**Software Quality Journal manuscript No.**
(will be inserted by the editor)

# A Systematic Literature Review on Quality Criteria for Agile Requirements Specifications

**Petra Heck · Andy Zaidman**

Received: date / Accepted: date

**Abstract** The quality of requirements is typically considered as an important factor for the quality of the end product. For traditional up-front requirements specifications a number of standards have been defined on what constitutes good quality: requirements should be complete, unambiguous, specific, time-bounded, consistent, etc. For agile requirements specifications no new standards have been defined yet and it is not clear yet whether traditional quality criteria still apply. To investigate what quality criteria for assessing the correctness of written agile requirements exist, we have conducted a systematic literature review. The review resulted in a list of 16 selected papers on this topic. These selected papers describe 28 different quality criteria for agile requirements specifications. We categorize and analyze these criteria and compare them with those from traditional requirements engineering. We discuss findings from the 16 papers in the form of recommendations for practitioners on quality assessment of agile requirements. At the same time we indicate the open points in the form of a research agenda for researchers working on this topic.

**Keywords** Agile Requirement · Just-in-Time · Quality Assessment · Systematic Literature Review · Quality Criteria · Correctness

P. Heck
Fontys Applied University, Eindhoven, the Netherlands
Tel.: +31-8850-89716
E-mail: p.heck@fontys.nl

A. Zaidman
Delft University of Technology, the Netherlands

## 1 Introduction

Requirements engineering has been perceived as one of the key steps in successful software development since the early days of software engineering (Sillitti and Succi 2005). Sillitti and Succi mention several standards for requirements elicitation and management, such as IEEE-830 Recommended Practice for Software Requirements Specification (IEEE 1998), that have been developed for traditional requirements engineering. They claim that agile methods do not rely on standards. Inayat et al. (2014) observe that agile requirements engineering solves the initial vagueness of agile requirements not by documenting according to standards, but by e.g., face-to-face communication or prototyping. On the other hand they remark that the practice of less documentation poses a challenge in many situations (e.g., distributed teams, large teams, complex projects) that require documented agile requirements that are fully elaborated in writing. As soon as agile teams cannot rely on face-to-face communication the correctness of written documentation becomes more and more important.

Standards such as IEEE-830 (IEEE 1998) define criteria for this correctness: requirement specifications should be complete, unambiguous, specific, time-bounded, consistent, etc. However, this standard focuses on traditional up-front requirements specifications. These are requirements sets that are completely specified (and used as a contract) before the start of design and development. As said, agile methods do not tend to follow such standards. However both research (Eberlein and Leite 2002; Inayat et al. 2014) and findings in practice (see e.g., Heck and Zaidman (2014) and the empirical study of Kassab (2014)) suggest that quality of requirements specifications is also an important topic for agile requirements engineering.

In previous work we have developed a framework for quality criteria for agile requirements specifications (Heck and Zaidman 2014). This resulted in a list of possible quality criteria for e.g. agile user stories. Working on this list we realized that we could not find any updated standards or best practices for agile requirements specifications, nor an extensive overview of quality criteria for agile requirements specifications. At the same time we felt the need to validate our list of quality criteria with what others have published on this topic. In our previous work (Heck and Zaidman 2014) we focus on open source feature requests and thus only included related work in the area of open source development. This is what made us see the necessity of conducting a systematic literature review (Kitchenham and Charters 2007) to make an inventory of quality criteria for agile requirements specifications that have been mentioned in literature. The result of such a literature review would be that we have a more thorough analysis of which of the traditional criteria are still applicable and which new quality criteria have been presented for agile requirements specifications. Furthermore, we take this opportunity to discuss the found literature from the viewpoint of both practitioners and researchers.

This leads us to the driving research question for our systematic literature review:

**[RQ]** Which are the known quality criteria for agile requirements specifications?

Note that we focus on requirements verification (have the requirements been written in a correct way) and not on requirements validation (do the requirements correctly reflect the need of the user). Moreover we consider formal requirement verification methods (that use mathematical models to derive specification correctness) out of our scope, since this requires specialist competence to apply.

The remainder of this paper is structured as follows. Section 2 introduces some background and related work. Section 3 details the selection process that we followed, while

Section 4 presents the classification of the resulting papers according to different dimensions. Section 5 summarizes the quality criteria for agile requirements specifications that are mentioned in the selected papers. Sections 6 includes recommendations for practitioners in the area of quality of agile requirements specifications. Section 7 discusses our findings and presents a research agenda. Section 8 concludes the paper.

## 2 Background and Related Work

This section sketches a short background on the role of requirements in agile development and introduces some related work. The absence of papers with quality of agile requirements as their main topic is what motivated us to conduct our research in the first place.

*Background on Agile Requirements* Agile development is a collective name for a family of software development processes that follow the so-called Agile Manifesto: "Individuals and interactions over processes and tools; working software over comprehensive documentation; customer collaboration over contract negotiation; responding to change over following a plan" (Beck et al. 2001). The main implications of this manifesto for agile requirements engineering are (Sillitti and Succi 2005; Inayat et al. 2014): software is developed incrementally with requirements being detailed and prioritized just before every iteration, requirements documentation is reduced in favor of face-to-face communication and prototyping. Grau et al. (2014) characterize agile requirements engineering as "collaborative, just enough, just in time, sustainable". Ernst and Murphy (2012) use the term "just-in-time requirements" (JIT requirements) for this. They observed that requirements are "initially sketched out with simple natural language statements", only to be fully elaborated (not necessarily in written form) when being developed. In this paper we use the term *agile requirements*, because this is the most widely used.

*Related Work on Agile, Requirements and Quality* For the area of agile development processes there is a body of work on either Requirements Engineering (Grau et al. 2014; Paetsch et al. 2003; Ramesh et al. 2010) or Quality Assurance (Bhasin 2012; Huo et al. 2004), but not specifically on the combination of the two. In this section we discuss some relevant papers.

Ramesh et al. (2010) present the results of a qualitative study of 16 organizations, to answer two questions: What Requirements Engineering (RE) practices do agile developers follow? What benefits and challenges do these practices present? Their study also included subjects in quality assurance roles. Their study presents a good overview of agile requirements engineering practices and challenges. One challenge they specifically mention is the absence of adequate requirements verification. This supports the main motivation for our research.

Inayat et al. (2014) conducted a systematic literature review on agile requirements engineering practices and challenges. They mention a few examples of other reviews on agile methods. None of them focus on requirements engineering. Inayat et al. (2014) focus on practices (process) instead of on the requirements (product) themselves. However, they mention minimal documentation and neglecting non-functional requirements as a challenge for agile requirements engineering.

Sfetsos and Stamelos (2010) conducted a review on quality in agile practices. However, they focus on quality aspects of the end product (maintainability, usability, etc.), not on quality of the requirements.

Grau et al. (2014) see that documentation formats in agile are a “continuous evolution of well-known requirements engineering concepts” (such as scenarios). They see this same continuous evolution in the definition of quality attributes for agile requirements. To illustrate this continuous evolution they mention SMART (Specific, Measurable, Achievable, Relevant, Time-Boxed) and INVEST (Independent, Negotiable, Valuable, Estimable, Small, Testable) (Wake 2003) as examples of quality attributes that have been defined in practice for agile requirements. However, they do not explain these acronyms and do not go into the subject to thoroughly investigate the applicable quality attributes for agile requirements.

*Related Work on Test- and Behaviour-Driven Development* There is a recent stream of agile development called Behavior-Driven Devevelopment, or BDD (North 2006). North suggested a template that takes the agile requirements one step further by specifying them following a strict template: GIVEN ... WHEN ... THEN .... Nowadays a number of tools, like Cucumber and JBehave, exist that help to translate this format into executable test cases. Using test cases as requirements is also done in the related area called Test-Driven Development, or TDD (Beck 2002).

Melnik et al. (2006) report on an experiment where customers used executable acceptance test (storytest)-based specifications to communicate and validate functional business requirements. To evaluate the quality of the acceptance test specifications they use the following criteria: credible (contain realistic and reasonable set of operations to be likely performed by the customer), appropriate complexity (involve many features, attributes, workflows, etc.), coverage of the major functionality, easy to read and informative, easy to manage (packaged in meaningfully structured suites, subsuites etc.).

This is an example that shows that using test cases as requirements results in quality criteria primarily related to the test cases themselves. Furthermore, in TDD verification of requirements is often done by implementing automated regression tests (Bjarnason et al. 2015), thus removing the need for further informal verification based on quality criteria. In fact, the creation of test cases is in itself a way of verifying the requirements, because if the requirements are not clear, we would not be able to specify them as test cases (Bjarnason et al. 2015). For this work we focus on quality criteria for agile requirements that are not in the form of test cases and consider BDD and TDD as out of scope.

*Related Work on Requirements Quality Criteria* Wake (2003) introduces SMART and INVEST in the context of Extreme Programming (XP). According to him tasks should be Specific, Measurable, Achievable, Relevant, Time-boxed and stories should be Independent, Negotiable, Valuable, Estimable, Small and Testable. Both acronyms have been taken over by several other papers and are being used in agile practice (see e.g. (Leffingwell 2011; Grau et al. 2014)). Our work takes these acronyms and places them in a larger framework of quality criteria.

