Improving user story practice with the Grimm Method: A multiple case study in the software industry

Garm Lucassen, Fabiano Dalpiaz, Jan Martijn E.M. van der Werf
and Sjaak Brinkkemper
{g.lucassen, f.dalpiaz, j.m.e.m.vanderwerf, s.brinkkemper}@uu.nl

Utrecht University, The Netherlands

Abstract. [Context and motivation] Previous research shows that a considerable amount of real-world user stories contain easily preventable syntactic defects that violate desired qualities of good requirements. However, we still do not know the effect of user stories' intrinsic quality on practitioners' work. **[Question/Problem]** We study the effects of introducing the Grimm Method's Quality User Story framework and the AQUSA tool on the productivity and work deliverable quality of 30 practitioners from 3 companies over a period of 2 months. **[Principal ideas/results]** Our multiple case study delivered mixed findings. Despite an improvement in the intrinsic user story quality, practitioners did not perceive such a change. They explained, however, there was more constructive user story conversation in the post-treatment period leading to less unnecessary rework. Conversely, project management metrics did not result in statistically significant changes in the number of comments, issues, defects, velocity, and rework. **[Contribution]** Introducing our treatment has a mildly positive effect but a larger scale investigation is crucial to decisively assess the impact on work practice. Also, our case study protocol serves as an example for evaluating RE research in practice.

Keywords: User Stories; Requirements Engineering; Agile Development; Empirical Study; Multiple Case Study.

1 Introduction

Thanks to the rapid adoption of agile development practices, approximately 50% of software practitioners capture requirements using the semi-structured natural language (NL) notation of *user stories* [25,8,12]: "As a ⟨*role*⟩, I want ⟨*goal*⟩, so that ⟨*benefit*⟩".

Despite the simplicity and wide-spread adoption of this requirements engineering (RE) method, practitioners make many mistakes when creating user stories. In previous work, we analyzed 1,000+ real-world user stories and found that 56% contain easily preventable syntactic defects [13]. Examples include the use of a non-standard template and the specification of multiple features in one user story.

Our solution for practitioners to create high-quality user stories was to introduce two artifacts: the Quality User Story (QUS) framework and the computational linguistics tool Automatic Quality User Story Artisan (AQUSA) [13]. Although we have empirically confirmed that numerous real-world user stories violate QUS characteristics [13], we still lack evidence of the impact of QUS and AQUSA on practitioners' work.

This paper studies the impact of QUS and AQUSA on team communication frequency, team communication effectiveness, work deliverable quality, and work productivity. We do this via a multiple case study where we compare the pre-treatment period with the experimental period (2 months each) with 30 practitioners from 3 companies.

The participants first attended a 2 hour training, received a summary of the training's content and applied their new skills in a training workshop. After we integrated AQUSA with the company's issue tracker, the participants applied QUS and AQUSA in their work activities for two months. During this period, we collected software development process metrics, practitioner perception of the impact, and inherent user story quality. To determine the effect of our treatment, we compared the results against data taken from the issue tracker in the preceding two months.

The results of our empirical study are summarized by the following findings:

- The intrinsic user story quality generally increased after the treatment, resulting in fewer violations of the user story qualities prescribed by the QUS framework;
- The perception on quality by practitioners shows a marginal improvement but without reaching statistical significance;
- Despite our accurate measurements of the impact on work practices, we could not identify any significant change, also due to changes in the organizational context.

The next section of this paper outlines the research method. We then present the results in Section 3 and review related literature in Section 4. Finally, Section 5 presents a discussion of the findings and concludes this paper with an outline of future work.

2 Research method

We study the impact of introducing a tool-supported quality framework for user stories in agile development. We do this through a multiple case study with three software product organizations. Over a span of two months, these organizations incorporated two artifacts into their user story practices: QUS and AQUSA.

QUS [13] defines the quality of user stories via three types of criteria: (i) *syntactic* (e.g., atomic and well-formed), concerning the textual structure of a user story; (ii) *semantic* (e.g., problem-oriented and unambiguous), which focus on the meaning of user stories; and (iii) *pragmatic* (e.g., unique and uniform), referring to how user stories are being used in RE. AQUSA automatically verifies some QUS criteria using computational linguistics algorithms. In particular, AQUSA supports those criteria for which we can reach close to 100% recall: unique, minimal, well-formed, uniform and atomic.

We combine QUS and AQUSA into a package called *Grimm Method*, which provides an easy-to-remember term for practitioners and explains our artifacts' use during user story creation, poker planning, and software development.

We investigate the impact of the Grimm Method (our independent variable) by presenting data that test the following four hypotheses (each defining one dependent variable): *"Applying the QUS framework accompanied by the AQUSA tool"* . . . :

H1 - *"increases communication frequency"*
H2 - *"fosters more effective communication"*
H3 - *"improves work deliverable quality"*
H4 - *"increases work productivity"*

2.1 Grimm Method Treatment Design

To minimize threats to the internal validity of our study and to ensure uniform data collection, we devised a stepwise treatment design. All the physical materials described in each of the steps below are available online [11]. Also, video/audio-recordings of the training sessions can be made available upon request.

