Challenges of Aligning Requirements Engineering and System Testing in Large-Scale Agile: A Multiple Case Study

Francisco Gomes de Oliveira Neto, Jennifer Horkoff, Eric Knauss, Rashidah Kasauli, Grischa Liebel
Chalmers and the University of Gothenburg
Gothenburg, Sweden
{gomesf,jenho,knauss,rashida,grischa}@chalmers.se

***Abstract*—As agile methods become more pervasive, agile practices are applied to more large-scale systems with a scope that goes beyond pure software. The expansion of agile in these contexts provides benefits, but creates new challenges. Widespread use of agile has changed the way we must think about practices both in Requirements Engineering (RE) and in System Testing (ST). Our experience shows that many challenges in the application of large-scale agile development relate to either RE or ST, and in particular to the alignment between these areas. In this paper we present large-scale agile-related challenges from a multiple case study which relate to REST alignment. We map our challenges to an existing framework for REST alignment, and make an initial attempt to suggest agile RE practices from the literature which may alleviate these challenges. Our results show that the interviewed companies need to first adopt more agile RE practices to enhance REST alignment and then leverage agile testing. Future work will look more towards evaluating these best practices.**

***Index Terms*—requirements engineering, system testing alignment, large-scale agile system development**

I. Introduction

Because of their benefits, agile methods, such as Scrum, have been extending their boundaries, both in terms of size, with application to increasingly large-scale projects, and in terms of scope, moving beyond pure software projects to more system-oriented projects [1, 2]. Incorporating agile practices and principles into other domains of development, besides software, has shown promise to different companies. Nonetheless, by moving towards a large-scale context with an increased scope and complexity, the adoption of agile methods produces several challenges for both processes of requirements engineering (RE) [1, 2, 3] and system testing (ST) [4, 5].

There are several trade-offs in RE for agile development with respect to the requirements artifacts (e.g., system requirements, non-functional requirements, system models) and the relation of those artifacts to customer value. Considering user stories, for instance, on one hand there is simplicity, but also limitations of effectively communicating requirements to off-site stakeholders. Similarly, the agile methods' emphasis on customer value can produce understated non-functional requirements [6]. Other challenges relate to longer lead times due to dependencies with hardware availability and interfacing with system engineering practices [6]. Such a combination of different disciplines and domains is a common source of challenges, since each contributes its own expertise and ways of communicating or performing activities.

In turn, different levels of testing in the V model (e.g., unit, integration and system) provide distinct challenges towards product development. Lower level artifacts provide feasible support to automated techniques since they are closer to the System Under Test (SUT) implementation, whereas, high level artifacts provide comprehensible information to different roles in the product development chain [7]. Thus, when adopting agile methods for testing in large-scale companies, one needs to even out the distinct ways that different experts communicate [5] in order to enable exchange of information and reach alignment between requirements and the related test artifacts (e.g., test cases, test reports, test executions) [8].

Others have explored the relationship between agile methods in testing [5] and RE [9, 4], as well as their alignment [4, 8]. Similarly, there are studies that investigate large-scale testing [10] and RE [2, 3]. Existing studies on combining large-scale agile and RE list several benefits to system development such as reduced development lead time and increased planning efficiency [6]. In line with these studies, we believe that exploring and exploiting the relationship between RE and test artifacts can foster agility within large-scale companies. Particularly, drawing on our case experience, we aim to answer:

RQ1: What are the RE-related challenges pertaining to large-scale agile testing in systems development?
RQ2: What are main challenges hindering alignment between requirements engineering (RE) and system testing (ST)?
RQ3: What existing requirements engineering practices could improve alignment between RE and ST?

