects simultaneously [59].

Unclassified challenges. Some challenges are not directly related to requirements engineering, but can be general to agile projects, hence they are put into this unclassified category. Inayat et al. [23] and Okesola et al. [24] reported contractual limitations as a challenge, where fixed-price contracts can prevent changes. An organisation working in a multi-disciplinary and regulated environment used traditional tools for requirements management and an agile tool 'JIRA' for development. The tools were disconnected [55]. Wagner et al. [38] suggest 'insufficient support by project lead', 'implementation of features without requirements', and 'not enough time in general' are challenges.

According to the frequency of these challenges in the papers, we depict the most prevalent (not necessarily the most important, but the commonality) challenges in Figure 2.

3.2. Agile Requirements Engineering Solutions

The reported solutions related to requirement engineering are summarised below. The papers for each solution are shown in Table A3.

Solution 1 (S1)—Provide the requirement information needed. This solution includes providing requirements in the form of test cases [13]; the RE-KOMBINE framework to specify requirements in a flexible way to support requirement change [23]; user goal-oriented stories and delivery stories [24,47]; W8 story cards [12]; creating use case diagrams [45,60]; and concise acceptance criteria in user stories [27]. The quality of user stories could be evaluated through a tool or by checking against a set of criteria [61]. A domain expert or business analyst role could help improve the quality and correctness of requirements [62]. Sprints of short duration with smaller requirements have clearer software requirements specifications [59].

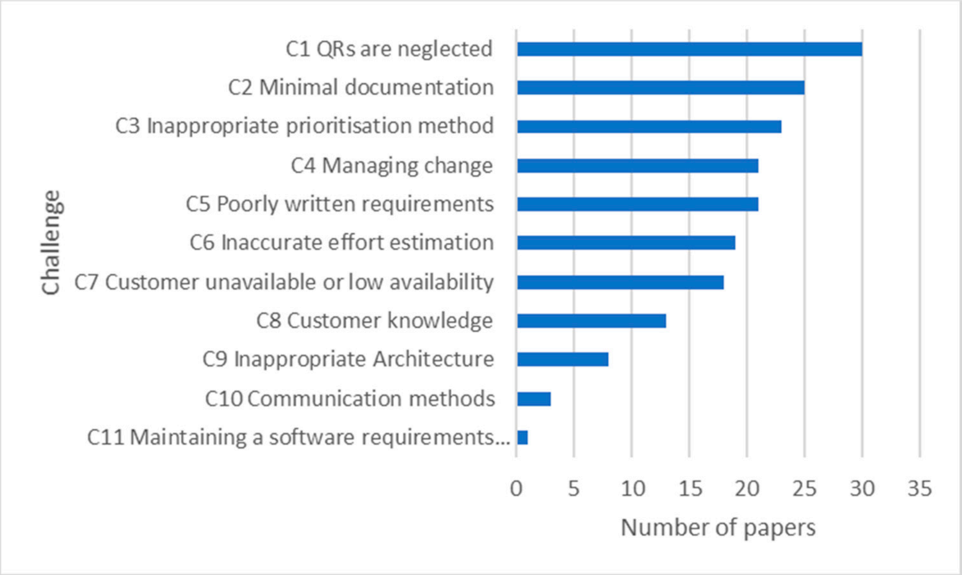

Figure 2. Common challenges.

Solution 2 (S2)—Manage the quality requirements (QRs). Timely integration testing of the whole system could help to identify any QR defects early. A project team could have a separate QR backlog and prioritise one QR for each sprint, or a whole sprint could be allocated to work on QRs. Some large, distributed projects give ownership of QRs to specialist teams, who ensure the QRs are implemented [27]. Elicit and manage QRs with their associated risks [63]. Maintaining a list of assumptions made about QRs by the agile team. The lightweight approach of the NORMATIC tool can achieve a balance between minimal documentation and neglecting QRs [24]; it supports FRs, NFR modelling and QRs. Follow guidelines developed for supporting quality requirements [37,64].

Solution 3 (S3)—Share knowledge about the requirements. Among global agile software development teams, Borrego et al. [32] found that instant messages and emails were preferred methods of communication and developed a method to extract architectural knowledge from these files. Other methods include a communications hub between developers and clients [58]; on-site customer representatives could improve user-developer interactions [65]. Frequent communication by means of face-to-face interactions, weekly meetings, and interview sessions [23,32,66] are recommend, using a surrogate customer if the customer is unavailable [24]. To share knowledge and synchronise development efforts, communities could be created around a shared interest [60].

Solution 4 (S4)—Manage a product backlog. This solution aims to cover different viewpoints, customer, architecture, FRs, and QRs using multiple backlogs [27,60]. Gaikwad and Joeg [47] suggest using the agile incremental-delivery model to renegotiate requirement priorities. All changes identified during a running sprint are reflected in the product backlog and considered for future sprints [43].

Solution 5 (S5)—Improve the estimation process. Estimation is a challenge for agile as the requirements are unstable at the start [24]. Solutions include eliciting rationales from each estimation to help the team achieve consensus for the estimate, setting aside regular time for estimating [43], and use of data-driven techniques to estimate effort [50].

Solution 6 (S6) Maintain requirements traceability. Firdaus et al. [29] propose a model to trace security and performance QRs. Urbieta et al. [67] propose an approach using Language Extended Lexicon to organise user stories and assist with traceability. Well-known tools such as JIRA could facilitate managing requirements [23].

Solution 7 (S7)—Identify the minimal documentation needed. A consolidated software requirements specification is recommended instead of various artefacts [59]. Architecture support is needed for agile, Gahyyur et al. [68] recommends that a sub-system model, reusable component list, architectural significant features, and a rationale are documented.

Solution 8 (S8)—Improve the method for requirements prioritisation. Consider three criteria in prioritisation: 'stakeholder satisfaction', 'risk of implementing', and 'available resource' [30].

Solution 9 (S9)—Share the product vision. Maintain a common product vision between the customer and development team to reduce requests for change, for a running sprint [43].

Unclassified solutions. Okesola et al. [24] recommend the adoption of legal measures for contracts to support the flexible nature of agile requirements engineering. Inayat et al. [23] suggest the use of fixed payments per release. These are not included in the classification as they are not directly RE-related.

According to the frequency of these solutions in the papers, the most prevalent (not necessarily the most important, but the commonest) solutions are shown in Figure 3.

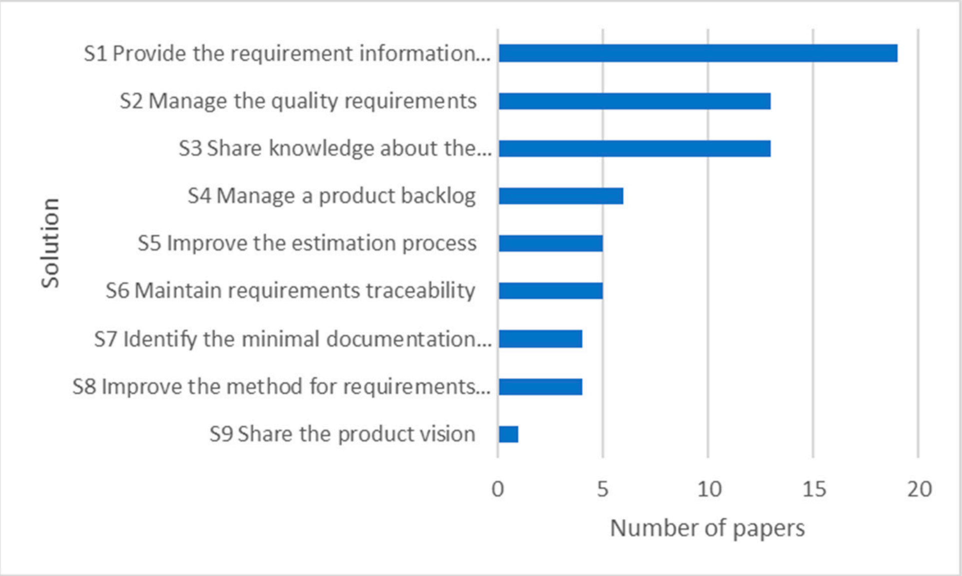

Figure 3. Common solutions.