## 3 Method

To answer our research question (*Which are the known quality criteria for agile requirements specifications?*) we needed to select articles that are relevant for the topic of quality criteria for agile requirements specifications. For the selection of the relevant articles we followed a structured process, according to the guidelines of Kitchenham and Charters (2007). The structured process they propose contains the following steps:

1. Define inclusion and exclusion criteria
2. Identify query string
3. Identify databases and other sources to search
4. Select candidates based on title and abstract
5. Refine candidate list based on full paper
6. Extend result set based on citations
7. Classify resulting papers

All steps were executed by the first author and validated by the second author. Candidate selection and refinement (step 4, 5 and 6) were repeated by the second author with a random sample of the articles. Where differences were found the outcome was adjusted according to the discussion between the two authors. The below paragraphs describe each of the steps in detail.

## 3.1 Step 1: Inclusion and Exclusion Criteria

*Inclusion Criteria* The inclusion criteria were defined by the two authors at the start of the review process based on the research question (*Which are the known quality criteria for agile requirements specifications?*) and on the type of literature we wanted to include:

– The paper is about *software* products. This is to make the distinction with agile/just-in-time in the areas of just-in-time manufacturing and systems engineering (more focused on hardware).
– *Agile* or similar just-in-time *requirements specifications* are the central topic of the paper. The paper can be focused on specific formats for requirements such as user stories.
– *Product quality* aspects of requirements specifications are an important part of the paper (we consider traceability also as a quality aspect in this sense); case studies can be included if they might deliver anecdotal evidence of requirements quality judging from the abstract.
– The quality aspects are discussed in a setting of *informal requirements verification*. As described in the introduction we consider papers focused on requirements validation and papers about formal methods for requirements verification out of scope.
– The paper is a self-contained article published in a journal or in the proceedings of a workshop/conference or as a book chapter.
– The paper went through an external peer review process.
– The paper is written in English.
– The paper is published between 2001 (Agile Manifesto) and 2014 (search was conducted in January 2015).

*Exclusion Criteria* During the selection process we sometimes ran into papers on the topic of requirements quality and agile, but with not exactly the right focus. Since it is difficult to define the exact focus with inclusion criteria only, we enhanced them with the following exclusion criteria to indicate which topic areas we considered out of scope for our review:

– does not meet inclusion criteria
– only contains a tool description
– focus on test driven development (TDD), where tests are used instead of requirements
– focus on User Experience (UX) requirements
– focus on architecture requirements
– focus on requirements prioritization
– focus on requirements (size or effort) estimation

## 3.2 Step 2: Query String

Based on the research question (*Which are the known quality criteria for agile software requirements specifications?*) we also defined a search string that includes the keywords and their most important synonyms. We took care of not making the search string too restrictive. We did not want to miss relevant papers on beforehand based on a difference in terminology. That is why we chose the key words for the search string to be "agile", "requirement" and "quality".

We included "just-in-time" as it has been coined by Ernst and Murphy (2012) as a term for requirements approaches that are characterized by the use of lightweight representations such as user stories, and a focus on evolutionary refinement. They note that the notion of "agile requirements" is in many ways analogous to "just-in-time requirements".

Some papers might not be discussing the general "quality" concept. Although we focus on informal requirements verification, we explicitly extended the query string with both "verification" and "validation" to ensure that we do not miss any candidate papers based on the confusion between verification and validation.

We did not explicitly include specific requirements formats such as "user story", "feature" or "epic" because we assume that any paper with the quality of those items as central topic would also mention "requirement".

((agile OR "just-in-time" OR "just in time") AND requirement AND
(quality OR verification OR validation))

The query string was defined by the two authors before executing it on selected sources (see step 3). The results of the query execution and the following steps did not give us reason to change the query string during any of the steps.

## 3.3 Step 3: Source Selection

Based on Kitchenham and Charters (2007) we selected five digital databases which index the most important venues in the software engineering research field. For the digital databases we executed the query string on title and abstract and noted the number of returned search results. We also determined for each digital database the number of unique search results (January 2015), resulting in 630 unique items in total:

– IEEE Xplore (`http://ieeexplore.ieee.org`) 271 items
– ACM Digital Library (`http://dl.acm.org`) 268 items, 120 unique items
– Scopus (`http://www.scopus.com`) 365 items, 219 unique items
– DBLP (`http://www.dblp.org`) 0 unique items
– ScienceDirect (`http://www.sciencedirect.com`) 38 items, 20 unique items

In this initial result set of 630 unique items we saw a number of venues that related to requirements or agile specifically. We cross-checked this list of venues with the list from the review of Davis and Hickey (2009). This resulted in a list of journals and conferences which have requirements engineering or agile as their main topic and which have digital publications:

– Agile Alliance Agile Conference (AGILE), 2003-2014
– International Conference on Agile Software Development (XP), 2003-2014
– IEEE Requirements Engineering Conference (RE), 2001-2014

– International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ), 2007-2014
– Requirements Engineering journal (REJ), 2001-2014
– First Agile Requirements Engineering Workshop (AREW'11), 2011

We decided to include all papers published in those venues (instead of executing the search string) for our manual search step described in the next paragraph, too reduce the chance of missing relevant papers based on the search string only. The starting year is the first year of digital proceedings for the conference or 2001 for other sources (see inclusion criteria).

And of course we include our previous work itself. This consists of a technical report (Heck and Zaidman 2014) which is being extended into a journal paper.

## 3.4 Step 4: Candidate Selection

A first step was to exclude items from the 630 unique items returned from the digital database that, based on their abstract and title, fulfilled the exclusion criteria. In case of doubt we included the paper in our candidate list. With this step we narrowed down the digital database items from 630 to 113, see Table 1.

A second step was to test the candidates for the inclusion criteria, based on title and abstract. We did this for the 113 candidates of the digital databases but also for all publications in the other sources mentioned in the previous section (the agile and requirements engineering forums and our previous work). In case of doubt we included the paper in our candidate list. Based on the inclusion criteria we decided to include 55 candidates from the digital databases and added 10 new ones from the other sources in the previous section, see Table 1.

Both steps were executed by the first author and a sample was validated by the second author (without knowing the results of the first author in advance). For the first step 20 papers from IEEE Xplore, 20 papers from Scopus and 10 from ACM (the first 10 percent of the query results in the order they were returned by each digital database) were handed to the second author. For the second step also all 2014 (i.e. the most recent year) publications

**Table 1:** Filtering publications on quality criteria for agile requirements

| Source | **Step 3** **Query** (Abstract) | **Step 4a** **Exclusion** (Abstract) | **Step 4b** **Inclusion** (Abstract) | **Step 5** **Inclusion** (Full Paper) | **Step 6** **References** (Full Paper) |
|---|---|---|---|---|---|
| ScienceDirect | 20 | 1 | 0 | | |
| IEEE Xplore | 271 | 57 | 26 | 4 | 5 |
| ACM DL | 120 | 18 | 9 | 0 | |
| Scopus | 219 | 38 | 21 | 4 | 6 |
| **Subtotal** | **630** | **113** | **55** | **8** | **11** |
| REJ | | | 0 | | |
| RE | | | 4 | 0 | |
| REFSQ | | | 1 | 0 | |
| XP | | | 4 | 2 | 4 |
| AGILE | | | 0 | | |
| AREW | | | 0 | | |
| Heck and Zaidman (2014) | | | 1 | 1 | 1 |
| **Subtotal** | | | **10** | **3** | **5** |
| **TOTAL** | **630** | **113** | **65** | **11** | **16** |

of the RE conference (67 papers) and XP conference (27 papers) were handed to the second author. Table 2 shows the different ratings that were given. This shows that in 93% of the cases both authors agreed on the in- or exclusion of the papers. In the cases where the rating was different a discussion lead to the judgment of the first author being kept. This was due to the fact that:

- The second author was more strict than the first author;
- The second author was not aware of the meaning of the term *Definition of Ready*;
- Upon reading the abstract a second time, the second author recalled his decision in the one case in which he included a paper that the first author excluded.

We found several papers on agile, requirements and quality that did not contribute in the area of quality criteria for agile requirements specifications. Some of these out of scope topics have been mentioned in the exclusion criteria. For reference purposes we would like to note that we also observed that a number of papers that were **not** included can be grouped around the following topics:

- The process of introducing agile methods in company or project XYZ
- Comparison of agile methods and CMM(I) or ISO
- Validation of the software product against user needs (by prototyping or active customer involvement)
- Quality assurance in agile methods (not product-oriented, but process-oriented)

This shows other topics within the area of quality and agile that have received attention in literature. This could be a starting point for other specialized systematic literature reviews.

## 3.5 Steps 5 and 6: Candidate Refinement and Citation Snowballing

Step 5 and 6 were executed simultaneously as we decided that the list we obtained by selecting candidates based on title and abstract was not too long (65 papers) to manually review all of them. By performing step 6 with 65 papers we increased the chance of finding relevant papers during this step.

We took the remaining 65 papers and applied snowballing according to the guidelines provided by Wohlin (2014). This means we checked all the references in the 65 papers, but also checked all citations of these 65 papers. For the backward checking (references in each of the 65 papers) we used the full version of each paper and for the forward checking (citations of each of the 65 papers) we used digital databases (IEEE and ACM include this info in their digital database and we used Google Scholar for the other papers). We repeated this snowballing process until no new papers are added.

In order to find the references in the 65 papers, we had to obtain the full paper. In doing so we at the same time reviewed the full papers for the inclusion criteria (would the paper help to answer our research question about quality criteria for agile requirements?). Based on this final review we decided to include only 11 out of 65 papers in the final result set.

