Requirements Engineering and Software Testing in Agile Methodologies: a Systematic Mapping

Jarbele C. S. Coutinho
jarbele.coutinho@ufersa.edu.br
Federal Rural University of the Semi-Arid (UFERSA)
Pau dos Ferros, RN, Brazil

Wilkerson L. Andrade
wilkerson@computacao.ufcg.edu.br
Federal University of Campina Grande (UFCG)
Campina Grande, PB, Brazil

Patrícia D. L. Machado
patricia@computacao.ufcg.edu.br
Federal University of Campina Grande (UFCG)
Campina Grande, PB, Brazil

ABSTRACT

The insertion of agile practices in software development has increased exponentially. Both industry and academic staff constantly face challenges related to requirements specification and testing in this context. In this study, we conducted a systematic mapping of the literature to investigate the practices, strategies, techniques, tools, and challenges met in the association of Requirements Engineering with Software Testing (REST) in the agile context. By searching seven major bibliographic databases, we identified 1.099 papers related to Agile REST. Based on the systematic mapping guidelines, we selected 14 of them for a more specific analysis. In general, the main findings include the fact that weekly meetings should be held to establish frequent communication with stakeholders. Also, most projects adopt use cases as conceptual models and perform use case detailing. Test cases are an important artifact with test case design as a Software Testing practice. For the automation of test cases, fit tables have been recommended as an artifact. Finally, proper project documentation constitutes a critical basis in Agile REST.

CCS CONCEPTS

• **Software and its engineering → Software creation and management**; Software verification and validation; Process validation; Software development process management; Software development methods; Agile software development; Designing software; Requirements analysis.

KEYWORDS

software requirements, software testing, agile, systematic mapping study

ACM Reference Format:
Jarbele C. S. Coutinho, Wilkerson L. Andrade, and Patrícia D. L. Machado. 2019. Requirements Engineering and Software Testing in Agile Methodologies: a Systematic Mapping. In *XXXIII Brazilian Symposium on Software Engineering (SBES 2019), September 23–27, 2019, Salvador, Brazil.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3350768.3352584

SBES 2019, September 23–27, 2019, Salvador, Brazil

ACM ISBN 978-1-4503-7651-8/19/09…$15.00
https://doi.org/10.1145/3350768.3352584

1 INTRODUCTION

Activities inherent to Requirements Engineering (RE) and Software Testing (ST), called by Unterkalmsteiner [4] as REST, are intended to support the development of products that meet customer expectations in terms of functionality and quality. In this context, software requirements and software testing processes must be aligned to avoid potential problems in the delivery of software products [15]. Poor communication with testers about requirements changes, for example, may result in a lack of verification of new requirements and/or incorrect verification of invalid old requirements, leading to software quality problems.

In this sense, software quality is one of the determining factors for the success of the project and, consequently, customer satisfaction. However, ensuring software quality is not a trivial task. Currently, software development teams have been focusing their practices on developing products that meet customer needs but in a scenario of continual changes and shorter deadlines. The adoption of agile methods in software development has been a common alternative adopted by companies to achieve the expected quality within this context.

Agile methods strive to respond to requirements changes by integrating requirements, design, implementation, and testing processes as simplified processes that focus on changes acceptance and adaptation [3, 16]. In agile methods, continuous and direct communication with the customer is prioritized concerning comprehensive documentation, for example. However, adopting such approaches implies adapting the development context of all phases of the project. This also includes requirements engineering and software testing activities, which are fundamental to software quality assurance.

With this perception, a few studies have been carried out to identify RE and ST practices or problems in agile projects [1, 17, 18]. However, such works do not present the synergy between these areas when executed in a context of agile methods, regarding the most common practices, strategies, techniques, tools or challenges in the alignment from Agile REST. Results concerning the relationship between acceptance tests and requirements with an emphasis on documentation are highlighted in [19], however, the methodology adopted involves traditional models besides agile methods, specifically. In this way, understanding such alignment, in an agile context, presents itself as an important topic for the agile software industry.

In this work, we raise and discuss important questions regarding RE and ST activities in agile development: 1) What are the most used practices in this context? 2) What techniques, strategies, and tools have been adopted? 3) What are the main challenges encountered

in the association of RE and ST? 4) What are the open problems identified? 5) What requirements and software testing artifacts are generated? To address these questions, aiming to complement research presented in the literature that relates REST Agile, we performed a Systematic Mapping (SM) to identify and analyze the state of the art concerning Agile REST. This method is appropriate for ample and still little-defined areas of research [20, 21]. Thus, the main contributions of this research are related to presenting the current panorama of the primary practices, challenges, and open problems presented in the context of Agile REST.

This paper is structured as follows. Section 2 presents a brief contextualization of Agile Requirements Engineering and Agile Software Testing and discusses related SM. Section 3 describes the research methodology of this SM, presenting our objective, our research questions, and our methodology, as well as the evaluation of the quality and the threats to the validity of this study. In Section 4, we present the general and specific results, in response to the research questions defined herein. In Section 5, we discuss the results obtained. Finally, Section 6 presents the conclusions of this study.

2 BACKGROUND AND RELATED WORK

In this section, we discuss important aspects related to Agile RE and Agile ST. Then we present and discuss a systematic literature review about Agile REST.

2.1 Agile Requirements Engineering

Requirements Engineering (RE) is a discipline used to describe activities related to production and requirements management [16, 23]. For this, several techniques, practices, and artifacts have been used in the development of these activities [16, 22, 23]. In agile methodologies the focus is on the software in operation and not in the generated artifacts [24], making RE a great challenge to professionals, since generating and maintaining the documentation of requirements is seen in agile approaches as a bureaucratic practice, which makes the process less agile [25].

Agile Requirements Engineering (Agile RE) has been a term used to define the "agile way" of planning, executing, and testing requirements specified during an agile process [26]. Systematic reviews of the literature show several problems related to Agile RE, such as stakeholder and user involvement, agile methodology integration, needs prioritization, multifunctional teams, requirements modeling and documentation, change management, and pre-coding tests [25–27, 42, 44].