3.3. Discussion—Three-Dimensional Classification of Challenges and Solutions

The reported challenges and solutions as summarised in the above sections tend to be patchy, anecdotal as each one is drawn from a specific context where the study has been conducted, and diverse, as they are not all rooted in the deficiencies of agile methodology. In order to answer our research questions, a further examination was conducted by skipping the specific context and focusing on the thematic meanings of these challenges and solutions. It becomes apparent that many challenges and solutions are rooted in organisational setting/business problems and project management issues, in addition to agile methodology. Hence, we use these three dimensions to map out all the challenges and solutions. The results are shown in Figure 4. It is noted that some challenges and solutions fall into multiple dimensions.

3.3.1. Dimension of Organisation/Business Challenges

The success of the agile approach relates to how well it fits into the organisation's structure, resources, processes, and culture. Some of the issues in this dimension adversely affect agile practice including:

Large organisational setting—where many levels exist between the real customer and the development team, the requirement of ownership and loss of information at each level is a problem. Real customers may not be effectively engaged with the agile team for requirements, knowledge sharing, and feedback [55].

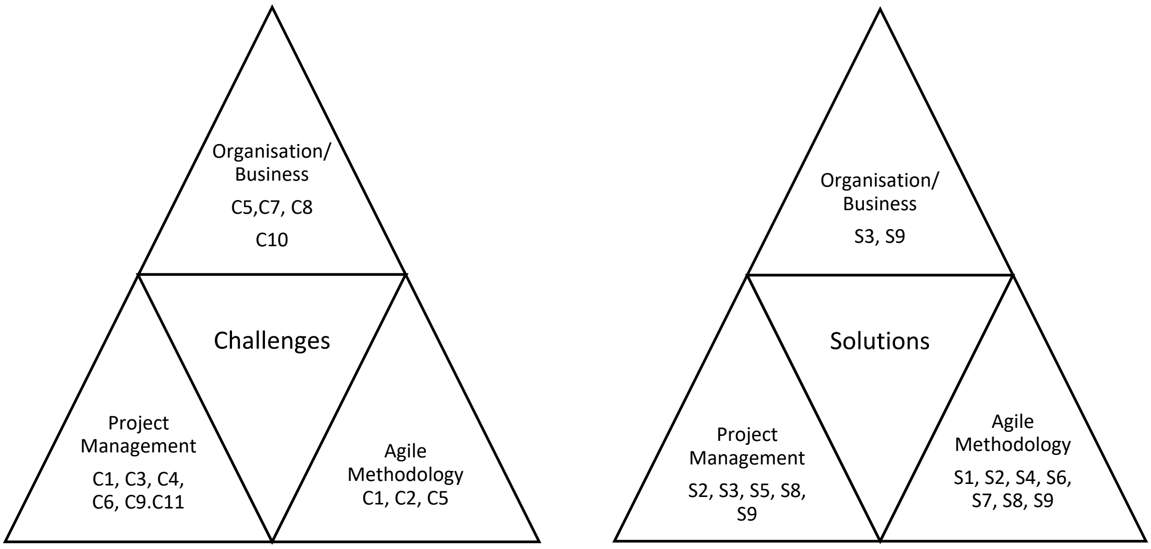

Figure 4. RE Challenges-Solutions in Three Dimensions.

Dynamic changes in business— changes in environment, objectives, people, etc. require continual re-planning and adjusting to new requirements, which could lead to misalignment in documentation, understanding, and coordination of actions by agile teams.

Motivation and communication—anxiety, motivation, mutual trust, and communication competence were found to be significantly correlated with human-related challenges in agile teams [1].

Coordination and shared understanding—multi-functional teams in organisations may not share common conceptualisation on issues and inform changes, even with conflicting requirements. Poor coordination between these teams, particularly in distributed geographical locations [52], is potentially the root cause of agile problems.

Our analysis shows that organisational setting/business problems are more likely to create challenges in customer availability, knowledge, and communication, hence leading to poor requirement quality and specification, i.e., C5 Poorly written requirements, C7 Customer unavailable or low availability, C8 Customer knowledge, and C10 Communication methods.

3.3.2. Dimensions of Project Management Challenges

Most of the agile challenges identified from this study are related to project management (PM) challenges. As seen from Figure 4, six challenges are classified into this dimension; they are C1 Quality requirements are neglected, C3 Inappropriate prioritisation method, C4 Managing change, C6 Inaccurate effort estimation, C9 Inappropriate architecture, and C11 Maintaining a software requirements specification. Large-scale agile projects would require an experienced project manager to be responsible for the whole project team and be accountable for the success or failure of the project [69]. The six challenges discovered in agile RE are actually rooted in the five PM processes: manage project execution, develop project structure plan, develop project schedule, estimate and define costs based on requirements, and develop and manage the team. An agile team under a project management matrix structure usually has its members from different functional areas and at different levels, with multiple stakeholders, external agile specialists, analysts and developers. As reviewed from this study, misunderstanding and conflicts in interest, requirements, costs and benefits, etc. can be the root causes of an agile project failure.

3.3.3. Dimensions of Agile Methodology Challenges

Three challenges are grouped under this dimension, due to direct links with agile methodology. They are C1 Quality requirements are neglected, C2 Minimal documentation and C5 Poorly written requirements. Agile is less documentation-oriented, which can cause difficulties for testing. Functional requirements (FR) are heavily reliant on user/customer involvement, whereas non-functional requirements | (NFR) cannot be proactively specified in the early sprint by customers, and hence tend to be ignored in many agile projects. It is clear that there is room for improving agile methods, but this is not the sole solution to cope with all the challenges.

3.3.4. Dimensions of Agile Methodology Solutions

Following the same approach, all the solutions are reviewed and classified into three dimensions as shown in Figure 4. It is interesting to note that seven of the identified solutions are most likely to fall into the category of agile methodology improvement, five of these are related to project management, and two are orientated to organisational setting improvement. A few solutions appear to cross the classification boundaries, i.e., S9—maintaining a common product vision, S8 on prioritisation methods, and S3—sharing knowledge are both organisational and project management solutions. This is elaborated in more detail below.

Organisation/Business Dimension has two solutions: Sharing knowledge (S3) and Maintaining a common product vision (S9). These are essential to the success of agile projects. Sharing customer knowledge among the agile project team reflects an organisation's norm and culture, which has been ranked the third most common solution. It also requires effective coordination and communication between distributed teams and divisions. Maintaining a common project vision requires a visionary leadership to articulate clearly the goal and targets across levels and teams, which will turn into explicit requirement specifications.

Project Management Dimension contains five solutions: Manage the quality requirements (S2), Improve the estimation process (S5), Improve the method for requirements prioritisation (S8), and cross-boundary solutions S9 and S3. These solutions are orientated towards better agile project management, for example, the project team to have a separate quality requirement backlog and prioritise one quality requirement in each sprint, also improved estimation of time, costs, and risks in order to achieve consensus. As aforementioned, sharing customer knowledge and vision can be considered an effective project management attribute in the context of agile development.

Agile Methodology Dimension solutions include seven solutions drawn from the literature—the largest from the three-typology point of view. They include providing the requirement information needed (S1), managing a product backlog (S4), maintaining requirements traceability (S6), identifying the minimal documentation needed (S7), and cross-boundary solutions S2, S8, and S9 as discussed. These are solutions aimed at specific agile methodology improvements.

Mapping the prevalence (commonality) between the challenges and the solutions shows an apparent mismatch between the dimensions of challenges vs. solutions, i.e., it appears to put more attention on agile method solutions than considerations of organisational/business/project management endeavour, despite the fact that many challenges are actually rooted in organisational/business and project management dimensions. This is explored further in the next section.