**Table 2:** Interrater agreement for candidate inclusion

| | | Second Author | |
|---|---|---|---|
| | | Include | Exclude |
| First Author | Include | 2 | 9 |
| | Exclude | 1 | 132 |

**Table 3:** Interrater agreement for final inclusion

| | | Second Author | | |
|---|---|---|---|---|
| | | Include | Exclude | Possible Include |
| First Author | Include | 2 | 1 | 1 |
| | Exclude | 1 | 4 | 1 |

In cases where we had multiple papers on the same research from the same authors, we decided to include only the most relevant one (i.e., the most recent one). Note that we also performed snowballing for the 54 papers that were not included in the final result set. The complete snowballing process had added 5 new papers to this final set, resulting in 16 papers in total (see Table 1).

To be more confident of our decision to only include 11 out of 65 papers in the final set, we also handed the top-10 most recent papers to the second author. Table 3 shows the results. In four cases the judgment of first and second author were different. After discussion the judgment of the first author was taken as the final judgment for each of the ten papers. In two cases (first author exclude, second author include) it turned out that the paper mentions quality of agile requirements, but does not elaborate on quality criteria. Therefore the decision to exclude them was kept. In one case (first author include, second author exclude) the paper ([P14]) mentions some elements that should be present in user stories. The first author considers this as part of quality, while the second author thought no quality aspects of agile requirements were mentioned. The decision was made to keep the paper. In the last case (first author include, second author probably exclude) the paper ([B14]) was indeed judged as low relevancy by both authors, as it only briefly mentions priority as a quality criterion. We decided that this brief mention was enough reason for us to keep it in the final set.

## 3.6 Step 7a: Classification by Meta-Data

To provide the reader with more background on the selected papers we classified them by the following meta-data. This list of meta-data was designed by the two authors based on what they thought to be interesting properties for the selected papers.

1. Author name, author affiliation, author country;
2. Year and venue of publication;
3. Publication type: Journal (J), Workshop (W), Book chapter (B), Conference (C), Other (O);
4. Number of pages;
5. Agile method: XP, Scrum or general;
6. Requirements format: user story or general;
7. Research type: see below;
8. Evaluation method: see below.

*Research Type* One aspect to know about the selected papers is the type of research that is included in each of them. This answers a number of questions we can ask about the selected papers: is it a new solution that is proposed? does it contain a full-blown validation? is it personal experience? is it just the author's opinion? This helps readers to understand the value of the paper for their own practice. For each paper we classify those aspects that touch on our topic (quality of requirements).

For the classification of the research type we follow the framework of Wieringa et al. (2005):

- **Evaluation Research**: investigates a problem in Requirements Engineering (RE) practice or an implementation of an RE technique in practice.
- **Proposal of Solution**: proposes a solution technique and argues for its relevance, without a full-blown validation.
- **Validation Research**: investigates the properties of a solution proposal that has not yet been implemented in RE practice.
- **Philosophical Paper**: sketches a new way of looking at things, a new conceptual framework, etc.
- **Opinion Paper**: contains the author's opinion about what is wrong or good about something, how we should do something, etc.
- **Personal Experience Paper**: the emphasis is on what and not on why. The experience may concern one project or more, but it must be the author's personal experience.

*Evaluation Method* For each paper we also classify the type of evaluation that was done for the quality-related aspects. To give an impression of the depth of evaluation we present a number of dimensions on the evaluation method used (adapted from Cornelissen et al. (2009)):

- **Preliminary**: Evaluation of proposed solution is of preliminary nature, e.g. toy example or informal discussion
- **Regular**: Evaluation is not of preliminary nature
- **Human Subjects**: Evaluation involved human subjects, e.g. questionnaires, interviews, observations
- **Industry**: Evaluation was performed in an industry setting
- **Quantitative**: Evaluation resulted in some quantitative data on the proposed solution
- **None**: No evaluation

## 3.7 Step 7b: Classification on Quality Criteria

To classify the papers based on quality criteria we build upon our previous work. We structure the inventory according to the three overall quality criteria (QC) explained in (Heck and Zaidman 2014):

**[QC1]** Completeness. In this category we place criteria that specify elements that should be present in (the description of) the agile requirements. An example would be the rule that each requirement should have a unique identifier. Note that our definition of completeness (all structure elements of a single requirement should be there) is different from the notion of completeness as in specifying all user needs.

**[QC2]** Uniformity. In this category we assign criteria that pertain to a standardized style or format of the agile requirements specification.

**[QC3]** Consistency and correctness. This category contains all other criteria that state something about the correctness of individual requirements or the consistency with other requirements.

We think that this is a general classification that holds for written requirements (in fact, for all written documentation). Specifications should be complete, follow certain templates or standards, be correct and consistent with other documentation.

We identify any mentions of quality criteria for agile requirements specifications. We include each of the mentioned quality criteria in our classification, regardless of the length of the discussion in the paper or the strength of the evidence presented in the paper. In the same way we also include all elements of agile requirements that are mentioned in the papers (QC1). We take the decision to include everything regardless because we want to provide a broad overview which points to specific papers for detailed information.

### 3.8 Step 7c: Classification on Recommendations for Practitioners and Researchers

Although we did not set out to specifically collect papers with recommendations for practitioners or researchers in our database query, we did think that the resulting papers enclosed some valuable information in this respect. That is why as a final step we classified recommendations for practitioners or researchers found in the papers.

For practitioners we look at recommendations found anywhere in the 16 resulting papers about how to apply the quality criteria in practice. Then we summarized these recommendations in a few paragraphs, to structure them and to connect similar recommendations.

We present the recommendations for researchers in the form of a research agenda. To construct this research agenda we use the future work as indicated in the selected papers and our own analysis of what we see missing in the selected papers.

## 4 Results: Meta-Data Classification

In this section we characterize the 16 selected papers based on the meta-data defined in section 3.6 such as author, venue, agile requirements format, research- and evaluation type. This sets the ground for a detailed analysis of the quality criteria in Section 5. Each paper is identified with a unique identifier [XXnn] based on the author and year of publication. For an overview of the unique identifiers see Table A in the appendix, that presents each of the 16 papers in more detail.

### 4.1 Author, Affiliation, Country

The 16 selected papers have been written by 34 authors. Two of those authors have co-authored 2 papers. These papers are similar but have both been included because their contributions slightly differ. Authors of eight papers originate from Europe, 7 from North-America, 1 is written by authors from both continents. Two papers are written by authors with an industrial affiliation, 14 are from researchers. This shows that the topic attracts interest from both researchers and practitioners.

### 4.2 Year, Venue, Type, Pages

The publication date of the selected papers is spread over different years between 2001 and 2014 (see Figure 1). However, the bulk of the publications come after 2009. It looks like the topic is gaining in popularity, but we can only be sure of this in the years to come. It is also worth noticing that the types (workshop, conference, journal, book chapter, other) of publication are very heterogeneous.

Table 4 shows the different venues the selected papers have been published in. This is important to know for persons that are new to the field and want to know where to start reading or publishing themselves. Unfortunately we cannot give them a definite answer. The number of different venues is almost as large as the number of papers. The Workshop on Software Measurement (IWSM-MENSURA) has 2 publications, but these come from the same research group. Only the Requirements Engineering Journal and the Agile Conference (XP) have two distinct publications. This at least shows that not only the traditional requirements engineering community is working on this topic, but also the agile community. This indicates that the topic is also considered important within agile environments. For the rest the communities are quite diverse, ranging from Human Aspects to Web Intelligence. Note that in Table 4 we have not included [HZ14] (technical report).

Table A in the appendix shows the number of pages for each paper. Most of them are longer papers of 8 pages or more. Half of the papers even have more than 10 pages.

### 4.3 Agile Method and Requirements Format

As described by De Lucia and Qusef (2010) many different agile methods exist and they all have slight differences from a requirements engineering perspective. That is why it is important to know for each of the selected papers which agile method they discuss, as this might influence their perspective on quality criteria. The same goes for the requirements format discussed, since not all quality criteria apply to all formats (e.g. there exist specific templates for some formats).

Most papers do not describe one specific agile methodology (see Table 5). Two papers specifically address eXtreme Programming (XP) and three papers specifically address Scrum.

According to Ernst and Murphy (2012) "just-in-time requirements refer to higher-order organizational constructs, including features to be added to the project, agile epic and user stories, improvements to software quality, and major initiatives such as paying down technical debt". In our selected papers, those papers that mention a specific type of requirements,

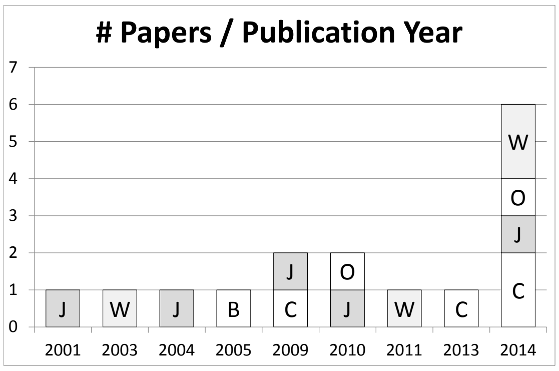

**Fig. 1:** Meta-data of selected papers (W=workshop, C=conference, B=book chapter, J=journal, O=Other)

**Table 4:** Venues

| Venue | Type | Paper |
|---|---|---|
| Requirements Engineering Journal | Journal (J) | [B14] |
| Journal of Emerging Technologies in Web Intelligence | Journal (J) | [DQ10] |
| Workshop on Software Measurement (IWSM-MENSURA) | Workshop (W) | [D11] + [DT14] |
| Journal of Defence Software Engineering | Journal (J) | [D01] |
| IEEE Southeastcon | Conference (C) | [FM13] |
| Journal of Object Technology | Journal (J) | [F04] |
| Better Software Magazine | Other (O) | [GG10] |
| Workshop on Traceability in Emerging Forms of Software Engineering | Workshop (W) | [L03] |
| Conference on Extreme Programming and Agile Processes in Software Engineering (XP) | Conference (C) | [L14a] + [P14] |
| Workshop on Cooperative and Human Aspects of Software Engineering | Workshop (W) | [L14b] |
| Journal of Software | Journal (J) | [PR09] |
| Engineering and Managing Software Requirements | Book Chapter (B) | [SS05] |
| Conference on Information Technology: New Generations (ITNG) | Conference (C) | [SL09] |