2.2 Agile Software Testing

Agile Software Testing (Agile ST) defines a set of practices that incorporates commonly used testing techniques and considers agile values to do so [28]. For Maia et al. [18], in Agile ST, the team is not only involved in identifying failures, but also preventing them. In this way, Agile ST is a challenge for testers accustomed to using traditional methodologies, mainly because they need to be proactive and start the tests from the beginning of the project together with the developers [18].

Systematic reviews of the literature in Agile ST have pointed out: 1) an emphasis on the use of development strategies oriented to unit and integration tests and acceptance tests with the client, as well as a concern with experimental studies that can measure the benefits and difficulties of using such strategies; 2) the need for testing tools in the context of agile methods; 3) the major technical, cultural, and managerial factors driving the testing in DevOps environments; 4) factors that limit the adoption of Test Driven Development (TDD) by industry; 5) among others [38, 39, 45].

2.3 Agile REST

Literature reviews about Agile REST have been little investigated as a research area. And, when explored, the focus is on the validation or comparison of specific Agile RE or Agile ST approaches, in particular contexts, as discussed in the previous sections. A related study performed by Park and Maurer [37] examines the results of a review of the Story Test Driven Development (STDD) literature. The findings point to evidence on benefits of agile development, disabilities and other issues driven by test history. Such discoveries were categorized into seven themes: cost, time, people, code design, test tools, what to test, and test automation. The results also point out the need for empirical research on these themes.

This research differs from others because it seeks to map and categorize studies directly related to Agile REST, to present the current panorama of the main practices, challenges, and problems presented in this area, and to share the gaps and opportunities of study in this field.

3 RESEARCH METHODOLOGY

The research methodology of this study is based on guidelines proposed by Kitchenham and Charters [20], and we partially implemented the mapping process provided by Petersen et al. [21], complemented by the recent quality assessment guidelines proposed by Ivarsson and Gorschek [41]. The general research process consisted of five steps: Research Methodology (see Subsection 3.1), Search Strategy (see Subsection 3.2), Data Filtering Strategy (see Subsection 3.3 and Subsection 3.4), Data Extraction and Analysis Processes (see Subsection 3.5). The search process is illustrated in Figure 1.

3.1 Objective and Research Questions

The objective of this work is to investigate what is reported in the main bibliographic databases on the alignment of REST in the agile context. Previous research (shown in Section 2) indicates that although these areas are often studied, the alignment between them is still poorly investigated. Studies that relate Agile REST were selected and analyzed in order to answer the following central research question: What is the current research landscape of English scientific publications on REST alignment in the agile context? This led us to the following specific research questions (RQs): ***RQ1:*** What are the most used practices in this context? ***RQ2:*** What techniques, strategies, and tools have been adopted? ***RQ3:*** What are the main challenges encountered in the association of RE and ST? ***RQ4:*** What are the open problems identified? ***RQ5:*** What requirements and software testing artifacts are generated?

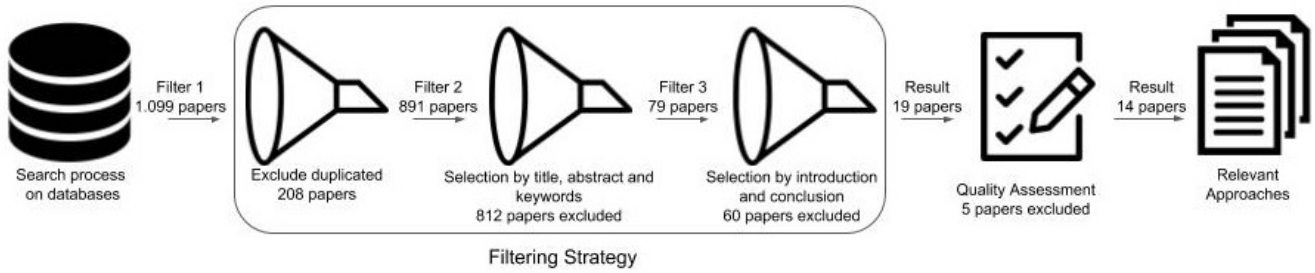

Figure 1: Search process.

3.2 Search strategy

The search string is based on the terms *"test AND software AND requirement AND agile"*, that were defined from a set of base articles related to the context of Agile RE and Agile ST [15, 17, 18, 24, 25, 27, 36, 39]. In this way, we noticed that some terms were standard of these areas, so they were adopted as search string search. In contrast, we noticed that some other synonyms (such as verification, validation, among others) were not commonly used as standard terms in these areas. The term "software" was used to exclude articles related to other areas of study beyond Software Engineering. Moreover, we did not identify cases where the use of specific agile methodologies returned works without the agile term.

As primary sources, we have used the following online digital libraries: ACM Digital Library, Compendex, Google Scholar, IEEE Xplore, Science Direct, Scopus, and Springer Link. These libraries were selected because they meet the following criteria: *(i)* the publication database is regularly updated; *(ii)* the manuscripts are available for download; *(iii)* the manuscripts are reviewed according to a peer-review process; *(iv)* and, for the most part, they are able to handle advanced queries. The search string has been applied to metadata (for example, titles, abstracts, and keywords) in the selected digital libraries.

3.3 Filtering Strategy

For the inclusion of an article, its relevance was determined in relation to the research questions, by analyzing the title, keywords, and summary. Each article was reviewed only by a single author in consultation with the other authors. Steps have been taken to evaluate the final set of included articles to reduce threats to validity (Section 3.6). The inclusion criteria (IC) and exclusion criteria (EC) are presented in Table 1.

The selection process of primary studies was conducted using a three-step screening procedure (Figure 1). In the first phase, only the titles, abstracts, and keywords of the studies were considered from the review based on the inclusion and exclusion criteria. In the second phase, the introduction and conclusion were read, also considering the inclusion and exclusion criteria. We emphasize that we use an inclusive approach in both phases to avoid premature exclusion from studies, that is, if there is a doubt about a study, such a study should be included in the next step. In the third phase, the full texts were read for quality evaluation of the study (see Subsection 3.4).

