Requirements engineering practice and problems in agile projects: results from an international survey

Requirements engineering (RE) is considerably different in agile development than in traditional processes. Yet, there is little empirical knowledge on the state of the practice and contemporary problems in agile RE. As part of a bigger survey initiative (Naming the Pain in Requirements Engineering), we build an empirical basis on such aspects of agile RE. Based on the responses from 92 people representing 92 organizations, we found that agile RE concentrates on free-text documentation of requirements elicited with a variety of techniques. Many manage explicit traces between requirements and code. Furthermore, the continuous improvement of RE is done because of intrinsic motivation. Important experienced problems include unclear requirements and communication flaws. Hence, agile RE is in several aspects not so different from RE in other development processes. We plan to investigate specific techniques, such as acceptance-test-driven development, in a future survey to better capture what is special in agile RE.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.2038v1 | CC-BY 4.0 Open Access | rec: 10 May 2016, publ: 10 May 2016

Requirements Engineering Practice and Problems in Agile Projects: Results from an International Survey

Stefan Wagner[1], Daniel Méndez Fernández[2], Michael Felderer[3], Marcos Kalinowski[4]

[1] University of Stuttgart, Germany

[2] Technical University of Munich, Germany

[3] Michael Felderer, University of Innsbruck, Austria

[4] Marcos Kalinowski, UFF, Brazil

Corresponding Author:

Stefan Wagner[1]

Universitätsstr. 38, 70569 Stuttgart, Germany

Email address: stefan.wagner@informatik.uni-stuttgart.de

ORCID ID: 0000-0002-5256-8429

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.2038v1 | CC-BY 4.0 Open Access | rec: 10 May 2016, publ: 10 May 2016

Requirements Engineering Practice and Problems in Agile Projects: Results from an International Survey

Stefan Wagner[1], Daniel Méndez Fernández[2], Michael Felderer[3], and Marcos Kalinowski[4]

[1] University of Stuttgart, Germany
ORCID ID: 0000-0002-5256-8429
stefan.wagner@iste.uni-stuttgart.de
[2] Technical University of Munich, Germany
ORCID ID: 0000-0003-0619-6027
daniel.mendez@tum.de
[3] University of Innsbruck, Austria
ORCID ID: 0000-0003-3818-4442
michael.felderer@uibk.ac.at
[4] UFF, Brazil
ORCID ID: 0000-0003-1445-3425
kalinowski@ice.ufjf.br

Abstract. Requirements engineering (RE) is considerably different in agile development than in traditional processes. Yet, there is little empirical knowledge on the state of the practice and contemporary problems in agile RE. As part of a bigger survey initiative (*Naming the Pain in Requirements Engineering*), we build an empirical basis on such aspects of agile RE. Based on the responses from 92 people representing 92 organisations, we found that agile RE concentrates on free-text documentation of requirements elicited with a variety of techniques. Many manage explicit traces between requirements and code. Furthermore, the continuous improvement of RE is done because of intrinsic motivation. Important experienced problems include unclear requirements and communication flaws. Hence, agile RE is in several aspects not so different from RE in other development processes. We plan to investigate specific techniques, such as acceptance-test-driven development, in a future survey to better capture what is special in agile RE.

Key words: requirements engineering, agile, state of the practice, survey

1 Introduction

We have seen a substantial change in the way requirements engineering (RE) is practiced in today's software engineering projects because of the success of agile methods: "No matter the specific method, agile's treatment of requirements is fundamentally different." [1] Furthermore, recent studies indicate that agile

practices are frequently adapted to the particularities of their individual environments [2]. A clear understanding of the practice of agile RE and its problems would be needed to steer future research. NaPiRE (Naming the Pain in Requirements Engineering) is an international initiative which tries to fill this gap and to establish a broad survey investigating the status quo of RE in practice together with problems respondents experience in their project environments. In this paper, we investigate RE practice and problems in agile projects based on data from NaPiRE.

1.1 Problem Statement

Although we can experience a growth in the body of knowledge about software engineering practices, little is known on the current state of practice in requirements engineering in general [3]. Moreover, despite the importance of agile practices, little is yet known how industrial environments conduct RE in an agile setting [4] and what problems they face.

1.2 Research Objective

We aim at investigating the current state of practice in agile requirements engineering and potentially resulting problems as an empirical basis for further investigations.

1.3 Contribution

We report on the results of an analysis of the state of practice and experienced problems in agile requirements engineering based on survey data from 92 organisations from a globally distributed family of surveys.

1.4 Outline

The remainder of this paper is structured as follows. Section 2 reviews related work with respect to empirical studies on agile RE and the NaPiRE initiative. Section 3 discusses the design of the study. Section 4 presents the results of the study. Finally, Section 5 concludes and presents future work.

2 Related Work

We briefly review the existing work on empirical studies on agile requirements engineering before we describe the context and previously published materials of the survey.

2.1 Empirical Studies on Agile RE

Heikkilä et al. [5] conducted a mapping study on requirements engineering in agile software development in 2015. Hence, it gives a great overview of the topic. They state that "the definition of agile RE is vague." This is also reflected in corresponding empirical studies that often do not specify the concrete process model used. The majority of the papers in their mapping study contained some kind of empirical evaluation. We refer to their paper for details.

Furthermore, there is a recent systematic literature review by Inayat et al. [6]. They summarise the results of 21 primary studies relating to agile requirements engineering. There is only one paper classified as survey research which in turn conducts interviews.

Cao and Ramesh [4] conducted a qualitative study of 16 software development organisations on their agile RE practices. They identified and rated detailed RE practices. They found, for example, that face-to-face communication, prototyping and reviews and tests are common agile RE practices. To some degree comparable is only the survey by Bustard, Wilkie and Greer [7]. They investigate the maturity of agile development principles and practices but also touch the topic of requirements. They found that their participants see a process benefit in agile requirements gathering and management. It is also mentioned that while quality requirements were all improved by agile methods, one company stated a "generally weaker treatment of non-functional requirements in an agile approach."