**Table 5:** Classification of selected papers

| | **Agile Method** | | | | **Research Type** | | | **Evaluation** | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | General | Scrum | XP | (User) Story | Evaluation Research | Proposal of Solution | Validation Research | Preliminary | Regular | Human Subjects | Industry | Quantitative | None |
| **[B14]** | x | | | | | x | | | x | | | | |
| **[DQ10]** | x | | | x | x | | | | | | | | x |
| **[D11]** | x | | | x | | x | | x | | | | x | |
| **[DT14]** | | x | | x | | | x | | x | | x | x | |
| **[D01]** | | | x | x | x | | | | | | | | x |
| **[FM13]** | x | | | | | x | | x | | | x | x | |
| **[F04]** | x | | | x | | x | | | | | | | x |
| **[GG10]** | x | | | x | | x | | | | | | | x |
| **[HZ14]** | x | | | x | | x | | x | | x | x | | |
| **[L03]** | x | | | | | x | | | | | | | x |
| **[L14a]** | x | | | x | | x | | | x | x | x | | |
| **[L14b]** | | x | | x | x | | | | x | x | | | |
| **[PR09]** | | | x | x | | x | | x | | x | x | | |
| **[P14]** | x | | | x | x | | | | x | x | x | | |
| **[SS05]** | x | | | x | x | | | | | | | | x |
| **[SL09]** | | x | | x | x | | | | x | x | x | | |

all mention (user) stories. User stories (US) are short sentences that represent the customer requirements at a high level, and the documentation for these stories includes the explanations of the requirements [D11]. Table 5 shows which other papers treat user stories. Note that some of them, such as [SS05], only mention user stories briefly.

Table 5 indicates that User Stories are not the only requirements format being discussed. Researchers investigating quality of agile requirements specifications should thus not only investigate user stories and practitioners working with agile requirements should bear in mind that there is no obligation to use user stories as their format.

### 4.4 Research Type

Out of the 16 selected papers, 9 of them have been classified as Proposal of Solution, 6 of them as Evaluation Research and only 1 as Validation Research (see Table 5). This shows that most papers report on a new solution that they propose, which makes it harder for practitioners to evaluate how to apply the solution in practice (i.e. validation research is missing).

### 4.5 Evaluation Method

Table 5 presents the evaluation method used in each of the 16 papers. Six papers lack an evaluation and four papers only contain a preliminary evaluation. We double-checked to see if newer papers of the same authors have been published, but this was not the case. There were not many (only 3) quantitative evaluations. In combination with our analysis in the previous section this shows that most papers report on an existing practice or new solution, without a full-blown validation. Note that for our own work [HZ14] we have executed a more extensive quantitative evaluation which is under submission. A good thing is that 70% of the evaluations (7 out of 10) involved industry projects or companies. This confirms our earlier remark that both industry and academia are working on this subject.

## 5 Results: Quality Criteria Used in Literature

We made an inventory of quality criteria for agile requirements specifications that are mentioned in each of the 16 selected papers. The method we used to create this inventory is described in Section 3.7.

In Figure 2 we present the classification of quality criteria and mention between brackets for each quality criterion which paper(s) mention them. In Table A in the appendix we briefly discuss the contribution of each of the papers to this inventory. Together, the 16 selected papers mention 28 different quality criteria for agile requirements specifications, of which most criteria are confirmed by more than one paper. Note that out of these 28 criteria, 11 criteria had not been mentioned in our previous work [HZ14] and four of them are only mentioned in [HZ14].

In what follows we will highlight criteria that are mentioned by three or more papers.

*Priority* Five papers mention that requirements should have a priority defined. [SS05] mentions Requirements Prioritization as an important technique for agile methods. [HZ14] mentions priority as an example of relative importance. [B14] calls this “a grade for the importance of each requirement” and describes a clustering algorithm based on this. [D01] calls

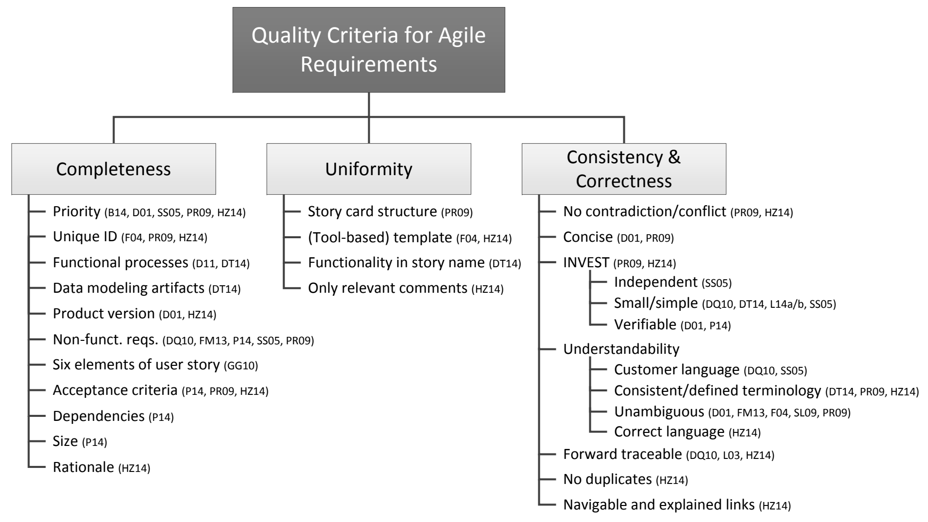
Quality Criteria for Agile Requirements
Completeness
Priority (B14, D01, SS05, PR09, HZ14)
Unique ID (F04, PR09, HZ14)
Functional processes (D11, DT14)
Data modeling artifacts (DT14)
Product version (D01, HZ14)
Non-funct. reqs. (DQ10, FM13, P14, SS05, PR09)
Six elements of user story (GG10)
Acceptance criteria (P14, PR09, HZ14)
Dependencies (P14)
Size (P14)
Rationale (HZ14)
Uniformity
Story card structure (PR09)
(Tool-based) template (F04, HZ14)
Functionality in story name (DT14)
Only relevant comments (HZ14)
Consistency & Correctness
No contradiction/conflict (PR09, HZ14)
Concise (D01, PR09)
INVEST (PR09, HZ14)
Independent (SS05)
Small/simple (DQ10, DT14, L14a/b, SS05)
Verifiable (D01, P14)
Understandability
Customer language (DQ10, SS05)
Consistent/defined terminology (DT14, PR09, HZ14)
Unambiguous (D01, FM13, F04, SL09, PR09)
Correct language (HZ14)
Forward traceable (DQ10, L03, HZ14)
No duplicates (HZ14)
Navigable and explained links (HZ14)

**Fig. 2:** Quality criteria for agile requirements (next to each quality criterion the papers that mention it, see Table A for more details).

this "Annotated by Relative Importance", based on (Davis et al. 1993). [PR09] suggests to prioritize story cards "based on agile software development values".

In agile development the priority of a requirement is very important to know, because the priority is used to plan iterations. In each iteration the open requirements with the highest priority are detailed and subsequently developed. Priority is allowed to change as long as the requirement is open. In this way agile development ensures that the customer receives what he needs most at any given moment. This also allows for the customer to change his/her mind and upgrade or downgrade requirements during the project by adjusting the priority attribute.

*Unique Identifier* Three papers mention that requirements need a unique identifier. [F04] recommends a template for agile requirements where ID is one of the columns. Since [PR09] is about story cards, they call this unique identifier a story card number. [HZ14] assumes that requirements are stored in an electronic tool and thus automatically assigned a unique ID.

Note that an unique identifier is also useful in oral communication of agile requirements. In oral communication we need an easy and unambiguous way to refer to requirements we are discussing, whereas in written communication we could use other means such as hyperlinks or paragraph numbers.

*Non-Functional Requirements* Five papers ([DQ10], [FM13], [P14], [SS05], [PR09]) indicate that non-functional requirements should not be overlooked in agile development. [P14] states that "architecture criteria (performance, security, etc.)" should be identified for a User Story to be considered Ready.

Recommendations are to arrange meetings to discuss the non-functional requirements as early as possible ([DQ10] and [SS05]), to include them as part of the story card [PR09], use quality metrics and a risk-driven algorithm to plan them [FM13].

*Acceptance Criteria* [P14], [PR09] and [HZ14] all mention acceptance criteria or acceptance tests as an important part of user stories.

In traditional requirements engineering acceptance tests are often developed together with the upfront requirements specification. Agile development does not recommend writing such elaborate test documentation upfront since there is a good chance that certain requirements do not get implemented or will change and thus the test cases will be obsolete. To replace comprehensive upfront test documentation, agile requirements should be elaborated with acceptance criteria (in Behaviour-Driven Development these acceptance criteria are even formalized according to a template).

*INVEST* An acronym introduced specifically for agile requirements quality is INVEST: user stories should be Independent, Negotiable, Valuable, Estimable, Small and Testable (Wake 2003). For readability of Figure 2 we have included INVEST as 1 sub-item, when in fact it is a collection of six criteria. INVEST is mentioned as-is in [HZ14]. [PR09] does not mention the acronym, but does mention each of the six criteria separately.