Table 1: Inclusion and exclusion criteria

Criteria	Id	Description
Inclusion	IC1	Papers focusing on the context of this protocol
	IC2	Presentation of complete results (with experiments and analysis of results)
	IC3	Papers applied in industry
Exclusion	EC1	Duplicate or repeated papers in more than one source
	EC2	Papers without access available
	EC3	Incomplete papers, drafts, slides or abstracts
	EC4	Papers with content irrelevant to research
	EC5	Papers not written in English
	EC6	Papers that do not address agile methodology
	EC7	Theoretical studies that do not present a practical validation
	EC8	Papers published before January 2008

Table 2: Reasons for exclusion of papers

Description	EC identifier	Number of papers
Lack of a focus on Agile REST	EC4, EC6	693 (63.05%)
Repetition	EC1	208 (18.92%)
Published until December 2007	EC8	53 (4.82%)
Paper format (editorial, seminar or discussion)	EC3	110 (10.00%)
Lack of access	EC2	28 (2.54%)
Lack of quality	EC7	05 (0.45%)
Not written in English	EC5	02 (0.18%)

At each stage, studies were excluded for several reasons related to the exclusion criteria (reasons are listed in Table 2). Several studies have appeared in more than one research source. We eliminated the repetitions and considered only one occurrence of a study. Details on repeated studies provide no meaningful information.

3.4 Study quality assessment

Quality evaluation was performed with the 19 primary studies included in this phase. After applying the quality criteria, 5 papers were excluded. The quality of this mapping was evaluated according to the quality criteria defined in the tertiary study carried out by Kitchenham [20].

All primary studies were ranked based on how well they met the quality criteria described in Table 3. The following points system was used to determine the individual scores: Yes (Y) = 1 point, Partial (P) = 0.5 points, No (N) = 0 point. The overall quality score was calculated by summing the score reported on the four individual criteria. Thus, the total quality score for each study ranged from 0 (very poor) to 4 (very good), as can be seen in Table 4.

3.5 Data Extraction and Analysis Processes

After completing the study selection process, we recorded the basic information of each article in the form of data extraction. Data extraction was performed using a structured extraction form in Microsoft Excel to capture all information, for each included article, needed for later synthesis. The data extracted from the included articles were analyzed with the objective of answering the research questions using descriptive analysis for the quantitative data and using the thematic analysis approach for the qualitative data. In Section 4 these results are exposed and discussed.

Kitchenham and Charters [20] and Kitchenham et al. [43] states that there are two main methods of data synthesis: descriptive (qualitative) and quantitative. The extracted data were analyzed using a qualitative method to answer our research questions, which leads to a synthesis of descriptive data. For the synthesis, we used the content analysis to structure the results. In addition, as answers to each research question are summaries and narrative summaries, we initially classify the extracted data using the scheme defined in Subsection 3.5. This led to the results detailed in Section 4.

3.6 Threats to validity

During the planning and execution of this mapping, we noticed that some factors were characterized as threats to the validity of the research. There is a threat to **construction validity** when deciding which studies should be included in the review. However, we believe that this aspect has been attenuated, since we executed a protocol [20, 21, 43], and supplemented it with the quality assessment guidelines proposed by Ivarsson and Gorschek [41]. The mapping protocol was also iteratively revised by two other researchers with practical experience in the research areas we studied.

Regarding **external validity**, the results of this review cannot be generalized because they are based on a specific set of keywords in the search sources that were used for data collection and at a particular time interval (last ten years). Therefore, our results may be limited. However, some strategies such as the definition of a broader search string and the inclusion of non-exclusive search sources in the area of Software Engineering have been taken to obtain a greater number of possible primary works.

Internal validity can be raised due to replications of similar works. To mitigate such a threat, some measures were taken: *(i)* a mapping protocol (described in Section 3) was defined and strictly followed, considering mainly the exclusion and inclusion criteria; *(ii)* where necessary, two researchers with extensive experience in this area of study were consulted until we reached a consensus on the acceptance of the identified studies. However, it may be possible that, if other researchers replicate this mapping, small variations in the identified studies can be observed due to differences in personal aptitude and reflection on the returned studies.

A potential threat to **theoretical validity** is the small sample size of the studies evaluated. This sample may not provide an accurate and complete explanation of all approaches used in Agile REST. Thus, to overcome this bias, we carried out a continuous discussion and revision of the mapping by two other experienced researchers.

The threat to **objectivity validity** comes from the inclusion and exclusion of studies during the selection process. The bias could lead to the exclusion of studies that should have been included. This threat was handled through the development of formal inclusion and exclusion criteria and through a random review of studies that were considered excluded. The inclusion and exclusion criteria were evaluated jointly by the authors.

To mitigate the threat to **interpretive validity**, inclusion and exclusion criteria were formulated to identify the studies, and a data collection form was developed to extract and analyze data systematically. The quality assessment criteria were developed to systematically evaluate the studies and reduce bias when evaluating them. However, the interpretation remains subjective to the view of the reviewers.

Finally, **repeatability** is ensured by describing the details of the steps that govern the protocol of this mapping, presented in Section 3. These steps ensure that the protocol of this study can be used to identify similar results and conclusions.

4 RESULTS

In this section, we describe the results of the mapping study. We present quantitative information on the identified studies (Subsection 4.1) followed by an analysis of the research questions formulated in Subsection 3.4.

4.1 General Results

Here we present some general results obtained with the analysis of the primary works.

4.1.1 Publications over time. We observed the frequency of studies published over the last ten years. Based on the year of publication, the identified studies were categorized. The first study identified in this time frame is from 2010. The year during which the largest number of Agile REST studies were published was in 2015, with 28.57% of the work identified in this mapping. The years of publication comprised from 2010 to 2017. We noticed a steady trend of studies published between the years 2010 to 2015.

Regarding the distribution of countries, according to the year in which they were published. From 2008 and 2017, Sweden concentrated 35.71% of research on Agile REST, United States concentrated 14.28% of the studies, while the other identified countries concentrated 7.14%, equivalent to 1 study per country.

