Computers in Human Behavior xxx (2014) xxx–xxx

Contents lists available at ScienceDirect

Computers in Human Behavior

journal homepage: www.elsevier.com/locate/comphumbeh

Review

A systematic literature review on agile requirements engineering practices and challenges

Irum Inayat [a,*], Siti Salwah Salim [a], Sabrina Marczak [b], Maya Daneva [c], Shahaboddin Shamshirband [d,e]

[a] Department of Software Engineering, Faculty of Computer Science and Information Technology, University of Malaya (UM), 50603, Malaysia
[b] School of Computer Science, Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS) University, Rio de Janerio, Brazil
[c] Information Science Research Group, University of Twente, Enschede, The Netherlands
[d] Department of Information Systems, Faculty of Computer Science and Information Technology, University of Malaya (UM), 50603, Malaysia
[e] Department of Computer Science, Chalous Branch, Islamic Azad University (IAU), 46615-397 Chalous, Mazandaran, Iran

A R T I C L E I N F O

Article history:
Available online xxxx

Keywords:
Agile software development methods
Agile requirements engineering
Collaboration
Traditional requirements engineering
Systematic review

A B S T R A C T

Unlike traditional software development methods, agile methods are marked by extensive collaboration, i.e. face-to-face communication. Although claimed to be beneficial, the software development community as a whole is still unfamiliar with the role of the requirements engineering practices in agile methods. The term "agile requirements engineering" is used to define the "agile way" of planning, executing and reasoning about requirements engineering activities. Moreover, not much is known about the challenges posed by collaboration-oriented agile way of dealing with requirements engineering activities. Our goal is to map the evidence available about requirements engineering practices adopted and challenges faced by agile teams in order to understand how traditional requirements engineering issues are resolved using agile requirements engineering. We conducted a systematic review of literature published between 2002 and June 2013 and identified 21 papers, that discuss agile requirements engineering. We formulated and applied specific inclusion and exclusion criteria in two distinct rounds to determine the most relevant studies for our research goal. The review identified 17 practices of agile requirements engineering, five challenges traceable to traditional requirements engineering that were overcome by agile requirements engineering, and eight challenges posed by the practice of agile requirements engineering. However, our findings suggest that agile requirements engineering as a research context needs additional attention and more empirical results are required to better understand the impact of agile requirements engineering practices e.g. dealing with non-functional requirements and self-organising teams.

Contents

1. Introduction 00
2. Related work. 00
3. Research method 00
 3.1. Planning the review 00
  3.1.1. Review objectives and research questions 00
  3.1.2. Search strategy 00
  3.1.3. Search criteria. 00
  3.1.4. Inclusion and exclusion criteria. 00
 3.2. Conducting the review 00
  3.2.1. Study search and selection. 00
  3.2.2. Data extraction and synthesis 00
  3.2.3. Methodological quality assessment. 00
4. Findings of our review 00
 4.1. Overview of studies. 00
 4.2. (RQ1) What are the adopted practices of agile RE according to published empirical studies? 00

* Corresponding author.
E-mail address: irum@siswa.um.edu.my (I. Inayat).

http://dx.doi.org/10.1016/j.chb.2014.10.046
0747-5632/© 2014 Elsevier Ltd. All rights reserved.

4.3. (RQ2) What are the challenges of traditional RE that are resolved by agile RE? . . . 00
4.4. (RQ3) What are the practical challenges of agile RE? . . . 00
5. Discussion of the results . . . 00
6. Conclusions . . . 00
6.1. Implications of the study . . . 00
6.2. Limitations of our study . . . 00
Acknowledgement . . . 00
Appendix A. List of reviewed studies . . . 00
References . . . 00

1. Introduction

The Agile Manifesto states that priority should be given to "individuals and interaction over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, and responding to changes over following a plan" (Beck et al., 2001). These agile principles incorporate flexibility by cordially receiving changes to project scope and requirements definitions (Bang, 2007). Overall, a high-level project scope is defined upfront and is revisited in each iteration. Therein, requirements are initially defined with the customer and listed in a customer wish list format; every couple of weeks they are discussed (e.g. in the Scrum method), better understood, and reprioritised, to define the scope of the next iteration. The customer works closely with the development team to achieve such definitions and to constantly validate the product being delivered. The development process is dynamic and open to changes in areas that can be identified at any given moment. Literature reports those projects that adopt agile methods exhibiting higher productivity (Eberlein & Julio Cesar, 2002), less rework (Bin, Xiaohu, Zhijun, & Maddineni, 2004), and more efficient defect fixing rates (Lagerberg & Skude, 2013). In addition, agile methods reduce risks in global software development (GSD) and diminish the need for coordination efforts, which result in an increase of productivity (Hossain, Babar, & Verner, 2009).

Requirements Engineering (RE) practices such as observations, interviews, workshops and strong team collaboration are embedded in iteration-based agile methods (Zhu, 2009). Likewise, RE practices such as customer involvement, requirements prioritisation (Cao & Ramesh, 2008; Ramesh, Baskerville, & Cao, 2010), requirements modelling (Boness & Harrison, 2007), requirements documentation (Wolfgang, 2011), have also been suggested to be used with agile methods.

Although the practices mentioned above provide an essence of the "agile way" of dealing with requirements, the software development community still knows little about the role of the RE processes and practices in such a flexible and dynamic way of working, and how such practices can resolve frequently reported issues in traditional RE processes. Although claimed to be beneficial, the adoption of agile methods might impact the way that RE activities are conducted and pose some new challenges to their realisation. We are motivated to close this gap of knowledge and embarked on mapping out the published evidence available about RE practices adopted and challenges faced by agile teams. The purpose is to learn how traditional RE issues are resolved by this new software development approach.

The remainder of this paper is structured as follows: Section 2 discusses previous literature reviews on agile software engineering, identifies a gap in literature and a need for a deeper investigation of RE processes in agile software engineering. Section 3 presents our research questions and the method followed for the review of contemporary practices in agile RE. Section 4 summarises the key findings of our study. Section 5 provides a discussion on the results. Section 6 concludes the article, provides implications for researchers and industry practitioners and defines the limitations of this study.

2. Related work

In the software engineering research literature, there are a few examples of reviews on agile methods, as (summarised in Table 1) usability issues (Hasnain, 2010) in agile methods and ways to resolve them (Silva da Silva, Martin, Maurer, & Silveira, 2011); agile methods in GSD (Hossain, Babar, & Paik, 2009; Jalali & Wohlin, 2011; Rizvi, 2013), and in open source software development (Gandomani, Zulzalil, Ghani, & Sultan, 2013).

Hossain et al. (2009) conducted a systematic literature review to focus on the practices used in the GSD projects using Scrum methods, the challenges that restrict the use of Scrum methodology and the solution to prevent them. The findings help researchers and practitioners to understand the challenges involved in using Scrum for GSD projects and the strategies available to deal with them.

Hasnain (2010) conducted a systematic literature review to identify the agile practices as well as the human and technical factors pointed out in agile studies, published within 2003–2007. The review revealed that agile RE practices had only been discussed in the literature from the overall perspective of agile methods and not in the context of any particular methods such as Scrum, test-driven development, etc. Hasnain's findings suggest that more empirical results are required on agile methods, in particular XP (Extreme Programming) (Beck, 1999) and Scrum (Schwaber & Beedle, 2001), in order to discuss the details from the practitioner's point of view.

Silva da Silva et al. (2011) conducted a systematic literature review on the topic of the integration of agile methods and user-centred design approaches. The review focused on usability issues in agile methods with respect to design. The findings show that usability issues in agile methods can be addressed by incorporating a user centred design specialist (UCDS) role in agile teams. The authors also defined practices to resolve usability issues in agile methods such as Little Design Up Front, Big Design Up Front, low fidelity prototypes, user testing, interaction models, and close collaboration.

Barlow et al. (2011) examined the effect of the usage of agile development practices in large organisations. The literature review contributed towards the formulation of a framework that provides guidelines to large organisations adopting agile methods. The findings of this review assist the practitioners to adopt software development methods in their organisations.

Jalali and Wohlin (2011) conducted a systematic literature review on studies comprising the combination of agile methods with global software engineering from 1999 to 2009. The review results showed that much attention had been given to agile methods from 2004 to 2009. In addition, the findings revealed that

Table 1
Summary of selected literature on agile software development reviews at large.

Reference	Goal	Research questions/goals
Hossain et al. (2009)	Review of Scrum methods used for GSD	RQ.1 What is currently known about the use of the Scrum practices in GSD projects? More specifically, this study focuses on the following two questions: RQ1. What challenges or risk factors restrict the use of Scrum practices in globally distributed projects? RQ2. What strategies or practices are being commonly used to deal with these challenging factors to support the use of Scrum practices in globally distributed projects?
Hasnain (2010)	Review of existing work on agile methods to identify the gaps	RQ.1 Do practitioners or academics publish in IEEE agile conferences and how have these changed over time? RQ.2 Are the published agile papers based on experience or empirical studies and how have these changed over time? RQ.3 Which agile methodology is the focus of research and how has this changed over time? RQ.4 Are human factors or technical factors the focus of agile papers and how have these changed over time?
Silva da Silva et al. (2011)	Integration of agile software development methods with user-centric design approaches	RQ.1 How are usability issues addressed in agile projects? RQ.1 What are the common practices used to address usability issues in agile methods?
Barlow et al. (2011)	Examination of the effects of agile development practices in large organisations	Research goals: 1. To examine the effects of agile development practices in large organisations 2. To further organise theory and observations into a framework with guidelines for large organisations considering agile methodologies
Jalali and Wohlin (2011)	Combining agile methods with global software development (GSE)	RQ.1 What is reported in the peer-reviewed research literature about agile practices in GSE? RQ.2 Which agile practices in which GSE settings, under which circumstances have been successfully applied?
Rizvi (2013)	Review of existing works to identify challenges and solutions of distributed agile software development in organisations (DASE)	RQ.1 What are the conditions under which organisations choose to adopt DASE? RQ.2 What are the biggest threats when adopting DASE? RQ.3 What flavour of the agile methodology is most adopted in DASE? RQ.4 What is the strength of evidence in supporting the findings of the above questions?
Gandomani et al. (2013)	Relationship between agile software development and open source software development (OSSD)	RQ.1 Is there any relationship between agile software development and OSSD? RQ.2 Are practices of one of them applicable to the second? RQ.3 Can they integrate with each other?

the focus of the majority of studies was on providing empirical results in the form of industrial experiences. Therefore, the authors emphasised the need for having a framework that incorporates agile methods in global software development. Findings also suggested that more empirical studies are needed on the subject of agile methods used by globally distributed teams.

Rizvi (2013) conducted a systematic literature review on distributed agile software development. The review aimed to study the way in which organisations adopted distributed agile software development. In addition, the review focused on the challenges and their solutions from 2007 to 2012. Rizvi's findings revealed communication, collaboration, coordination and cultural differences as major challenges of distributed agile development. The review also emphasised the importance of having an infrastructure for communication and collaboration to address the identified challenges.