2.2 The NaPiRE Initiative

The NaPiRE (Naming the Pain in Requirements Engineering) initiative was started in 2012 as a reaction to the lack of a general empirical basis for RE research. The idea was to establish a broad survey investigating the status quo of RE in practice together with contemporary problems practitioners encounter. This should lead to the identification of interesting further research areas as well as success factors for RE.

We created NaPiRE as a means to collaborate with researchers from all over the world to conduct the survey in different countries. This allows us to investigate RE in various cultural environments and increase the overall sample size. Furthermore, we decided to run the survey every two years so that we can cover slightly different areas over time and have the possibility to observe trends. NaPiRE aims to be open, transparent and anonymous while yielding accurate and valid results.

At present, the NaPiRE initiative has members from 14 countries mostly from Europe but also North-America, South-America and Asia. There have been two runs of the survey so far. The first was the test run performed only in Germany and in the Netherlands in 2012/13. The second run was performed in 10 countries in 2014/15. All up-to-date information on NaPiRE together with links to all publications and the data is available on the web site `http://www.re-survey.org`.

The first run in Germany together with the overall study design was published in [8] with the detailed data and descriptive analysis available as technical report [9]. It already covered the spectrum of status quo and problems. Overall, we were able to get full responses from 58 companies to test a proposed theory on the status quo in RE. We also made a detailed qualitative analysis of the experienced problems and how they manifest themselves.

For the second run, we have published three papers [10, 11, 12] so far, concentrating on specific aspects and the data from only one or two countries. An analysis of the data with a focus on the state of practice of RE in agile projects as well as a study based on data from all 10 countries has not been published so far.

3 Survey Design

This paper uses a part of the overall NaPiRE design: We focus on the descriptive analysis of the state of the practice and potential problems in agile requirements engineering. For that, we analyse the data from the second NaPiRE run conducted in 2014/15. In the following, we detail the information on the study design relevant to the analysis presented in this paper. Details on the overall principles and process followed in NaPiRE, the team involved, and the full instrument used can be taken from our project website `http://www.re-survey.org`.

3.1 Research Questions

We aim at understanding the state of practice of requirements engineering in agile projects. This cannot be exhaustive as there are too many aspects potentially relevant to agile requirements engineering. Our objective is to be generic to be able to apply the same instrument to non-agile projects. To this end, we formulate the following five research questions, shown in Table 1, to steer the design of our study:

Table 1. Research questions

RQ 1	How are requirements elicited and documented?
RQ 2	How are requirements changed and aligned with tests?
RQ 3	Why and how is RE improved?
RQ 4	Is there an RE standard and how is it applied?
RQ 5	What are common problems in agile RE?

The first question aims to capture the most basic activities in RE: elicitation and documentation. Yet, a key principle in agile development is that requirements are not stable. Hence, we want to understand how agile projects deal in particular with changing requirements. A further key principle in agile development is the continuous improvement of the development process itself. This

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.2038v1 | CC-BY 4.0 Open Access | rec: 10 May 2016, publ: 10 May 2016

Table 2. Questions (simplified and condensed excerpt)

Parts	No.	Question	Type
Demographics	Q 1	What is the size of your company?	Closed(SC)
	...	...	...
	Q 8	Which process model do you follow (or a variation of it)?	Closed(MC)
Status Quo	Q 9	How do you elicit requirements?	Closed(MC)
	Q 10	How do you document functional requirements?	Closed(SC)
	Q 11	How do you document non-functional requirements?	Closed(SC)
	Q 12	How do you deal with changing requirements after the initial release?	Closed(SC)
	Q 13	Which traces do you explicitly manage?	Closed(MC)
	Q 14	How do you analyse the effect of changes to requirements?	Closed(MC)
	Q 15	How do you align the software test with the requirements?	Closed(MC)
	...	...	...
	Q 22	How is your RE standard applied in your projects?	Closed(MC)
	Q 23	Is your RE continuously improved?	Closed(SC)
	Q 24	Why do you continuously improve your RE?	Closed(MC)
	...	...	...
Problems	Q 28	Considering your personal experiences, how do the following (more general) problems in requirements engineering apply to your projects?	Likert
	...	...	...

should also hold for the RE process. Therefore, we are interested in whether agile projects perform continuous improvement and what is their motivation. Next, while agile methods emphasise flexibility, they still can have a standard way of doing RE. We investigate what kinds of standards agile projects use and how they apply them. Finally, after gathering an understanding about the state of the practice, we want to understand how important various potential problems for RE are in agile projects and what are their causes and effects.

3.2 Instrument

The instrument used in NaPiRE constitutes in total 35 questions used to collect data on topics including the demographics, how practitioners elicit and document requirements and finally what problems practitioners experience in their RE. In this study, we focus on the status quo using the demographics only as context and to select the companies working in an agile manner. We will also discuss the main problems as rated by these companies, but we will not go into a more detailed problem analysis. Table 2 summarises the excerpt of our questionnaire in scope of this study.

For these areas, we only use closed questions. The answers can be mutually exclusive single choice or multiple choice answers. Most of the closed multiple choice questions include a free text option, e.g. “other”, so that the respondents can express company-specific deviations. We furthermore use Likert-type scales on an ordinal scale of 5 and define for each a maximum value (e.g., “agree”, or “very important”), a minimum value (e.g., “disagree”, or “very unimportant”), and the middle (“neutral”). These are used to answer the last question on the

problems where we let the respondents rate the extent to which a given set of typical RE problems apply to their agile project environments.

3.3 Data Collection

The survey is conducted by invitation only to have a better control over the distribution of the survey among specific companies and also to control the response rate. The responses were, however, anonymous to allow our respondents to freely share their experiences made within their respective company. For each company, we invited one respondent as a representative of the company. In case of large companies involving several autonomous business units working each in a different industrial sector, we selected a representative for a unit. For the data collection, each country representative defined an invitation list including contacts from different companies and initiated the data collection independently as an own survey project. All surveys relied on the same survey tool[1] hosted and administrated by the authors. The data collection phases in each country is shown in Table 3.