We explore these research questions in a multiple case study with three large companies based on focus groups with topic matter experts. Here we report initial results of our work-in-progress study. We begin by investigating RE-related challenges stated by each company during our data collection to address RQ1. Then, we use that data to identify (RQ2) RE and ST (REST) alignment challenges for each company, followed by (RQ3) practices which address these challenges. For RQ2, we use a list of REST alignment challenges observed in studies with other companies from Bjarnason et al. [4];

whereas for RQ3, we use Inayat et al.'s systematic literature review (SLR) [9], which includes a list of 17 agile RE practices from empirical studies described in 21 papers. Note that our assumption is that agile RE practices foster agility within large-scale RE [9, 6] and testing [4, 8], hence contributing to REST alignment. We provide two main contributions:

- We identify a variety of challenges related to both RE and ST, and propose solutions to help stakeholders address those challenges.
- We find the challenges [4], and practices [9] identified in related work were applicable to large-scale agile system development in our investigated companies. Therefore, we offer strong indication that existing results on RE and Software Testing alignment can be generalized to large-scale agile system development.

Our conclusions show that the investigated companies face similar challenges regarding large-scale agile system development and, in order to leverage agile testing, the companies need to improve requirements management, prioritization and continuous planning. In other words, by encouraging agile RE practices, companies can leverage their large-scale agile testing, as requirements information can then be used to automate tests (e.g., test generation or selection), supporting both the product and the development teams. Even though our findings are limited to the companies involved, we use our experience in industrial cases to give insights on these questions, framing future work.

Section II presents existing research comprising large-scale and agile methods pertaining to both RE and testing, followed by methodology in Section III. We discuss our results concerning REST alignment (Section IV) and expose the limitations in our study (Section V). Finally we draw conclusions and describe future work in Section VI.

II. Related Work

A. REST alignment frameworks

REST alignment is the process of adjusting RE and testing activities to coordinate and optimize product development [8]. Often perceived as very similar concepts, we refer to alignment and traceability as two different instruments, where traceability is related to the connection of artifacts through trace links, and alignment includes coordination of practices, artifacts, and roles [8, 4]. Therefore, traceability becomes one element of alignment to document the relation between, for instance, requirements specification and test cases. There are several benefits in investing in the alignment between RE and testing, such as improved requirements coverage, risk management [11], and better communication between stakeholders [12].

Bjarnason et al. [4] perform a systematic literature review (SLR) to collect the set of proposed REST alignment challenges reported in literature. Unterkalmsteiner et al. [8] advance the contributions towards REST-alignment into a taxonomy and a process linking the respective areas through dyads and network constructs that can identify opportunities to improve REST alignment. At this stage in our case studies, we could not collect enough details on each company's tests, requirements artifacts and tacit knowledge to build a dyad network and benefit from Unterkalmsteiner et al.'s taxonomy [8]. Instead, we begin by identifying specific challenges and hope to apply the taxonomy in future work as more data collection is performed with practitioners.

Regarding practices to improve REST alignment, Uusitalo et al. [13] interview six companies and present alignment practices aiming to improve communication and interactions between different testing and RE roles, whereas Kukkanen et al. [12] proposes a set of practices to align RE and ST such as use of metrics and traceability with tool support. In our analysis, we use agile RE practices presented in an SLR by Inayat et al. [9] to address REST alignment challenges. Their list of practices focuses on supporting agile RE instead of REST alignment, thus providing a new viewpoint to existing analyses, especially considering large organizations. Below, we briefly describe their 17 practices (in italics), but refer to their SLR for a more detailed description.

Face-to-face communication refers to communication between clients and team members, emphasizing minimization of documentation. *Customer involvement and interaction* involves identification of appropriate customer contacts or representatives to extract requirements. *User stories* are the agile form of requirements specification, while *iterative requirements* refer to an emergence of the requirements over time via frequent interactions with stakeholders. During each agile iteration, *requirements are prioritized*, often by customers with a focus on risk and value.

Change management is a broad category in traditional RE, but in the agile context refers to adding or dropping features and refining requirements via frequent client communication. *Cross-functional teams* combine team members such as testers, designers and product owners, helping to alleviate communication gaps between roles. *Prototyping* allows for quick feedback for high-priority requirements, while *testing before coding* means writing tests before starting to implement functionality in code. *Requirements modeling* in an agile setting is more lightweight compared to traditional RE modeling, focusing on easy-to-read sketches, such as goal-sketching. *Requirements management* in an agile setting refers to maintaining a product backlog, feature list, or index cards, while *review meetings* continuously review the status of the backlog, and acceptance tests provide a pass/fails results for individual user stories.