Gandomani et al. (2013) conducted a systematic literature review on the relationship between agile methods and open source software development. These authors found that agile software development supports open source software development since both share several principles and practices such as self-organised teams and shared goals. However, the authors claimed that additional empirical evidence is required to demonstrate the validity of agile methods for open source development.

Although the reviews highlight that agile methods have been the focus of intense research activity, little is known about requirements engineering in agile methods (also called agile requirements engineering), their processes, practices and challenges. The table below summarises some studies that are found to be specifically dedicated to agile RE. The purpose of carrying out a detailed systematic literature review is to aggregate the evidence on agile RE and hence, close the research gap or at least narrow it.

3. Research method

In our research process, we followed the guidelines proposed by Kitchenham and Charters (2007). Subsequently, we presented the main steps of our systematic review, namely planning, conducting and reporting the review results (Kitchenham & Charters, 2007).

3.1. Planning the review

We planned this review by proposing research questions relevant to our research objectives. We defined the search strategy, search string and inclusion/exclusion criteria. We present these in more detail below.

3.1.1. Review objectives and research questions

With the increased use of agile methods in software development, it is of utmost importance to study the role of RE in agile methods. As defined in RE textbooks, e.g. Kotonya and Sommerville (1997), the RE process comprising five main activities, which are requirement elicitation, analysis and negotiation, documentation, validation, and management, is used in traditional software development life-cycle models, i.e. waterfall models. When RE is considered in such context, using traditional development methods, it is termed "traditional RE" (Cao & Ramesh, 2008). Although there have been several contributions to the field of incorporating traditional RE activities into agile software development, there is a lack of a coherent and consolidated views on the topic. Therefore, the main goal of this work is to develop an understanding of RE practices in agile methods, and about the challenges that teams face when doing RE in such a context. We also sought to find the agile RE practices that can resolve the challenges of tradi-

Table 2
Search sources.

Electronic databases	ACM Digital library IEEE Xplore SpringerLink EI Compendex Inspec ISI Web of Knowledge ScienceDirect
Searched items	Journal, workshop and conference papers
Search applied on	Full text—to avoid missing any of the papers that do not include our search keywords in titles or abstracts, but are relevant to the review object
Language	English
Publication period	From January 2002 to June 2013

tional RE. To fulfil these objectives, we formulated the following research questions:

RQ1. What are the adopted practices of agile RE according to published empirical studies?
RQ2. What are the challenges of traditional RE that may get alleviated by agile RE?
RQ3. What are the challenges of agile RE?

3.1.2. Search strategy

The study by Kitchenham and Charters (2007) was used as a guideline for carrying out the research. After defining our research goals and questions, we started with the formulation of a formal search strategy to analyse all available empirical materials specific to the objective of this review. The plan involved defining the search space, which included electronic databases and printed proceedings as shown in Table 2. The studies were initially retrieved from the electronic databases and then analysed to identify other meaningful studies through reference searches (snowballing). In addition, we also consulted the related publications of the authors of the papers identified in DBLP (Digital Bibliographic Library Browser) database. This supplementary strategy aimed to add any potential works that might have been left out. Then the inclusion and exclusion criteria were applied on the retrieved studies in two distinct rounds, involving a different number of researchers as explained in Section 3.1.4.

3.1.3. Search criteria

The search criteria used for this review consist of two parts—C1 and C2, defined as follows:

- C1 is a string made up of keywords related to agile software development methods such as agility, agile, Scrum, XP (Extreme Programming), FDD (Feature-Driven Development), TDD (Test-Driven Development), Lean, Kanban;
- C2 is a string made up of keywords related to requirements engineering such as “requirements engineering”, “requirement”, “user story”, and “feature”.

Eq. (1). Boolean expression search criteria

$$C1 \textit{ AND } C2 \tag{1}$$

An example of a search done in the electronic databases is shown below:

Software AND (agile OR agility OR scrum OR “XP” OR “extreme programming” OR fdd OR “feature-driven development” OR “feature-driven” OR tdd OR “test-driven development” OR “test-driven” OR lean OR kanban) AND (“requirements engineering” OR “requirements” or “user story” OR “feature” OR “prioritisation”).

We composed the search string in each database manually based on the search functionality offered by that database. We treated the search in each database as a process of learning and experimentation.

3.1.4. Inclusion and exclusion criteria

To determine whether a study should be included, the following inclusion and exclusion criteria were used:

Inclusion criteria: (I1) the study is a peer-reviewed publication; (I2) the study is in English; (I3) it is relevant to the search terms defined in Section 3.1.3; (I4) it is an empirical research paper, an experience report, or workshop paper; and (I5) the study is published between January 2002 and June 2013.
Exclusion criteria: (E1) studies that do not focus explicitly on agile methods, but only refer to agile software development methods as a side topic (e.g. studies that cite agile as an adjective); (E2) studies that do not discuss RE in agile methods; (E3) studies that do not meet inclusion criteria; and (E4) opinion, viewpoint, keynote, discussions, editorials, comments, tutorials, prefaces, and anecdote papers and presentations in slide formats without any associated papers.

3.2. Conducting the review

In this section, we present the findings of our search and extraction of information from relevant sources and databases.

3.2.1. Study search and selection

By following the search strategy (previously explained in Section 3.1.2), the selected electronic databases were searched and the studies retrieved. In this original search, we retrieved 543 studies as shown in Table 3. It is important to note that we only selected databases that publish peer-reviewed papers (I1). An extensive inspection of the studies’ titles and abstracts was made by one of the researchers (Round 1) by applying the inclusion criteria. Most of the retrieved studies fell within the inclusion criteria I2, I3 and I5. Due to limitations of the search engines in applying the search string to the entire body of text of the paper, a substantial number of results retrieved were then discarded. As a result of this first round of classification, we ended up with 51 candidate studies. We also made sure that the retrieved papers were not discussions, editorials, comments, tutorials, prefaces and presentations (I4). Then, in Round 2, the pre-selected studies were assessed by a second (one of the co-authors) and a third (independent and experienced) researcher in order to apply the exclusion criteria (E1, E2, E3, and E4). To review the agreements and disagreements raised by the researchers in their assessments, we conducted a face-to-face consensus meeting. For the papers where consensus was not reached, the three researchers read the entire paper and then excluded the studies based on the defined exclusion criteria. Out of the 51 studies pre-selected after the application of the inclusion criteria, 23 were excluded on the ground that they did not discuss any topic directly related to the scope of our investigation (E1 to E4). Twenty-two of them referred to methods, models, and frameworks for agile requirements engineering and one of them discussed the role of a requirements engineer as a liaison officer for agile requirements engineering. Therefore, our final selection consists of 21 studies (see the two rightmost columns in Table 3). The complete list of studies is available in Appendix A, found at the end of this paper.

3.2.2. Data extraction and synthesis

According to the guidelines provided by Kitchenham and Charters (2007), we defined a data extraction process to identify

Table 3
Number of identified studies during the distinct rounds of our systematic search.

Database	Retrieved	Round 1		Round 2	
		Included	Excluded	Included	Excluded
ISI Web of Knowledge	63	10	53	3	7
Wiley	12	4	8	4	0
Emerald	19	3	16	1	2
Springer Link	27	5	22	4	1
Taylor & Francis Online	12	1	11	1	0
Science Direct	63	7	56	3	4
IEEE Xplore	168	14	154	4	10
ACM	179	7	172	1	6
Total	543	51	492	21	30

relevant information from the 21 included primary studies that pertain to our research questions. Our data extraction process includes the following: First, we set up a form to record ideas, concepts, contributions, and findings of each of the 21 studies. Using this form ensures subsequent higher-order interpretation. The following data were extracted from each publication: (i) review date; (ii) title; (iii) authors; (iv) reference; (v) database; (vi) relevance to the theme, i.e. agile requirements engineering issues, challenges, practices, models, methods, techniques; (vii) methodology (interview, case study, report, survey); (viii) data analysis; (ix) validation techniques; (x) future work; (xi) limitations; (xii) country/location of the analysis; and (xiii) year of publication.

Once the extraction was completed, we used content analysis (Elo & Kyngäs, 2007; Hsieh & Shannon, 2005) to characterise the focus (e.g. shared recommendations and lessons learnt) of each study. Finally, we assessed the results of our data extraction by using an inter-rater agreement among the researchers (Fleiss, Levin, & Paik, 2003). To find the inter-rater agreement among the three researchers, the Kappa coefficient, which is a statistical measure, was used. The value of the Kappa coefficient was calculated to be 0.67, which indicates good or substantial agreement. For the sake of clarity, we note that values <0 indicate no agreement, while values in the range of 0–0.20 indicate slight agreement; values in the range of 0.21–0.40 indicate fair agreement, values in the range of 0.41–0.60 indicate moderate agreement, values in the range of 0.61–0.80 indicate substantial agreement, and values in the range of 0.81–1 indicate nearly perfect agreement (Landis & Kosh, 1977). Subsequently, independent quality assessments were conducted for the 21 studies, as explained in Section 3.1.4, and disagreements were resolved through a discussion.

3.2.3. Methodological quality assessment

This systematic review used the quality criteria initially proposed by Guyatt, Rennie, Meade, and Cook (2008) for assessing the methodological quality of the primary studies selected for review; the criteria were later used by Dybå and Dingsøyr (2008) for assessing the quality of empirical studies discussing agile software development methods. These quality criteria (presented in Table 4) comprise questions that provide a measure of the extent to which a study is satisfactory and will contribute to the scope of the investigation. The criteria cover thoroughness, trustworthiness, and significance of the studies. We chose these criteria because (i) they can be used to investigate the usefulness of synthesis findings and result interpretation (Kitchenham & Charters, 2007) and (ii) the quality measures associated with these criteria were previously employed in several recent systematic reviews (Dybå & Dingsøyr, 2008; Pacheco & Garcia, 2012).

We evaluated each study according to the quality assessment criteria presented in Table 4. For a better categorisation and rating of the studies, we utilised an ordinal scale that was based on our quality assessment criteria (Table 4), instead of a dichotomous scale. The first criterion (C1) involved assessing the objective of each study. This question was answered positively in 92% of the studies. The second criterion (C2) assessed whether the research context was properly addressed and described. This question was answered positively in 87% of the studies. In the third criterion (C3), we inquired about a clear statement of findings in each study. This question was answered positively in 89% of the studies. The heuristic scores for the quality measures (C4) were established by the three above-mentioned researchers. The normalised scores of the selected studies, which are based on their quality scores, are shown in Fig. 1.

4. Findings of our review

In this section, we describe the findings of our review in light of our research questions.

4.1. Overview of studies

As previously mentioned, we identified 21 studies. Table 5 presents the distribution of the studies' publication sources. Of the 21 studies, about 57% (12 of them) were published in conferences, 19% (4 of them) in journals, 19% (4 of them) in workshops, and 5% (1 only) in a magazine.

Our results in Table 5 also suggest that the studies are equally distributed across the publication venues. In fact, each of the publication sources has one or two papers published in it. This means there is no single source that is preferred by agile RE authors.