Table 3. Data collection phase (overview)

Area	Country	Data Collection Phase
Central Europe		
	Austria	2014-05-07 to 2014-09-15
	Germany	2014-05-07 to 2014-08-18
	Ireland	2014-05-07 to 2014-12-31
North America		
	Canada	2014-05-07 to 2015-08-15
	USA	2014-05-07 to 2015-05-01
Northern Europe		
	Estonia	2014-05-07 to 2014-10-31
	Finland	2015-06-01 to 2015-08-28
	Norway	2014-05-07 to 2014-09-15
	Sweden	2014-05-07 to 2014-09-15
South America		
	Brasil	2014-12-09 to 2015-03-31

We conducted the survey in North America (Canada, USA), South America (Brazil), Central Europe (Austria, Germany, Ireland) and Northern Europe (Estonia, Finland, Norway, Sweden). The corresponding response rates are shown in Table 4.

3.4 Data Analysis and Validity Procedures

In the subset of NaPiRE that we will discuss in this paper, we conduct two types of analysis: The first analysis is frequency counting for questions in which the respondents choose one or more options. This is, for example, the case when they

[1] We implemented the survey as a Web application using the *Enterprise Feedback Suite*.

Table 4. Countries and response rates

Area	Country	Response Rate
Central Europe	Austria	72.0 %
	Germany	36.8 %
	Ireland	39.7 %
North America	Canada	75.0 %
	USA	60.0 %
Northern Europe	Estonia	89.0 %
	Finland	83.0 %
	Norway	59.0 %
	Sweden	34.0 %
South America	Brazil	63.0 %

have to choose which requirements elicitation techniques they use. We extract the counts using an R script which also creates bar charts from it.

The second analysis is necessary for the question about contemporary problems. We analyse the Likert type data by transforming the answers to numbers from 1 to 5. Then, we calculate and report the median and the median absolute deviation (MAD) for each problem also using an R script. We refrain from a detailed qualitative analysis and coding of the free-text answers, because this is out of scope of this paper. Yet, we use them to substantiate the discussion and interpretation of the ranking of importance of the problems.

The overall NaPiRE endeavour includes several procedures for checking validity, i.e., concerning the data collection and analysis phases, as described in detail in our previously published material [8].

4 Results

In the following, we summarise our results structured according to the research questions and beginning with an overview of the study population.

4.1 Study Population

Overall, we received 354 answers to the second NaPiRE run in 2014/15 out of which 228 completed the questionnaire. Out of these, we selected the 92 organisations that answered “Scrum” and/or “XP” as their development process model, but not “Waterfall”, “V-Model XT” or “Rational Unified Process”. Hence, the following results represent the situation of 92 different companies or business units (in case of large companies).

To better illustrate the study population, we grouped organisations into small, medium, and large ones. For this grouping, we relied on the number of employees. Organisations with up to 50 employees were considered small, with 51 to 250 medium, and organisations with more than 250 were considered large.

Table 5 summarises the distribution of the responses according to the different company sizes and the areas where they are situated.

Table 5. Responding organisations by size and region

Size	Central Europe	North America	Northern Europe	South America	Total
Small	6	4	6	14	30
Medium	4	0	8	10	22
Large	12	8	11	8	39
Unknown	0	0	1	0	1
Total	22	12	26	32	92

4.2 Elicitation and Documentation (RQ 1)

We start answering RQ 1 by looking at how agile projects elicit requirements. We used the elicitation technique classification as provided in the SWEBOK.[2] How often these elicitation techniques have been selected by our respondents is shown in Fig. 1. The most frequently used techniques are interviews, prototyping, and facilitated meetings. Scenarios are employed by about half of the respondents, observations by less than a third. Additional answers for *Others* included "Created personas and presented them to our stakeholders", "Questionnaires"/"Surveys" and "It depends on the client."

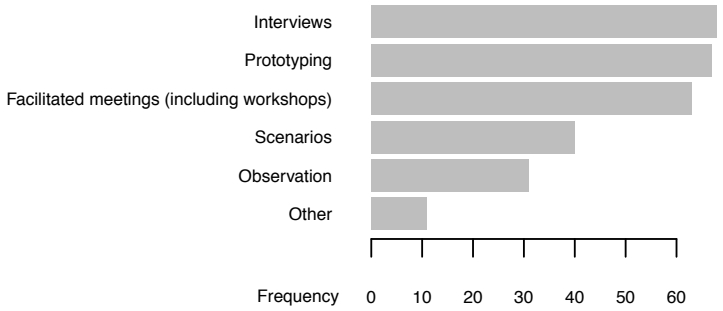

Fig. 1. How do you elicit requirements?

We believe these answers fit very well to the expectation to agile projects. Roles like a product owner in Scrum would use interviews to understand the overall product requirements while the further elicitation is done in workshops with stakeholders and the sprint planning. Prototyping is usually not an explicit part of agile methods but building minimal viable products could be seen as a form of prototyping. Furthermore, paper prototypes or wire frames of user interfaces can also be useful in agile projects. Observations are not frequently used. Maybe there is potential to explore this kind of elicitation in more detail. But it might also be caused by a lack possibilities for the developers.

[2] `www.swebok.org`

Next, we asked about the documentation of the most frequent type of requirements: functional requirements. The respondents could choose multiple items from different description techniques, namely

- structured requirements lists,
- domain/business process models,
- goal models, data models, and
- and use-case models

as well as their degree of formality, namely

- free form,
- textual,
- textual with constraints, and
- semi-formal or formal.

As shown in Fig. 2, the three most frequent ways to document requirements are as free-form textual domain/business process models, free-form textual structured requirements lists and use case models as text with constraints. But also structured requirements lists as texts with constraints and free-form textual use case models are used by almost a third of the respondents. Data models are almost only used in a semi-formal notation such as the UML. Goal models are rarely used overall. Formal notations for requirements are also rarely used.