Seven other papers mention that agile requirements should be independent [SS05], should be kept small or simple [DQ10, DT14, L14a/b, SS05] and that the requirements should be verifiable (= testable) [D01, P14]. [DT14] states that user stories should be kept small in the sense that "It is expected that each user story be mapped to only one functional process.", [L14a] and [L14b] state that the "Expected Implementation Duration" should be kept low. [DQ10] suggests to split requirements that are considered too complex by the team. [GG10] dedicated their whole paper to how to perform what they call "story slicing". [P14] phrases verifiability as "team knows what it will mean to demo the User Story".

Practitioners working with agile requirements should also keep in mind that INVEST/SMART could also be applied to other types of agile requirements.

*Understandability* We have used the term Understandability to group several criteria related to the choice of wording for the requirements specification. Defining clear requirements can save a lot of time in discussion and question answering during the implementation.

Two papers [DQ10, SS05] mention that this can be achieved with requirements *written in the language of the customer*. [PR09] states that story cards should "use language simply, consistently and concisely". Two more papers deem it important to use a *consistent and defined terminology* throughout all requirements: [DT14] and [HZ14]. [HZ14] advocates the use of a glossary for this purpose and also recommends the use of *correct language* (i.e. spelling- and grammar-wise). In total five papers mention that it is important for the requirements to be *unambiguous* in general [D01, FM13, F04, SL09, PR09].

*Forward Traceable* Three papers mention that agile requirements should be forward-traceable [DQ10, L03, HZ14].

It is important to know to which source code and test cases the requirements trace (forward traceability) to be aware where things must be changed when requirements change, since agile development embraces change as a given factor (Beck et al. 2001).

## 6 Results: Recommendations for Practitioners

In this section we discuss how practitioners should use the quality criteria in practice. We support our discussion with references to the relevant papers.

*Use a list of quality criteria* We have found 16 papers that contain quality criteria for agile requirements specifications. We advise practitioners to consider the full list of criteria summarized in Section 5 and establish a list of quality criteria appropriate for their own project, team or environment.

[HZ14] describes how a given list of quality criteria can be customized by a team: 1) decide which criteria are not relevant for the team; 2) add missing criteria by interviewing team members, by re-evaluating old requirements or by applying the criteria in practice and improve them on-the-fly.

The quality criteria should be applied to the agile requirements specifications, but do not have to hold from the beginning. According to [DQ10] "any documents produced in the early stages can quickly become irrelevant because the agile principles encourage requirements change". This is confirmed by e.g. Ernst and Murphy (2012) who coined the term "just-in-time requirements" for this. The recommendation is to only document what is relevant at a given moment, and postpone all other requirements documentation to as late as possible. Analogue to this we can also say that the quality criteria should hold just-in-time: for each criterion it should be decided at which point in time it should hold (e.g. at creation time of the requirement or just before development starts).

*Checklist or Definition of Ready* [HZ14] calls the resulting list of quality criteria a checklist. [PR09] also promotes the use of a so-called validation checklist to assess correctness as part of a high maturity level for agile requirements engineering. [P14] advises to include such criteria in a so-called Definition of Ready. A Definition of Ready (DoR) is a sort of checklist that an agile team uses to determine when a user story is ready for the developers to start implementing it. According to [P14] "not having a definition of ready can seriously impede the flow of work through your system". A DoR is something that can also be implemented for other types of agile requirements.

*Use of a tool* To simplify the process of applying checks the requirements could be stored in a tool. According to [B14] "tools that allow collaboration and provide tracing of changes and allow recording of requirements in standardized formats, along with a proper plan for team coordination, are considered as essential". This same finding is supported by [HZ14] that already assumes agile requirements are stored in a tool, resulting in quality criteria that are inherent to the tool (e.g. unique identifiers). [DQ10] recommends the use of tools not only for storing requirements but also for storing traceability information between requirements, tests and code.

## 7 Discussion

In this discussion we will revisit our research question, present the research agenda and discuss threats to validity.

### 7.1 [RQ1] Which are the known quality criteria for agile requirements specifications?

Based on the selected papers of our literature survey, we identified 28 quality criteria that have been mentioned for agile requirements specifications. These quality criteria are listed in Figure 2. In this section we analyze which criteria come from traditional requirements engineering and which new quality criteria have been presented.

*Quality criteria for traditional requirements specifications* When investigating the 28 quality criteria, we find that most of them are also applicable to traditional up-front requirements specifications. Criteria specific for agile requirements specifications are: six elements of a user story [GG10], story card structure [PR09], functionality in story name [DT14] and INVEST [PR09, HZ14]. Of course the interpretation of traditional criteria might be slightly different for agile requirements. For example, acceptance criteria should not be full-blown test scenarios but short statements indicating when the implementation of a requirement can be accepted; priority is paramount for agile requirements as it is the basis for all planning activities and determines when to spend time on detailing the requirement; understandability might be less important for agile requirements specifications in environments where conversation with the customer can be used to clarify ambiguities.

The papers that we selected do not provide enough evidence to completely answer the question which traditional criteria still apply to agile requirements specifications. However, [D01] contains an analysis where it is shown which traditional quality criteria are not or less applicable to eXtreme Programming. A similar analysis is made in [HZ14] for feature requests in open source systems.

*New quality criteria for agile requirements specifications* The INVEST model seems to be the only new quality criterion that is mentioned in the context of agile requirements specifications, since the other three new ones (six elements of a user story [GG10], story card structure [PR09], functionality in story name [DT14]) can be seen as translations of existing criteria to the agile requirements format. The goal of INVEST is to divide the system to be developed in small deliverables that can be delivered independently, one of the key principles of agile development. INVEST is currently tied to user stories, but we think it is also valuable for other types of (agile) requirements.

*Just-in-time quality* In our own continued research we decided to add a timing dimension to quality criteria for agile requirements specifications. Since the requirements are specified or detailed just-in-time, some quality attributes also do not have to hold from the creation of the requirement, but should hold just-in-time. For example an initial specification might be ambiguous as long as any unclear terms or wordings have been resolved just before development starts.

### 7.2 Research Agenda

In this section we discuss the 16 papers with respect to open points for researchers to work on in the area of quality of agile requirements:

1. *More than user stories.* Most selected papers investigate or mention user stories (see Table 5). However, we think that the characteristics of a good user story also hold for other types of agile requirements. An example of this is given in [HZ14] for feature requests in open source projects. We would like to see more research papers on the topic of suitable requirements formats for agile environments with of course a focus on the quality aspects of each of those formats. This would help practitioners to select the proper requirements format for their environment and to be aware of the caveats for each of the possible formats.
2. *Validation research.* There are few papers with a thorough validation of the proposed solution (see Table 5). The selected papers are one-off publications, without a continuation

in future work. Existing and new frameworks or methodologies for creating good quality requirements should be validated more extensively by the research community. This makes it easier for practitioners to see if the method could work in their daily practice.

3. *More case studies*. The generalizability of results is a threat to validity for most of the selected papers. Almost all of them mention that more case studies should be done in future work to validate the results in other situations. For practitioners it is extremely valuable to know to what extent the described results can be applied in other situations. Therefore, as a research community we need to publish more case studies in the area of agile requirements engineering with real-life data sets or industrial setting. This need for more case studies is also confirmed by other publications on research agendas for agile development (Dingsøyr et al. 2008; Dybå and Dingsøyr 2008). In addition, we would also want to advocate the need for *longitudinal studies* (e.g., see Runeson et al. (2012)) in agile requirements engineering to study how agile requirements engineering quality practices change or vary over time.
4. *Tooling*. [DQ10], [HZ14], [L03] and [PR09] mention that tools could be useful to automate some of the checks for quality criteria. We think this is true. Researchers could develop prototypes of such tools, test them and make them available as e.g. plug-ins for requirements tools. In this way practitioners can use the guidelines for good quality agile requirements without the extra burden of checking things manually.
5. *Cooperation*. The types of venues that have published papers on the topic of quality of agile requirements are very heteregenuous (see Table 4). Both the agile community and the requirements engineering community have published papers. Next to that, a whole scala of different smaller and bigger venues welcome papers on the topic (from the measurement community to the web intelligence community to the human aspects of software engineering community). A valuable contribution can be made if researchers working on the topic would collaborate more across communities. In 2015 we see two good examples of this: the Just-in-Time Requirements Engineering workshop (JITRE)[1] in conjunction with the Requirements Engineering conference (RE'15), and the Workshop on Requirements Engineering in Agile Development (READ)[2] in conjunction with the Conference on Agile Software Development (XP2015). Also here it could be remarked that it would have been even better if both communities would have joined in one single workshop. Although in this case at least Neil Ernst and Maya Daneva are involved in both workshops (organizing the one and serving as a Program Committee member for the other).
6. *Also start from the problems.* Next to investigating sources that mention quality criteria for agile requirements specifications, we should also start from the other side. What are quality problems with software that has been developed in an agile way and which of those problems link back to quality problems in the agile requirements specifications? When we investigate the quality problems in practice, we can subsequently define quality criteria for agile requirements specifications that will help us to avoid or prevent those problems in practice. In that way we would let problems in agile practice guide us towards the most important quality criteria for agile requirements specifications. This could also lead to newly discovered quality criteria.
7. *Investigate necessity and impact of agile requirements correctness.* In our introduction we have stated that correctness of agile requirements specifications is important. We have gathered further support for this claim with the publications in this survey. However

[1] `https://jitre.wordpress.com/`
[2] `http://www.xp2015.org/1st-international-workshop-on-requirements-engineering-in-agile-development/`

a lot of questions around this topic still remain open. Do written requirements play an important role in agile? What is the difference between distributed and non-distributed teams for the importance of written requirements correctness? What is the impact of non-correct requirements on final product quality? We would like to see more research in this area to answer these important questions and further confirm our claims.