Regarding the years of publication, we did not find any significant studies related to our research topic prior to 2002. The distribution of reviewed papers, which were published from 2002 to 2013, is presented along with the topics of interest of our investigation (see Fig. 2a and 2b, respectively).

Regarding the topics that formed the focus of the 21 studies, Fig. 2b indicates that 29% are on agile RE practices, 28% are on newly proposed ideas in the form of methods. With respect to techniques and models for agile RE, only 5% are based specifically on the comparison of traditional RE with agile RE while the remaining 38% of the studies discuss agile RE in general.

In our set of 21 studies, we observed that for most of them, the co-authors were affiliated to different countries within a single study. However, it is not possible to determine authorship per geographical distribution per study. However, if we consider the location of each author individually, we can note that most of them are from North America and Europe (refer to Fig. 3). It is also possible

Table 4
Quality criteria for study selection.

Criteria	Response grading	Grade obtained
(C1) Is the research aim/objective clearly defined?	{1, 0.5,0}(Yes, nominally, No)	22 studies, 92%
(C2) Is the context of research well addressed?	{1, 0.5,0}(Yes, nominally, No)	20 studies, 87%
(C3) Are the findings clearly stated?	{1, 0.5,0}(Yes, nominally, No)	20 studies, 89%
(C4) Based on the findings, how valuable is the research?	>80% = 1, <20% = 0, in-between = 0.5	

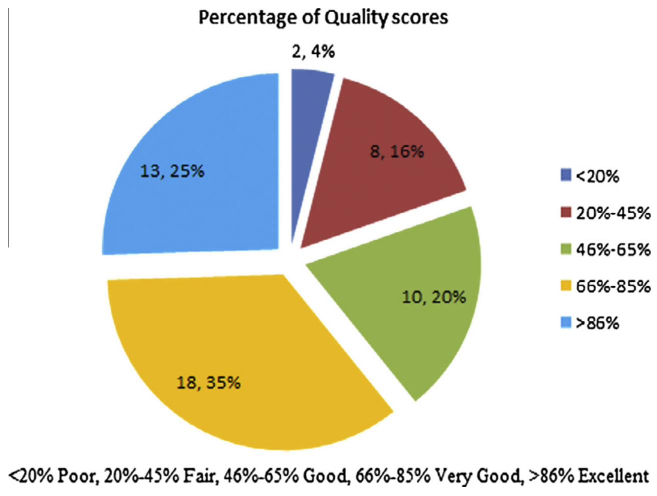

Fig. 1. Percentage scores for quality assessments of studies.

Table 5
Distribution of studies according to the publication channel.

Publication source	Type	Number
IEEE Software	Magazine	1
Information and Software Technology	Journal	1
Systems & Software	Journal	1
Information Systems	Journal	2
Time Constrained Requirements Engineering	Workshop	2
IEEE International Software Metrics Symposium	Conference	1
Requirements Engineering	Conference	1
Systems Engineering	Conference	1
Second World Congress of Software Engineering	Conference	1
Empirical Software Engineering and Measurement	Conference	1
Agile Requirements Engineering	Workshop	2
Malaysian Software Engineering Conference	Conference	1
International Conference on Communication and Information Technology	Conference	1
Hawaiian International Conference on Systems Science	Conference	1
IEEE SoutheastCon: Student and Technical Conference	Conference	2
Annual Systems Engineering Conference	Conference	1
IEEE Conference on Research Challenges in Information Science	Conference	1

to see that there are relatively few studies on agile RE from Asian countries and there are none from South American or African countries.

Fig. 4 presents the research methods that were adopted in the 21 selected studies. First, we found that the majority of studies are exploratory in nature. Out of the 21 studies in this category, some of the empirical studies are "empirically-evaluated" (6 out of 21), based on the evaluation of methods or tools without any experimental or real-life investigation; meanwhile, the others used qualitative research methods (e.g. survey and interview) and are considered "empirically-based" (15 out of 21) and are exploratory in nature (Smite, Wohlin, Gorschek, & Feldt, 2009). Out of the 15 "empirically-based" studies, we found that interviews and experiments were used as sub-methods in 40% (6 of 15) of the studies. In the other 63% of studies within the group of 15, the sub-methods used in case studies were not mentioned. For the purpose of this research, we use the term "sub-research method" when a research technique is applied as a part of research process, e.g. a case study may include interviews and surveys as data collection techniques. Interviews and surveys are considered sub-research methods. Other research methods used in empirically based studies include interviews (1) (with grounded theory as a sub-method for data analysis), surveys (1) (with interviews as a sub-method), ethnography (1), observation (1) and focus-group discussion (1). We have listed the research methods as reported by the authors of selected studies such as ethnography and observation separately. Moreover, in one of the studies, multiple cases were used to improve the generalisability of the results (e.g. Ramesh et al., 2010) considered 16 software development organisations to investigate the agile RE practices). Second, we found 18% of studies were empirically evaluated studies based on new ideas, i.e. tools/methods and their experimental evaluation. These include tools for modelling non-functional requirements (Farid & Mitropoulos, 2012a, 2012b), a mind-mapping technique to facilitate agile RE (Mahmud & Veneziano, 2011), a model for agile requirements prioritisation (Racheva, Daneva, & Herrmann, 2010), goal-sketching technique for agile RE (Boness & Harrison, 2007) and agile requirements modelling through para-consistent reasoning (Ernst, Borgida, Jureta, & Mylopoulos, 2013). These studies focus on overall agile RE (8 out of 21), on the difference between traditional and agile RE (1 out of 21), and on agile RE practices (6 out of 21).

4.2. (RQ1) What are the adopted practices of agile RE according to published empirical studies?

Below, we describe the 17 RE practices that we found to be adopted in agile software development. For each practice, we identify its potential respective challenges. It is important to note that the list below is inclusive, i.e. it reflects what we have collectively found in the 21 studies. Frequency of occurrences and the studies reporting each of the practices can be found in Table 6.

1. *Face-to-face communication* between team members and client representatives is a characteristic of agile RE. Agile processes advocate minimal documentation (Cao & Ramesh, 2008; Zhu, 2009) in the form of user stories and discourage long and complex specification documents. Frequent face-to-face communication helps the client steer the project in an unexpected direction according to his or her own understanding of the project. The frequent meetings lead to informal communication among

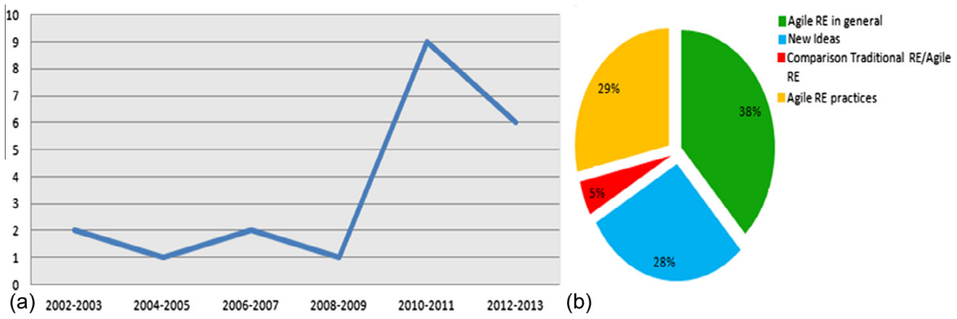

Fig. 2. (a) Year-wise distribution of selected studies. (b) Categorisation of study basis.

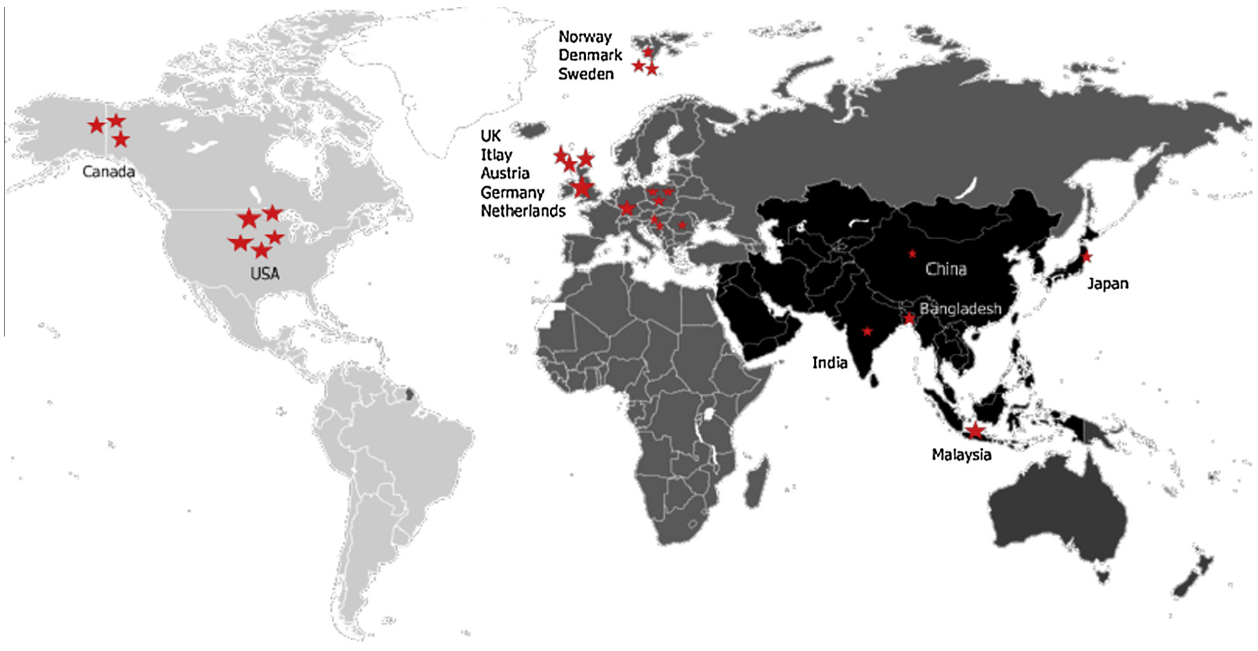

Fig. 3. Authorship geographic distribution of the selected studies.