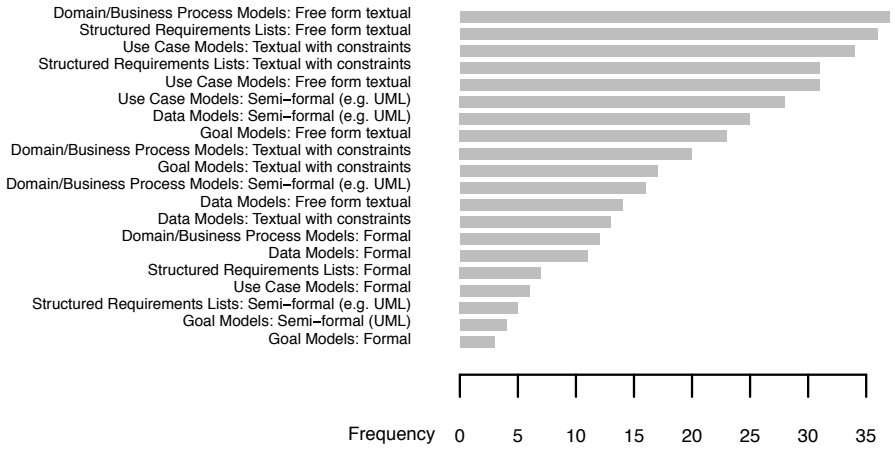

Fig. 2. How do you document functional requirements?

This again fits to the expectation of common agile methods: requirements are usually written down as text either in a free form or with some constraints (such as the role/feature/reason schema for user stories). Only data models are documented with a class diagrams or a variation of them. More semi-formal and formal documentation methods are probably too heavy-weight or unnecessary in the presence of automated tests for requirements. Especially the role of automated tests would be interesting to follow-up in further studies.

Finally, we briefly touched also the topic of non-functional requirements (such as security or performance requirements). As shown in Fig. 3, we found that most respondents document non-functional requirements with text. About half of those document non-functional requirements either in a quantified manner, e.g., by defining concrete measurements, or in a non-quantified manner, e.g., by linking to external reference models or style guides. Answers for *Other* included "on our user stories" and "user story acceptance criteria". Overall, there seems to be no clear consensus if non-functional requirements should be documented in a quantified or non-quantified way in the given settings.

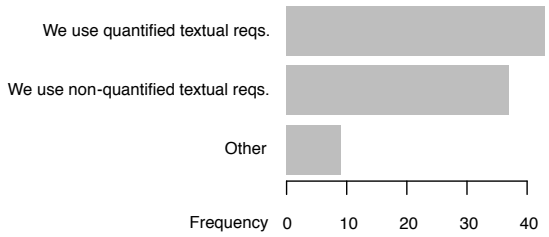

Fig. 3. How do you document non-functional requirements?

4.3 Changing Requirements (RQ 2)

In RQ 2, we are interested in how agile projects document changes in requirements. First, we asked how the respondents deal with changing requirements

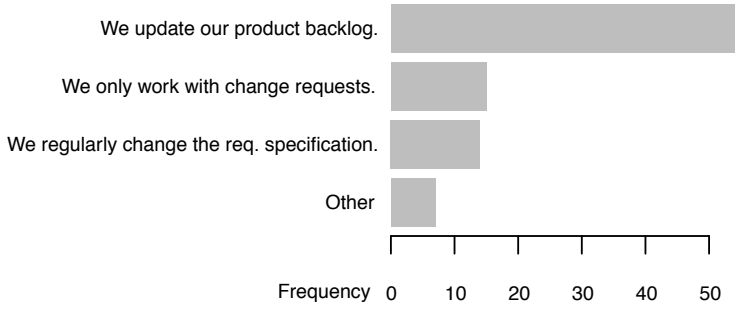

Fig. 4. How do you deal with changing requirements after the initial release?

after the initial release. The answers are shown in Fig. 4. As to be expected, the overwhelming majority updates the product backlog when requirements change. Yet, 16 % only work with change requests and 15 % even have a requirements specification they regularly change. Answers for *Other* include "all methods,

depends on the project" and "we mix product backlog and change requests". Overall, the product backlog seems to be the common way to work with changing requirements in agile projects, but is not always clear how it works together with change requests.

Next, we were interested in how the respondents analyse the effect of changes to requirements. As shown in Fig. 5, most respondents do impact analysis between requirements. More than a third analyse the impact of requirement changes on the code. A fifth do no analysis of the effect of changes to the requirements. Answers for *Other* include "test-driven analysis for TDD projects", "rerun test suites", "we discuss with users and decide the best approach" and "team-based discussion before change". Therefore, besides looking at requirements and code, the test suites and direct discussions with stakeholders seem important for impact analyses in agile projects.

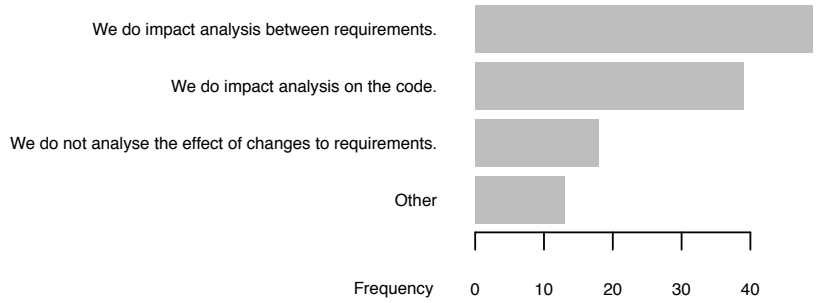

Fig. 5. How do you analyse the effect of changes to requirements?

A help for impact analysis are traces between requirements and code or between requirements and design documents. As can be seen in Fig. 6, more than half of the respondents answered that they explicitly managae traces between the requirements and the code. A third manages explicitly the traces between requirements and design documents. Unfortunately, we did not ask in detail how these traces are implemented. More than a fifth of the respondents do not explicitly manage traces at all. For the *Other* answer, several respondents mentioned traces between tests and requirements: "Traces between requirements and functional/system tests are most common for us."

Finally, we had a question relating requirements and tests. We asked how the respondents align the tests with the requirements. As shown in Fig. 7, in agile RE, it is common to define acceptance criteria.