4. A Framework of Orchestrating Agile Challenges

The findings emerged from this study helped us develop the following insights:

(i) In agile software development literature, there is overwhelming research interest in challenges, but disproportionately little in exploration and exploitation of solutions;

(ii) Of the solutions summarised by this study, the largest portion is aimed at finding ways to improve agile approaches, and to some extent, to improve project management. Challenges rooted in the organisational setting and business context have been neglected;

(iii) Challenges faced by agile projects (teams) are rooted in the aforementioned three dimensions, hence, improving agile methods would not solve all the problems, and any endeavour to improve agile practice has to consider the organisational/business setting and project management practice. We echo Al-Zewairi et al.'s [70] assertion that agile is ideal for projects that need to deliver in short intervals of time, with high capability of change in requirements, the capabilities of people working on the projects and technologies being used;

(iv) Organisational setting and business are complex, dynamic, and context-specific, which creates many challenges that demand different solutions. The fit (or alignment) between these complexities and agile suitability should be assessed before taking an agile approach. This reaffirms Neto et al.'s [14] assertion that organisational context is key in RE because agile methods emphasise reactivity and informal communication that are hard to guarantee across multidisciplinary teams within a large organisation;

(v) The agile project and the team(s) are the visible entity in the centre to execute various agile tasks (e.g., RE, testing, implementation. . .) and deliver the desired outputs. The operations of an agile project are affected by four driving forces: (a) the organisational/business settings determine its governance, member composition, roles and responsibilities, resources and stakeholders; (b) project management competence and experience determine the effectiveness of initiation, planning, control, coordination and communication; (c) the business requirements and the expected functionalities of the software (FR and NFR); (d) the ability of the agile methods. Hence an orchestrated approach is needed.

The relationships and the driving forces, as suggested above, are depicted in Figure 5 as a framework for considering agile challenges and solutions.

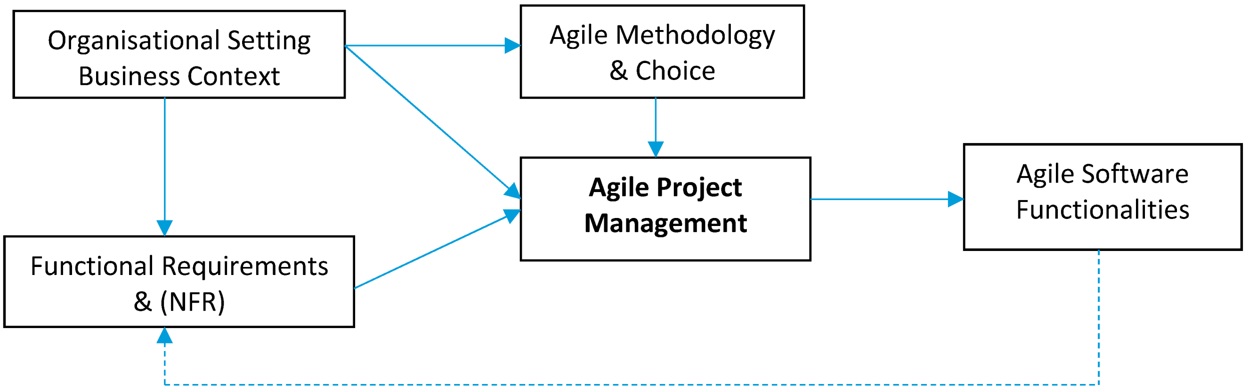

Figure 5. A Framework of researching agile challenges and solutions.

This framework suggests several possible relationships that can be extended and validated by future research. We assume that the agile software functionality is the deliverables (in various versions during the development process) as the dependent variable, which is determined by the four factors as independent variables. Among these independent variables, inter-correlations exist. To give a detailed account:

- Organisational setting/business context determine the functional and non-functional requirements, it also influences the governance, composition, management and operations of the project, as mentioned before, the levels of customers, the geographic locations of teams, the thin or multi business objectives, requirements, frequency of changes, can lead to different PM team structure and management mechanism. This also determines the suitability of taking agile approaches and the choice of the agile methods;
- Requirements engineering is key to the agile process. Functional requirements can be identified and explicitly specified with the right customers (with knowledge and availability); however, we doubt that non-functional requirements can be accurately specified in the early sprints, because these are systems performance features that cannot be pre-specified by customers until they experience how the systems would

function. Thereby, there is a backward loop to enable feedback from customers of early version functionalities, in order to stimulate NFR specification and improvement;

- Agile methodology as an independent variable covers different agile approaches along the development process, e.g., Scrum, Kanban, Scrumban, eXtreme Programming [12], each with its own capabilities, conditions, and limitations, hence, the choice could affect the agile project performance;
- Finally, the agile project team and management assemble and orchestrate the people, resources, skills, technologies, and methodologies to deliver the outputs.

Comparing our results to a most recent study by Gupta, Poels and Bera [46] where SLR (2009–2019) method and interviews were adopted, shows that our results confirm most of the challenges discovered by Gupta et al. [46], but significantly differentiated from that study by classifying the challenges to include organisational challenges, and mapping the solutions to develop a framework to address the gaps. Gupta et al. [46] have systematically searched for the challenges to requirements engineering with agile methods. Using 57 selected papers, they found 22 challenges, and added three from their own empirical study, which confirm eight previously found challenges [10,20].

Our study confirms all the project management and customer involvement-related challenges found by Gupta at al. [46], e.g., the team's lack of involvement and motivation, breakdown of team communication and coordination, difficulty in managing dispersed teams, not sharing knowledge, lack of management involvement, difficulty in customer interaction, customer inability and disagreement, and difficulty in estimating time and costs. Our study partially confirms Gupta et al.'s study [46] on the agile methodology-related challenges, but not at the level of details, e.g., minimal documentation, detailed user stories not being created or integrated, difficulties in decomposing user stories, availability of testing resources, reduced testing and testing coverage, and incomplete NFR.

Some challenges identified in Gupta et al.'s study [46] are not actually challenges, but the repercussions of not dealing with the challenges—for example, incomplete and missing requirements, ambiguous requirements, and inadequate requirement verification. Although Gupta et al.'s study [46] also identified external visibility on project tasks and requirement volatility as challenges, these are not considered from an organisational perspective. Furthermore, the study proposes the use of conceptual models to address the challenges, but does not map out the gaps in solutions.

One of the implications of this framework is that the agile methodology does not cause all the challenges, and therefore it should not be expected to solve all the problems. Challenges rooted in other dimensions need to be considered for alternative solutions. As proposed in this framework, a coordinated, orchestrated approach centred around agile project management with sufficient consideration of the organisational setting and business context could deliver the intended software functionalities. For theoretical development, we propose that:

(1) A shift from studying agile challenges to influential elements and their relationships. Studies on challenges show a sign of reaching saturation, as evidenced from a recent empirical case-based study by Pertti Karhapää et al. [17] (p. 40): "not all of the challenges found in literature were identified in our study, nor did we find any new challenges to add". Shifting the focus would help explore more solutions relevant to their root causes;

(2) More studies to examine effectiveness and efficiency of agile practice (including each method) in different context. Effectiveness of agile practice means satisfaction (and more sustainability) of meeting both FR and NFR requirements; efficiency means resource utilisation of the agile project—both intangible and tangible (time, costs. . .). This will help exploitation of existing agile methods.