1. **Baseline measurement -** Two months before applying the treatment we start collecting software development process metrics. We refer to this time frame as the pre-treatment period.
2. **Intake survey -** On the treatment application day, each experiment participant fills out a brief intake survey to learn more about his/her knowledge of user stories and professional experience.
3. **Training -** The experimental treatment consists of a 2-hour training session during which the participants: (1) attend a presentation by the first author on the Grimm Method [11], (2) discuss analysis of their stories by AQUSA, (3) apply the QUS framework in a training workshop and (4) receive a summary of the training's content as reference material.
4. **Experimental period -** Upon completing the training, the team kicks off the two-month experimental period by integrating AQUSA with their issue tracker. From this moment, the team applies QUS in their daily work activities, supported by AQUSA that automatically analyzes any updated or newly added user stories and collects data on company's software development processes.
5. **Exit survey -** At the end of the experimental period each team member completes an exit survey on the impact of introducing the Grimm Method. The exit survey consists of 24 questions to capture the participant's perception of H1–H4, subdivided in four distinct parts. The first and second part include questions on the pre-treatment period and experimental period in isolation, the third part has questions that directly compare the two periods, and the fourth part includes four open questions for respondents to provide further feedback.
6. **Project Manager Evaluation -** Together with the team's project manager we evaluate the experimental period. The goal of this conversation is to validate our interpretation of the data. In particular, we ask the project manager to clarify outliers and to explain context-specific responses to the exit survey.
7. **Interviews -** We go through the responses to the exit survey to identify participants who give inconsistent, particularly positive, remarkably negative and/or opinionated answers. We invite these participants to clarify their responses in a follow-up interview to gather in-depth qualitative data.

These steps result in three types of data to test the hypotheses: (1) the intrinsic user story quality as reported by AQUSA, (2) the practitioners' perception of the Grimm Method's impact on work processes and user story quality, and (3) metrics about the software development process. We detail each data type in the following subsections. We employ methodological triangulation to reduce bias and to strengthen the validity of our results to form a more detailed and balanced picture of the actual situation [18].

2.2 Measures

User Story Quality. The first type of collected data is the intrinsic user story quality reported by AQUSA. As a direct measure of the impact of introducing the Grimm Method, the results indicate whether the experiment participants actually started creating syntactically better user stories. We apply AQUSA analyses to user stories created during the pre-treatment period and experimental period to detect the total number of violations for five quality criteria: well-formed, atomic, minimal, uniform and unique [13]. Note that this is a subset of the full QUS framework based on the characteristics that can be automatically checked with high precision and recall accuracy. We expect the total number of violations to decrease after introducing the Grimm Method.

Perceived Impact on Work Practice. We collect the second type of data by means of the exit survey and follow-up interviews. In both cases, the experiment participants self-report how they perceive the impact of the Grimm Method on their work. In total, the exit survey includes 12 Likert-Type statements on whether the respondents perceive the user stories to contribute to H1, H2, H3 and H4. Examples include "The user stories improved my productivity" and "User story conversation occurs more frequently since the Grimm Method training". Furthermore, the survey contains 4 questions and statements on respondents' perception of the intrinsic quality of the user stories themselves such as "How would you rate the quality of the user stories?" and "The quality of our user stories is better since the Grimm Method training". For the complete list of survey questions we refer the reader to the online materials [11]. Note that similar to our previous work [12], the survey relies on the participants' own understanding of *productivity* and *work deliverable quality* rather than enforcing our own definition.

Process Metrics. Over the years, a large corpus of software development metrics has been proposed in the literature. The available metrics span from inherent code quality metrics such as *cyclomatic complexity* [15] to team well-being metrics like the *Happiness Index* [10]. For RE alone, a literature review by Holm et al. [6] distilled 298 unique dependent variables from 78 articles. In RE research, however, it is unclear when and why which specific metrics are applicable [6].

Although the quality of user stories can potentially impact code quality metrics, the primary intention of user stories is to streamline the processes facilitating software development. By capturing and communicating the discussion around the features to be implemented, user stories aim to enable developers to produce software that stakeholders actually want [2] and practitioners perceive a key quality of user stories to be enabling creation of the *right software* [12]. For this reason this study looks at a specific type of metrics related to the (human) *processes* around software development [14].

We select the following five metrics from literature for their relevance to our hypotheses and their availability from project management software like Jira[1]:

Formal communication - We measure the frequency of communication in a team (H1) as the number of comments added to the stories completed in a 2-week sprint.

Rework - We measure the amount of rework in a project by calculating the *recidivism rate*: the rate at which user stories move backward in the software development

[1] https://www.atlassian.com/software/jira

process [4]. When a developer assigns the status 'done' to a user story, but it is not complete or has bugs, it is re-assigned 'to-do' or 'in progress', causing the recidivism rate to go up. A high recidivism rates is an indicator of ineffective communication causing misunderstanding among the stakeholders. We calculate recidivism rate as: $200 * (Backward/(Forward + Backward))$. Note that we include a multiplier of 200 to get a natural 0-100% range, instead of 0-50 as in [4]. This metric measures communication effectiveness thereby relating to H2.

Pre-release defects - The number of bugs added to the issue tracker during a 2-week sprint. Inspired by defect prediction literature [19,26,20], which base their metrics on all the defects in a six-months period before a new release. Unlike these works, we focus on the narrow 2-week period of a sprint as Scrum prescribes every sprint to be a potentially *releasable* increment. This metric measures work deliverable quality, which is used to assess if H3 holds.

Post-release defects - The number of bugs related to user stories in a sprint, as reported in the two sprints after that sprint. This choice is inspired by defect prediction literature, which counts all defects in the first six months after a release. Again, we substantially shorten this time-frame because of Scrum's quick feedback cycle. Moreover, summing the defects as reported in twelve sprints would make the defect count differences between consecutive sprints negligibly small. This metric measures work deliverable quality that is used to test H3.