This is what we would expect because specific test-driven practices which have become popular in and through agile methodologies like test-driven development [13] and behavior-driven development [14] as well as the common user story practices demand to make acceptance criteria explicit. Furthermore, also coverage of requirements by tests is considered in a remarkable number of agile projects. Also in this case, test-driven practices linked to agile methodologies

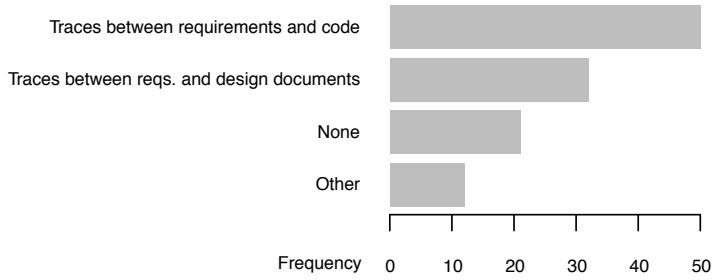

Fig. 6. Which traces do you explicitly manage?

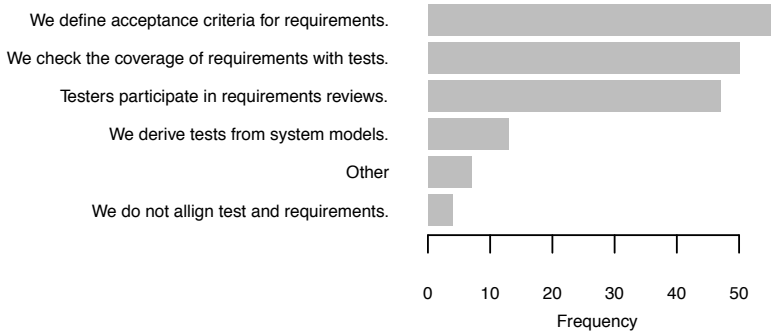

Fig. 7. How do you align the software test with the requirements?

may be a trigger for that. In about half of the projects of the respondents, the testers participate in requirements reviews. This also means that half of the projects do have requirements reviews, which we would not expect from all agile projects as it is not demanded in common agile development processes. Finally, The derivation of tests from system models either automatically [15] or manually [16] is only rarely done. Answers for *Other* include specific techniques ("BDD - Behavior Driven Development") as well as references to the used process model and that it prescribes the relation of requirements and tests ("The user story of the backlog is a good basis for creating a test case to test the base requirement(s) of that user story - Agile/Scrum").

4.4 RE Improvement (RQ 3)

Also, and maybe in particular, requirements engineering processes need to be improved. In an agile context, we would expect this improvement to be done continuously. We asked whether the organisations improve their RE continuously and who is responsible for this improvement. The results in Fig. 8 show that in more than half of the responding organisations, the RE is continuously improved and this improvement is under sovereignty of the project team. This is in tune

with our expectations because of the deeply entrenched idea to regularly work on the development processes with, for example, retrospectives in Scrum.

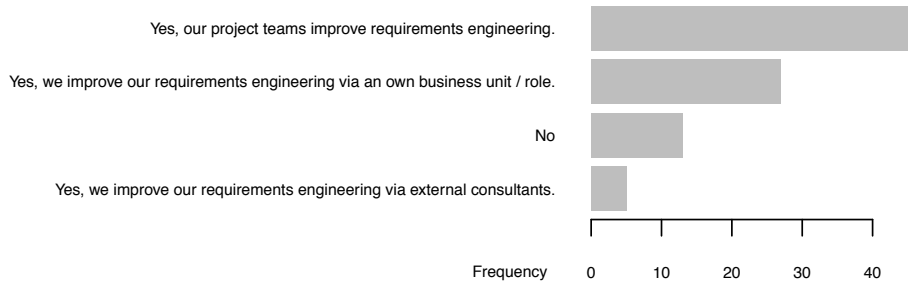

Fig. 8. Is your requirements engineering continuously improved?

Yet, also almost a third of the respondents have an own business unit or role responsible for the continuous improvement. Only few respondents use external consultants for that. Still, about 14 % of the respondents do not continuously improve their RE.

At this point, we wanted to dig deeper and understand the reasoning behind doing a continuous improvement. As shown in Fig. 9, most of the respondents who do continuous improvement do it because it helps them to determine their individual strengths and weaknesses and to act accordingly. Hence, the motivation is mostly intrinsic. Only a quarter or below give extrinsic reasons such as the expectation of the customer, certifications or regulations. Answers for the category *Other* include "Better efficiency", "To improve the quality in project development" and "We adopt an agile method that has inspection as one of the principles and adopt them to promote continuous improvement."

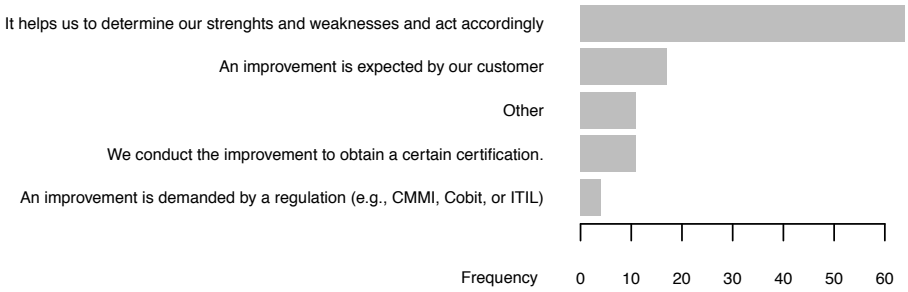

Fig. 9. Why do you continuously improve your requirements engineering?

Hence, continuous improvement in general as well as in RE is widespread in agile projects in practice. The motivation is intrinsic based on a perceived improved efficiency and because it is postulated by agile process models.

4.5 RE Standard (RQ 4)

As an RE standard, we consider both formal norms but also more informal conventions that guide the way of working in requirements engineering. Classical RE is often aligned with organisational or industrial software development or software life cycle standards to guarantee the quality of specified requirements and the overall RE process. In an agile context, we would expect that standards play a minor role.