4.1.2 Types of research. We found out that 42.85% refers to case studies. While another 35.71% performed experiments as a research method to obtain data. The other types of research identified were

Table 3: Quality assessment criteria

	Yes (1.0 score)	Partially (0.5 score)	No (0 score)
Q1. Does the primary study describe the research methodology employed?	Research methodology described in detail	Research methodology is implicit	The methodology adopted was not informed in the text and can not be inferred
Q2. Does the study highlight at least one practice, strategy, tool or technique adopted in agile requirements and/or tests?	One or more approaches (practice, strategy, tool or technique) in both areas (agile requirements and tests)	Only approaches adopted in one of the areas (requirements or agile testing), but not both	It does not emphasize or point out any
Q3. Do the presented results refer to studies carried out in real scenarios of agile teams?	Four or more real scenarios were studied	Between two and three real scenarios were studied	Only one or no real scenario was studied
Q4. Have the data / results been clearly described?	Results presented and discussed clearly	Results presented and discussed superficially	The results were presented but not discussed

Table 4: Score by quality range

Low	Average	High
$0 \leq N \leq 1.0$	$1.5 \leq N \leq 2.5$	$3.0 \leq N \leq 4.0$

perceived in only 1 article each. Thus we had a survey, a tool validation and a quantitative research, equivalent to 7.14% each, in relation to the totality of the work of this mapping. Table 5 shows the distribution by type of research carried out.

4.1.3 Type and forums of publication. The Agile REST studies identified were published in workshops, conferences, and journals. Thus, we found that conferences are the most used type of publication since 57.14% of the studies were published in this medium. The next most popular type was publication in journals, with reference to 28.57% of the studies. Finally, 14.28% of the studies were published in specific workshops in the area of Requirements Engineering or Testing. Table 6 shows the distribution by publication types.

Analysis of the primary study publication forums reveals that several different forums have published research on Agile REST. Each identified study was published in a specific publication forum such as 2nd International Workshop on Requirements Engineering and Testing, Information and Software Technology, Transactions on Software Engineering and Methodology, 5th ACM/SPEC International Conference on Performance Engineering, among others.

4.2 Specific Results

In this section, we answer the research questions of the study and discuss the results achieved.

4.2.1 **RQ1***: What are the most used practices in the context?* Some studies highlight common practices among them, such as: holding weekly meetings to establish frequent communication with project stakeholders [1, 3, 13]; the adoption of conceptual models of Use Cases and the detailing of Use Cases [1, 6, 7, 9]; the description of acceptance test cases [3, 7, 19] or execution of test cases [6]; and, the use of FitNess tables [1, 13] for the automation of the tests. Other more specific practices are also pointed out in the studies:

- The use of tools to define the flow of activities and artifacts, estimates of time and effort provided directly by the teams involved, and periodic reviews of generated Gantt diagrams; the demonstration of artifacts or software versions developed at the end of each sprint is also emphasized [1].
- The obtaining test cases from models, called Model-Based Testing (MBT), in order to verify if what was implemented in the code conforms to its specification in the model [2].
- The communication direct and frequent among stakeholders, the specification of requirements as acceptance test cases, for project documentation [3].
- Test Driven Development (TDD) for incremental coding [5].
- Specifics notations are used to generate the test case of a performance requirement, considering an activity sequence that starts with the identification of the functional requirements and closes with system optimization, after analyzing the test output of performance [6].
- The use of conceptual models such as the BPMN (Business Process Model and Notation) notation, the Use Case Diagram, and a traceability matrix were used as the basis for the execution of the Action-Triad method; and to generate test cases it highlights the use of the Unified Modeling Language (UML) Activity Diagram [7].
- Requirements specification, the use of User Stories [8].
- Other common practices are: writing acceptance test cases based on specification requirements, which are not always complete; writing acceptance tests mostly manually, using a specific format; the acceptance testing documentation is updated less frequently on teams with good communication; and, the number of requirements and tests have a strong correlation in the design [19].
- Developers create FitNess tests from requirements in Fit table format, discussed twice a week with the client [13].
- Practices pointed out from the perspective of Requirements Engineering, to mention: management of requirements through a traceability tool that supports version control; prioritization and updating of requirements, along with ongoing planning [14].

Table 5: Distribution of studies by type of survey

Approach	Studies	Quantity (%)
Case Study	[3], [7], [19], [11], [13], [14]	42.85%
Survey	[1]	7.14%
Tool Validation	[2]	7.14%
Quantitative Research	[4]	7.14%
Experimental Research	[5], [6], [8], [9], [10]	35.71%

Table 6: Publication types

Publication	Number of studies	Studies
Workshop	2	[1], [14]
Conference	8	[2], [5], [6], [7], [8], [9], [19], [13]
Journal	4	[3], [4], [10], [11]

4.2.2 **RQ2***: What techniques, strategies, and tools have been adopted?* To answer this research question, we identify the techniques, strategies and tools adopted in this context. We consider a technique, the whole set of procedures based on a scientific knowledge inherent to the Agile REST activities; and, a strategy, every set of procedures that follows a plan or a method, where steps must be followed. Below, we present and discuss each of these approaches.

Through the analysis of the 14 studies returned in this mapping, we identified some of the main techniques adopted in Agile REST in 6 studies, to mention: Model V [1, 10] a REST taxonomy [4], the use of conceptual models [7], the automated approach to generate test cases [8], and Acceptance Test Driven Development (ATDD) [13].

Exactly 7 studies highlighted the adoption of specific strategies in Agile REST, to cite the use of: an agile methodology called PARFAIT [1]; agile simulation guided by tests [2]; prioritization of tests [5]; a notation for performance testing [6]; a method called Action-Triad Method [7]; text mining [8]; and, the FitNesse method [1, 13].

To develop or implement some of the techniques and strategies described above, it is necessary to use specific software tools (see Table 7). For the development of activities related to RE, the following tools were highlighted: Jira, PapyrusUML, Syntony Framework, and Microsoft Office Word. For the development of ST related activities, the following tools were highlighted: Confluence, Veritas, JMeter, Text2Test, Framework for Integrated Testing (FIT), Microsoft Office Excel, Test Rail, Automation Server, TestLink, Cucumber, and Selenium. Among the studies analyzed, only one of them [10] pointed to the adoption of a tool to coordinate activities in REST of agile projects, called REST-bench.

In some cases, studies describe similar approaches. In others, either they point to specific approaches or they do not present all the approaches investigated in this mapping. A few studies have highlighted, simultaneously, the technique, strategy, and tool adopted or indicated for the activities carried out in Agile REST.