Team productivity - To measure the productivity of the development team (H4) we sum the story points a team completes in a sprint: the so-called *velocity* [2].

2.3 Experiment Participants

We announced the experiment within our professional networks. Based on organizational details and selection criteria such as development sprint length and compatibility of the issue tracker with AQUSA, we invited 11 companies to participate and sent them the exact details of the experiment. 5 companies with headquarters in the Netherlands registered for participation. Unforeseen technical difficulties integrating AQUSA with firewalled enterprise editions of Jira reduced the number of companies to 3.

eCommerce Company is a large company with over 2000 employees. One team working on a next generation edition of the platform consisting of 6 developers, 2 UX designers and 3 project managers participated in the study.

Health Company is a medium-sized software producing organization that has multiple products for delivering digital healthcare. A team working on a new product consisting of 4 software developers, 1 software architect, 1 UX designer, 1 scrum master and 1 project manager participated in the study.

RealEstate Company is a medium-sized software producing organization that develops a product for a housing cooperative. The participating team of 11 employees has 6 software developers, 3 testers, one product owner and one product manager.

In total, 30 practitioners from six different countries participated in the experiment. The roles of these participants are: 16 software developers, 7 managers, 3 testers, 3 UX designers and 1 CTO. All the teams use Scrum as their primary software development

method and employ user stories. All practitioners report they capture user stories in the Connextra template *"As a ⟨role⟩ , I want ⟨goal⟩, [so that ⟨benefit⟩]"*, 11 of which ensure their quality by applying the INVEST framework, while 8 defined their own guidelines instead and the remaining 11 participants do not use any guidelines. Note that the majority of practitioners from RealEstate company indicated they did not use quality guidelines, while 2 selected INVEST and 2 others chose self-defined guidelines. For the eCommerce company, it was a tie between INVEST and self-defined guidelines with 4 each, followed by 3 saying they are not aware of any guidelines. The Health company participants are more in agreement with 6 employing INVEST and just 1 each choosing self-defined or no guidelines at all. The average participant has 3.1 years of experience in working with user stories while his or her organization has 4.7 years of experience. Concerning their expertise with user stories, 2 participants indicate they are at the novice stage, 5 are beginner, 15 are intermediate and 8 are advanced.

3 Results

This section presents the results of our multiple case study. We first look at the change in intrinsic quality between the pre-treatment period and the experimental period (Section 3.1). We then investigate the participants' perception of the effectiveness of the Grimm Method (Section 3.2). Finally, we analyze the software development process itself by comparing the metric computed on the basis of issue trackers data (Section 3.3).

3.1 Intrinsic User Story Quality

We collected all user stories created by the companies during the pre-treatment period and the experimental period. We analyzed them running AQUSA's defect detection algorithms. We observe a reduced number of defects in the second period in Table 1.

In particular, there are 38.3% fewer defects per user story (see the last row) for eCommerce company and 47.5% fewer defects for RealEstate company in the experimental period. The number of defects remained practically stable for Health company (one additional defect: 23 instead of 22) (4.5% more). Aggregating all the results, the post-period reveals 116 fewer defects for nearly as many user stories. The companies produced 241 user stories in the pre-treatment period with 266 defects versus 239 user stories with 150 defects in the experimental period. Due to the small number of defects for Health company, there is a substantial difference between the 27% macro-average and 43.14% weighted micro-average defect reduction [22].

Looking beyond the averages, the distribution of defects mostly remained the same for eCommerce and RealEstate companies, while they changed for Health company. The number of uniformity and uniqueness defects (almost) doubled whereas the number of well-formed and atomic defects (more than) halved.

The results suggest that applying the Grimm Method treatment generally had a positive effect on the intrinsic quality of user stories. On the other hand, the improvement is to be expected considering we measure quality with the exact same criteria as prescribed by the treatment. To further test the quality of the textual requirements, we considered using algorithms that assess single sentence complexity. Unfortunately, this field is still

immature [24]; while many metrics for readability of texts exist, they require passages of text with 100+ words. Nevertheless, we applied multiple readability metrics to our user story collections but found no substantial change between the pre-treatment period and the experimental period. For example: the *Gunning fog index*es [5] of our three sets are 11.0 vs. 10.8, 12 vs. 12.1, and 9.3 vs. 8.7, while the *New Dale-Chall* scores [1] are 5 vs. 5.3, 5.3 vs. 5.4, and 5.5 vs. 4.9 for eCommerce, Health and RealEstate, respectively.

	eCommerce		Health		RealEstate	
	Pre	Exp	Pre	Exp	Pre	Exp
Number of user stories	105	71	33	33	103	135
Defects breakdown						
- *Well Formed*: each story has at least a role and a means	63	27	7	4	33	22
- *Atomic*: each story expresses a requirement for *one* feature	10	1	6	2	6	8
- *Minimal*: each story contains only role, means, and ends	20	11	0	0	30	16
- *Uniform*: all written stories employ the same template	38	19	7	13	24	18
- *Unique*: each story is unique, duplicates are avoided	20	5	2	4	0	0
Total defects	151	63	22	23	93	64
Defects per user story	1.44	0.89	0.67	0.70	0.90	0.47

Table 1. Intrinsic user story quality analysis by AQUSA, comparing the pre-treatment period (*pre*) and the experimental period (*exp*).