For this reason, we asked what RE company standards are established in the respective organisation. The results in Figure 10 show that in about half of the companies the RE standard is predefined by the development process. Taking a closer look into typical agile processes, this is not surprising because requirements artefacts play a prominent role to link customers and development and customer collaboration is even an important principle in the Manifesto for Agile Software Development [17]. In Scrum, for instance, the product backlog typically consists of user stories which often follow a specific format and which can be considered as requirements artefacts.

Standards that are predefined by a regulation, own standards that define artefacts, milestones and phases, as well as own standards that define artefacts and offer document templates are each equally used by about one fifth of the organisation. These more plan-driven types of standards seem therefore rather seldom in light-weight agile requirements engineering.

Using other standards or not using a standard at all has been stated rather rarely.

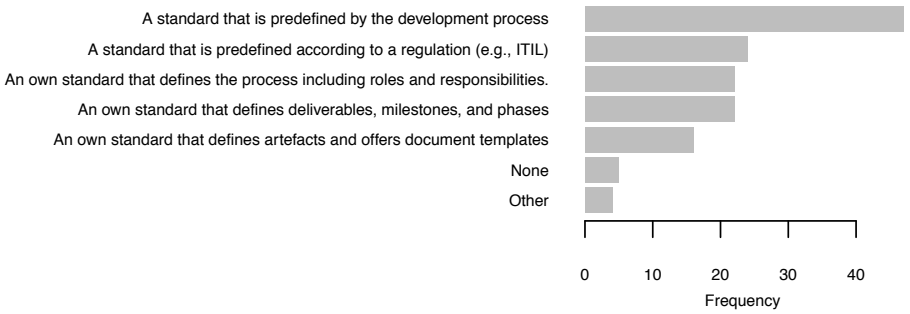

Fig. 10. What RE company standard have you established in your company?

We then asked whether the applied requirements engineering standards are mandatory and practiced. The results in Figure 11 shows that about two-third of the RE standards are actually practiced, although only about half of them are

also mandatory. Standards in requirements engineering of agile projects are thus more commonly practiced than one might assume. On the contrary, only about 10 percent of the RE standards are mandatory but not practiced. This additionally emphasises that RE standards are actually practiced in agile projects even without external pressure as they seem to be beneficial for the overall project success. About 13 percent of the RE standards are neither mandatory nor practiced.

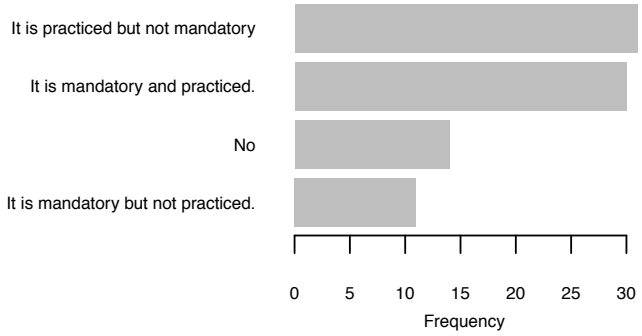

Fig. 11. Is the requirements engineering standard mandatory and practiced?

As RE standards are widely practiced, we further investigated how they are applied and tailored in regular projects. The results in Figure 12 show that the most common variant of application performed by about one-third of the organisations is that the requirements engineer tailors the standard based on own experiences. About half the number of that have a tailoring approach that continuously guides the application of the standard, and approximately the same number of organisations even has tool support for tailoring. About 20 percent of the organisations does not consider a tailoring approach. Tailoring seems therefore very common, but primarily guided by experience.

Finally, we asked how the application of requirements engineering standards is actually checked. The results in Figure 13 show that this is most commonly done via project assessment. For instance, Scrum explicitly considers a retrospective where also the application of RE standards could be assessed. In about 30 percent of the organisations, constructive quality assurance measures are applied, for instance via checklists or templates, and in about 25 percent of the organisations analytical quality assurance techniques like reviews are applied. Other checks are applied in five percent of the organisations.

4.6 Problems in Agile RE (RQ 5)

Finally, after getting an overview of the current state of practice (RQ1–4), we wanted to know what common problems the respondents experience in their respective project environments. To this end, we presented a list of common RE

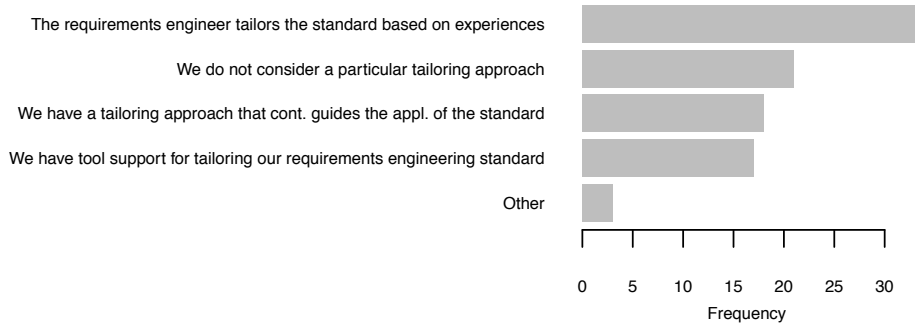

Fig. 12. How is your requirements engineering standard applied in your regular projects?

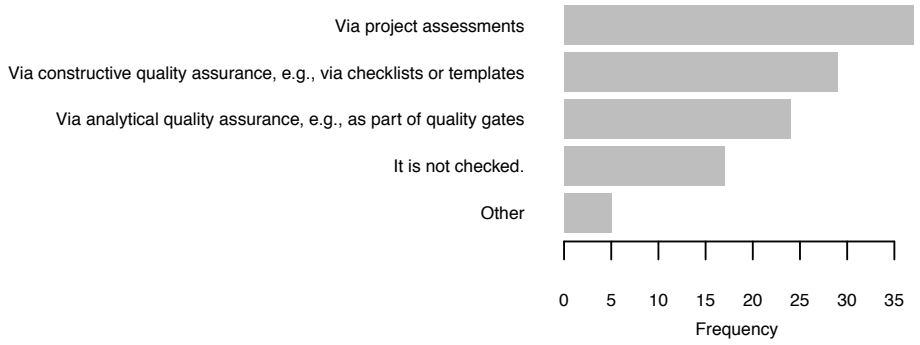

Fig. 13. How do you check the application of your requirements engineering standard?

problems and asked the respondents whether they agree that these problems occur in their setting. Table 6 summarises the problems, ordered from top to bottom according to the agreement by the respondents.