## 7.3 Threats to Validity

Threats to the validity of the systematic review are analyzed according to the following taxonomy: construct validity, reliability, internal validity and external validity (Runeson et al. 2012).

### *7.3.1 Construct Validity*

Construct validity reflects to what extent the phenomenon under study really represents what the researchers have in mind and what is investigated according to the research questions.

Requirements engineering in general and requirements engineering in an agile context are broad subjects. We explicitly confined our survey to agile requirements engineering research with a special focus on *quality criteria* for agile requirements specifications. In order to be as objective as possible on which papers to include or exclude in our survey, we followed the advice of Kitchenham (2004) and Brereton et al. (2007) to use predefined selection criteria that clearly define the scope of the survey.

Our process of collecting relevant articles was based on keyword searches in well-established scientific digital libraries (IEEE Xplore, ACM Digital Library, Scopus, DBLP and ScienceDirect). However, as Brereton et al. (2007) point out, “current software engineering search engines are not designed to support systematic literature reviews”. This observation is confirmed by Staples and Niazi (2007). In order to mitigate this threat we also manually processed relevant venues in a certain period of time, in particular the Agile Alliance Agile Conference (AGILE), International Conference on Agile Software Development (XP), IEEE Requirements Engineering Conference (RE), International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ), the Requirements Engineering journal (REJ) and the Agile Requirements Engineering Workshop (AREW’11). In an additional step, we also involved Google Scholar[3]. Our query string returned more than 98 000 results with Google Scholar. Of course it is not feasible to manually check this amount of items, so we decided to include the top-100 of Google Scholar as a double-check that we do not miss any important publications not indexed by the main databases mentioned above. No new articles were found with this double-check.

We used a single query string when querying the aforementioned digital libraries. It might be that we missed papers, because different terms are used by different authors. We tried to mitigate this issue by performing the aforementioned manual search, but also through citation snowballing (Wohlin 2014). We performed the snowballing recursively until we could no longer add relevant literature to the set of papers under consideration.

In Section 3.1 we have discussed and justified the inclusion and exclusion criteria for the selection strategy of publications. However, it is still possible to miss some relevant literature. One such instance is the existence of grey literature (Auger 1994) such as technical reports and PhD theses (Kitchenham and Charters 2007). In this literature review, we did

[3] `http://scholar.google.com`, last visited July 18th, 2015

not include such information. On the other hand, we did include book chapters for which we know that they underwent an external review process ([SS05]).

This work is centred around agile requirements *specifications*, thus leaving out face-to-face communication or prototyping, both elements which are very important in an agile setting (Beck et al. 2001). While we acknowledge this importance, Inayat et al. (2014) explicitly mention situations (e.g., distributed teams, large teams, complex projects) that require documented agile requirements that are fully elaborated in writing. As soon as agile teams cannot rely on face- to-face communication the correctness of written documentation becomes more and more important.

Inconsistent terminology or use of different terminology with respect to the exercised search string (see Section 3.2) may have biased the identification of primary studies. The manual search and recursive snowballing approach we followed should mitigate the risk that we have missed important articles.

Agile requirements engineering in general and quality criteria for these agile requirements are young research areas. This shows in the papers that we surveyed as many are preliminary in nature and have the purpose to launch and discuss ideas. These papers typically do not contain a full-blown evaluation. In fact, by looking at Table 5 we observe that 6 out of 16 papers contain no evaluation at all, while another 3 have a preliminary evaluation. There is no immediate mitigation for this risk, but it is our opinion that each of these papers has an important contribution.

Another threat to validity is the strength of evidence with regard to the quality criteria as mentioned in the papers that we surveyed. We acknowledge that not all papers have quality criteria for agile requirements as their primary goal of investigation. While there is no immediate mitigation for this risk, it is our view that each of the insights generated in these papers is an important piece of information for the overall area.

#### *7.3.2 Reliability*

Reliability focuses on whether the data is collected and the analysis is conducted in a way that it can be repeated by other researchers with the same results.

In this paper we have presented a series of findings based on the papers that we selected during our systematic literature review. Since conducting a survey is a largely manual task, most threats to validity relate to the possibility of researcher bias, and thus to the concern that other researchers might come to different results and conclusions. One remedy we adopted is to follow, where possible, guidelines on conducting systematic reviews as suggested by, e.g. Kitchenham (2004) and Brereton et al. (2007). In particular, we documented and reviewed all steps we made in advance (per pass), including selection criteria and attribute definitions.

The first author of this paper did the actual paper selection and refinement. The application of the inclusion and exclusion criteria might however be subjective. Therefore, the second author was involved in a series of checks in which the second author re-applied the inclusion and exclusion criteria. The results showed some discrepancies, which were discussed among the authors and resulted in a common understanding of why a paper should be included or excluded from the set.

We acknowledge that the classification of the articles in Table 5 is probably the most subjective step in our approach. This step is also subject to researcher bias as the interpretation may seek for results that the reviewers were looking for. Our countermeasure has been a systematic approach towards the analysis, including the use of an established framework for research types as proposed by Wieringa et al. (2005).

### *7.3.3 Internal Validity*

Internal validity is concerned with the analysis of the data. Since no statistical analysis was done considering the small sample size, the analysis is mostly qualitative and thus subject to researcher bias (also see Section 7.3.2).

### *7.3.4 External Validity*

External validity is about generalizations of the findings derived from the primary studies.

As the field of quality criteria for agile requirements is flourishing, the observations we have made in this paper are valid at the time of writing, but might not generalize into the future. More specifically, it might be that future work extends upon the 28 quality criteria that we have currently listed.

Similarly, while we have paid a lot of effort on finding all relevant literature (also see Section 7.3.1), it might still be that the list of 28 quality criteria is incomplete.

We are aware of the fact that agile requirements engineering in general is a very actively debated topic among practitioners through for example blog posts. We excluded such non-reviewed material in our study. The insights that are thus generated might not have been reported upon in an academic paper. Future work might want to investigate this particular aspect, in a similar way to the structured mapping study by Kabbedijk et al. (2015) who combined the academic and industrial perspective in their particular area.

## 8 Conclusion

For traditional up-front requirements specifications quality is considered important and standards define what quality of requirements specifications means. Our main research question is which quality criteria apply to agile requirements specifications, since no standards have been defined for them.

In this paper we report on a systematic literature review on quality criteria for agile requirements specifications. Through a structured process we selected 16 articles. From these articles we devised an overview of existing quality attributes that can be used in informal requirements verification. We also derived (1) recommendations for practitioners that want to perform informal verification of requirements specifications in an agile setting and (2) a research agenda for academics that highlights the need for further research in this area. In addition, the resulting systematic overview is useful as a reference work for researchers in the field of informal requirements verification and helps them identify both related work and new research opportunities in this field.

Figure 2 shows the list of 28 quality criteria that we collected. Most of these criteria are not new for agile requirements specifications, although the way to apply them to the specification might be slightly different from a traditional setting. The only new criterion we encountered is INVEST (Independent, Negotiable, Valuable, Estimable, Small and Testable).

Practitioners working with agile requirements can take the recommendations in Section 6 as a starting point. Our main recommendation is to follow the list of quality criteria in this paper. When a team includes the applicable criteria in the Definition of Ready for their agile requirements, the assessment of quality criteria on those requirements will be an undeniable part of the daily agile development process.

In Section 7.2 we present our research agenda for the area of quality criteria for agile requirements. Our overall observation is that only few papers focus on quality criteria for agile requirements. However, the topic is deemed important by both researchers (Inayat et al. 2014) and practitioners that we encountered during our own investigations. Our most important recommendation is thus a call to arms for academia to put more effort in this area in general.

In short, this paper makes the following contributions:

– A classification of existing quality criteria that can be used for the assessment of agile requirements specifications;
– Recommendations for practitioners for quality assessment of agile requirements specifications;
– A research agenda in the area of quality of agile requirements containing 7 important avenues for future research.

This literature review shows that the number of publications on informal verification of agile requirements specifications has increased over the last few years. This indicates that there should be coming more in the next few years. We plan to keep track of new publications in this area with the goal of updating this literature review. Furthermore we plan to further evolve and validate our own framework with quality criteria and we hope that others will also build on our work. Additionally, another avenue of future work is to check which quality criteria for agile requirements specifications are described in non-academic blogs, white papers, etc.