3.2 Participant Perception

At the end of the experimental period each participant completed an exit survey on the perceived impact of introducing the Grimm Method. 27 participants submitted a complete and valid survey. Due to the limited sample size we mostly report on the aggregate answers of all respondents; a per-company discussion is presented at the end of the subsection. We first analyze the participants' responses, illustrated with quotes from 6 follow-up interviews. After perceived user story quality, we focus on the treatment's impact (H1–H4). Note that we use H#a to highlight questions that directly compare the pre-treatment and experimental periods, and H#b for questions on an individual period.

User story quality. The exit survey asks the participants to rate user story quality on a scale from 0-10 twice: first for the pre-treatment period and then again for the experimental period. Overall, the responses show a small user story quality improvement after applying the treatment: an average score of 6.96 vs. 7.15 with standard deviations of 1.34 vs. 1.29. In total, 9 respondents report a positive user story quality change after the experimental period, 12 report no change at all and 6 indicate the quality of the user stories decreased. However, when analyzing the distribution of responses via a Wilcoxon signed-rank test, we obtain no statistically significant difference between the two periods ($Z = -.984, p = .325$). Thus, concerning the introduction of the Grimm Method's QUS and AQUSA, we could not observe statistically significant changes in practitioner's perception of user story quality.

Yet, when asked to directly compare the pre-treatment period with the experimental period the responses are less neutral. The majority of respondents (16 or 59%) state they agree with the Likert-Type statement "The quality of our user stories is better since the

Grimm Method training", while 10 respondents neither agree nor disagree (37%) and just 1 respondent disagrees (4%). Also, 14 respondents (52%) agree with the statement "My satisfaction of our user stories is higher since the Grimm Method training", 11 neither agree nor disagree (41%) and 2 respondents disagree (7%).

In follow-up interviews, we asked respondents to motivate their answers and to explain why they did not report a change in the user story quality score, yet they do agree that the quality of their user stories improved. Two eCommerce Company and two RealEstate Company interviewees emphasize that their reported increase in user story quality was moderate, if present at all. They indicate that introducing the Grimm Method did not result in a meaningful, lasting process change, but that they themselves and their colleagues did become more aware of the relevance of capturing user stories in a diligent manner. The UX designer from eCommerce Company notes *"I believe that someone coming from outside the organization to tell us how you are supposed to do this has had the most influence. It aligns everyone's ideas on user stories. It helps that the method presents everything in a piecemeal fashion."*

These results reveal a discrepancy between the *intrinsic* and *perceived* change in user story quality. While the number of defects dropped by 43.14%, just 9 participants reported a (marginal) positive change in user story quality. Although respondents did respond positively to the Likert-Type statements, follow-up interviews clarified that the respondents do not believe user story quality improved substantially.

One possible explanation for this mismatch is AQUSA's focus on highlighting simple, easy to detect issues. This results in highly accurate yet seemingly trivial output. The importance of fixing a uniformity mistake, for example, can be difficult to comprehend, and those improvements may be regarded as too small to lead to an improvement.

Communication Frequency and Communication Effectiveness. We study participants' perception of conversation by breaking it down into two dimensions: frequency and constructiveness. Respondents do not report a significant change in H1b: communication frequency between the pre-treatment period and experimental period (see Fig. 2). 21 respondents did not change their answer after the training (78%), 4 reported a decrease in conversation frequency (15%) and 2 reported an increase (7%). The majority of respondents still experienced excessive communication around user stories; however, 3 out of the 4 participants from eCommerce Company that reported a communication decrease did no longer perceive the amount of conversation to be *excessive*.

The number of respondents that agree with statement H2b "The user stories contributed to constructive conversation concerning the software to be made" after the treatment grew from 14 to 18 (52% to 67% as per Fig. 2). Notably, the respondent that indicated he strongly disagreed with the statement for the pre-treatment period agreed with the statement after the treatment. When directly comparing the pre-treatment period with the experimental period, most respondents agreed that conversation was more frequent (H1a, 13 or 48%) *and* was more effective (H2a, 14 or 52%, see Fig. 1).

Again, the data exhibits a discrepancy. Participants self-report small communication differences between the pre-treatment period and experimental period, yet agree with statements that communication frequency and effectiveness improved after attending the Grimm Method training. Follow-up interviewees gave diverse motivations of their answers and clarifications of this discrepancy. The answers regarding whether conversa-

tion frequency increased or decreased after applying the treatment varied in particular. One respondent indicated the amount of conversation increased by up to 40% while another thought the amount of conversation decreased substantially.

Regardless of the increase or decrease, however, the interviewees agreed on the positive impact on conversation effectiveness. The same UX designer from eCommerce Company reported a positive change in communication frequency and effectiveness: *"Previously we lost a lot of time talking about trivial things. By trying to create better user stories as a team the discussion became more focused which resulted in more in depth conversation on why and how to approach a problem. Although this means more conversation it also saved time"*.

A software developer from eCommerce Company who contributed to the exhibited discrepancy explained his choice as follows: *"I think the conversation effectiveness did change positively, but the Grimm Method training made me more critical of what to expect from user story conversation in terms of effectiveness"*. This explanation highlights an unexpected phenomenon: although the treatment achieved its intended effect, it simultaneously influenced the way we measure that effect. These consequences cancel each other out, leading to a nullification of the impact measurement.