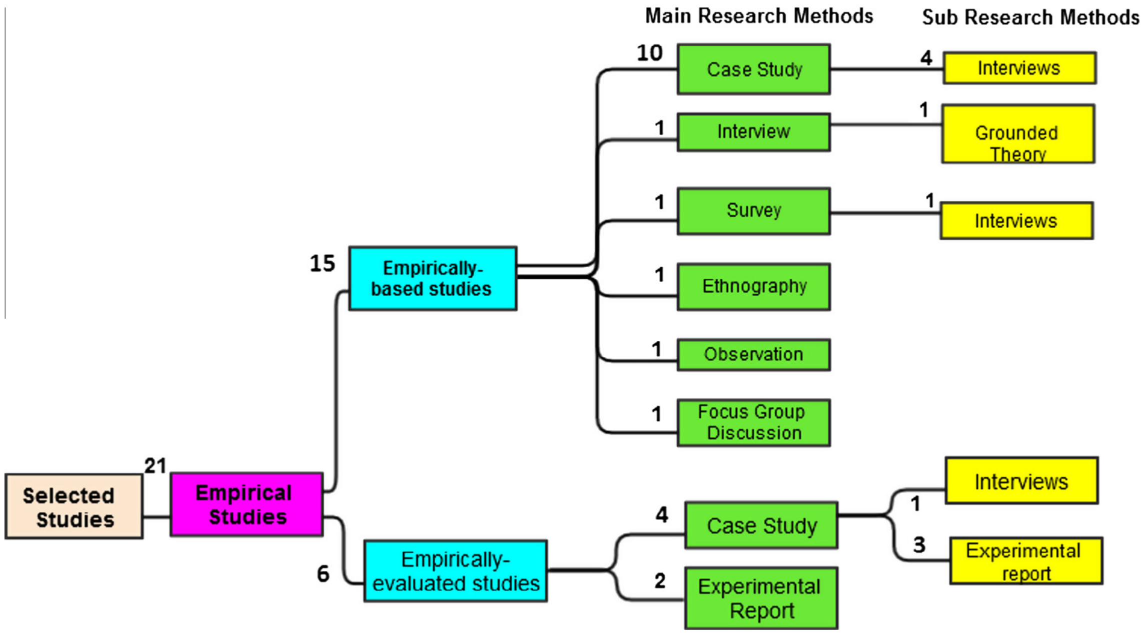

Fig. 4. Research-method-based distributions of research studies included.

stakeholders, which aids in the evolution of the requirements. The frequency of communication depends on the availability and willingness of team members. Customers may be accustomed to traditional methods and are unable to comprehend and trust agile methods.

2. *Customer involvement and interaction* were declared the primary reasons for project success and limited failure (Eberlein & Julio Cesar, 2002). There is an urgent need for identification of customers or representatives from business groups to ensure that requirements are appropriately defined, clarified and prioritised (Daneva et al., 2013). Failure to identify customer representatives can lead to disagreement and differing viewpoints on a variety of issues. Therefore, agile methods rely on frequent collaboration (e.g. face to face, Cao & Ramesh, 2008) with an accessible and available onsite customer (Beck et al., 2001).

3. *User stories* are created as specifications of the customer requirements. User stories facilitate communication and better overall understanding among stakeholders (Daneva et al., 2013). The user stories shift the concentration from written documentation to communication (Carlson & Matuzic, 2010). User stories are believed to be capable of eradicating the challenge of constant updating of requirements specification documents in traditional requirements engineering (Bjarnason, Wnuk, & Regnell, 2011a) by keeping team members updated. User stories emphasise “user goals”, briefly explain the user perception, focus on “what” is

Table 6
Summary of practices and the respective studies that have investigated them.

Practice	Freq.	Studies that reported the practice
1. Face-to-face communication	3	Cao and Ramesh (2008), Ramesh et al. (2010), Jun et al. (2010)
2. Customer involvement	3	Daneva et al. (2013), Cao and Ramesh (2008), Ramesh et al. (2010)
3. User stories	2	Paetsch, Eberlein, and Maurer (2003), Bjarnason et al. (2011a)
4. Iterative requirements	3	Cao and Ramesh (2008), Ramesh et al. (2010), Jun et al. (2010)
5. Requirements prioritisation	5	Cao and Ramesh (2008), Ramesh et al. (2010), Daneva et al. (2013), Jun et al. (2010), Racheva, Daneva, and Buglione (2008)
6. Change management	2	Cao and Ramesh (2008), Ramesh et al. (2010)
7. Cross-functional teams	1	Bjarnason et al. (2011a)
8. Prototyping	2	Cao and Ramesh (2008), Ramesh et al. (2010)
9. Testing before coding	4	Cao and Ramesh (2008), Ramesh et al. (2010), Jun et al. (2010), Haugset and Stalhane (2012)
10. Requirements modelling	2	Boness and Harrison (2007), Ernst et al. (2013)
11. Requirements management	2	Cao and Ramesh (2008), Ramesh et al. (2010)
12. Review meetings and acceptance tests	2	Cao and Ramesh (2008), Ramesh et al. (2010)
13. Code refactoring	1	Berry (2002)
14. Shared conceptualisations	1	Abdullah et al. (2011)
15. Pairing for requirements analysis	1	Yu and Sharp (2011)
16. Retrospectives	3	Cao and Ramesh (2008), Ramesh et al. (2010), Jun et al. (2010)
17. Continuous planning	1	Jun et al. (2010)

needed to be done, and support collaborative and iterative development (Carlson & Matuzic, 2010).

4. *Iterative requirements*, requirements unlike in traditional software development methods, emerge over time in agile methods (Ramesh et al., 2010). Frequent interaction among stakeholders leads to this iterative requirements approach. Such approach makes requirements clearer over time, strengthens relationships with customer and allows requirements to evolve with less investment of time (Cao & Ramesh, 2008). This gradual detailing of requirements (Bjarnason et al., 2011a) yields a finalised document that is closer to development, which makes it more subtle and less fragile.

5. *Requirement prioritisation* is part of each iteration in agile methods. In traditional RE, prioritisation is performed only once before development commences; in contrast, in agile methods, requirements are prioritised continuously in each development cycle by customers who focus on business value (Cao & Ramesh, 2008), or on risk (Daneva et al., 2013). The greatest challenge is that only focusing on business value can become problematic; moreover, allowing the customer to prioritise the requirements is not always right because there can be other factors to consider, for instance, the software architecture might not be scalable. Following business value, risk was defined as the most important requirements prioritisation criteria in the context of large outsourced agile projects (Daneva et al., 2013).

6. *Change management* has proven to be a significant challenge for traditional approaches thus far. For agile RE, its dynamic nature offers the greatest benefit. The main reported changes in requirements are to add or to drop features (Cao & Ramesh, 2008). Frequent face-to-face communication between development teams and clients preclude the need for changes in subsequent stages. The clarity gained by clients helps them to refine their requirements, which contributes to less rework and changes in subsequent stages. Rapid validation of prototypes by clients in informational sessions simplifies the management of subsequent changes.

7. *Cross-functional teams* include members from different functional groups who have similar goals. In agile methods, developers, testers, designers, and managers sit and work together. This concept helps to reduce challenges such as overscoping requirements and communication gaps (Bjarnason et al., 2011a). Team members intermingle with one another and share their knowledge, which enhances their level of trust and ultimately generates further communication.

8. *Prototyping* is perceived as a simple and straightforward way to review requirements specifications with clients and to gain timely feedback prior to moving to subsequent iterations. Prototyping starts with simple requirements that are completely understood and have high priority (De Lucia & Qusef, 2010). It promotes quicker feedback and enhances customer anticipation of the product. With prototyping, there are no wrong requirements; the only requirements are those waiting to be discovered (Tomayko, 2002). However, this seems to be problematic in the case of an exceptional increase in client requirements (Thayer & Dorfman, 1997). In addition, the development of multiple high-fidelity prototypes can be prohibitively expensive.

9. *Testing before coding* means to write tests prior to writing functional codes for requirements. It promotes feedback in the case of test failures. Another approach proposed in this domain is automated acceptance test-driven development (ATDD) (Haugset & Stalhane, 2012). It combines features of both agile RE practices and traditional RE, i.e. detailed documentation with iterative frequent communication.

10. *Requirements modelling* is performed in agile software development methods, but it is different from RE models developed in traditional software development methods. A technique used in modelling agile requirements is goal-sketching, which intends to provide intuitive and easy-to-read goal graphs for project managers, sponsors and team members (Boness & Harrison, 2007). The goals are refined constantly for each iteration, leaving no room for vague intention. This technique empowers decision-making while requirements negotiating process is carried out. Likewise, another technique is proposed for agile requirements modelling through para-consistent reasoning (Ernst et al., 2013).

11. *Requirements management* is performed by maintaining product backlog/feature lists and index cards (Cao & Ramesh, 2008; Ramesh et al., 2010). In the Scrum method, product backlog can be used to keep track of requirement changes.

12. *Review meetings and acceptance tests* are the developed requirements and product backlogs that are constantly reviewed in meetings (Carlson & Matuzic, 2010); they are a form of checks and balances of the user stories completed and still in hand. Similarly, acceptance tests are just like unit tests, resulting in binary results of "pass" or "fail" for a user story. These acceptance tests increase team, customer and domain expert collaboration as well as reduce the severity of defects and regressions.

13. *Code refactoring* is meant to revisit and modify developed code structure, to improve on the structure and accommodate

Table 7
Summary of challenges of traditional RE resolved by agile RE practices.

No.	Challenge	Practice	Description
1.	Communication issues (Bjarnason et al., 2011a; Carlson & Matuzic, 2010)	Frequent face-to-face meetings (Bang, 2007; Sillitti et al., 2005)	Agile RE promotes regular interaction with customer and among teams. It is the predominant method for eradicating communication gaps
		Collocated teams (Highsmith & Fowler, 2001)	Agile principles prefers collocated teams for better communication and collaboration
		Onsite customer (Cao & Ramesh, 2008; Lundh & Sandberg, 2002; Pichler et al., 2006)	Customer and development teams should be located in the same place to enhance informal communication, to enable timely feedback, to facilitate agreement, to develop ownership, and to create a sense of responsibility
		Alternate customer representations (Bjarnason et al., 2011a; Fraser, Mellon, Dunsmore, & Lundh, 2001; Hoda, Noble, & Marshall, 2011)	In industry, having business representatives to be present at the development site is expensive and impossible at times. RE alternatives such as proxy customers can serve the same purpose
		Cross-functional agile teams (Bjarnason et al., 2011a)	Cross-functional agile teams aid in the clarification and understanding of requirements
		Integrated RE process (Bjarnason et al., 2011a)	Locating RE process closer to development activities enhances developer's understanding and reduces communication lapses
2.	Overscoping (Bjarnason et al., 2011a)	One continuous scope flow (Bjarnason et al., 2011a)	In agile methods, developers receive a list of features that are constantly prioritised by the customer. Thus, the chance of having to repeat allocation in projects is reduced
		Gradual detailing (Bjarnason et al., 2011a)	Gradual detailing of requirements helps to reduce overscoping and contributes to a feasible scope
		Cross-functional teams (Bjarnason et al., 2011a)	The team can focus more on important features when sharing responsibilities and working closely together in a cross-functional structure
3.	Requirements validation (Carlson & Matuzic, 2010)	Requirements prioritisation (Racheva et al., 2010)	Customer continues with prioritisation requirements in every iteration; thus, less important requirements remain on hold
		Prototyping (Cao & Ramesh, 2008; Ramesh et al., 2010)	Prototyping helps in providing the customer with a blueprint of the product, and therefore helps in validating the requirements
4.	Requirements documentation (Bjarnason et al., 2011a)	User stories (Bjarnason et al., 2011a; Carlson & Matuzic, 2010)	User stories are precise and provide to-the-point explanation of user demands, prevent the need for maintaining long SRS documents as well as constant updating and traceability
		Face-to-face communication (Cao & Ramesh, 2008; Ramesh et al., 2010)	More face-to-face communication reduces ambiguities and the need for maintaining long documents
5.	Rare customer involvement (Carlson & Matuzic, 2010)	Requirements prioritisation by the customer (Racheva et al., 2010)	Prioritisation of the requirements for all iterations ensures, to a large extent that the customer goals will be met

Table 8
Summary of challenges of agile RE.