4.2.3 **RQ3***: What are the main challenges encountered in the association of RE and ST?.* We extract the perceived and highlighted challenges in each study, and categorize them according to each reference. Some papers highlight specific challenges in each area (RE and ST). In RE, for example, the following challenges are identified: elicitation and verification of requirements; changing management; maintaining documentation; maintaining diagrams, such as the activity diagram; the use of User Stories is not enough to support dependent tasks; Fit tables are not useful for all types of requirements; and, creating and maintaining the generated artifacts in the requirements phase. In ST, few challenges were cited, although most of them are related to the activities carried out in the requirements phase. The automation of the tests was therefore highlighted; the need to define the prioritization of test cases; the involvement of testers in requirements activities; acceptance tests are performed manually; as model-based tests increase demand in design, they become more complex and difficult to control; and, to achieve the maturity of the testing processes. Table 8 associates the challenges to the studies that reported them.

Finally, through the analysis of the studies of this mapping, we identified that a great part of them (approximately 78.5%) do not mention the challenges faced in the Agile REST association, but separately emphasize the challenges encountered when performing RE and ST activities. Few papers point to more than one challenge

Table 7: Tool adopted in Agile REST

Tool for Requirements	Studies
Jira	[1], [19]
PapyrusUML	[2]
Framework Syntony	[2]
Word	[19]
Tool for Tests	**Studies**
Confluence	[1], [19]
Veritas	[2]
JMeter	[6]
Text2Test	[9]
Fit	[13]
Excel, Test Rail, Automation Server, TestLink, Cucumber, Selenium	[19]

Table 8: Challenges with agile activities in Requirements Engineering (RE) and Software Testing (ST)

Challenges with RE	Studies
Elicitation and verification	[3]
Change Management	[3]
Keep documentation	[4]
Maintaining the Activity Diagram	[7]
User Stories are not enough to support dependent tasks	[19]
Create and maintain the requirements artifacts	[14]
Challenges with ST	**Studies**
Automation of tests	[1]
Define prioritization of test cases	[5]
Involve testers in requirements activities	[19]
Manually accepted acceptance tests	[19]
Maturity of testing processes	[14]
Control of the model-based tests	[2]

Table 9: Challenges in alignment Agile REST

Challenges in Agile REST	Studies
Communication between heterogeneous teams Combine agility and process control Allocate resources and estimate time When to close a sprint and when to iterate for previous sprints?	[1]
Lack of communication about documentation Consider the vision document in the project	[19]
Fit tables not useful for all types of requirements Maintain iterative communication with the customer	[13]

in each area (RE and ST). Others point to specific challenges due to a technique or strategy. Or, challenges due to the adoption of agile methodologies.

A few studies (approximately 21.4%) highlighted the perceived challenges of aligning REST activities in an agile context, to mention: communication between heterogeneous teams; combining agility and process control; allocating resources and estimate time; when to close a sprint and when to iterate to previous sprints; lack of communication on documentation; considering the vision document in the project; and, maintaining iterative communication with the customer. Table 9 presents the main challenges highlighted in this association, related to the respective study that presents them.

4.2.4 **RQ4***: What are the open problems identified?* To address this question we considered possible future studies, research lines not yet explored and/or problems still to be resolved. The primary studies analyzed present several themes of proposals for future studies. The authors report, for example, studies focusing on: automation of test activities, validation of use case effectiveness, strategies to maintain requirements documentation and acceptance test, strategies for consolidation of creation and maintenance of requirements artifacts, among other topics. It is important to mention that some papers did not present evidence of future research, while others suggested more than one theme. In the sequel, we present possibilities of categorized study and we discuss research themes:

- *Conduct studies focused on the automation of testing activities.* Tonella and Tiella [1] propose the development of studies focusing on the connection between the constraints expressed in the conceptual model of the software and the properties to be tested, in order to also automate the activity of testing based on restrictions, which is still a quite demanding and humane task. Another possibility of study is to research on the automation of GUI (Graphical User Interface) testing.
- *Investigate factors, limits, and artifacts generated in test cases as requirements.* Bjarnason et al. [3] propose documentary studies on the artifacts generated in the practice of test cases used as requirements (TCR), as well as the investigation of factors, relationships and limits when introducing TCR and how these aspects are affected by different contexts (such as project size, volatility of requirements, number and types of stakeholders involved), with the aim of providing more specific recommendations to teams and improving the effectiveness of agile development.
- *Conduct experiments that validate the effectiveness of use cases.* It is suggested in Sinha et al. [9] the experimentation of use cases in agile development processes, in order to evaluate the effectiveness of this approach.
- *Explore challenges from the agile testing perspective.* Neto et al. [14] propose to expand the case study based on the challenges and practices of Agile REST, through the analysis of an agile testing perspective, since the work focused more on an agile requirements perspective.
- *Investigate strategies for consolidating the creation and maintenance of requirements artifacts.* Neto et al. [14] also propose the investigation of strategies to consolidate the creation and maintenance of requirements artifacts, through the collection of data in companies, to obtain more RE.
- *Explore strategies to maintain documentation of requirements and acceptance testing*. Hotomski et al. [19] propose to explore the existing communication gap between the professionals involved with the requirements documentation and acceptance testing (since they are performed as two separate tasks and by different people) for a deeper understanding of the risks and challenges that exist during document development and maintenance.
- *Automatically generate test cases from different UML diagrams.* Elghondakly et al. [8] propose to investigate strategies to automatically generate test cases using different UML diagrams, besides to use the requirements specification.
- *Tool Development*. Hotomski et al. [19], suggest the development of a tool that supports the documentation update since it is still a manual task. And, Tiwari and Gupta [5] propose the development of a tool that supports the construction of an actor-oriented activity diagram from the use-case specification.
- *Expand the validation of a specific approach adopted in the research.* Some studies propose: the reapplication of the Action-Triad method with other functionalities in order to verify

its viability [7]; validate the usability of the REST taxonomy, supported by other tools for mapping the artifacts [10]; and to investigate the use of ATDD from an RE perspective, since there are few studies on this topic [13].
- *Improve search results.* Djanatliev et al. [2] and Unterkalmsteiner et al. [4] propose to continue the search for the improvement and extension of the results.

Finally, some works do not highlight proposals for future work such as Sorensen [6] and Bjarnason et al. [11].