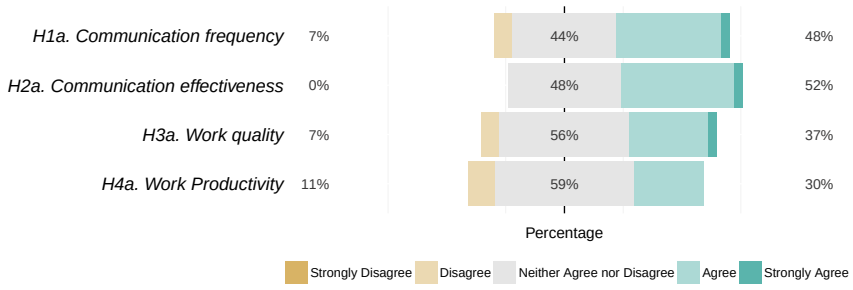

Fig. 1. Answers to Likert-Type statements directly comparing pre-treatment and experimental periods. The shown percentages (left-to-right) refer to Strongly Disagree + Disagree, Neither Agree nor Disagree and Agree + Strongly Agree, respectively.

Work Deliverable Quality and Work Productivity. Although respondents reported little change in work deliverable quality (H3b) after the experimental treatment in Fig. 1, 10 respondents (37%) agreed with H3a "The quality of my work deliverables is better since the Grimm Method training" in Fig. 2. In follow-up interviews, we asked respondents to clarify this discrepancy. Interviewees' responses were unanimous: the technical quality of the developed software did not improve, whereas the treatment clarified the goals of user stories thereby making it easier to develop software with less rework. This sentiment is illustrated by a software engineer from RealEstate Company: *"More effective and efficient conversation on user stories has reduced the amount of surprises later on. As a developer you have the responsibility to continue posing questions until you know what you are working on. The actual work deliverable quality is the same."*

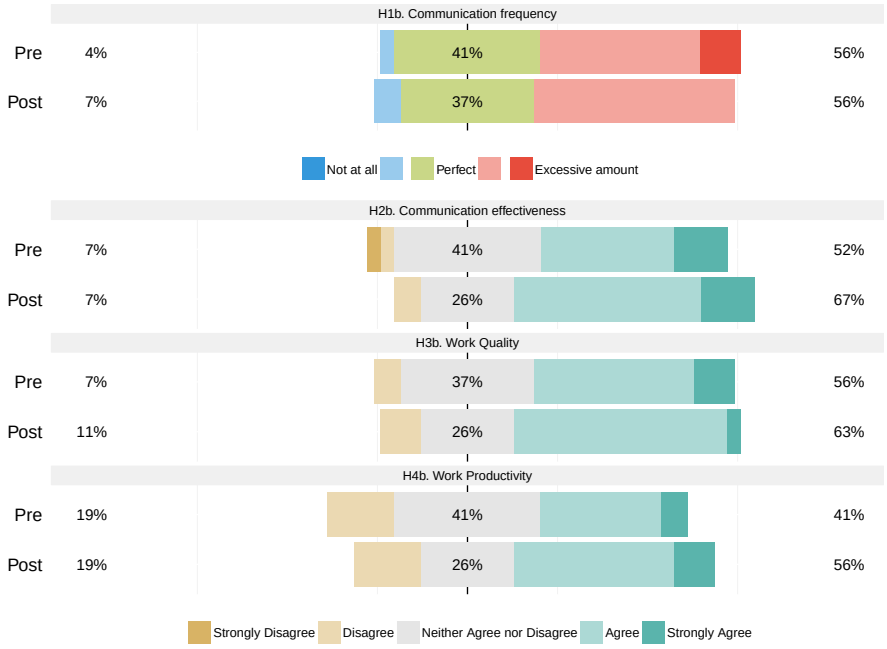

Fig. 2. Answers to Likert-Type statements for pre-treatment and experimental periods.

Responses were more consistent for work productivity. For H4b, 7 respondents indicated a positive change (26%), 16 did not change their answer (59%) and 4 experienced a decrease in work productivity (15%). Similarly, 8 or 30% of respondents agreed with H4a "My work productivity is higher since the Grimm Method training" (Fig. 2). Again, interviewees did not find user stories to have a direct impact on developers productivity because programming tasks do not substantially change due to better user stories. The interviewees, however, perceived more efficient processes around software development. The product manager of Health Company says *"I do not think the productivity itself increased, but we are more efficient in getting to the stage of being productive"* and a software developer from eCommerce Company *"I do not think my personal productivity increased, but for the entire team it did. As a team you can collaborate and communicate more efficiently and effectively which improves productivity"*.

Note that we found no correlation between a participant's role and his or hers perceived change in work deliverable quality and work productivity. It is likely, however, that this is due to the study's sample size. A future study at a larger scale could confirm our earlier results [12] that project managers perceive a more positive impact than software developers. Similarly, there is no statistically significant difference between the three case companies concerning their change in perception due to the limited sample size. Inspecting the Likert-Type graphs per company available online [11] reveals that:

– **RealEstate** Company employees clearly indicated a positive impact in all questions concerning both H#a and H#b.

– **Health** Company employees are mostly neutral concerning H#a: slightly positive on communication frequency (H1a) and communication effectiveness (H2a), and mildly negative on work quality (H3a) and work productivity (H4a). In H#b questions, they report no change concerning communication effectiveness (H2b) and work quality (H3b), and a minor positive change about work productivity (H4b).
– **eCommerce** Company employees are in between. They report both positive and negative change after introducing the Grimm Method, with more positive results concerning communication (H1a, H2a, H2b), slightly negative results on work quality (H3b), and neutral results for the other questions.

3.3 Software process metrics

The raw metric data is presented in Tables 2, 3 and 4, one for each participating company. The columns capture the following data: sprint identifier, number of comments, issues and defects within the sprint, number of post-release defects, recidivism rate and velocity. Note that we separate each table in two: the pre-treatment period corresponds to sprints 1-4 and the experimental period spans across sprints 5-8.