## References

Auger CP (1994) Information Sources in Grey Literature. Bowker-Saur

Beck (2002) Test Driven Development: By Example. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA

Beck K, Beedle M, van Bennekum A, Cockburn A, Cunningham W, Fowler M, Grenning J, Highsmith J, Hunt A, Jeffries R, Kern J, Marick B, Martin RC, Mellor S, Schwaber K, Sutherland J, Thomas D (2001) Manifesto for agile software development. URL `http://agilemanifesto.org`, last visited: April 16th, 2015

Belsis P, Koutoumanos A, Sgouropoulou C (2014) Pburc: a patterns-based, unsupervised requirements clustering framework for distributed agile software development. Requirements Engineering 19(2):213–225

Bhasin S (2012) Quality assurance in agile: A study towards achieving excellence. In: AGILE India (AGILE INDIA), 2012, pp 64–67

Bjarnason E, Unterkalmsteiner M, Engstrm E, Borg M (2015) An industrial case study on test cases as requirements. In: Lassenius C, Dingsyr T, Paasivaara M (eds) Agile Processes, in Software Engineering, and Extreme Programming, Lecture Notes in Business Information Processing, vol 212, Springer International Publishing, pp 27–39, DOI 10.1007/978-3-319-18612-2_3

Brereton P, Kitchenham BA, Budgen D, Turner M, Khalil M (2007) Lessons from applying the systematic literature review process within the software engineering domain. J Syst Software 80(4):571–583

Cornelissen B, Zaidman A, van Deursen A, Moonen L, Koschke R (2009) A systematic survey of program comprehension through dynamic analysis. IEEE Transactions on Software Engineering 35(5):684–702

Davis A, Hickey A (2009) A quantitative assessment of requirements engineering publications 1963-2008. In: Glinz M, Heymans P (eds) Requirements Engineering: Foundation for Software Quality, Lecture Notes in Computer Science, vol 5512, Springer Berlin Heidelberg, pp 175–189

Davis A, Overmyer S, Jordan K, Caruso J, Dandashi F, Dinh A, Kincaid G, Ledeboer G, Reynolds P, Sitaram P, Ta A, Theofanos M (1993) Identifying and measuring quality in a software requirements specification. In: Int'l Software Metrics Symposium, Baltimore (MD), pp 141–152

De Lucia A, Qusef A (2010) Requirements engineering in agile software development. Journal of Emerging Technologies in Web Intelligence 2(3)

Desharnais JM, Kocaturk B, Abran A (2011) Using the cosmic method to evaluate the quality of the documentation of agile user stories. In: Joint Conference of the 21st Int'l Workshop on Software Measurement and 6th Int'l Conference on Software Process and Product Measurement (IWSM-MENSURA), Nara (Japan), pp 269–272

Dingsøyr T, Dyba T, Abrahamsson P (2008) A preliminary roadmap for empirical research on agile software development. In: AGILE Conference, Toronto (ON), pp 83–94

Dumas-Monette JF, Trudel S (2014) Requirements engineering quality revealed through functional size measurement: An empirical study in an agile context. In: Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement (IWSM-MENSURA), Rotterdam (The Netherlands), pp 222–232

Duncan R (2001) The quality of requirements in extreme programming. CrossTalk, June 19:22–31

Dybå T, Dingsøyr T (2008) Empirical studies of agile software development: A systematic review. Inf Softw Technol 50(9-10):833–859

Eberlein A, Leite J (2002) Agile requirements definition: A view from requirements engineering. In: Proceedings of the International Workshop on Time-Constrained Requirements Engineering (TCRE02), Essen (Germany), pp 4–8

Ernst NA, Murphy G (2012) Case studies in just-in-time requirements analysis. In: Int'l Workshop on Empirical Requirements Engineering, IEEE, Chicago (IL), pp 25–32

Farid W, Mitropoulos F (2013) Norplan: Non-functional requirements planning for agile processes. In: Proceedings of IEEE Southeastcon, Jacksonville (FL), pp 1–8

Firesmith D (2004) Generating complete, unambiguous, and verifiable requirements from stories, scenarios, and use cases. Journal of Object Technology 3(10):27–40

Gottesdiener E, Gorman M (2010) Slicing requirements for agile success. Better Software 2010(04)

Grau R, Lauenroth K, Bereza B, van Veenendaal E, van der Zee S (2014) Requirements engineering and agile development - collaborative, just enough, just in time, sustainable. IREB

Heck P, Zaidman A (2014) A quality framework for agile requirements: A practitioner's perspective. Tech. Rep. TUD-SERG-2014-006, Software Engineering Research Group, Delft University of Technology

Huo M, Verner J, Zhu L, Babar M (2004) Software quality and agile methods. In: Int'l Computer Software and Applications Conference, pp 520–525, vol.1, DOI 10.1109/CMPSAC.2004.1342889

IEEE (1998) IEEE recommended practice for software requirements specifications. IEEE Std 830-1998

Inayat I, Salim SS, Marczak S, Daneva M, Shamshirband S (2014) A systematic literature review on agile requirements engineering practices and challenges. Computers in Human

Behavior

Kabbedijk J, Bezemer CP, Jansen S, Zaidman A (2015) Defining multi-tenancy: A systematic mapping study on the academic and the industrial perspective. Journal of Systems and Software 100:139–148

Kassab M (2014) An empirical study on the requirements engineering practices for agile software development. In: 40th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA), Verona (Italy), pp 254–261

Kitchenham B, Charters S (2007) Guidelines for performing systematic literature reviews in software engineering. Tech. rep., EBSE-2007-01

Kitchenham BA (2004) Procedures for performing systematic reviews. Tech. rep.

Lee C, Guadagno L, Jia X (2003) An agile approach to capturing requirements and traceability. In: Proceedings of the 2nd International Workshop on Traceability in Emerging Forms of Software Engineering, Montreal (Canada), pp 17–23

Leffingwell D (2011) Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise, 1st edn. Addison-Wesley Professional

Liskin O, Pham R, Kiesling S, Schneider K (2014a) Why we need a granularity concept for user stories. In: Cantone G, Marchesi M (eds) Agile Processes in Software Engineering and Extreme Programming, Lecture Notes in Business Information Processing, vol 179, Springer International Publishing, pp 110–125

Liskin O, Schneider K, Fagerholm F, Münch J (2014b) Understanding the role of requirements artifacts in kanban. In: Proceedings of the 7th International Workshop on Cooperative and Human Aspects of Software Engineering, Hyderabad (India), CHASE 2014, pp 56–63

Melnik G, Maurer F, Chiasson M (2006) Executable acceptance tests for communicating business requirements: customer perspective. In: Agile Conference, 2006, pp 12 pp.–46, DOI 10.1109/AGILE.2006.26

Paetsch F, Eberlein A, Maurer F (2003) Requirements engineering and agile software development. In: Int'l Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, Linz (Austria), pp 308–308

Patel C, Ramachandran M (2009) Story card maturity model (smm): A process improvement framework for agile requirements engineering practices. JSW 4(5):422–435

Power K (2014) Definition of ready: An experience report from teams at cisco. In: Cantone G, Marchesi M (eds) Agile Processes in Software Engineering and Extreme Programming, Lecture Notes in Business Information Processing, vol 179, Springer International Publishing, pp 312–319

Ramesh B, Cao L, Baskerville R (2010) Agile requirements engineering practices and challenges: an empirical study. Information Systems Journal 20(5):449–480

Runeson P, Alexandersson M, Nyholm O (2007) Detection of duplicate defect reports using natural language processing. In: Proc. Int'l Conf. on Software Engineering (ICSE), IEEE, pp 499–510

Runeson P, Host M, Rainer A, Regnell B (2012) Case Study Research in Software Engineering: Guidelines and Examples. Wiley

Sfetsos P, Stamelos I (2010) Empirical studies on quality in agile practices: A systematic literature review. In: Seventh International Conference on the Quality of Information and Communications Technology (QUATIC), Porto (Portugal), pp 44–53

Sillitti A, Succi G (2005) Requirements engineering for agile methods. In: Aurum A, Wohlin C (eds) Engineering and Managing Software Requirements, Springer Berlin Heidelberg, chap 14, pp 309–326

Srinivasan J, Lundqvist K (2009) Using agile methods in software product development: A case study. In: Sixth International Conference on Information Technology: New Generations. ITNG '09, Las Vegas (NV), pp 1415–1420

Staples M, Niazi M (2007) Experiences using systematic review guidelines. J Syst Software 80(9):1425–1437

Wake B (2003) INVEST in good stories, and SMART tasks. `http://xp123.com/articles/invest-in-good-stories-and-smart-tasks/`, accessed Dec-2015

Wieringa R, Maiden N, Mead N, Rolland C (2005) Requirements engineering paper classification and evaluation criteria: A proposal and a discussion. Requir Eng 11(1):102–107

Wohlin C (2014) Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: Proceedings of the International Conference on Evaluation and Assessment in Software Engineering (EASE), London (UK), pp 38:1–38:10

## Appendix A Details of Selected Papers on Quality Criteria for Agile Requirements

This appendix contains detailed information on each of the 16 selected papers. We repeat author, title and venue and include number of pages. We provide a short summary of the paper based on the paper abstract. We briefly explain the relevance of the paper for quality criteria for agile requirements (in italic) and we sum up each of the quality criteria mentioned in the paper. For an overview of all quality criteria from all papers, see Figure 2.