The problems ranked as low where not surprising to us considering the agile RE setting. For instance, volatility in the customer's business domain seems not to be a critical problem. Yet, one respondent described the "Lack of business organization to have the system ready." as a cause of this problem which then led to "Project failure". Overall, however, changes in processes or requirements are what agile processes are designed for at this shows in the little relevance of this problem in practice.

Similarly, unclear responsibilities are experienced as a problem rarely. The clear roles in agile processes seem to provide a good understanding here. Respondents who experienced this problem give problems on the developer side or customer side to really understand and live up to their corresponding roles ("Our developers did not involve in the development process", "Customer does not understand the responsibilities of a PO, responsibilities unclear inside cus-

Table 6. Considering your personal experiences, how do the following problems in requirements engineering apply to your projects? (from 1: I disagree to 5: I agree)

	Median	MAD
Underspecified reqs. that are too abstract and allow for various interpretations	4	1
Unclear / unmeasurable non-functional requirements	4	1
Communication flaws within the project development team	4	1
Communication flaws between developers and the customer	4	1
Moving targets (changing goals, business processes and / or requirements)	4	1
Incomplete and / or hidden requirements	4	1
Implicit requirements not made explicit	4	1
Stakeholders with difficulties in separating reqs from previously known solutions	4	1
Inconsistent requirements	4	1
Insufficient support by project lead	3	1
Insufficient support by customer	3	1
Missing traceability	3	1
"Gold plating" (implementation of features without corresponding requirements)	3	1
Weak access to customer needs and / or (internal) business information	3	1
Weak knowledge of customer's application domain	3	1
Weak relationship between customer and project lead	3	1
Time boxing / Not enough time in general	3	1
Discrepancy between high innovation and need for formal acceptance of reqs	3	1
Volatile customer's business domain regarding	3	1
Terminological problems	3	1
Unclear responsibilities	3	1
Technically unfeasible requirements	2	1

tomer organization"). Hence, overall the roles in agile RE seem to support clear responsibilities but they need to be clearly explained and lived.

Some of the top ranked problems in turn can be argued based on the current state resulting from the agile process models used, such as unclear / unmeasurable non-functional requirements or underspecified requirements. The latter is caused by general problems in the capabilities of the involved people either on the development team ("Ability to write requirements and analyze customer needs") or on the customer side ("Customer clueless about functions of the system"). But also communication between the developers and the customers seem to cause these problems ("Developer may do their own wrong interpretation"). Furthermore, also in agile projects, it seems to be problematic to rush too quickly through defining what needs to be done ("Not enough time spent defining to the level of detail required").

For unclear or unmeasurable non-functional requirements, the cause seems to be mostly the reliance on experience ("Boils down to experience, on both ends. Non-functional requirements are easy to miss.") and unsuccessful communication ("think we talk about the same thing but not"). Agile RE might be able to provide more structure and terminology in this area to avoid consequences such as "Lots of surprises in deployment" and "unhappy end-users".

Some of the top-ranked problems, however, are at the same time surprising to us given that those problems also form a natural condition for agile projects. We refer in particular to moving targets and incomplete requirements which are stated as problems and which should motivate the use of agile practices. Even more, communication flaws within the project development team as well

as communication flaws between developers and the customer are stated under the top problems.

The moving target problem is caused by "Changing priorities", "Changes and instability in the customer organization are not isolated from our process. Their problems leak through." and "Mostly the reason is that the business is constantly learning at the same time or changes in management." This leads to "already specified requirements may become obsolete". Hence, fixing requirements during a sprint as, for example, emphasised in Scrum seems important to address these causes. Yet, overall the effects can be small: "If parties are on board that things are changing then the project won't have problems in term of budget, timeline etc. because everybody knows these are flexible as long as targets are moving. It will cause stress for dev team though."

In many agile RE approaches, requirements are not meant to be complete but a cause for discussions with the customer. Hence, incomplete requirements are to be expected. When does this become a problem? It is a problem if the effect is "Rework or delivery that does not fully meet the customer's need." or "customer dissatisfaction (delivery that does not meet customer expectations)". It is caused by "Hidden requirements that are obvious to the customer" and inexperience of the product owner and customer ("Lack of experience of the Product Owner; Lack of clarity / understanding of the client"). Hence, the role of the on-site customer or product owner is a central one that needs to be filled with a person being able to understand the customers and elicit all important requirements.

The communication flaws seem to be mostly caused by missing time ("Also related to an attempt to gain time in developing.", "high need for meetings and documentation versus time") and more general communication problems ("Lack of open dialogue on the team.", "Our developers don't know the flows to generate questions to other teams."). These communication problems can lead to unnecessary work ("we waste time trying to develop new features that were developed by other teams previously") and generating unnecessary risks ('Unsolved problems due to the lack of dialogue between people.").

We interpret this as that the prerequisite on which agile RE relies, i.e. human-intensive continuous exchange, can quickly manifest itself as a critical problem. That is, agile RE does not necessarily solve all problems plan-driven process models often have, but they become explicit once key prerequisites for successful RE are not met: human-intensive exchange and collaboration. Yet, it would be interesting to understand in more detail the causes and effects of these problems in agile projects.

5 Conclusions and Future Work

In this paper, we reported on the results from an analysis of the current state of practice and potential problems in agile requirements engineering. Our analysis is based on data gathered from a globally distributed family of practitioner surveys (called NaPiRE). We shed some light on how requirements are elicited

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.2038v1 | CC-BY 4.0 Open Access | rec: 10 May 2016, publ: 10 May 2016

and documented, how our respondents deal with changing requirements, and why and how RE is improved.