A visual, qualitative inspection of the results reveals three main concerns:

1. The total number of comments, issues, defects and velocity varies greatly between companies; compare, for example, the number of comments in the eCommerce company and in the Health company.
2. Some sprints include data outliers such as the number of defects in RE7.
3. The impact of the Grimm Method is likely to be small and can hardly be detected by visual inspection.

Sprint	*Issues*	*Comments*	*Def*	*Post Def*	*Recidivism*	*Velocity*
EC1	50	1	15	24	0.00	24
EC2	67	1	12	18	4.11	34
EC3	59	2	11	10	4.62	8
EC4	71	3	8	6	2.21	15
Treatment applied						
EC5	21	6	3	7	8.00	18
EC6	21	4	2	22	0.00	18
EC7	28	14	4	21	4.84	16
EC8	48	9	17	6	3.06	16

Table 2. eCommerce Metric Data

Sprint	*Issues*	*Comments*	*Def*	*Post Def*	*Recidivism*	*Velocity*
HE1	56	54	2	7	14.32	47
HE2	42	64	4	4	9.36	63
HE3	17	26	4	2	3.23	23
HE4	19	46	1	5	7.96	33
Treatment applied						
HE5	20	24	1	12	3.69	32
HE6	21	56	3	12	0.93	50
HE7	30	36	9	3	2.56	37
HE8	26	40	1	0	9.55	42

Table 3. Health Metric Data

To learn the actual impact of the Grimm Method we investigate whether the pre-treatment period and experimental period produce statistically different means. To reduce the numbers' variety we normalize them to a 0 to 100 scale, where 100 is the highest score per columns per company. For example: eCommerce Company's sprint EC7 has the most comments, 14, and is assigned 100. EC6 has 4 comments and gets normalized to $4/14 * 100 = 28.57$.

To test if the mean ranks differ between the pre-treatment and the experimental period we apply the Wilcoxon signed-rank test for non-parametric data (see Table 5).

Sprint	*Issues*	*Comments*	*Def*	*Post Def*	*Recidivism*	*Velocity*
RE1	130	5	9	15	3.57	18
RE2	136	7	1	15	4.05	14
RE3	137	10	4	15	8.23	7
RE4	117	5	2	44	6.67	8
Treatment applied						
RE5	113	12	8	106	3.31	22
RE6	96	2	18	94	7.14	18
RE7	268	18	59	85	8.13	33
RE8	230	9	20	27	6.77	22

Table 4. RealEstate Metric Data

	Issues	*Comments*	*Def*	*Post Def*	*Recidivism*	*Velocity*
Z	-1.177	-1.490	-.178	-1.557	-.275	-.628
Sig.	.239	.136	.859	.120	.784	.530

Table 5. Wilcoxon signed-rank tests

None of the tests produce a significant difference between the pre-treatment period and experimental period groups with all $Z < \pm 1.960$. This indicates that introducing the Grimm Method and Tool did not produce a statistically significant change in number of comments, issues, pre-release defects, post-release defects, recidivism rate nor velocity.

Although no statistically significant effect is observable, outliers in the data suggest that some change did occur. The outliers' significant divergence from the means make them eligible for removal in case the data has been unfairly influenced. For instance: the number of pre-release defects in EC7 is 3 times as high as the second highest value. We asked the in-charge project managers to clarify the outliers in their data. Their explanations reassured us of their validity. Two of the three project managers attribute the outliers to company-specific irregularities. In summary:

- The eCommerce company team lost productivity starting from EC3 due to internal discussions concerning the product direction.
- For the RealEstate company the merger of two project teams into one in RE3 resulted in productivity loss. Furthermore, an upcoming release in RE7 resulted in the reporting of many defects.
- The Health company project manager did notice a steady increase in productivity during the experimental period. However, he did not believe the Grimm Method has been the primary driver of this improvement, which he attributed to how the team started achieving what it is capable of after a period of under performance.

Yet, each project manager agreed that the frequency and effectiveness of user story conversation improved after the Grimm Method training. A replication of this study on a larger scale is necessary to confirm whether the phenomenon of experiencing a positive change without actively attributing it to the treatment is universal.

4 Related Literature

Although many evaluations exist on the impact of new RE methods, we could not identify any study where an RE treatment was experimented by comparing months-long periods with companies. A review of empirical papers on software process improvement [23] shows that just 8 out of 148 studies applied experimentation as their research method. None of these 8 both (1) relate to RE and (2) apply a pre-post comparison. However, the literature includes several closely related case study reports.

Kamata and Tamai investigated the relationship between requirements quality and project success or failure [7]. They analyzed 32 projects completed in a Japanese company which produces thorough quality reports for all requirements. They detected a weak relationship between SRS quality and project outcome, with five SRS criteria having a strong impact: overview, product perspective, apportioning of requirement, functions and purpose. Similarly, Knauss et al. found that in 40 student projects', success relates to the quality of the SRS they produce. For their specific context, they were even able to define a quality threshold that can be used to predict risk of failure [9].

Damian et al. introduced a formal RE process to the daily work processes of 31 project members of one Australian company [3]. They collected data over 6 months via a questionnaire, interviews and document inspection to measure practitioner's perception and development performance in terms of estimated effort vs. expended effort. Their analysis of the data indicated a positive effect of improving requirements management process in industry on downstream software development, especially in terms of more accurate estimations, improved project planning, and enhanced project scoping.