4.2.5 **RQ5***: What requirements and software testing artifacts are generated?* The identified artifacts were generated for both requirements documentation and subsequent testing, based on the approaches (techniques, strategies, and tools) highlighted in the responses to RQ2. In RE related activities, the following artifacts were pointed out: a detailed use case document [1, 5, 6, 9]; UML (Unified Modeling Language) diagrams, such as class diagram, packet diagram and state diagram [2], activity diagram [5, 7] and use case diagram [7]; business models, using BPMN (Business Process Modeling Notation) notation and traceability matrix [7]; and, User Stories [8, 19]. While in the ST related activities, the following artifacts were highlighted: description of test cases [1–3, 5–9] and Fit tables [13]. In some cases, studies highlight similar artifacts. In others, they point to specific artifacts. Or, they do not inform the type of artifact generated in each activity, RE or ST [10, 11, 13].

5 DISCUSSION

In this section, we relate and discuss the results presented in Section 4 addressing key findings, contributions, and opportunities for further research.

5.1 Main findings and their implications

The synthesis of the analyzed data reveals five findings (F) derived from the relationship between the practices and the other approaches identified in this mapping. Such findings result in implications for software organizations that adopt agile methodologies and attempt to manage REST activities in this context.

F1. *Weekly meetings should be held to establish frequent communication with project stakeholders.* For this, some approaches are adopted to support such practice, for instance, Model V, the PARFAIT process and the use of the Jira and Confluence tools, both cited in Tonella and Tiella [1], to support the parallel development of requirements and tests. Moreover, Accepted Test Driven Development (ATDD) in conjunction with the FitNesse method and the Framework for Integration Testing (FIT), highlighted in Haugset and Stalhane [13], to support writing the tests from the business requirements defined by the stakeholders. Furthermore, the use of test cases as requirements to support the elicitation, validation, and management of requirements that constantly change due to customer requests [3]. In addition, some common artifacts generated with this practice are Detailed Use Cases, Test Cases, and Fit Tables. Thus, to ensure productivity in the face of the increasing pace of change that can occur with the practice of this discovery, we reinforce some recommendations that anticipate the challenges and risks of this initiative:

(1) Fit tables are not useful for all types of requirements;
(2) It is necessary to estimate the time allocated for each meeting, as well as to allocate resources accurately;
(3) It is necessary to define the sprints based on the client's requests, but focusing on the set of requirements and goals that can be developed by the team, that is, to define reachable sprints for the team;
(4) to define the closing of a sprint, as well as the need to iterate for an earlier sprint;
(5) It is important to maintain iterative communication with the client;
(6) It is necessary to seek strategies that help communication between heterogeneous teams;
(7) It is necessary to select suitable test tools for the automation of tests.

F2. *Adoption of conceptual models of use cases and the detailing of use cases* has been successful in the Agile REST domain [1, 6, 7, 9]. Used in some cases as design artifacts, such models have contributed to better compliance of requirements and higher quality of test cases generated. Model V, the PARFAIT process and the use of the Jira and Confluence tools, are highlighted in Tonella and Tiella [1] as approaches that support the parallel development of requirements and tests, as mentioned above. The use case models also support the detailing of the performance requirements used to generate performance test cases, with the aid of a specific [6] notation and the JMeter tool, for the automation of these test cases. Another common strategy is the use of a method that concentrates conceptual modeling on a triad of models, called Action-Triad Method [7]. In this sense, associated with Use Case we have the BPMN model and the traceability matrix to express the requirements and support the creation of the test cases. The Activity Diagram is also referred to as an artifact that complements this method. Some key elements for specifying a Use Case [9], must also be considered: Use Case models must contain syntactic and functional information; the refinement of the Use Case should be done through a cycle of critical analysis; and, the Use Case must meet the quality properties in automated mode. To assist in the automatic inspection of the use case specification the Text2Test tool was indicated more suitable. Finally, we reinforce some recommendations:

(1) the Action-Triad Method captures requirements systemically and assists the challenge of developing test cases before writing the code;
(2) maintaining the Activity Diagram becomes a complex task as the application undergoes frequent changes in its functionalities.

F3. *Test cases have been widely adopted in Agile REST*. In some cases, used as project artifacts, the test cases are presented as system requirements, supporting the elicitation, validation, and management of new requirements or of requirements that change constantly [3]. The prioritization of test cases, according to the priority given by the client, is also an important aspect in this context [5]. Besides, for incremental coding, obtaining and prioritizing test cases is a prerequisite for TDD [5]. User Stories also support the development of [8] test cases, as well as the Action-Triad Method, supported by Activity Diagrams [7]. One tool adopted in automating test cases is JMeter [6]. Other tools are listed in Section 4.2.2.

F4. *Fit tables for the automation of tests* have also been adopted as an artifact in ST. The tests are written in the form of simple tables (Fit Tables) based on a strategy called FitNesse [1, 13]. A specific framework is used, Fit (Framework for Integrated Test), where the acceptance test cases are expressed. The ATDD technique is adopted as a complement to the use of FitNesse. However, not all requirements can be written through Fit Tables. Another observation to consider is if developers neglect to upgrade the tests, the acceptance testing framework becomes difficult to maintain and will offer few benefits.

F5. *Project documentation provides a critical basis for the alignment of Agile REST*, since we realize that in agile development, the generated documentation is treated as a result, not as an input. Hotomski et al. [19] point out that human aspects, such as cooperation and communication between staff, are central to the development of clear documentation. The lack of communication between the professionals involved with the documentation of requirements and acceptance tests poses a challenge to be explored in this context. Another critical factor related to such alignment is the practice of documenting requirements well. We note that in Agile REST, requirements are commonly documented in the form of User Stories [8, 19], while the view document is not made [19] and there are difficulties in integrating documentation and ST. However, the results of the cited researches show that User Stories are not sufficient to support dependent tasks in the requirements specification. The Action-Triad Method also generates conceptual models that can be used to document requirements, and tests, in the form of test cases. We then realized that agile development not only changed the format of requirements, but also brought new documentation practices. However, practices and tools that support the documentation update need to be incorporated or developed to support this activity, since it is still a manual task.