5.1 Summary of Conclusions

Overall, we found that most of the responding organisations conduct RE in a way we would expect in agile projects. The documentation of requirements is dominated by free-text documents with some constraints. The backlog is the central means to deal with changing requirements. Code and traces are explicitly linked and RE is continuously improved because of an intrinsic motivation.

Yet, for all these aspects there is also a considerable number of projects claiming to follow Scrum or Extreme Programming and not work in that way. To some degree, this supports or findings from [2] that many companies claiming that they do Scrum actually deviate heavily from it. Therefore, in future surveys, we will need to differentiate in more detail.

In terms of RE standards and how they are applied (as well as in several other aspects such as the documentation of requirements), agile RE is not so different from classical RE after all. Our concluding analysis of contemporary problems in RE revealed that some of the problems often seen to come along the use of plan-driven process models are not seen as critical anymore. Others, however, which are often seen as a motivation for agile RE, e.g., moving targets, can still become dominant. We will have to dig deeper into specifics of agile RE to better understand what agile RE practices are related to which problems or their mitigation.

5.2 Limitations

Although our analysis is based on a broad family of surveys, we are aware that our study has limitations. First, our results emerge from a reasonable but still limited sample with a limited context model. We therefore cannot make concrete statements about how generalisable the results eventually are, let alone because we still are not able to estimate the representativeness of our population. Therefore, we need to follow our design of a family of surveys and further steer our continuous replications while capturing the context more precisely.

Also, inherent to survey research is that surveys can only reveal stakeholders' perceptions on current practices rather than empirically backed-up knowledge about those practices. Although we were interested in revealing those perceptions, the answers given by our respondents might still be biased. We mitigated this threat by conducting the survey anonymously, but need to apply further empirical methods in the future to further explore the field based on project data.

Finally and most importantly, NaPiRE was not intentionally designed to explore RE in agile contexts. This has two implications. First, the selection of our sample was based on the self-assessment of the respondents based on a pre-defined list of options in the process models. This means that although the

respondents might have selected, for instance, Scrum, we still have no guarantee that the process model followed was indeed agile, but it could have also been a plan-driven, iterative model baptised as "Scrum" in that organisational context. Second, although we could analyse different variations in the status quo of how our respondents do their RE, our instrument does not yet capture the particularities of agile practices (e.g. considering agile artefacts such as user stories). As future work, we plan to steer re-design of our instrument to capture more precisely the particularities of agile practices.

5.3 Future Work

A richer investigation of facets important to agile RE forms part of future work where we redesign the instrument to give more attention to agile practices. We further plan to complement our survey research with other empirical methods such as case study research based on project data. Besides more detailed investigations of the current state of practice in RE, we plan to elaborate a first holistic theory on RE practices to define a reliable basis for future problem-driven research which by now too often relies on conventional wisdom only.

Acknowledgements

We are grateful to the whole NaPiRE team as well all respondents taking the time for the survey.

References

1. Leffingwell, D.: Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise. Addison-Wesley (2011)
2. Diebold, P., Ostberg, J., Wagner, S., Zendler, U.: What do practitioners vary in using Scrum? In: Proc. 16th International Conference Agile Processes, in Software Engineering, and Extreme Programming (XP), Springer (2015) 40–51
3. Cheng, B., Atlee, J.: Research directions in requirements engineering. In: Proc. Future of Software Engineering (FOSE'07), IEEE (2007) 285–303
4. Cao, L., Ramesh, B.: Agile requirements engineering practices: An empirical study. IEEE Software **25**(1) (2008) 60–67
5. Heikkilä, V.T., Damian, D., Lassenius, C., Paasivaara, M.: A mapping study on requirements engineering in agile software development. In: Proc. 41st Euromicro Conference on Software Engineering and Advanced Applications (SEAA), IEEE (2015) 199–207
6. Inayat, I., Salim, S.S., Marczak, S., Daneva, M., Shamshirband, S.: A systematic literature review on agile requirements engineering practices and challenges. Computers in Human Behavior **51** (2015) 915–929
7. Bustard, D., Wilkie, G., Greer, D.: The maturation of agile software development principles and practice: Observations on successive industrial studies in 2010 and 2012. In: Proc. 20th IEEE International Conference and Workshops on Engineering of Computer Based Systems (ECBS), IEEE (2013) 139–146

8. Méndez Fernández, D., Wagner, S.: Naming the pain in requirements enginering: A design for a global family of surveys and first results from germany. Information and Software Technology **57** (2014) 616–643
9. Méndez Fernández, D., Wagner, S.: Naming the Pain in Requirements Engineering – NaPiRE Report 2013. Technical Report TUM-I1326, Technische Universität München (2013)
10. Kalinowski, M., Spinola, R., Conte, T., Prikladnicki, R., Mendez Fernandez, D., Wagner, S.: Towards building knowledge on causes of critical requirements engineering problems. In: Proc. 27th International Conference on Software Engineering and Knowledge Engineering (SEKE). (2015)
11. Kalinowski, M., Mendes, E., Travassos, G.: Automating and evaluating the use of probabilistic cause-effect diagrams to improve defect causal analysis. In: Proc. International Conference on Product Focused Software Process Improvement, Springer (2011)
12. Kalinowski, M., Felderer, M., Conte, T., Spinola, R., Prikladnicki, R., Winkler, D., Mendez Fernandez, D., Wagner, S.: Preventing incomplete/hidden requirements: Reflections on survey data from Austria and Brazil. In: Proc. Software Quality Days, Springer (2016)
13. Beck, K.: Test-driven development: by example. Addison-Wesley Professional (2003)
14. Chelimsky, D., Astels, D., Helmkamp, B., North, D., Dennis, Z., Hellesoy, A.: The RSpec book: Behaviour driven development with Rspec, Cucumber, and friends. Pragmatic Bookshelf (2010)
15. Schieferdecker, I.: Model-based testing. IEEE software **29**(1) (2012) 14
16. Felderer, M., Herrmann, A.: Manual test case derivation from uml activity diagrams and state machines: A controlled experiment. Information and Software Technology **61** (2015) 1–15
17. Fowler, M., Highsmith, J.: The agile manifesto. Software Development **9**(8) (2001) 28–35