Sommerville and Ransom investigated whether improvements in RE process maturity lead to business improvements [21]. Over a period of 18 months, 9 case study companies incorporated advice on RE process improvement and self-reported on business key performance indicators (KPIs). After the experimental period, for each company both the RE process maturity and relevant business KPIs had improved. In spite of this, the authors could not statistically correlate the two due to incompatible KPIs.

Napier et al. explored the feasibility of an RE improvement process based on the RE Good Practice Guide that considers stakeholders' perception of which problems are most relevant [17]. The authors evaluate this method during a three-year action research process with one company. The results show a 69% increase in the number of implemented RE guidelines and unanimous positive perception by the participants.

Méndez Fernández and Wagner [16] explored the effect of the RE improvement approach *ArtREPI* that applies the RE best practice database AMDiRE. The impact of ArtREPI in two case studies is measured via two post-treatment questionnaires: one on the support of process engineers and one about project participants' rating of ArtREPI's output. ArtREPI meets practitioners' process improvement needs when problem and artifact orientation are important, but the authors call for larger-scale replications.

Our research incorporates elements from each of the aforementioned studies. We consider the relevance of little direct author involvement [16], allow the companies to choose themselves which quality criteria to apply [17], introduce our treatment to multiple companies [21] and collect both qualitative perception data and quantitative metrics [3]. Differently from Damian's study [3], our metrics include quantitative and qualitative data on indirectly related human processes around software development.

There are many other possible metrics. Holm et al.'s systematic literature review [6] provides an overview of how previous literature conceptualizes and operationalizes RE. Their examination of 78 studies reveals that in total researchers used 298 dependent variables corresponding to 37 unique classes and they find there is no agreement on how to measure RE success and unclarity in the choice of the variables. However, RE validation studies like ours do perform best: 60% of all dependent variables are of the type *defects found* and the majority of dependent variables include a motivation.

5 Discussion and Outlook

We studied the effect of applying the Grimm Method's QUS framework and the AQUSA tool into existing user story practices through a multiple case study. Although the number of user story quality defects decreased by 43.14% after applying the treatment, participants did not perceive a meaningful change in user story quality. Yet, the respondents agreed that communication frequency and effectiveness improved.

On the negative side, we found no statistically significant difference in communication frequency, communication effectiveness, work quality and work productivity perception between the pre-treatment and experimental periods. Furthermore, our study of the impact on work practices by measuring software process metrics (number of comments, issues, defects, velocity and recidivism rate) did not lead to statistically significant results, perhaps also due to organizational changes between the two periods.

Taking our results into account, we cannot accept our hypotheses H1–4 from Section 2. Further investigation is required to obtain more decisive results. However, the results make us hypothesize that improving user stories' intrinsic quality in itself is not essential, but that highlighting quality criteria defects seems to stimulate relevant and meaningful discussion around user stories (a key activity according to Cohn [2]). For example, the quality criterion *minimal* is seldom resolved by simply removing the text between two brackets, as the in-brackets text is often important: during Grimm Method workshops, defects of this type often indicated an insufficiently refined user story to be further discussed prior to assigning it to the sprint backlog. The consequence of these dynamics is not necessarily a higher quality user story or a direct increase in productivity, but a team that may more quickly agree upon the requirements.

Another explanation for the outcome is that the treatment is wrong, incomplete or even overcomplete. Although the QUS framework describes 13 quality characteristics, the individual relevance is unknown. Replacing or removing some criteria could improve the results. Also, in this study we focused on just the 5 (out of 13) characteristics that the AQUSA tool automatically detects. It would be interesting to conduct a study on all of QUS framework's criteria and to assess their individual impact.

Threats to validity. Multiple human factors should be considered and many aspects of the study design are hard to control for. Two important internal validity threats are the *Hawthorne Effect* and *good participant response bias* which causes participants that are aware of being observed to (sub)conciously modify their behavior. Related is the first *confounding* variable: simply instructing participants to pay attention to user story quality when creating them could be enough to explain the change in intrinsic user story quality. Additionally, there is a risk of *regression toward the mean*: if the user stories' quality was very poor, they would have improved regardless of the applied quality criteria. Although we tried to control for the latter two validity threats by selecting case companies with multiple years of user story experience, their relevance persists due to the absence of control groups. Note that all four of these threats could explain why we did not detect statistically significant positive changes, yet participants do agree that communication frequency and effectiveness improved after applying the treatment.

Our application of the treatment and measurement of the results introduce two other validity threats. After attending the training, the participants could have attempted

hypothesis guessing: knowing the desired result changes their actions. In a larger scale study it is possible to control for this threat by leaving participation in the study open ended and not informing participants of data collection. A large scale study also allows taking into account other potentially confounding variables such as the number of people in the project or the company size. For example, in the large eCommerce company, frequent team composition changes could influence the team's acceptance of new methods. Furthermore, there is an evident *history* threat: events outside of the researchers' control affect the outcome of the results. In the chaotic environment of software companies, many unforeseen events occur over a two month period that can affect the collected data. Although unavoidable in empirical research, substantially increasing the scale of the study is likely to reduce the impact of on the data.

Finally, there is a potential *bias in the experimental design*: the number of comments in Jira is not sufficient to measure the intended effect of increasing communication. In agile software development, much of the communication is verbal and informal. Unfortunately, it is extremely hard to accurately measure and record verbal communication.

Future work and outlook. The mildly positive participant perception is not confirmed by software development process metrics. As in similar studies [16,21], a large scale replication is necessary to generalize our conclusions, also to reduce threats to validity by controlling for confounding variables as company and project size. Additionally, we want to study the implications of specific quality criteria; what are the consequences of a minimality or uniformity defect in isolation? Also, we aim to devise tools that learn how to suggest improvements based on the most common resolution strategies.