5.2 Contributions pointed out in primary studies

The primary studies also indicated some general contributions of the practices and approaches used in the alignment of Agile REST, as:

(1) Conceptual models can be systematically validated iteratively during the development phase;
(2) Obtaining and prioritizing test cases for incremental coding becomes a prerequisite for TDD;
(3) The actor-oriented activity diagram approach is a good artifact for understanding the dependencies between use cases belonging to the same actor or different actors;
(4) Prioritization of the tests, based on actor-oriented activity diagrams, can be used to improve change management and control changes made at the use case level;
(5) Performance requirements can also be assigned to User Stories, although individual steps in use-case scenarios can be useful to indicate response time and interaction;
(6) Some models such as BPMN, Use Case, Traceability Matrix and Activity Diagram assist in the creation of test cases;
(7) A good specification of use cases, increases productivity and a higher quality of test cases;
(8) Human aspects such as, cooperation and communication, are central factors in the alignment of Agile REST;
(9) RE can benefit from the greater involvement of testers;
(10) ATDD provides detailed documentation of requirements in an iterative manner.

5.3 Opportunities for further research

The results found in this Mapping point to several Agile REST research opportunities. Some possibilities of study were highlighted in Section 4.2.4, extracted from the primary studies analyzed. However, we perceive a need to expand these results, both from the point of view of empirical studies and in the execution of theoretical researches that consolidate the alignment of the approaches adopted in these areas.

Practices, techniques, strategies, tools, and challenges should also be investigated through empirical studies with agile teams working in the industry to verify the use of other approaches in these categories and especially to investigate the most common design problems in the development of software that tries to align REST activities in an Agile context. In addition, human and organizational problems can be investigated, as they also characterize one of the main challenges in this association, according to [1, 13, 19].

Another research that can be performed is the definition of metrics to more objectively evaluate some of the strategies of elicitation, specification and documentation of requirements, as well as the strategies of generation and automation of cases of acceptance tests, considering that these practices are considered the most common in Agile REST [1, 3, 14, 19]. From these metrics, some experiments can be performed to empirically investigate the relationship between practices in the context of agile development.

It is important to carry out detailed theoretical research, aiming at proving and/or discussing the approaches indicated in these areas. For this, we emphasize that other studies, similar to ours, can be carried out in different contexts and objectives, both to increase confidence in the external validity of these results and to help identify more specific perceptions of Agile REST alignment - that cannot be covered, completely, with the analysis of the primary studies of this mapping.

Furthermore, a comparative analysis of results from theoretical research and empirical research becomes crucial for the understanding and alignment of the activities developed and the artifacts generated in this context, to provide guidance to professionals working with Agile REST.

6 CONCLUSION

In this paper, we report the results of a systematic mapping on the alignment between the areas of Requirements Engineering and Software Testing in the context of Agile methodologies, with the objective of mapping and categorizing studies directly related to this theme to later support the development of more specific research.

The initial search returned 1.099 unique results. The application of the screening and validation process (see Section 3) of each study resulted in the inclusion of a final set of 14 primary studies, which were then thoroughly analyzed. We emphasize in each of them: the practices, techniques, strategies and tools adopted; the main challenges encountered in the association of these areas; the

possibilities of study in this context; and the software requirements and software artifacts generated in order to answer the research questions (see Section 3.1) that led to this mapping.

In general, the synthesis of the data led us to the five findings derived from the relationship between the practices and the other approaches identified in the studies. Such findings result in implications for software organizations that adopt agile methodologies and attempt to manage REST activities in this context. We conclude, therefore, that in Agile REST: weekly meetings should be held to establish frequent communication with stakeholders; conceptual models of use cases and the detailing of use cases are adopted in most software development projects; test cases are also adopted as ST artifacts and practices in this context; Fit tables have been recommended for the automation of the tests; and good project documentation becomes essential to alignment.

We have observed that, in recent years, Agile REST has been an area of study that has not been explored in an associated way, since few studies show such alignment. Therefore, we hope that the findings of this research contribute to a better understanding of the current panorama of the main practices, approaches, and challenges of Agile REST, as well as the evidence that indicates new research opportunities in this area.

ACKNOWLEDGMENTS

The second author is supported by National Council for Scientific and Technological Development (CNPq)/Brazil (process 315057/2018-1).