For practice, we suggest the following:

(1) The business problems and the organisational setting attributes—multiple objectives, dispersed teams, distributed problem ownership and decision-making, customer

resources and knowledge availability, etc.—shall be considered as the preconditions for taking an agile approach. That implies that although agile offers many advantages, it may not fit complex organisational situations where requirements vary significantly across different levels and geo-locations. We refer to this as agile eligibility evaluation;

(2) In the case that an agile project has been approved, it remains critical for the leadership to articulate clearly the vision and goals to all the problem owners, team members, developers, agile specialists, and stakeholders. This is to ensure consistency in understanding the requirements, making changes in a dynamic manner, and developing/testing the functionalities before, during, and after the agile development process;

(3) Once an agile project has been initiated, it is critical to ensure a competent project team has been established with adequate resources and control power. More importantly, customers as participants in the agile team shall have sufficient knowledge, time, responsibilities, and positive attitude to work effectively with agile specialists for functional requirements specification, feedback, and co-designing desirable functionalities;

(4) With regard to NFR specifications, these are unlikely to be known explicitly before users and problem owners see the earlier sprint results in different phases. Hence, agile developers should initiate NFR in demonstratable fashion, so as to engage users and problem owners for verification and improvements. The rationale for this is that the rapid development of digital technologies in security, scalability, and architecture means that business users are not expected to foresee the potential of the advanced technologies during agile development;

(5) Lastly, but not the least, for agile developers, programmers, and researchers in this area, there are rooms for agile methodological improvements, particularly on issues with minimal documentation, lack of traceability, difficulty in testing and prioritisation, and coping with implicit requirements and constant changes in requirements.

5. Conclusions

This study examined the challenges and solutions of agile requirement engineering based on a systematic literature review. Through thematic analysis, the reported challenges and the solutions are classified into three dimensions: organisational/business context, project management, and agile methodology. This answers research question 1 (RQ1). Figure 4 identified a mismatch between the solutions and the challenges. It showed a concentration of solutions for agile methodologies, but a lack of solutions to address challenges arising from organisational setting and business context and agile project management. This answers research question 2 (RQ2). The conclusion drawn from the insight is that agile requirement engineering challenges/problems are not all rooted in the deficiencies of agile methodologies, although there is room for improvements. Agile can only be effective when there is a supporting organisational setting coupled with appropriate project management. Organisational setting/business context shapes how agile projects shall be initiated and run, hence a coordinated approach addressing organisational/business and project challenges is required. A framework establishing the key dimensions and the inter-relationships has been suggested, which shifts attention beyond the current focus on agile methodology. This opens a new avenue for future research endeavour in seeking agile project solutions.

A number of limitations should be noted. First, the key evidence is based on a systematic literature review with limited scope, as described in the introduction, and a small sample. Second, although a structured and guided approach was followed for this study, unintentional oversight and bias in search and reading the literature can still happen. However, our results tend to be reliable, as the challenges and solutions are found to be similar in a recent empirical study conducted by Pertti Karhapää et al. [17]. Third, the classification of the challenges and the solutions into three-dimensional typologies are driven by the search results, but the analysis is subjective in nature. Fourth, after an extensive literature search, 13 Scopus indexed journal articles from the initial search results and one from the initial backwards search could not be sourced. However, this is a very small sample of the 440 articles which were analysed in our study, therefore the impact is negligible. Fifth, we have only selected papers published in English. This was not considered a limitation [71] as there were only 10 Scopus results for articles published in other languages for 2009 to 2023, meeting our search terms. Last, the framework illustrates a conception drawn from the analysis of this study. It can be a useful guide for theoretical development and for agile practice, but needs further studies to validate it. These limitations can be addressed by future empirical studies following the research avenue set out earlier.

Author Contributions: Conceptualization, Z.H.; methodology, Z.H.; formal analysis, Z.H.; writing-original draft preparation, Z.H.; writing-review and editing, Z.H. and M.X.; supervision, M.X. All authors have read and agreed to the published version of the manuscript.

Funding: The authors have no relevant financial or non-financial interests to disclose. All authors certify that they have no affiliations with or involvement in any organisation or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript. The authors have no financial or proprietary interests in any material discussed in this article.

Data Availability Statement: The dataset used for this study is available from the University of Portsmouth Research Portal [72].

Conflicts of Interest: The authors have no conflict of interest to declare that are relevant to the content of this article.

Appendix A

Table A1. Distribution of journal outlets.

Journal Outlets	Count	Percentage
ACM Transactions on Management Information Systems	1	1.00
Artificial Intelligence Review	1	1.00
CLEI Electronic Journal	1	1.00
Computers and Electrical Engineering	1	1.00
Computers and Operations Research	1	1.00
Computers in Human Behavior	1	1.00
Empirical Software Engineering	4	5.00
Future Computing and Informatics Journal	1	1.00
IAENG International Journal of Computer Science	1	1.00
IEEE Access	5	6.00
IEEE Transactions on Software Engineering	3	4.00
IET Software	3	4.00
Information (Switzerland)	1	1.00
Information and Software Technology	10	13.00
Information Systems Journal	1	1.00
International Journal of Advanced Computer Science and Applications	4	5.00
International Journal of Applied Engineering Research	1	1.00
International Journal of Computer Applications	1	1.00
International Journal of Computer Science and Applications	1	1.00
International Journal of Engineering and Technology (UAE)	1	1.00
International Journal on Advanced Science, Engineering and Information Technology	1	1.00
Issues in Informing Science and Information Technology	1	1.00
Journal of Industrial Information Integration	2	3.00
Journal of Information Technology Case and Application Research	1	1.00
Journal of Software Evolution and Process	3	4.00
Journal of Software Maintenance and Evolution	1	1.00
Journal of Systems and Software	11	14.00

Table A1. *Cont.*

Journal Outlets	Count	Percentage
Journal of the Brazilian Computer Society	1	1.00
Journal of the Chinese Institute of Engineers, Transactions of the Chinese Institute of Engineers, Series A/Chung-kuo Kung Ch'eng Hsuch K'an	1	1.00
Journal of Theoretical and Applied Information Technology	1	1.00
Jurnal Teknologi	1	1.00
Requirements Engineering	5	6.00
Scandinavian Journal of Information Systems	1	1.00
Soft Computing	1	1.00
Software Quality Journal	1	1.00
Tehnicki Vjesnik	1	1.00
TQM Journal	1	1.00
VINE	1	1.00
	78	100

Note—The distribution of the 78 included studies is shown in Figure A1. The years 2017, 2018, 2019 and 2022 yielded the highest numbers of publications, with 10, nine, 12, and 13 in total. There were no publications in 2013. The years 2009, 2010, 2011, and 2012 gave the lowest number of publications, with one, two, two, and one, respectively. The results show an increasing trend for papers published since 2014, a drop in 2020 and a second increasing trend from 2021, with the highest number of publications in 2022.

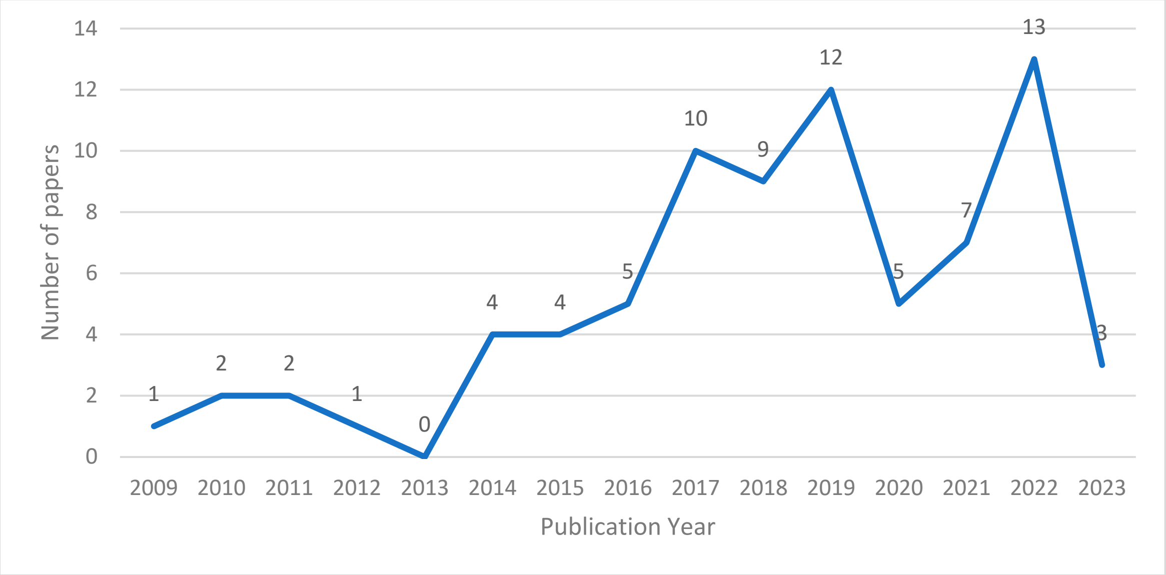

Figure A1. Distribution of journal articles by year of publication.