Finally, we invite the RE community to undertake similar studies with the aim of measuring the *actual* effect of RE methods on work practices; our mixed results emphasize the importance of replication studies but also highlight some of the problems that other researchers could encounter in the evaluation of their own solutions.

References

1. J. S. Chall and E. Dale. *Readability revisited: The new Dale-Chall readability formula*. Brookline Books, 1995.
2. M. Cohn. *User stories applied: For agile software development*. Addison Wesley, 2004.
3. D. Damian, J. Chisan, L. Vaidyanathasamy, and Y. Pal. Requirements engineering and downstream software development: Findings from a case study. *Empirical Software Engineering*, 10(3):255–283, 2005.
4. C. W. H. Davis. *Agile metrics in action: Measuring and enhancing the performance of agile teams*. Manning Publications Co., Greenwich, CT, USA, 1st edition, 2015.
5. R. Gunning. *Technique of clear writing*. McGraw-Hill, 1968.
6. H. Holm, T. Sommestad, and J. Bengtsson. Requirements engineering: The quest for the dependent variable. In *Proceedings of the IEEE International Requirements Engineering Conference (RE)*, pages 16–25, 2015.
7. M. I. Kamata and T. Tamai. How does requirements quality relate to project success or failure? In *Proceedings of the IEEE International Requirements Engineering Conference (RE)*, pages 69–78, 2007.
8. M. Kassab. The changing landscape of requirements engineering practices over the past decade. In *Proceedings of the International Workshop on Empirical Requirements Engineering (EmpiRE)*, pages 1–8. IEEE, 2015.

9. E. Knauss, C. El Boustani, and T. Flohr. Investigating the impact of software requirements specification quality on project success. In *Proceedings of the International Conference on Product-Focused Software Process Improvement (PROFES)*, volume 32 of *LNBIP*, pages 28–42. Springer, 2009.
10. Kniberg, Henrik. What is Crisp? `http://blog.crisp.se/2010/05/08/henrikkniberg/what-is-crisp`, 2010. Accessed: 2016-05-25.
11. G. Lucassen. Experimental materials QUS and AQUSA evaluation. `http://www.staff.science.uu.nl/~lucas001/qus_aqusa_eval_materials.zip`, 2016. Accessed: 2016-10-02.
12. G. Lucassen, F. Dalpiaz, J. M. van der Werf, and S. Brinkkemper. The use and effectiveness of user stories in practice. In *Proceedings of the International Working Conference on Requirements Engineering: Foundations for Software Quality (REFSQ)*, volume 9619 of *LNCS*, pages 205–222, 2016.
13. G. Lucassen, F. Dalpiaz, J. M. E. M. van der Werf, and S. Brinkkemper. Improving agile requirements: The quality user story framework and tool. *Requirements Engineering*, 21(3):383–403, 2016.
14. L. Madeyski and M. Jureczko. Which process metrics can significantly improve defect prediction models? An empirical study. *Software Quality Journal*, 23(3):393–422, 2015.
15. T. J. McCabe. A complexity measure. *IEEE Transactions on Software Engineering*, SE-2(4):308–320, Dec 1976.
16. D. Méndez Fernández and S. Wagner. A case study on artefact-based RE improvement in practice. In *Proceedings of the International Conference on Product-Focused Software Process Improvement (PROFES)*, volume 9459 of *LNCS*, pages 114–130. Springer, 2015.
17. N. P. Napier, L. Mathiassen, and R. D. Johnson. Combining perceptions and prescriptions in requirements engineering process assessment: An industrial case study. *IEEE Transactions on Software Engineering*, 35(5):593–606, 2009.
18. P. Runeson and M. Höst. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2):131–164, 2009.
19. A. Schröter, T. Zimmermann, R. Premraj, and A. Zeller. If your bug database could talk. In *Proceedings of the International Symposium on Empirical Software Engineering (ISESE)*, pages 18–20, 2006.
20. E. Shihab, Z. M. Jiang, W. M. Ibrahim, B. Adams, and A. E. Hassan. Understanding the impact of code and process metrics on post-release defects: A case study on the Eclipse project. In *Proceedings of the International Symposium on Empirical Software Engineering and Measurement (ESEM)*, pages 4:1–4:10. ACM, 2010.
21. I. Sommerville and J. Ransom. An empirical study of industrial requirements engineering process assessment and improvement. *ACM Transactions on Software Engineering and Methodolology*, 14(1):85–117, 2005.
22. J. Tague-Sutcliffe. The pragmatics of information retrieval experimentation, revisited. *Information Processing & Management*, 28(4):467—490, 1992.
23. M. Unterkalmsteiner, T. Gorschek, A. K. M. M. Islam, C. K. Cheng, R. B. Permadi, and R. Feldt. Evaluation and measurement of software process improvement-a systematic literature review. *IEEE Transactions on Software Engineering*, 38(2):398–424, 2012.
24. S. Vajjala and D. Meurers. Readability-based sentence ranking for evaluating text simplification. *ArXiv e-prints: 1603.06009*, 2016.
25. X. Wang, L. Zhao, Y. Wang, and J. Sun. The role of requirements engineering practices in agile development: An empirical study. In *Proceedings of the Asia Pacific Requirements Engineering Symposium (APRES)*, volume 432, pages 195–209. 2014.
26. T. Zimmermann, R. Premraj, and A. Zeller. Predicting defects for Eclipse. In *Proceedings of the PROMISE 2007 Workshop*, 2007.