Challenge	Description	Impact	Solutions
Minimal documentation (Cao & Ramesh, 2008)	User stories and product backlogs are the only documents in agile methods (Zhu, 2009)	Traceability issues (Zhu, 2009)	
Customer availability (Ramesh et al., 2010)	Availability of customer for requirements negotiation, clarification and feedback	Increase in rework	Surrogate customers (Ramesh et al., 2010)
Inappropriate architecture (Ramesh et al., 2010)	Inadequate infrastructure can cause problems during later project stages	Increase in cost	Code refactoring (Berry, 2002)
Budget and time estimation (Cao & Ramesh, 2008)	Initial estimates of time and cost are changed substantially by a change in requirements in subsequent stages	Project delays	Frequent communication
		Over-budgeting	Accurate modelling of user story
Neglecting non-functional requirements (NRFs)	User stories only satisfy system/product features	System security, usability, performance at stake	NRF modelling approach (Farid & Mitropoulos, 2012b). The NORMATIC tool (Farid & Mitropoulos, 2012a)
Customer inability and agreement (Daneva et al., 2013; Ramesh et al., 2010)	Incomplete domain knowledge and in consensus among customer groups	Increase in rework	Creation of delivery stories to accompany user stories (Daneva et al., 2013)
		Increase in cost	Frequent communication Iterative RE (Ramesh et al., 2010)
Contractual limitations (Cao & Ramesh, 2008)	Fixed-price contracts do not allow changes	Increase in cost	
Requirements change and its evaluation	To find the consequences of requirements change	Increase in work delay	RE-KOMBINE framework (Ernst et al., 2013)

changes (Fowler, Beck, Brant, Opdyke, & Roberts, 2011). Code refactoring is a practice to accommodate changes in requirements and cater to requirements volatility in subsequent stages (Berry, 2002). Therefore, it is one of the flexible features offered by agile methods to handle dynamically changing requirements. Nevertheless, it generates code wastage, which increases cost by including wastage data warehouse management.

14. *Shared conceptualisations* is a supporting concept to carry out RE activities related to gathering, clarifying and evolving for agile methods (Abdullah, Honiden, Sharp, Nuseibeh, & Notkin, 2011). The concepts are built and stored in each individual's memory through communication and collaboration during RE activities. The co-located agile teams constantly rearticulate their shared conceptualisations during development, which helps in problem solving.

15. *Pairing for requirements analysis* is a practice that encourages the stakeholders to perform multiple roles as well (Yu & Sharp, 2011). A single stakeholder playing different roles can introduce efficient task sharing due to minimal communication delay. Pairing practice in requirements analysis is one of the ways to close communication gaps between agile teams.

16. *Retrospectives* are the meetings held after completion of an iteration (Carlson, Matuzic, & Simons, 2012). These meetings often review the work completed so far and determine future steps and rework. The customer proposes new requirements in the form of changes in the deliverables.

17. *Continuous planning* is a routine task for agile teams (Jun, Qiuzhen, & Lin, 2010). The team never sticks to fixed plans and adapts to the upcoming changes from customers as the project progresses. This flexibility facilitates changing requirements in later stages of projects.

4.3. (RQ2) What are the challenges of traditional RE that are resolved by agile RE?

Changes that agile methods promote in relation to traditional software development also impact requirements engineering activities to a certain extent. Many researchers have been keen to gain a better understanding of such impacts and how agile methods and RE are now related; the investigation of these researchers has focused on this interconnection. The agility and dynamic nature that allow changes based on constant feedback from stakeholders cause the emergence of requirements throughout the development process (Cao & Ramesh, 2008). Unlike the emergence of requirements in traditional software development, agile methods do not support phase-driven approaches; thus, changes are continual (Wolfgang, 2011). Agile methods seek to counter many challenges of traditional RE practices, which remain an area of interest to software practitioners and researchers. The challenges and their resolution approaches identified in our review are discussed below. Tables 7 and 8 summarise these challenges and highlight the studies that identified them.

1. *Communication gaps* occur when there is a lapse in providing required information to relevant people. It occurs due to gaps in roles over time and unclear vision of goals (Bjarnason, Wnuk, & Regnell, 2011b). Agile methods have the capability to deal with RE challenges such as communication lapse or gap and overscoping (Bjarnason et al., 2011a; Carlson & Matuzic, 2010). These challenges are overcome by frequent face-to-face communication (Sillitti, Ceschi, Russo, & Succi, 2005) among cross-functional teams through agile methods, gradual detailing of requirements and integrated RE processes (Bjarnason et al., 2011a).

2. *Overscoping* is defined as setting the scope of the project that is too large for the resources provided. It occurs due to communication gaps, changes after finalising project scope, quality issues, and several other reasons (Bjarnason et al., 2011b). The customer's intervention and interaction at each step of the process enables him or her to clearly recommend changes or to give approval for further progress. This process prevents doing work for unwanted and out-of-scope requirements (Bjarnason et al., 2011a). This helps to lessen the overscoping of project.

3. *Requirements validation* is one of the biggest challenges of traditional RE (Carlson & Matuzic, 2010). Agile methods deal with the validation of requirements through continuous prioritisation (Hasnain & Hall, 2009). At the end of every iteration, the user story is demonstrated to the product owner and customer representatives. In this way, changes are suggested and scrutinised to ensure user story satisfies what was intended.

4. *Requirements documentation* means agile methods rely on tacit knowledge and frequent face-to-face communication in place of documentation for explicitly supporting knowledge sharing and decisions (Cao & Ramesh, 2008). Unreliable and lengthy documentation of requirements specifications is another challenge of traditional requirements engineering (Bjarnason et al., 2011a; Jun et al., 2010). Furthermore, lengthy specification documents are difficult to compile and consume excessive amounts of time, which can cause delays. Agile methods discourage lengthy documentation and support face-to-face communication with constant feedback and reviews. Often, the only documentation involved in agile methods includes user stories, product backlogs, burn down charts, etc.

5. *Rare customer involvement* is where the customer involvement plays an important part in agile methods (Lundh & Sandberg, 2002). In traditional software development like the waterfall model, customers receive the product after development and testing (Carlson & Matuzic, 2010). Conversely, with agile methods, customers remain aware of the progress that is made throughout the development life cycle. As a result, they provide timely feedback and have the means of expressing their enthusiasm for the final product. Timely customer feedback reduces cost of rework and increases common ownership and agreement (Verner & Babar, 2004).

4.4. (RQ3) What are the practical challenges of agile RE?

Our literature review found that while agile RE practices help counter the challenges experienced in traditional RE, they also introduce several limitations for achieving adequate balance between agility and stability, and ensuring sufficient competence of cross-functional development teams (Bjarnason et al., 2011a). Likewise, the use of agile RE opens up several challenges for the software industry, e.g. lack of approaches to deal with non-functional requirements such as security and scalability (Ramesh et al., 2010), user stories prioritisation performed by customers based on business value, customer inability, and a lack of harmony among customers (Ramesh et al., 2010). The challenges of agile RE are described in detail below. Their respective solutions, also found in the literature, are listed in Table 6.

1. *Minimal documentation* is a vital challenge that agile methods pose to development teams (Cao & Ramesh, 2008; Ramesh et al., 2010). Agile methods replace the conventional requirements documentation with to-the-point, precise and "user goal" oriented user stories (Carlson & Matuzic, 2010). Whenever there is a communication lapse due to sudden changes in requirements, unavailability of appropriate client representatives, project complexity and the lack of documentation, multiple problems crop up. Moreover, if the requirements are supposed to be communicated to customers at distributed geographical locations and not collocated or onsite, it becomes cumbersome to tackle such a situation with little or no documentation (Goetz, 2002). In some cases, a team's workspace, i.e. room or office, is not large enough to accommodate all members; in that case, verbal communication of requirements is insufficient. This challenge is even bigger in large projects (for example, those investigated by Daneva et al. (2013). Organisations are actively searching for solution strategies to work around this challenge (see Table 6), e.g. user stories are complemented with more detailed artefacts (called delivery stories) that help developers make the right implementation choices in the coding stage of a sprint (Daneva et al., 2013).

2. *Customer availability* is assumed and advocated by agile methods. However, this assumption is often unrealistic as empirical studies confirmed customer availability and access to be overall a challenge (Pichler, Rumetshofer, & Wahler, 2006; Ramesh et al., 2010). While no study disputes that changes in requirements can be determined directly by the customer to accelerate the process, customer availability is deemed challenging and highly scarce because of many factors from a business perspective such as time,

cost and workload of customer representative (Racheva et al., 2010). In practice, most of the agile teams have surrogates or proxy customers to play the role of a real customer (for example, product owners (Daneva et al., 2013). Other organisations implement the practice of "onsite developer" by moving a developer representative to the customer site (Racheva et al., 2010).

3. *Budget and schedule estimation* is a challenge for organisations that follow agile methods. Although practising agile methods enables the initial valuations of a project, it is not possible to make upfront estimations due to volatile requirements, dynamic planning and design (Ramesh et al., 2010). The extent of a project is based on known user stories and some of these may be discarded in forthcoming iterations. Thus, the estimations are sometimes significantly affected (Cao & Ramesh, 2008; Ramesh et al., 2010).

4. *Inappropriate architecture* finalised by the team in earlier stages of the project becomes inadequate in later stages with new requirements (Ramesh et al., 2010). On the other hand, refactoring is an ongoing activity among agile teams, which keeps on changing the code. However, if refactoring were avoided and stopped, changes occurring during later stages would add to the cost and become cumbersome to deal with.

5. *Neglecting non-functional requirements* is where non-functional requirements (NRFs) focus on system quality, including its internal quality, i.e. maintainability, testability and external quality usability, and security (Cardinal, 2011). It is considered as a major challenge for agile methods (Ramesh et al., 2010) and can be the reason for massive lapse and rework. Farid and Mitropoulos (2012a) have proposed novel and slight artefacts such as agile use cases, agile loose cases and agile choose cases for NRF modelling by using NFRs Modelling for agile processes (Farid & Mitropoulos, 2012b).

6. *Customer inability and agreement* are the two main issues apart from availability according to Daneva et al. (2013). Customer inability refers to incompetence of customer in terms of decision-making and complete domain knowledge, and customer agreement is about consensus of more than one customer group involved in a project (Daneva et al., 2013; Ramesh et al., 2010). The disagreement between customer groups affects the performance, especially in short development cycles (Ramesh et al., 2010).