Note—Figure A2 shows the distribution of the research methods used in the included papers. The types of paper identified by Alavi and Carlson [73] for MIS research were used. Most of the papers were empirical (55 out of 78), and the results were based on data and observations; others were non-empirical and were based on ideas. Of the 55 empirical works, the majority were case studies (28), followed by surveys (13) and laboratory experiments (eight). Of the 23 non-empirical papers, the majority focused on a conceptual model (seven), followed by a conceptual overview (seven) and a conceptual framework and its applications (six). As most of these articles presented case studies, this indicated that the researchers sought to gain a rich understanding of the study context. There were only 13 surveys, meaning that there is an opportunity for future research in which more surveys are conducted, as these provide opportunities to explore possible relationships between variables and to produce models of these relationships for agile requirements engineering.

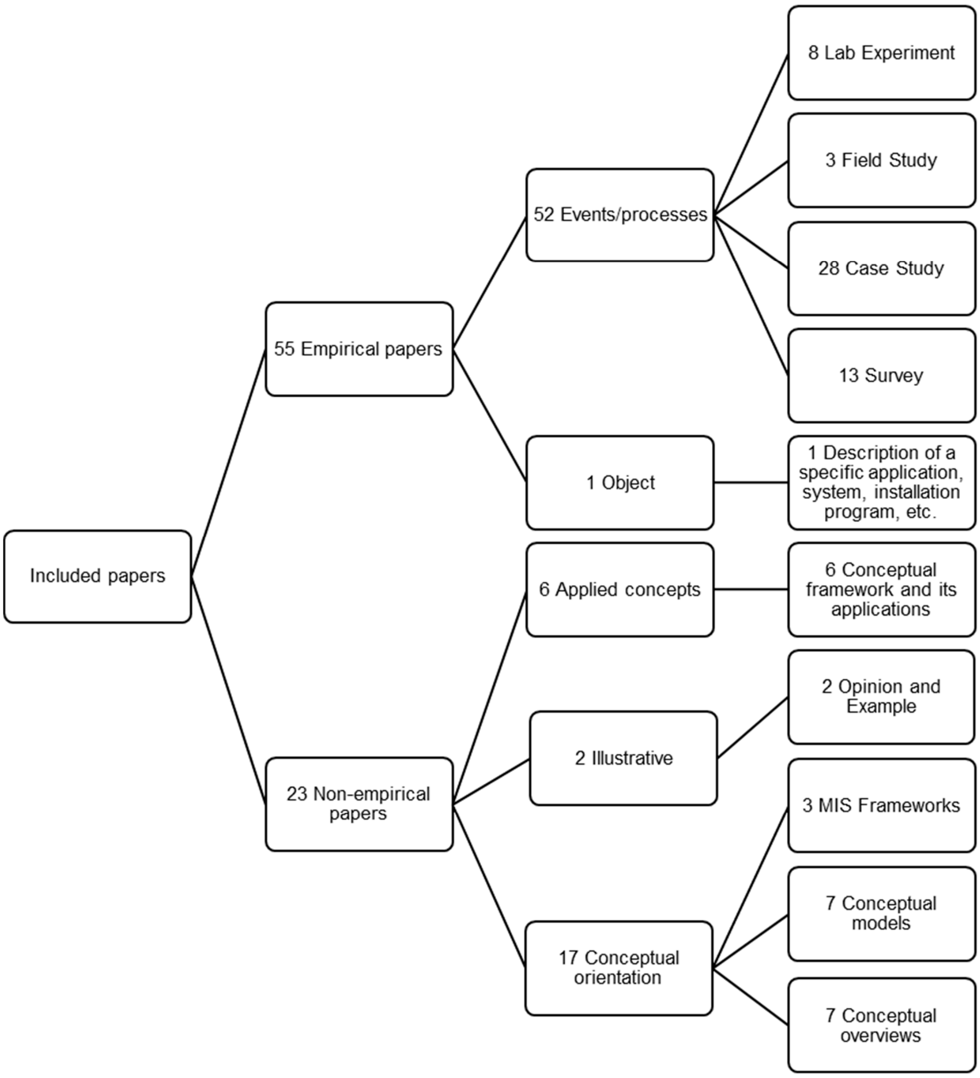

Figure A2. Distribution of research methods used in the included papers.

Table A2. Literature sources for each challenge.

Challenges	Sources	No.
C1 QRs are neglected	[6,10,12,17,20,21,23–29,37,38,40,46,53,54,63,64,74–82]	30
C2 Minimal documentation	[6,10,17,21,23,24,27,28,31–34,37,41,46,47,51,53–56,64,65,77,82]	25
C3 Inappropriate prioritisation method	[10,20,21,23,25,27,30,34–37,39,40,46,47,54,59,60,62,74,77,83,84]	23
C4 Managing change	[20,21,23,24,28,34,35,38–43,47,53,65,82–86]	21
C5 Poorly written requirements	[20,21,27,28,34,37,38,40,44–47,54,55,59,61,71,74,78,87,88]	21
C6 Inaccurate effort estimation	[10,20,21,23,24,27,31,40,43,46,48–54,62,89]	19
C7 Customer unavailable or low availability	[10,20,21,23,24,27,28,34,38,40,47,53–55,59,65,74,77]	18
C8 Customer knowledge	[21,23,24,31,38,46,47,54–56,76,77,90]	13
C9 Inappropriate Architecture	[21,23,27,34,46,54,57,76]	8
C10 Communication methods	[40,46,58]	3
C11 Maintaining a Software Requirements Specification	[59]	1

Table A3. Literature sources for each solution.

Solutions	Sources	No.
S1 Provide the requirement information needed	[13,17,23,24,27,28,42,45–47,53,59–61,71,74,83,91,92]	19
S2 Manage the QRs	[12,22–24,27,37,63,64,74,75,81,93,94]	13
S3 Share knowledge about the requirements	[22–24,28,32,39,47,58,60,64–66,95]	13
S4 Manage a product backlog	[23,27,43,47,60,65]	6
S5 Improve the estimation process	[10,24,43,50,89]	5
S6 Maintain requirements traceability	[23,29,65,67,92]	5
S7 Identify the minimal documentation	[28,46,59,68]	4
S8 Improve the method for requirements prioritisation	[25,30,84,95]	4
S9 Share the product vision	[43]	1

Note–The review was not registered, however the PRISMA 2020 item checklist [19] was used.