| [ID] Reference | Title and Venue | #pp. | Summary and Relation to Quality Criteria |
|---|---|---|---|
| **[B14]** Belsis et al. (2014) | PBURC: A Patterns-Based, Unsupervised Requirements Clustering Framework for Distributed Agile Software Development<br><br>(Requirements Engineering Journal) | 13 | This paper presents a patterns-based, unsupervised requirements clustering framework, which makes use of machine-learning methods for requirements validation, being able to overcome data inconsistencies and determine appropriate requirements clusters for the definition of software development sprints.<br><br>*[B14] is of low relevance for quality criteria as it focuses on a clustering algorithm for requirements.*<br><br>*Quality criteria: priority.* |
| **[D01]** Duncan (2001) | The Quality of Requirements in Extreme Programming<br><br>(Journal of Defence Software Engineering) | 4 | This paper describes and evaluates the quality of requirements generated by using XP and discusses how the XP process can assist or hinder proper requirements engineering.<br><br>*[D01] compares eXtreme Programming to a list of quality criteria for traditional requirements from Davis et al. (1993).*<br><br>*Quality criteria: priority, product version, verifiable, concise, unambiguous.* |
| **[D11]** Desharnais et al. (2011) | Using the COSMIC Method to Evaluate the Quality of the Documentation of Agile User Stories (IWSM-MENSURA) | 4 | In the research reported here, the COSMIC method is used to analyze and report on the quality of the documentation of user stories. The functional size of evolving requirements can be measured with COSMIC measurement method. To support this measurement activity, the quality of the documentation is important to be interpreted correctly.<br><br>*See [DT14].* |
| **[DT14]** Dumas-Monette and Trudel (2014) | Requirements Engineering Quality Revealed through Functional Size Measurement: An Empirical Study in an Agile Context (IWSM-MENSURA) | 11 | This paper reports preliminary results related to a software development organization. The functional size of the software was measured and compared with development and measurement effort, taking into account the quality rating of requirements. The results led to recommendations for the organization and recommendations for planning any software measurement project. |

| [ID] Reference | Title and Venue | #pp. | Summary and Relation to Quality Criteria |
|---|---|---|---|
| | | | *Both [D11] and [DT14] focus on size measurement of agile requirements (using COSMIC). They claim that quality of those requirements is a necessary condition for accurate estimation. Quality in this sense mainly means completeness: descriptions of functional processes and data modeling artifacts are needed to base the COSMIC size measurements on. [DT14] contains a case study that also highlights other quality issues related to size measurement: user story names should at least contain the name of the functional process it maps to in order to be correctly measured; it is expected that each user story be mapped to only one functional process; consistent terminology should be used for data groups/objects.*<br><br>*Quality criteria: functional processes, data modeling artifacts, functionality in story name, small/simple, consistent/defined terminology.* |
| **[DQ10]** De Lucia and Qusef (2010) | Requirements Engineering in Agile Software Development (Journal of Emerging Technologies in Web Intelligence) | 9 | This paper discusses problems concerned with requirements engineering in agile software development processes and suggests some improvements to solve some challenges caused by large projects. The paper also discusses the requirements traceability problem in agile software development.<br><br>*[DQ10] is a general overview paper on requirements engineering for agile methods. The focus is not specifically on quality, but [DQ10] mentions some criteria: non-functional requirements should not be forgotten, requirements should be in the language of the customer, requirements should not be too complex, requirements should be forward-traceable (i.e. to source code and tests).*<br><br>*Quality criteria: non-functional requirements, small/simple, customer language, forward traceable.* |
| **[F04]** Firesmith (2004) | Generating Complete, Unambiguous, and Verifiable Requirements from Stories, Scenarios, and Use Cases<br><br>(Journal of Object Technology) | 13 | This paper shows how to transform incomplete and vague stories, scenarios, and use cases into a proper set of textual requirements.<br><br>*[F04] promotes a template for writing textual requirements: ID, trigger, precondition, action, postcondition.*<br><br>*Quality criteria: unique ID, template, unambiguous.* |

| [ID] Reference | Title and Venue | #pp. | Summary and Relation to Quality Criteria |
|---|---|---|---|
| **[FM13]** Farid and Mitropoulos (2013) | NORPLAN: Non-functional Requirements Planning for Agile Processes (IEEE Southeastcon) | 8 | This research proposes project management and requirements quality metrics. NORPLAN proposes two prioritization schemes, Riskiest-Requirements-First and Riskiest-Requirements-Last, for planning release and sprint cycles using a risk-driven approach. The approach is validated through visual simulation and a case study.<br><br>*[FM13] focuses on non-functional requirements planning for agile processes. It contains a table with agile requirements quality metrics. However, these metrics are mainly process-related, except for ambiguity.*<br><br>*Quality criteria: non-functional requirements, unambiguous.* |
| **[GG10]** Gottesdiener and Gorman (2010) | Slicing Requirements for Agile Success<br><br>(Better Software Magazine) | 8 | This paper presents different options for slicing user stories.<br><br>*[GG10] does not specifically focus on quality, but presents a way of slicing (splitting) user stories. By explaining the ways of slicing, they also mention six important elements of a user story: user roles, actions, data objects, business rules, interfaces, quality attributes (i.e. non-functional requirements).*<br><br>*Quality criteria: six elements of a user story.* |
| **[HZ14]** Heck and Zaidman (2014) | A Quality Framework for Agile Requirements: A Practitioner's Perspective (TU Delft Computer Science Report) | 11 | This paper presents a quality framework for informal verification of agile requirements and instantiates it for feature requests in open source projects. The framework was derived based on an existing framework for traditional requirements specifications, literature about agile and open source requirements and the authors' experience with agile and open source requirements.<br><br>*[HZ14] presents a framework for agile requirements quality criteria, which is instantiated for feature requests in open source projects and user stories. It presents a large number of quality criteria. The use of a template is indirectly also promoted because they claim that a tool should be used to store the requirements. This tool, e.g. an issue tracker, would have predefined fields for each requirement, thereby functioning like a template.*<br><br>*Quality criteria: priority, unique ID, product version, acceptance criteria, rationale, template, only relevant comments, no contradiction/conflict, INVEST, consistent/defined terminology, correct language, forward traceable, no duplicates, navigable and explained links.* |

| [ID] Reference | Title and Venue | #pp. | Summary and Relation to Quality Criteria |
|---|---|---|---|
| **[L03]** Lee et al. (2003) | An Agile Approach to Capturing Requirements and Traceability<br><br>(Workshop on Traceability in Emerging Forms of Software Engineering) | 7 | This paper presents a tool-based approach that provides for the implicit recording and management of relationships between conversations about requirements, specifications, and design decisions.<br><br>*[L03] is of low relevance for our research question as it focuses on a tool for traceability. However, it mentions forward traceability (e.g., to tests and source code) as an important quality aspects of agile requirements.*<br><br>*Quality criteria: forward traceable.* |
| **[L14a]** Liskin et al. (2014a) | Why We Need a Granularity Concept for User Stories (XP) | 16 | This paper investigates Expected Implementation Duration of a user story as a characteristic of granularity by conducting a study with software engineering practitioners. Many developers have experienced certain problems to occur more often with coarse user stories. The findings emphasize the importance to reflect on granularity when working with user stories.<br><br>*[L14a] and [L14b] both focus on the usefulness of Expected Implementation Duration (EID) as a quality criterion for user stories. They observe that user stories should be kept small in terms of this EID.*<br><br>*Quality criteria: small/simple.* |
| **[L14b]** Liskin et al. (2014b) | Understanding the Role of Requirements Artifacts in Kanban (Workshop on Cooperative and Human Aspects of Software Engineering) | 8 | This paper explores how to utilize requirements artifacts effectively, what their benefits and challenges are, and how their scope granularity affects their utility. It studies a software project carried out in the University of Helsinki. The requirements artifacts were investigated and then developers and the customer were interviewed about their experiences.<br><br>*See [L14a].* |
| **[P14]** Power (2014) | Definition of Ready: An Experience Report from Teams at Cisco (XP) | 8 | This paper presents an example of definition of ready used by agile teams in Cisco. These teams have developed three levels of ready that apply for user stories, sprints and releases. The paper describes how definition of ready provides a focus for backlog grooming, and some consequences of not meeting definition of ready. The paper finishes with perspectives from different roles in the organization.<br><br>*[P14] gives an example of the definition of ready (when are user stories ready to start implementing them?) for Cisco teams. This list of criteria mainly contains process-related items, but also some product-related: non-functional requirements should be identified, acceptance criteria should be identified, dependencies should be identified, size should have been estimated, team knows what it will mean to demo the user story (i.e. verifiable).* |

| [ID] Reference | Title and Venue | #pp. | Summary and Relation to Quality Criteria |
|---|---|---|---|
| | | | *Quality criteria: non-functional requirements, acceptance criteria, dependencies, size, verifiable.* |
| **[PR09]** Patel and Ramachandran (2009) | Story Card Maturity Model (SMM): A Process Improvement Framework for Agile Requirements Engineering Practices (Journal of Software) | 14 | This paper describes an ongoing process to define a suitable process improvement model for story cards based requirement engineering at agile software development environments: the SMM (based on CMM). It also presents how the identified areas of improvement from assessment can be mapped with best knowledge based story cards practices for agile software development environments.<br><br>*[PR09] focuses on story cards. The appendix contains a list of guidelines for story cards, unfortunately without a detailed description of each guideline. The guidelines are placed in a maturity framework.*<br><br>*Quality criteria: priority, unique ID, non-functional requirements, acceptance criteria, story card structure, no contradiction/conflict, concise, INVEST, consistent/defined terminology, unambiguous.* |
| **[SL09]** Srinivasan and Lundqvist (2009) | Using Agile Methods in Software Product Development: A Case Study (ITNG) | 6 | This paper presents an in-depth case study of agile methods adoption in a software product development firm. Using a mix of interviews, observation and archival data, the evolution of agile adoption within the firm is reconstructed.<br><br>*[SL09] is a short case study of the use of agile software development in a small company. They do not specifically mention requirements quality criteria, but they observe some problems that occurred because of the ambiguity of the requirements.*<br><br>*Quality criteria: unambiguous.* |
| **[SS05]** Sillitti and Succi (2005) | Requirements Engineering for Agile Methods (Engineering and Managing Software Requirements) | 18 | This paper introduces Agile Methods as the implementation of the principles of the lean production in software development. It discusses several agile practices that deal with requirements. These practices focus on a continuous interaction with the customer to address the requirements evolution over time, prioritize them, and deliver the most valuable functionalities first.<br><br>*[SS05] is a general overview paper on requirements engineering for agile methods. The focus is not specifically on quality, but [SS05] mentions some criteria.*<br><br>*Quality criteria: priority, non-functional requirements, independent, small/simple, customer language.* |