7. *Contractual limitations and requirements volatility* are important by not allowing changes in requirements after the signing of contract; the changes can cause an increase in cost and sometimes failure of projects. Therefore, legal measures should be taken to avoid such a situation and appropriately handle the flexible nature of agile RE. Nevertheless, these issues can be resolved by fixed payment per release, which protects investments and prevents volatility of requirements. Fixed payment contracts are also likely to reduce project-specific risks at vendor level in agile-based outsourced projects (Daneva et al., 2013). Similarly, the removal of misleading and incorrect requirements based on changes requires extra cost and effort to manage waste. This situation can be handled by increasing communication and involving customers to prevent subsequent rework.

8. *Requirements change and change evaluation* is an important aspect of agile methods. The flexible nature of agile methods welcomes changes, but it can create trouble when evaluating the consequences of these changes. Recently, a framework named RE-KOMBINE has been proposed to deal with para-consistent requirements specification (Ernst et al., 2013). This approach allows the requirements to be formally specified, yet flexible enough to accommodate changes. In addition, there are agile RE specific tools as JIRA (JIRA, 2013), which is recommended for usage in challenging projects (Sarkan, Ahmad, & Bakar, 2011).

5. Discussion of the results

An important aspect highlighted in the analysis of our 21 selected studies is the geographic locations of authors. It is observed that nearly 1/3 of all contributions were from North American countries (authors based in the US and Canada). This is unsurprising, considering the fact that the Agile Manifesto was created by North-American software development practitioners. European countries took second place in the contributions, involving the UK, Italy, the Netherlands, Belgium, Germany, Austria and Denmark. Meanwhile, Asian countries produced the least number of studies regarding agile RE. The only contributions from South Asian countries were those from Bangladesh, China, India, Malaysia and one Far-East Asian country, Japan. The uneven distribution of authors across geographic regions means that the empirical evidence reported by the 21 studies could not be considered generalisable. As most of the evidence is provided through empirical research in organisations in North America and Western Europe, it is hard to predict the similarity in results if agile RE is practised in Asian or South American organisations. There are definitely differences in organisational culture, country-specific culture and social norms across organisations globally (e.g. North American in contrast to Asian, Al-Ani et al., 2013). We therefore expect the findings of our review to vary with the possible findings reported in empirical studies on Asian companies. Based on our findings, it can be concluded that many organisations in Asia are still in the process of adopting agile methods and are in maturation. Therefore, it is recommended that researchers conduct more empirical studies on Asian organisations to report the findings from different parts of the world, where the culture in each location varies. In addition, this opens up an avenue to conduct comparative studies on the agile RE implementations across the continents.

We were surprised that we could not find any studies from South American and African countries. In view of the number of agile companies in certain South-American countries, e.g. Brazil, we would expect studies from these countries too. In the process of preparing this paper, we checked local venues such as the Annual Brazilian Symposium on Software Engineering but we did not find any study on agile RE. We deduced that local conferences often publish papers in their local languages (e.g. Portuguese for Brazil, Spanish for Argentina, etc.) and therefore because they are not in English, they were not considered in our study. Another hypothesis is that smaller local conferences are often not indexed in large databases like the ones we investigated, and as such, these papers might not be located. One other possible reason is that there are no researchers investigating agile RE in these countries.

We found that most of the studies reported in this review have used a case study-based research approach. This establishes the fact that researchers acknowledge agile methods as a social process and thus investigate it in its real-life context (e.g. through case studies) and not reproducing it in classrooms. The case studies speak of real-world examples with some of them having discussed multiple cases of agile methods in software development organisations. This helps increase the generalisability of the results and scale up to bigger settings (Wieringa, 2014). In 7 cases of study-based studies, the sub-research method was not mentioned explicitly. This refers to the problem of clarity in presentation of the studies. 75% of the studies were exploratory in nature based on empirical investigations, newly proposed ideas and literature review papers. This finding reveals the fact that most of the efforts are focused on understanding and identifying agile RE, defining agile RE practices, issues, and challenges. Examples of exploratory studies include improving our understanding of requirements definition, requirements prioritisation, and dealing with non-functional requirements in agile methods. These actions pave ways for the research community to use these practices and report empirical results in diverse settings. We also found five studies in which new methods and ideas are proposed to handle agile RE, i.e. studies by Boness and Harrison (2007), Racheva et al. (2010), Mahmud and Veneziano (2011), Farid and Mitropoulos (2012a, 2012b), and Ernst et al. (2013). However, there is a need to implement these ideas in real-world scenarios and gather empirical evidence to improve on them.

Furthermore, we have identified the current practices of agile RE in response to RQ1, which ensure the effectiveness of agile ways of dealing with requirements. However, we noticed that we could not always trace our findings to particular project contexts, for example large and very large projects versus small projects, or projects in specific industry sectors, e.g. government, healthcare. Hence, we feel that more research is needed to define specific practices for specific industry-based scenarios. An increased number of empirical studies are required to implement these practices and contribute findings to the existing body of knowledge.

We have identified the challenges of traditional RE resolved by agile RE practices in response to RQ2. The traditional RE has several challenges such as communication gaps, overscoping, requirements validation, documentation and customer participation. The agile practices introduced to resolve these challenges include face-to-face communication for reducing documentation and communication gaps, gradual detailing of requirements for reducing overscoping, requirements prioritisation by customer based on business value to deal with requirements validation, and close team and customer interaction to avoid the lack of customer participation. However, while matching the list of 17 practices that answered RQ1 with the list of challenges that agile RE helped to solve (the answer to RQ2), a few discrepancies were noted: In answering RQ1, we mentioned change management as a practice. However, in answering RQ2, no study proves the use of agile RE practices to have successfully solved the problem of requirements change (a challenge in traditional RE). This is clearly an inconsistency and researchers need to further investigate it by means of fieldwork and explain how exactly agile RE overcomes the change management challenge. In our opinion, a larger number of empirical results are needed to provide evidence that agile RE practices resolve traditional RE challenges. Likewise, another important aspect of agile methods is self-organising teams (Carlson et al., 2012), which has also been overlooked in agile RE studies. A self-organising team distributes workload among the team members and takes part in decision-making. Having a self-organising team is an important aspect of agile project organisation and warrants attention from researchers.

Overall, it is obvious that agile RE has resolved various challenges of traditional RE. Customer interaction, team collaboration and change acceptability are some of the features that make agile methods feasible. The interaction principle of agile RE helps to reduce communication lapses. It aids teams in developing mutual trust and rapport, which simplifies knowledge sharing. The cross-functional agile teams help to reduce the overscoping and over-allocation phenomena by integrating the RE process with development tasks and fostering close interactions among teams. Moreover, agile methods work well for requirements validation through customer feedback and reduce the overhead required to maintain lengthy requirements documents. In traditional RE, writing requirements specifications is a cumbersome and protracted process; however, maintenance of documentation is also equally burdensome. Agile methods discourage excessive documentation and utilise only user stories and product backlogs as isolated documents. Requirements validation is an important challenge of traditional requirements engineering, which is resolved by continuous requirements prioritisation in agile RE. However, other methods proposed for requirements validation in agile RE include continuous user story demonstration sessions (Hasnain & Hall, 2009). Therefore, more research should be conducted in this direction to cover such intricate issues.

We have also identified seven challenges of agile RE practices in response to RQ3 and identified practices to resolve them. The challenges that agile RE poses to project organisations include minimal documentation, budget and schedule estimation, inappropriate architecture, neglect of non-functional requirements, waste management, customer unavailability and contractual issues. Our review found that to tackle budget and time estimation constraint, frequent communication and story prioritisation are deemed suitable. To deal with NRFs, once believed to be undiscovered for agile methods, several methods are proposed like testing of functional requirements (e.g. feature acceptance tests and system acceptance tests). At the same time, exploratory tests, scenario tests, and system quality tests (performance tests, load testing) have also been proved suitable (Leffingwell, 2010). In addition, independent inspective testing throughout the life cycle ensures that the system appropriately addresses the NFRs (Ambler, 2008a, 2008b; Leffingwell, 2010) offer solutions to the problem related to NFR; however there is no existing empirical study that explored the use of these solutions in agile RE and their effectiveness. Similarly, to handle the problem of minimum documentation, an approach presented by Rubin and Rubin (2010) employs active domain documentation. However, no solution has been suggested in the selected studies. Meanwhile, to solve the problem of fixed contracts, some of the solutions should be considered with reference to business value in order to save capital and time. This opens up an opportunity for more research in the dimension of using inspective testing while dealing with NRFs, minimal documentation and fixed contractual issues in agile methods. Further empirical evidence needs to be collected from real-world cases over an extended period of time to assess the usefulness of these practices in overcoming these challenges. Furthermore, the methods and tools proposed to deal with NRFs also open the doors for tool vendors to design new features for agile RE tools.

6. Conclusions

This paper presents a systematic review of literature on practices and challenges of agile RE. This review was conducted by following available guidelines for conducting systematic literature reviews (Kitchenham & Charters, 2007) to search and categorise all existing and available literature on agile RE. Of the 543 initial papers located in well-known electronic research databases, 21 relevant papers were extracted through a multistage sifting process with independent validation in each step. These papers were then assessed for the quality of the evidence they produced and further analysed and categorised into the following thematic groups based on the research questions: (i) commonly used practices of agile RE; (ii) challenges of traditional requirements engineering resolved by agile RE; and (iii) practical challenges of agile RE. The findings in our research provides future dimensions to industry and research practitioners for further work on agile RE.

We have determined that there is a need for further research on agile RE and its real-world impact and applications. The promising features of agile RE such as lesser documentation, quick feedback, and prototyping may help organisations in the industry to benefit from this flexible yet time-constrained approach.

Our review of agile RE studies shows that this field is still immature and needs further empirical evaluation of practices in industrial cases. We have discovered that a large number of empirical studies focus on overall management of agile software development methods and very few focus particularly on agile RE practices. The review shows that agile RE practices can compensate for some of the shortcomings of traditional RE methods. The agile RE practices like customer involvement, change in requirements management, cross-functional teams, review meetings and sessions, are the distinct features missing in the traditional way of dealing with requirements; thus, they can outperform traditional RE practices. Hence, the usage of these practices in projects can remove the impediments of traditional RE and improve the quality and success rate of outcomes. Therefore, it can be concluded that traditional RE practices can be combined with agile methods for better performance on the same continuum. The review shows that most of the studies were exploratory and conducted to provide more knowledge of the topic. The number of studies with empirical evidences from the real-world industry cases is scarce and needs attention. There are five studies focusing on new idea/method proposals. This calls for more infrastructures in terms of frameworks and models to incorporate agile RE practices in global software engineering, open source software development and outsourcing domains. Furthermore, the study of overall use of RE tools designed for agile methods in large-sized industrial projects is also a promising dimension. We have addressed the benefits and limitations of adopting agile RE as well as their solution practices, which can help motivate practitioners concerning the use of agile methods for handling requirements.