References

1. Luong, T.T.; Sivarajah, U.; Weerakkody, V. Do agile managed information systems projects fail due to a lack of emotional intelligence? *Inf. Syst. Front.* **2021**, *23*, 415–433. [CrossRef]
2. Xu, Y.; Koivumäki, T. Digital business model effectuation: An agile approach. *Comput. Hum. Behav.* **2019**, *95*, 307–314. [CrossRef]
3. Janssen, M.; van der Voort, H. Agile and adaptive governance in crisis response: Lessons from the COVID-19 pandemic. *Int. J. Inf. Manag.* **2020**, *55*, 102180. [CrossRef]
4. Torrecilla-Salinas, C.J.; Sedeño, J.; Escalona, M.J.; Mejías, M. Estimating, planning and managing agile web development projects under a value-based perspective. *Inf. Softw. Technol.* **2015**, *61*, 124–144. [CrossRef]
5. Manifesto for Agile Software Development. Available online: http://agilemanifesto.org/ (accessed on 14 April 2023).
6. Alsaqaf, W.; Daneva, M.; Wieringa, R. Agile quality requirements engineering challenges: First results from a case study. In Proceedings of the IEEE International Symposium on Empirical Software Engineering and Measurement, Toronto, ON, Canada, 9–10 November 2017; pp. 454–459. [CrossRef]
7. Mersino, A. Agile Project Success Rates Are 2X Higher than Traditional Projects. Available online: https://vitalitychicago.com/blog/agile-projects-are-more-successful-traditional-projects/ (accessed on 14 April 2023).
8. Rehman, A.U.; Nawaz, A.; Ali, M.T.; Abbas, M. A Comparative Study of Agile Methods, Testing Challenges, Solutions & Tool Support. In Proceedings of the 14th International Conference on Open Source Systems and Technologies (ICOSST), Lahore, Pakistan, 16–17 December 2020; pp. 1–5. [CrossRef]
9. Govil, N.; Sharma, A. Information extraction on requirements prioritization approaches in agile software development processes. In Proceedings of the 5th International Conference on Computing Methodologies and Communication, Erode, India, 8–10 April 2021; pp. 1097–1100. [CrossRef]
10. Ramesh, B.; Cao, L.; Baskerville, R. Agile requirements engineering practices and challenges: An empirical study. *Inf. Syst.* **2010**, *20*, 449–480. [CrossRef]
11. Razali, R.; Anwar, F.; Rahman, M.A.; Ismail, F.F. Mixed Methods Research: Insights from Requirements Engineering. *J. Bus. Res.* **2016**, *14*, 125–134. Available online: https://academic-publishing.org/index.php/ejbrm/article/view/1347 (accessed on 14 April 2023).
12. Jarzębowicz, A.; Weichbroth, P. A qualitative study on non-functional requirements in agile software development. *IEEE Access* **2021**, *9*, 40458–40475. [CrossRef]
13. Bjarnason, E.; Unterkalmsteiner, M.; Borg, M.; Engström, E. A multi-case study of agile requirements engineering and the use of test cases as requirements. *Inf. Softw. Technol.* **2016**, *77*, 61–79. [CrossRef]
14. Neto, F.G.D.O.; Horkoff, J.; Knauss, E.; Kasauli, R.; Liebel, G. Challenges of Aligning Requirements Engineering and System Testing in Large-Scale Agile: A Multiple Case Study. In Proceedings of the IEEE 25th International Requirements Engineering Conference Workshops, Lisbon, Portugal, 4–8 September 2017; pp. 315–322. [CrossRef]
15. Hoy, Z. Requirements engineering for agile teams and startups: A challenge-solution gap analysis from a systematic literature review. In *Software Engineering Practices for Start-Ups*; CRC Press (Taylor & Francis): Boca Raton, FL, USA, 2023; *in press*.
16. Webster, J.; Watson, R.T. Analyzing the Past to Prepare for the Future: Writing a Literature Review. *MIS Q.* **2002**, *26*, xiii–xxiii. Available online: https://www.jstor.org/stable/4132319 (accessed on 14 April 2023).
17. Karhapää, P.; Behutiye, W.; Rodríguez, P.; Oivo, M.; Costal, D.; Franch, X.; Aaramaa, S.; Choraś, M.; Partanen, J.; Adherve, A. Strategies to manage quality requirements in agile software development: A multiple case study. *Empir. Softw. Eng.* **2021**, *26*, 28. [CrossRef]
18. Dybå, T.; Dingsøyr, T. Empirical studies of agile software development: A systematic review. *Inf. Softw. Technol.* **2008**, *50*, 833–859. [CrossRef]

19. Page, M.J.; McKenzie, J.E.; Bossuyt, P.M.; Boutron, I.; Hoffmann, T.C.; Mulrow, C.D.; Shamseer, L.; Tetzlaff, J.M.; Akl, E.A.; Brennan, S.E.; et al. The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. *BMJ* **2021**, *372*, n71. [CrossRef] [PubMed]
20. Alam, S.; Shah, S.A.A.; Bhatti, S.N.; Jadi, A.M. Impact and Challenges of Requirements Engineering in Agile Methodologies: A Systematic Review. *Int. J. Adv. Comput. Sci. Appl.* **2017**, *8*, 411–420. [CrossRef]
21. Curcio, K.; Navarro, T.; Malucelli, A.; Reinehr, S. Requirements engineering: A systematic mapping study in agile software development. *J. Syst. Softw.* **2018**, *139*, 32–50. [CrossRef]
22. Elghariani, K.; Kama, N. Review on Agile requirements engineering challenges. In Proceedings of the 3rd International Conference on Computer and Information Sciences, Kuala Lumpur, Malaysia, 15–17 August 2016; pp. 507–512. [CrossRef]
23. Inayat, I.; Salim, S.S.; Marczak, S.; Daneva, M.; Shamshirband, S. A systematic literature review on agile requirements engineering practices and challenges. *Comput. Hum. Behav.* **2015**, *51*, 915–929. [CrossRef]
24. Okesola, J.; Adebiyi, M.; Okokpujie, K.; Odepitan, D.; Goddy-Worlu, R.; Iheanetu, O.; Omogbadegun, Z.; Adebiyi, A. A Systematic Review of Requirement Engineering Practices in Agile Model. *Int. J. Mech. Eng.* **2019**, *10*, 671–687. Available online: http://eprints.lmu.edu.ng/3119/1/Agile%20model_Okesola%20IJMET.pdf (accessed on 14 April 2023).
25. Muhammad, A.; Siddique, A.; Mubasher, M.; Aldweesh, A.; Naveed, Q.N. Prioritizing Non-Functional Requirements in Agile Process Using Multi Criteria Decision Making Analysis. *IEEE Access* **2023**, *11*, 24631–24653. [CrossRef]
26. Harvie, D.P.; Agah, A. Targeted Scrum: Applying Mission Command to Agile Software Development. *IEEE Trans. Softw.* **2016**, *42*, 476–489. [CrossRef]
27. Alsaqaf, W.; Daneva, M.; Wieringa, R. Quality requirements challenges in the context of large-scale distributed agile: An empirical study. *Inf. Soft. Technol.* **2019**, *110*, 39–55. [CrossRef]
28. Hess, A.; Diebold, P.; Seyff, N. Understanding information needs of agile teams to improve requirements communication. *J. Ind. Inf. Integr.* **2019**, *14*, 3–15. [CrossRef]
29. Firdaus, A.; Ghani, I.; Abg Jawawi, D.N.; Wan Kadir, W.M.N. Non functional requirements (NFRs) traceability metamodel for agile development. *J. Teknol.* **2015**, *77*, 115–125. [CrossRef]
30. Alrezaamiri, H.; Ebrahimnejad, A.; Motameni, H. Software requirement optimization using a fuzzy artificial chemical reaction optimization algorithm. *Soft Comput.* **2019**, *23*, 9979–9994. [CrossRef]
31. Drury-Grogan, M.L.; Conboy, K.; Acton, T. Examining decision characteristics & challenges for agile software development. *J. Syst. Softw.* **2017**, *131*, 248–265. [CrossRef]
32. Borrego, G.; Morán, A.L.; Palacio, R.R.; Vizcaíno, A.; García, F.O. Towards a reduction in architectural knowledge vaporization during agile global software development. *Inf. Softw. Technol.* **2019**, *112*, 68–82. [CrossRef]
33. Martínez-García, J.R.; Castillo-Barrera, F.-E.; Palacio, R.R.; Borrego, G.; Cuevas-Tello, J.C. Ontology for knowledge condensation to support expertise location in the code phase during software development process. *IET Softw.* **2020**, *14*, 234–241. [CrossRef]
34. Heikkilä, V.T.; Paasivaara, M.; Lasssenius, C.; Damian, D.; Engblom, C. Managing the requirements flow from strategy to release in large-scale agile development: A case study at Ericsson. *Empir. Softw. Eng.* **2017**, *22*, 2892–2936. [CrossRef]
35. Rojas, L.A.; Macías, J.A. Toward collisions produced in requirements ranking: A qualitative approach and experimental study. *J. Syst. Softw.* **2019**, *158*, 110417. [CrossRef]
36. Mishra, D.; Mishra, A. Complex software project development: Agile methods adoption. *J. Softw. Maint.* **2011**, *23*, 549–564. [CrossRef]
37. Behutiye, W.; Rodriguez, P.; Oivo, M. Quality Requirement Documentation Guidelines for Agile Software Development. *IEEE Access* **2022**, *10*, 70154–70173. [CrossRef]
38. Wagner, S.; Fernández, D.M.; Kalinowski, M.; Felderer, M. Agile Requirements Engineering in Practice: Status Quo and Critical Problems. *CLEI Electron. J.* **2018**, *21*, 1–15. [CrossRef]
39. Al-Ta'ani, R.H.; Razali, R. Process Model for Systematic Requirements Prioritisation Process in an Agile Software Development Environment Based on 5S Approach: Empirical Study. *J. Theor. Appl. Inf. Technol.* **2017**, *95*, 1715–1736. Available online: http://www.jatit.org/volumes/Vol95No8/15Vol95No8.pdf (accessed on 14 April 2023).
40. Martins, H.F.; de Oliveira, A.C.; Canedo, E.D.; Kosloski, R.A.D.; Paldês, R.A.; Oliveira, E.C. Design thinking: Challenges for software requirements elicitation. *Information* **2019**, *10*, 371. [CrossRef]
41. Mathrani, A.; Wickramasinghe, S.; Jayamaha, N.P. An evaluation of documentation requirements for ISO 9001 compliance in scrum projects. *TQM J.* **2022**, *34*, 901–921. [CrossRef]
42. Madampe, K.; Hoda, R.; Grundy, J.A. Faceted Taxonomy of Requirements Changes in Agile Contexts. *IEEE Trans. Softw.* **2022**, *48*, 3737–3752. [CrossRef]
43. Hoda, R.; Murugesan, L.K. Multi-level agile project management challenges: A self-organizing team perspective. *J. Syst. Softw.* **2016**, *117*, 245–257. [CrossRef]
44. Mishra, D.; Mishra, A. Managing requirements in market-driven software project: Agile methods view. *Tech. Gaz.* **2010**, *17*, 223–229. Available online: https://hrcak.srce.hr/en/clanak/84354 (accessed on 14 April 2023).
45. Chen, P.-S.; Chen, G.Y.-H.; Lien, S.-F.; Huang, W.-T. Using Scrum and unified modelling language to analyze and design an automatic course scheduling system. *J. Chin. Inst. Eng.* **2019**, *42*, 534–543. [CrossRef]
46. Gupta, A.; Poels, G.; Bera, P. Using Conceptual Models in Agile Software Development: A Possible Solution to Requirements Engineering Challenges in Agile Projects. *IEEE Access* **2022**, *10*, 119745–119766. [CrossRef]