REFERENCES

[1] P. Tonella, R. Tiella, Weekly Round Trips from Norms to Requirements and Tests: an Industrial Experience Report, in: 2nd RET, pp. 20-26, 2015.
[2] A. Djanatliev, W. Dulz, R. German, V. Schneider, Veritas-a versatile modeling environment for test-driven agile simulation, in: WSC, 2011, pp. 3657-3666.
[3] E. Bjarnason, M. Unterkalmsteiner, M. Borg, E. Engström, A Multi-Case Study of Agile Requirements Engineering and the Use of Test Cases as Requirements, in: Information and Software Technology, vol.77, 2016, pp. 61-79.
[4] M. Unterkalmsteiner, R. Feldt, T. Gorschek, A taxonomy for requirements engineering and software test alignment, ACM Trans. Softw. Eng. Methodol. 23 (2) (2014) 16:1-16:38.
[5] S. Tiwari, A. Gupta, An approach of generating test requirements for agile software development, in: ISEC, 2015, pp. 186-195.
[6] M. H. Sørensen, Use case-driven performance engineering without concurrent users, in: ICPE, 2013, pp. 3-12.
[7] P. Bera, A. Gupta, A Proposed Method in Agile Practices to Create Requirements Documentation and Test Cases, in: 29th CAiSE, 2017, pp. 113-121.
[8] R. Elghondakly, S. Moussa, N. Badr, Waterfall and agile requirements-based model for automated test cases generation, in: ICICIS, 2015, pp. 607-612.
[9] A. Sinha, S. M. Sutton, A. Paradkar, Text2Test: Automated inspection of natural language use cases, in: ICST, 2010, pp. 155-164.
[10] M. Unterkalmsteiner, T. Gorschek, R. Feldt, E. Klotins, Assessing requirements engineering and software test alignment - Five case studies, Journal of Systems and Software 109 (2015) 62-77.
[11] E. Bjarnason, P. Runeson, M. Borg, M. Unterkalmsteiner, E. Engström, B. Regnell, G. Sabaliauskaite, A. Loconsole, T. Gorschek, R. Feldt, Challenges and practices in aligning requirements with verification and validation: a case study of six companies, Empirical Software Engineering 19 (2014) 1809-1855.
[12] S. Hotomski, E. B. Charrada, M. Glinz, An Exploratory Study on Handling Requirements and Acceptance Test Documentation in Industry, in: 24th RE, 2016, pp. 116-125.
[13] B. Haugset, T. Stalhane, Automated acceptance testing as an agile requirements engineering practice, in: 45th HICSS, 2012, pp. 5289-5298.
[14] F. G. O. Neto, J. Horkoff, E. Knauss, R. Kasauli, G. Liebel, Challenges of Aligning Requirements Engineering and System Testing in Large-Scale Agile: A Multiple Case Study, in: 25th REW, 2017, 315-322.
[15] A. Bertolino, Software testing research: Achievements, challenges, dreams, in: Future of Software Engineering, IEEE Computer Society, 2007, pp. 85-103.
[16] I. Sommerville, Software Engineering, Addison-Wesley, 2011.
[17] S. Wagner, D. M. Fernandez, M. Felderer, M. Kalinowski ,Requirements engineering practice and problems in agile projects: results from an international survey, in: XX CIBSE, 2017.
[18] N. Maia, G. Macedo, E. Collins, A. D. Neto, Aplicando Testes Ágeis com Equipes Distribuídas: Um Relato de Experiência, in: SBQS, 2012, pp. 365-372.
[19] S. Hotomski, E. B. Charrada, M. Glinz, An exploratory study on handling requirements and acceptance test documentation in industry, in: 24th RE, 2016, pp. 116-125.
[20] B. Kitchenham, S. Charters, Guidelines for performing systematic literature reviews in software engineering, 2007.
[21] K. Petersen, S. Vakkalanka, L. Kuzniarz, Guidelines for conducting systematic mapping studies in software engineering: An update, Journal of Information and Software Technology 64 (2015) 1-18.
[22] R. H. Thayer, M. Dorfman, Software Requirements Engineering, IEEE Computer Society Press, 1997.
[23] R. S. Pressman, Software engineering: a practitioner's approach, Palgrave Macmillan, 2005.
[24] R. Hoda, N. Salleh, J. Grundy, H. M. Tee, Systematic literature reviews in agile software development: A tertiary study, Information and Software Technology 85 (2017) 60-70.
[25] J. Medeiros, D. Alves, A. Vasconcelos, C. Silva, E. Wanderley, Requirements engineering in agile projects: a systematic mapping based in evidences of industry, in: CIBSE, 2015.
[26] I. Inayat, S. S. Salim, S. Marczak, M. Daneva, S. Shamshirband, A systematic literature review on agile requirements engineering practices and challenges, Computers in human behavior 51 (2015) 915-929.
[27] E. M. Schon, J. Thomaschewski, M. J. Escalona, Agile Requirements Engineering: A systematic literature review, Computer Standards & Interfaces 49 (2017) 79-91.
[28] L. Crispin, J. Gregory, Agile testing: A practical guide for testers and agile teams, Pearson Education, 2009.
[29] C. Wang, F. Pastore, A. Goknil, L. Briand, Z. Iqbal, Automatic generation of system test cases from use case specifications, in: ISSTA, 2015, pp. 385-396.
[30] M. Zhang, T. Yue, S. Ali, H. Zhang, J. Wu, A systematic approach to automatically derive test cases from use cases specified in restricted natural languages, in: SAM, 2014, pp. 142-157.
[31] E. Sarmiento, J. C. S. P. Leite, E. Almentero, C&l: Generating model based test cases from natural language requirements descriptions, in: 1st RET, 2014, pp. 32-38.
[32] G. Carvalho, D. FalcÃčo, F. Barros, A. Sampaio, A. Mota, L. Motta, M. Blackburn, Test case generation from natural language requirements based on SCR specifications, in: SAC, 2013, pp. 1217-1222.
[33] M. Escalona, J. Gutierrez, M. Mejías, G. Aragón, I. Ramos, J. Torres, F. Domínguez, An overview on test generation from functional requirements, Journal of Systems and Software 84 (8) (2011) 1379-1393.
[34] C.-W. Ho, M. J. Johnson, L. Williams, E. M. Maximilien, On agile performance requirements specification and testing, in: Agile Conference, 2006, pp. 6-pp.
[35] F. Ricca, M. Torchiano, M. D. Penta, M. Ceccato, P. Tonella, Using acceptance tests as a support for clarifying requirements: A series of experiments, Information and Software Technology 51 (2) (2009) 270-283.
[36] E. C. dos Santos, P. Vilain, Automated acceptance tests as software requirements: An experiment to compare the applicability of fit tables and gherkin language, in: XP, 2018, pp. 104-119.
[37] S. Park, F. Maurer, A literature review on story test driven development,in: XP, 2010, pp. 208-213.
[38] A. Vicente, M. E. Delamaro, J. C. Maldonado, Uma revisÃčo sistemática sobre a atividade de teste de software em métodos Ágeis, in: CLTM, 2009.
[39] J. Angara, S. Prasad, G. Sridevi, The factors driving testing in devops setting a systematic literature survey, Indian Journal of Science and Technology 9 (48) 2017.
[40] E. Sarmiento, J. CSP Leite, N. Rodriguez, A. von Staa, An automated approach of test case generation for concurrent systems from requirements descriptions, in: ICEIS, 2014, pp. 339-347.
[41] M. Ivarsson, T. Gorschek, A method for evaluating rigor and industrial relevance of technology evaluations, Empirical Software Engineering 16 (3) (2011) 365-395.
[42] K. Curcio, T. Navarro, A. Malucelli, S. Reinehr, Requirements Engineering: a systematic mapping study in agile software development, The Journal of Systems & Software 139 (1) (2018) 32-50.
[43] B. Kitchenham, O. P. Brereton, D. Budgen, M. Turner, J. Bailey,S. Linkman, Systematic literature reviews in software engineering a systematic literature review, Information and software technology 51 (1)(2009) 7-15.
[44] V. T. Heikkilä, C. Lassenius, D. Damian, M. Paasivaara, A mapping study on requirements engineering in agile software development, in: SEAA, 2015, pp. 199-207.
[45] A. Causevic, D. Sundmark, S. Punnekkat, Factors limiting industrial adoption of test driven development: A systematic review, in: ICST, 2011, pp. 337-346.