6.1. Implications of the study

This review has several implications for both researchers and practitioners. In terms of research, the review shows that there is a need for more empirical studies that incorporate agile RE approaches using various variants of agile methods, e.g. Scrum, XP, lean, and Kanban. With the advancements in GSD, the examination of the performance of agile RE in outsourced projects with distributed teams and customers remains challenging. Thus, there is a need to implement agile RE in large distributed and outsourced projects to gain more insights. At present, there are only a few studies available on distributed outsourced projects using agile methods like Daneva et al. (2013). Hence, it is a potential domain for more empirical investigation of outsourced distributed projects using agile methods to gain deeper understanding of the phenomena.

The practical examples of agile RE applied to real-world projects remain scarce. Empirical studies of agile method variants such as test-driven development, feature-driven development, lean and Kanban are needed. Currently, there are only a few studies that discuss XP and Scrum methods for studying agile RE in real-world cases. Nonetheless, further research on other aspects of management-oriented methods, such as Scrum, which is considered the most under-researched method (Dybå & Dingsøyr, 2008) and newly emerged methods, i.e. lean and Kanban, are necessary to assess their response to the agile way of managing requirements.

The advent of the agile methods has sparked interest in the software industry for agile RE, which has fallen behind in the past few years. The compensations and flexibility of the agile methods are also expected from the agile way of managing requirements. Thus, this should prompt the industry to adopt agile RE practices for even distributed large-scale projects. Active research can only be possible with industry participation. Thus, the industry should participate in research related to agile RE and target research goals that are highly related and beneficial for the future of the software industry. The empirical results generated from communication patterns and collaboration studies among agile teams should be utilised by project managers to orient their teams towards the implementation of certain collaboration structures, i.e. hierarchal or centralised among them for maximum performance.

Moreover, present research models and frameworks developed for managing requirements in an agile way should be implemented and practically used to prove their efficiency and performance. The models and techniques should be developed for commercial off-the-shelf tools to aid the requirements engineering process with regard to the agile method. Supporting tools for flexible porting should be introduced to facilitate and simplify the use of these tools.

6.2. Limitations of our study

The basic limitations of any systematic review are the bias in selection of studies and the possible imprecision in data extraction from the variable sources. To eliminate this bias and ensure precision and accuracy in study selection, we implemented the following steps when developing our research strategy. First, we treated the search-string-building process as a learning process that included experimentation. We subsequently followed our research questions to define keywords for extensive search in electronic databases. As Dybå and Dingsøyr (2008) indicate, search strings in software engineering are language dependent; thus, there is a possibility of missing relevant studies during each search. In addition, the alternative terms used for requirements in agile methods like user stories, features and tasks are not considered in the search. These terms might lead to discovery of several other studies as well.

Next, our study encompassed only articles that primarily focused on the agile method of addressing requirements and not on the agile studies, which include requirements as a small part of it. In that context, it might have been possible to include more data and draw more conclusions. To reduce bias due to personal preferences in study selection or missing out any detail, we used a multistage process, and the studies were examined by two researchers for their relevance based on inclusion and exclusion criteria (Kitchenham & Charters, 2007) (defined in Section 3.1.4). Then, the corresponding *Kappa Coefficient* was calculated for both the researchers' results; the kappa coefficient was 0.69, which indicates good agreement. In addition, we found that many articles lacked sufficient information for inclusion in our study. More specifically, we found that the level of detail at which the research method was described in the 21 papers, varied widely across the studies. For example, some studies included more complete evaluation of validity threats than others. These variations across the studies lead to the possibility that the data extraction process may have resulted in inaccuracies.

Furthermore, we are aware that for any of the papers that described challenges (be it in agile RE or in traditional RE), the underlying causes of the challenges reported might not have been discussed in detail. It is inherently hard to change this situation because the researchers who authored the respective papers felt it was unnecessary to find the cause of the challenge. We also assume that authors of some studies might have chosen the challenges reported for specific reasons that are tacit and not explicitly stated in the papers. For example, a RE challenge might well has been very important to the organisation where the authors were executing their case study. However, the authors were focused on their specific research goals and questions, and the underlying reasons for the challenge itself might have been overlooked.

Acknowledgement

This research is supported by UM High Impact Research Grant UM.C/625/1/HIR/MOE/FCSIT/05 from the Ministry of Higher Education, Malaysia. We are also thankful to the Brightsparks Unit, University of Malaya for their support in this research.

Appendix A. List of reviewed studies

1. Abdullah, N. N. B., Honiden, S., Sharp, H., Nuseibeh, B., & Notkin, D. (2011). Communication Patterns of Agile Requirements Engineering. *Proceedings of the 1st Workshop on Agile Requirements Engineering* (pp. 1–4). ACM. http://dx.doi.org/10.1145/2068783.2068784.
2. Berry, D. M. (2002). The Inevitable Pain of Software Development, Including of Extreme Programming, Caused by Requirements Volatility University of Waterloo. *Proceedings of the International Workshop on Time Constrained Requirements Engineering* (pp. 1–11).
3. Bjarnason, E., Wnuk, K., & Regnell, B. (2011). A Case Study on Benefits and Side-Effects of Agile Practices in Large-Scale Requirements Engineering. *Agile RE 2011*. Lancaster, UK: ACM.
4. Boness, K., & Harrison, R. (2007). Goal Sketching: Towards Agile Requirements Engineering. *International Conference on Software Engineering Advances* (pp. 1–6).
5. Cao, L. C. L., & Ramesh, B. (2008). Agile Requirements Engineering Practices: An Empirical Study. *IEEE Software*, *25*(1), 60–67. http://dx.doi.org/10.1109/MS.2008.1.
6. Carlson, D., & Matuzic, P. (2010). Practical Agile Requirements Engineering. *13th Annual Systems Engineering Conference*. San Diego, CA.
7. Daneva, M., Van der Veen, E., Amrit, C., Ghaisas, S., Sikkel, K., Kumar, R., Ajmeri, N., et al. (2013). Agile requirements prioritization in large-scale outsourced system projects: An empirical study. *Journal of Systems and Software*, *86*(5), 1333–1353. http://dx.doi.org/10.1016/j.jss.2012.12.046.
8. Ernst, N. a., Borgida, A., Jureta, I. J., & Mylopoulos, J. (2013). Agile requirements engineering via paraconsistent reasoning. *Information Systems*, 1–17. http://dx.doi.org/10.1016/j.is.2013.05.008.
9. Farid, W. M., & Mitropoulos, F. J. (2012a). NORMATIC: A visual tool for modeling Non-Functional Requirements in agile processes. *2012 Proceedings of IEEE Southeastcon*, (978), 1–8. http://dx.doi.org/10.1109/SECon.2012.6196989.
10. Farid, W. M., & Mitropoulos, F. J. (2012b). Novel lightweight engineering artifacts for modeling non-functional requirements in agile processes. *2012 Proceedings of IEEE Southeastcon*, 1–7. http://dx.doi.org/10.1109/SECon.2012.6196988.
11. Haugset, B., & Stalhane, T. (2012). Automated Acceptance Testing as an Agile Requirements Engineering Practice. *45th Hawaii International Conference on System Sciences* (pp. 5289–5298). Hawaii, US: Ieee. http://dx.doi.org/10.1109/HICSS.2012.127.
12. Jun, L., Qiuzhen, W., & Lin, G. (2010). Application of Agile Requirement Engineering in Modest-Sized Information Systems Development. *2010 Second World Congress on Software Engineering* (pp. 207–210). Ieee. http://dx.doi.org/10.1109/WCSE.2010.105.
13. Lundh, E., & Sandberg, M. (2002). Time Constrained Requirements Engineering with Extreme Programming – An Experience Report. *Proceedings of the International Workshop on Time Constrained Requirements Engineering.*
14. Mahmud, I., & Veneziano, V. (2011). Mind-mapping: An effective technique to facilitate requirements engineering in agile software development. *Proceedings of 14th International Conference on Computer and Information Technology (ICCIT 2011) 22–24 December, 2011, Dhaka, Bangladesh* (pp. 22–24). Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6164775.
15. Martakis, A., & Daneva, M. (2013). Handling Requirements Dependencies in Agile Projects: A Focus Group with Agile Software Development Practitioners. *IEEE Seventh International Conference on Research Challenges in Information Science (RCIS)* (pp. 1–11). Paris: Ieee. http://dx.doi.org/10.1109/RCIS.2013.6577679.
16. Pichler, M., Rumetshofer, H., & Wahler, W. (2006). Agile Requirements Engineering for a Social Insurance for Occupational Risks Organization: A Case Study. *14th IEEE International Requirements Engineering Conference (RE'06)*, 251–256. http://dx.doi.org/10.1109/RE.2006.8.
17. Racheva, Z., Daneva, M., & Herrmann, A. (2010). A Conceptual Model of Client-driven Agile Requirements Prioritization: Results of a Case Study. *IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)* (pp. 1–4). Bolzano-Bozen, Italy.
18. Ramesh, B., Baskerville, R., & Cao, L. (2010). Agile requirements engineering practices and challenges: an empirical study. *Information Systems Journal*, *20*(5), 449–480. http://dx.doi.org/10.1111/j.1365-2575.2007.00259.x.
19. Sarkan, H. M., Ahmad, T. P. S., & Bakar, A. A. (2011). Using JIRA and Redmine in requirement development for agile methodology. *2011 Malaysian Conference in Software Engineering*, 408–413. http://dx.doi.org/10.1109/MySEC.2011.6140707.
20. Sillitti, A., Ceschi, M., Russo, B., & Succi, G. (2005). Managing Uncertainty in Requirements: a Survey in Documentation-driven and Agile Companies. *11th IEEE International Software Metrics Symposium (METRICS 2005).*
21. Yu, Y., & Sharp, H. (2011). Analysing requirements in a case study of pairing. *Workshop on Agile Requirements Engineering, July 2011*. Lancaster, UK. Retrieved from http://dl.acm.org/citation.cfm?id=2068787.

References

Abdullah, N. N. B., Honiden, S., Sharp, H., Nuseibeh, B., & Notkin, D. (2011). Communication patterns of agile requirements engineering. In *Proceedings of the 1st workshop on agile requirements engineering* (pp. 1–4).

Al-Ani, B., Bietz, M. J., Wang, Y., Koehne, B., Marczak, S., Redmiles, D., et al. (2013). Globally distributed system developers: Their trust expectations and processes. In *Computer supported cooperative work (CSCW)* (pp. 563–573).

Ambler, S. W. (2008a). Beyond functional requirements on agile projects. *Dr. Dobb's*. <http://www.drdobbs.com/architecture-and-design/beyond-functional-requirements-on-agile/210601918?pgno=4> Accessed 01.11.13.

Ambler, S. W. (2008b). Beyond functional requirements on agile projects. *Dr. Dobb's*.

Bang, T. J. (2007). An agile approach to requirement specification. In G. Concas, E. Damiani, M. Scotto, & G. Succi (Eds.), *XP 2007, LNCS 4536* (pp. 193–197). Berlin, Heidelberg: Springer-Verlag.