47. Gaikwad, V.; Joeg, P. A Case Study in Requirements Engineering in Context of Agile. *Int. J. Appl. Eng.* **2017**, *12*, 1697–1702. Available online: http://www.ripublication.com/ijaer17/ijaerv12n8_31.pdf (accessed on 14 April 2023).
48. Tanveer, B.; Guzmán, L.; Engel, U.M. Effort estimation in agile software development: Case study and improvement framework. *J. Softw. Evol. Process* **2017**, *29*, e1862. [CrossRef]
49. Bhalerao, S.; Ingle, M. Incorporating Vital Factors in Agile Estimation through Algorithmic Method. *Int. J. Comput. Sci.* **2009**, *6*, 85–97. Available online: https://d1wqtxts1xzle7.cloudfront.net/34071468/v6i17-libre.pdf?1404103564=&response-content-disposition=inline%3B+filename%3DIncorporating_Vital_Factors_in_Agile_Est.pdf&Expires=1685553636&Signature=Ou-Js2eyMgOG7c-rMy4z8A5LJRc2mKx90BpFjMn9wwM8peXwiWnAR4fMdhk1Oz-0UOD4F2PUSIOqOWQF4ZiS7dsJIpm2VM84E0umQiHvMy0JNoLQ~CJBx0YtKszUjjQJ2XG~lUi-iHIFeJiwFO4wQf14xsXhLvNLjTbbjd7Qlbf0w8u-uQgXvKjr4NL0Xv7V-djaaHzAPF-01dMMyK6G~e7Xzl05OVE6hNJ8kSOge4rjWzWqCeAfGHEHV-r6wvhSWtnxvk35H51DgMXhDqojTpQX3h0SvXWoFKgeaT4No9Vc9zeTrRnXXpX3Oqh6p72h~r3TZq2sDtaq9wFwVboE6A__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA (accessed on 14 April 2023).
50. Alsaadi, B.; Saeedi, K. Data-driven effort estimation techniques of agile user stories: A systematic literature review. *Artif. Intell. Rev.* **2022**, *55*, 5485–5516. [CrossRef]
51. Dragicevic, S.; Celar, S.; Turic, M. Bayesian network model for task effort estimation in agile software development. *J. Syst. Softw.* **2017**, *127*, 109–119. [CrossRef]
52. Usman, M.; Britto, R.; Damm, L.-O.; Börstler, J. Effort estimation in large-scale software development: An industrial case study. *Inf. Softw. Technol.* **2018**, *99*, 21–40. [CrossRef]
53. Elghariani, K.; Kama, N.; Mohd Azmi, N.F.; Abu bakar, N.A. Implicit thinking knowledge injection framework for Agile requirements engineering. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *9*, 141–146. Available online: https://core.ac.uk/download/pdf/333872558.pdf (accessed on 14 April 2023). [CrossRef]
54. Vithana, V.N. Scrum requirements engineering practices and challenges in offshore software development. *Int. J. Comput. Appl.* **2015**, *116*, 43–49. Available online: https://pdfs.semanticscholar.org/d8e7/a8d3e7ff978a94002dacb2d50320035167d9.pdf (accessed on 14 April 2023).
55. Kasauli, R.; Liebel, G.; Knauss, E.; Gopakumar, S.; Kanagwa, B. Requirements Engineering Challenges in large-scale agile system development. In Proceedings of the 25th International Requirements Engineering Conference, Lisbon, Portugal, 4–8 September 2017; pp. 352–361. [CrossRef]
56. Saeeda, H.; Dong, J.; Wang, Y.; Abid, M.A. A proposed framework for improved software requirements elicitation process in SCRUM: Implementation by a real-life Norway-based IT project. *J. Softw. Evol. Process.* **2020**, *32*, e2247. [CrossRef]
57. Santos, P.O.; de Carvalho, M.M. Exploring the challenges and benefits for scaling agile project management to large projects: A review. *Requir. Eng.* **2022**, *27*, 117–134. [CrossRef]
58. El-Najar, T.; Ahmad, I.; Alkandari, M. Easycomm: A framework and tool to solve client communication problem in Agile development. *IAENG Int. J. Comput. Sci.* **2019**, *46*, 90–101. Available online: https://www.iaeng.org/IJCS/issues_v46/issue_1/IJCS_46_1_11.pdf (accessed on 14 April 2023).
59. Medeiros, J.; Vasconcelos, A.; Silva, C.; Goulão, M. Quality of software requirements specification in agile projects: A cross-case analysis of six companies. *J. Syst. Softw.* **2018**, *142*, 171–194. [CrossRef]
60. Kasauli, R.; Knauss, E.; Horkoff, J.; Liebel, G.; de Oliveira Neto, F.G. Requirements engineering challenges and practices in large-scale agile system development. *J. Syst. Softw.* **2021**, *172*, 110851. [CrossRef]
61. Levy, Y.; Stern, R.; Sturm, A.; Mordoch, A.; Bitan, Y. An impact-driven approach to predict user stories instability. *Requir. Eng.* **2022**, *27*, 231–248. [CrossRef]
62. Thomas, M.; Senapathi, M. Agile requirements engineering: An empirical analysis and evidence from a tertiary education context. *Issues Inf. Sci. Inf. Technol.* **2019**, *16*, 97–112. [CrossRef] [PubMed]
63. Muntés-Mulero, V.; Ripolles, O.; Gupta, S.; Dominiak, J.; Willeke, E.; Matthews, P.; Somosköi, B. Agile risk management for multi-cloud software development. *IET Softw.* **2019**, *13*, 172–181. [CrossRef]
64. Behutiye, W.; Rodríguez, P.; Oivo, M.; Aaramaa, S.; Partanen, J.; Abhervé, A. Towards optimal quality requirement documentation in agile software development: A multiple case study. *J. Syst. Softw.* **2022**, *183*, 111112. [CrossRef]
65. Jain, R.; Cao, L.; Mohan, K.; Ramesh, B. Situated boundary spanning: An empirical investigation of requirements engineering practices in product family development. *ACM Trans. Manag. Inf. Syst.* **2014**, *5*, 1–29. [CrossRef]
66. Bernier, C.; Dubé, L.; Roy, V. An Agile Method, a Contractual Relationship, and Distance: A Triad of Challenging Conditions for a Successful System Development Project. *J. Inf. Technol. Case Appl. Res.* **2012**, *14*, 30–57. [CrossRef]
67. Urbieta, M.; Antonelli, L.; Rossi, G.; do Prado Leite, J.C.S. The impact of using a domain language for an agile requirement management. *Inf. Softw. Technol.* **2020**, *127*, 106375. [CrossRef]
68. Gahyyur, S.A.K.; Razzaq, A.; Hasan, S.Z.; Ahmed, S.; Ullah, R. Evaluation for feature driven development paradigm in context of architecture design augmentation and perspective implications. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *9*, 236–247. [CrossRef]
69. Rosenberger, P.; Tick, H.J. Agile enhancement of critical PMBOK V6 processes. *J. Mod. Proj. Manag.* **2021**, *9*, 190–203. [CrossRef]
70. Al-Zewairi, M.; Biltawi, M.; Etaiwi, W.; Shaout, A. Agile Software Development Methodologies: Survey of Surveys. *J. Comput. Commun.* **2017**, *5*, 74–97. [CrossRef]
71. Amna, A.R.; Poels, G. Ambiguity in user stories: A systematic literature review. *Inf. Softw. Technol.* **2022**, *145*, 106824. [CrossRef]

72. Hoy, Z. *Dataset for Agile Requirements Engineering Challenges-Solutions Updated May 2023*; University of Portsmouth Research Portal: Portsmouth, UK, 2023. [CrossRef]
73. Alavi, M.; Carlson, P. A review of MIS research and disciplinary development. *J. Manag. Inf. Syst.* **1992**, *8*, 45–62. [CrossRef]
74. Medeiros, J.; Vasconcelos, A.; Silva, C.; Goulão, M. Requirements specification for developers in agile projects: Evaluation by two industrial case studies. *Inf. Softw. Technol.* **2020**, *117*, 106194. [CrossRef]
75. Martini, A.; Bosch, J. On the interest of architectural technical debt: Uncovering the contagious debt phenomenon. *J. Softw. J. Softw. Evol. Process* **2017**, *29*, e1877. [CrossRef]
76. Lohan, G.; Conboy, K.; Lang, M. Examining customer focus in IT project management: Findings from Irish and Norwegian case studies. *Scand. J. Inf. Syst.* **2011**, *23*, 29–58. Available online: https://aisel.aisnet.org/sjis/vol23/iss2/2/ (accessed on 14 April 2023).
77. Liebel, G.; Knauss, E. Aspects of modelling requirements in very-large agile systems engineering. *J. Syst. Softw.* **2023**, *199*, e1877. [CrossRef]
78. Canedo, E.D.; Calazans, A.T.S.; Bandeira, I.N.; Costa, P.H.T.; Masson, E.T.S. Guidelines adopted by agile teams in privacy requirements elicitation after the Brazilian general data protection law (LGPD) implementation. *Requir. Eng.* **2022**, *27*, 545–567. [CrossRef]
79. Werner, C.; Li, Z.S.; Lowlind, D.; Elazhary, O.; Ernst, N.; Damian, D. Continuously Managing NFRs: Opportunities and Challenges in Practice. *IEEE Trans. Softw.* **2022**, *28*, 2629–2642. [CrossRef]
80. Brataas, G.; Martini, A.; Hanssen, G.K.; Ræder, G. Agile elicitation of scalability requirements for open systems: A case study. *J. Syst. Softw.* **2021**, *182*, 111064. [CrossRef]
81. Oriol, M.; Martínez-Fernández, S.; Behutiye, W.; Farré, C.; Kozik, R.; Seppänen, P.; Vollmer, A.M.; Rodríguez, P.; Franch, X.; Aaramaa, S.; et al. Data-driven and tool-supported elicitation of quality requirements in agile companies. *Softw. Qual.* **2020**, *28*, 931–963. [CrossRef]
82. Traini, L. Exploring Performance Assurance Practices and Challenges in Agile Software Development: An Ethnographic Study. *Empir. Softw. Eng.* **2022**, *27*, 74. [CrossRef]
83. del Sagrado, J.; del Águila, I.M.; Orellana, F.J. Multi-objective ant colony optimization for requirements selection. *Empir. Softw. Eng.* **2015**, *20*, 557–610. [CrossRef]
84. Al-Ta'ani, R.H.; Razali, R. A framework for requirements prioritisation process in an agile software development environment: Empirical study. *Int. J. Adv. Sci. Eng. Inf. Technol.* **2016**, *6*, 846–856. [CrossRef]
85. Trkman, M.; Mendling, J.; Krisper, M. Using business process models to better understand the dependencies among user stories. *Inf. Softw. Technol.* **2016**, *71*, 58–76. [CrossRef]
86. Trkman, M.; Mendling, J.; Trkman, P.; Krisper, M. Impact of the conceptual model's representation format on identifying and understanding user stories. *Inf. Softw. Technol.* **2019**, *116*, 106169. [CrossRef]
87. de Souza, P.L.; de Souza, W.L.; Ferreira Pires, L. ScrumOntoBDD: Agile software development based on scrum, ontologies and behaviour-driven development. *J. Braz. Comput. Soc.* **2021**, *27*, 10. [CrossRef]
88. Raharjana, I.K.; Siahaan, D.; Fatichah, C. User Stories and Natural Language Processing: A Systematic Literature Review. *IEEE Access* **2021**, *9*, 53811–53826. [CrossRef]
89. Usman, M.; Petersen, K.; Börstler, J.; Santos Neto, P. Developing and using checklists to improve software effort estimation: A multi-case study. *J. Syst. Softw.* **2018**, *146*, 286–309. [CrossRef]
90. Rahy, S.; Bass, J.M. Managing non-functional requirements in agile software development. *IET Softw.* **2022**, *16*, 60–72. [CrossRef]
91. Ernst, N.A.; Borgida, A.; Jureta, I.J.; Mylopoulos, J. Agile requirements engineering via paraconsistent reasoning. *Inf. Syst.* **2014**, *43*, 100–116. [CrossRef]
92. Sarkan, H.M.; Ahmad, T.P.S.; Bakar, A.A. Using JIRA and Redmine in Requirements Development for Agile Methodology. In Proceedings of the 5th Malaysian Conference in Software Engineering, Johor Bahru, Malaysia, 13–14 December 2011; pp. 408–413. [CrossRef]
93. Farid, W.M.; Mitropoulos, F.J. NORMATIC: A visual tool for modeling non-functional requirements in agile processes. In Proceedings of the 2012 IEEE Southeastcon, Orlando, FL, USA, 15–18 March 2012; pp. 1–8. [CrossRef]
94. Farid, W.M.; Mitropoulos, F.J. Novel lightweight engineering artifacts for modeling non-functional requirements in agile processes. In Proceedings of the 2012 IEEE Southeastcon, Orlando, FL, USA, 15–18 March 2012; pp. 1–7. [CrossRef]
95. Anand, R.V.; Dinakaran, M. Handling stakeholder conflict by agile requirements prioritization using Apriori technique. *Comput. Electr. Eng.* **2017**, *61*, 126–136. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.