Barlow, J. B., Giboney, J. S., Keith, M. J., Wilson, D. W., Schuetzler, R. M., Lowry, P. B., et al. (2011). Overview and guidance on agile development in large organizations. *Communications of the Association for Information Systems, 29*(July), 25–44.

Beck, K. (1999). *Extreme programming explained: Embrace change*. Addison-Wesley Professional.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., et al. (2001). *Manifesto for agile software development*. <http://agilemanifesto.org/>.

Berry, D. M. (2002). The inevitable pain of software development, including of extreme programming, caused by requirements volatility University of Waterloo. In *Proceedings of the international workshop on time constrained requirements engineering* (pp. 1–11).

Bin, X., Xiaohu, Y., Zhijun, H., & Maddineni, S. R. (2004). Extreme programming in reducing the rework of requirements change. In *Canadian conference on electrical and computer engineering* (pp. 1567–1570).

Bjarnason, E., Wnuk, K., & Regnell, B. (2011a). A case study on benefits and side-effects of agile practices in large-scale requirements engineering. In *Agile RE*.

Bjarnason, E., Wnuk, K., & Regnell, B. (2011b). Requirements are slipping through the gaps—A case study on causes & effects of communication gaps in large-scale software development. In *2011 IEEE 19th international requirements engineering conference* (pp. 37–46).

Boness, K., & Harrison, R. (2007). Goal sketching: Towards agile requirements engineering. In *International conference on software engineering advances* (pp. 1–6).

Cao, L. C. L., & Ramesh, B. (2008). Agile requirements engineering practices: An empirical study. *IEEE Software, 25*(1), 60–67.

Cardinal, M. (2011). *Addressing non-functional requirements with agile practices*. <agile101.net>.

Carlson, D., & Matuzic, P. (2010). Practical agile requirements engineering. In *13th Annual systems engineering conference*.

Carlson, R., Matuzic, P. J., & Simons, R. L. (2012). Applying scrum to stabilize systems engineering execution. *CrossTalk*, June (pp. 1–6).

Daneva, M., van der Veen, E., Amrit, C., Ghaisas, S., Sikkel, K., Kumar, R., et al. (2013). Agile requirements prioritization in large-scale outsourced system projects: An empirical study. *Journal of Systems and Software, 86*(5), 1333–1353.

De Lucia, A., & Qusef, A. (2010). Requirements engineering in agile software development. *Journal of Emerging Technologies in Web Intelligence, 2*(3), 308–313.

Dybå, T., & Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. *Information and Software Technology, 50*(9–10), 833–859.

Eberlein, A., & Julio Cesar, S. do P. L. (2002). Agile requirements definition: A view from requirements engineering. In *Proceedings of the international workshop on time constrained requirements engineering*.

Elo, S., & Kyngäs, H. (2007). The qualitative content analysis process. *Journal of Advanced Nursing, 62*(1), 107–115.

Ernst, N. a., Borgida, A., Jureta, I. J., & Mylopoulos, J. (2013). Agile requirements engineering via paraconsistent reasoning. *Information Systems* (June), 1–17.

Farid, W. M., & Mitropoulos, F. J. (2012a). NORMATIC: A visual tool for modeling non-functional requirements in agile processes. In *2012 Proc. IEEE Southeastcon*, no. 978, March 2012 (pp. 1–8).

Farid, W. M., & Mitropoulos, F. J. (2012b). Novel lightweight engineering artifacts for modeling non-functional requirements in agile processes. In *2012 Proc. IEEE Southeastcon*, March 2012 (pp. 1–7).

Fleiss, J. L., Levin, B., & Paik, M. C. (2003). *Statistical methods for rates and proportions* (pp. 598–626). John Wiley & Sons, Inc.

Fowler, M., Beck, K., Brant, J., Opdyke, W., & Roberts, D. (2011). *Refactoring: Improving the design of existing code*. Addison Wesley.

Fraser, S., Mellon, B., Dunsmore, K., & Lundh, E. (2001). XP requirement negotiation workshop. In *XP requirement negotiation workshop co-located with XP2001 – The second international conference on eXtreme programming and agile processes in software engineering*, Villasimius, Cagliari, Italy, 23rd May 2001 (pp. 26–31).

Gandomani, T. J., Zulzalil, H., Ghani, A. A. A., & Sultan, A. B. M. (2013). *A systematic literature review on relationship between agile methods and open source software development methodology*. CoRR.

Goetz, R. (2002). How agile processes can help in time-constrained requirements engineering. In *Proceedings of the international workshop on time constrained requirements engineering*.

Guyatt, G., Rennie, D., Meade, M., & Cook, D. (2008). *Users' guide to the medical literature essentials of evidence-based clinical practice* (2nd ed., Vol. 270(21)). Mc Graw Hill.

Hossain, E., Babar, M. A., & Paik, H. (2009). Using scrum in global software development: A systematic literature review. In *4th international conference on global software engineering* (pp. 175–184). IEEE, http://dx.doi.org/10.1109/ICGSE.2009.25.

Hasnain, E. (2010). An overview of published agile studies: A systematic literature review. In *Proceedings of the national software engineering conference* (pp. 1–6).

Hasnain, E., & Hall, T. (2009). Introduction to stand-up meetings in agile methods. *Transactions on Engineering Technologies, II*, 110–121.

Haugset, B., & Stalhane, T. (2012). Automated acceptance testing as an agile requirements engineering practice. In *45th Hawaii international conference on system sciences* (pp. 5289–5298).

Highsmith, J., & Fowler, M. (2001). The Agile Manifesto. *Software Development, 9*(8), 29–30.

Hoda, R., Noble, J., & Marshall, S. (2011). The impact of inadequate customer collaboration on self-organizing agile teams. *Information and Software Technology, 53*(5), 521–534.

Hossain, E., Babar, M. A., & Verner, J. (2009). How can agile practices minimize global software development co-ordination risks? In R. V. O'Connor, N. Baddoo, J. C. Gallego, R. R. Muslera, K. Smolander, & R. Messnarz (Eds.). *Software process improvement communications in computer and information science* (Vol. 42, pp. 81–92). Berlin, Heidelberg: Springer.

Hsieh, H.-F., & Shannon, S. E. (2005). Three approaches to qualitative content analysis. *Qualitative Health Research, 15*(9), 1277–1288.

Jalali, S., & Wohlin, C. (2011). Global software engineering and agile practices: A systematic review. *Journal of Software: Evolution and Process, 24*(6), 643–659.

JIRA (2013). *JIRA issue and project*. Atlassian Pvt Ltd.

Jun, L., Qiuzhen, W., & Lin, G. (2010). Application of agile requirement engineering in modest-sized information systems development. *In 2010 Second world congress on software engineering*, no. 1 (pp. 207–210).

Kitchenham, B., & Charters, S. (2007). *Guidelines for performing systematic literature reviews in software engineering*. UK: Keele.

Kotonya, G., & Sommerville, I. (1997). *Requirements engineering*. John Wiley & Sons.

Lagerberg, L., & Skude, T. (2013). *The impact of agile principles and practices on large-scale software development projects*. Linköpings Universitet.

Landis, J. R., & Kosh, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics, 33*(1), 159–174.

Leffingwell, D. (2010). *Agile software requirements lean requirements practices for teams, programs, and the enterprise* (p. 560). Boston, MA: Addison-Wesley Professional.

Lundh, E., & Sandberg, M. (2002). Time constrained requirements engineering with extreme programming – An experience report. In *Proceedings of the international workshop on time constrained requirements engineering*.

Mahmud, I., & Veneziano, V. (2011). Mind-mapping: An effective technique to facilitate requirements engineering in agile software development. In *Proceedings of 14th international conference on computer and information technology (ICCIT 2011)*, 22–24 December, 2011, Dhaka, Bangladesh, Iccit (pp. 22–24).

Pacheco, C., & Garcia, I. (2012). A systematic literature review of stakeholder identification methods in requirements elicitation. *Journal of Systems and Software, 85*(9), 2171–2181.

Paetsch, F., Eberlein, A., & Maurer, F. (2003). Requirements engineering and agile software development. In *Proceedings of twelfth IEEE international workshops on enabling technologies: Infrastructure for collaborative enterprises (WETICE)* (pp. 308–313).

Pichler, M., Rumetshofer, H., & Wahler, W. (2006). Agile requirements engineering for a social insurance for occupational risks organization: A case study. In *14th IEEE int. requir. eng. conf.*, September 2006 (pp. 251–256).

Racheva, Z., Daneva, M., & Buglione, L. (2008). Supporting the dynamic reprioritization of requirements in agile development of software products. In *Second international workshop on software product management (ISWPM'08)*.

Racheva, Z., Daneva, M., & Herrmann, A. (2010). A conceptual model of client-driven agile requirements prioritization: Results of a case study. In *IEEE international symposium on empirical software engineering and measurement (ESEM)* (pp. 1–4).

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

Rizvi, B. (2013). *A systematic review of distributed agile software engineering*. Alberta: Athabasca University.

Rubin, E., & Rubin, H. (2010). Supporting agile software development through active documentation. *Requirements Engineering, 16*(2), 117–132.

Sarkan, H. M., Ahmad, T. P. S., & Bakar, A. A. (2011). Using JIRA and Redmine in requirement development for agile methodology. In *Malaysian conf. softw. eng.*, December 2011 (pp. 408–413).

Schwaber, K., & Beedle, M. (2001). *Agile software development with scrum*. New York, United States: Prentice Hall.

Sillitti, A., Ceschi, M., Russo, B., & Succi, G. (2005). Managing uncertainty in requirements: A survey in documentation-driven and agile companies. In *11th IEEE international software metrics symposium (METRICS 2005)*, Metrics.

Silva da Silva, T., Martin, A., Maurer, F., & Silveira, M. (2011). User-centered design and agile methods: A systematic review. In *Agil. conf.*, August 2011 (pp. 77–86).

Smite, D., Wohlin, C., Gorschek, T., & Feldt, R. (2009). Empirical evidence in global software engineering: A systematic review. *Empirical Software Engineering, 15*(1), 91–118.

Thayer, R., & Dorfman, M. (1997). *Software requirements engineering*. Los Alamitos, CA: IEEE Computer Society Press.

Tomayko, J. E. (2002). Engineering of unstable requirements using agile methods. In *Proceedings of the international workshop on time constrained requirements engineering*.

Verner, J., & Babar, M. a. (2004). Software quality and agile methods. In *Proceedings of the 28th annual international computer software and applications conference, 2004* (pp. 520–525).

Wieringa, R. (2014). Empirical research methods for technology validation: Scaling up to practice. *Journal of Systems and Software, 95*, 19–31. http://dx.doi.org/10.1016/j.jss.2013.11.1097.

Wolfgang, E. (2011). Working with user stories. In *Agile requirements engineering workshop*, July 2011.

Yu, Y., & Sharp, H. (2011). Analysing requirements in a case study of pairing. In *Agile requirements engineering workshop*, July 2011.

Zhu, Y. (2009). *Requirements engineering in an agile environment*. Uppsala University.