Code refactoring has the typical software meaning, particularly important in light of changing requirements. *Shared conceptualizations* refer to the shared understanding of agile teams, fostered by frequent communication and necessary due to reduced documentation. *Pairing for requirements analysis* refers to a stakeholder performing multiple roles, similar to cross-functional teams. *Retrospective* meetings review the work and find new requirements after an iteration, while *continuous planning* refers to the general agile way of working, continually re-planning and adjusting to new requirements.

B. Large-scale agile RE and testing

We use the term *large-scale agile* to refer to large organizations adopting or implementing agile methods in their system development process [6]. When considering agile methods, usually researchers relate improvements to a "traditional" approach, such as agile RE versus traditional RE. The same applies to testing. Adopting agile RE and testing methods has lead to benefits for companies, such as decreasing process overhead and increased flexibility for change [6].

However, there are both organizational and process changes involved when becoming agile, transitioning from a traditional waterfall development. From an RE perspective, the roles of requirements must change to emphasize customer value and avoid overwhelming specification documents. For instance, in their SLR, Inayat et al. [9] map challenges from traditional RE that can be addressed by their list of agile RE practices. They further provide a list of 8 challenges from agile RE itself that can also be addressed by agile RE practices. In our research we see that many of those practices can, in fact, help companies overcome large-scale agile REST-alignment, especially since Inayat et al. emphasize both test and requirements artifacts in their practices, even though they primarily focus on RE.

In turn, testing also has its own set of challenges and solutions pertaining to agile methods. ST relies on interaction with the SUT in order to provide verification and validation (V&V) assessment to stakeholders [5]. Test cases represent that interaction, and can be done within a continuum, having automatic and manual activities as opposite ends, each with their own trade-offs. Even though automation is one of the goals when adopting agile testing [5], most processes do not support it since requirements and development artifacts are often scattered among, or not properly managed by, stakeholders.

Both RE and testing can be found in existing processes that scale agile, such as the Scaled Agile Framework (SAFe)[1] and Large-Scale Scrum (LeSS)[2]. SAFe is composed of 4 to 5 levels (team, program, value stream, portfolio and foundation), each with its own purpose such that, combined, they support 4 core values (alignment, built-in quality, transparency and program execution) to promote a lean-agile mindset. In turn, LeSS is similar to Scrum, but instead, shifts the team's focus from "my part" to the product as a whole so that all teams act as a team itself, thus providing stakeholders with guidelines on how to coordinate activities within and across teams in order to foster sharing and cooperation. The participating companies do not use SAFe and LeSS rigorously. Thus, we did not consider either frameworks in this study, but plan to explore them in future work with the same companies.

III. Methodology

We perform a case study with the objective of exploring and describing the RE challenges that companies face when moving towards large-scale agile. For each company, we conducted interviews with focus groups (FG) as part of a workshop. In this way, we collect data regarding their artifacts and how they are used. Since this is an exploratory case study, our objective, at this stage, is not to assess the quality of the artifacts and methods that the companies apply in their software development, rather we use that information to describe each case and point to particular aspects of their RE or testing processes.

TABLE I
Our case study planning according to guidelines presented by Runeson and Höst [14].

Objective	Explore
The context	RE and testing in large-scale agile system development
The cases	Telecommunication case (FG = 2 people) Manufacturing case (FG = 7 people) Automotive case (FG = 5 people)
Theory	Challenges in REST-alignment [4] and agile RE practices [9].
Research questions	RQ1, RQ2, RQ3 (see Section I)
Methods	Semi-structured group interviews
Selection strategy	Large companies adopting agile methods

We conducted the workshops with three companies from different domains: i) telecommunication (Tele), ii) package manufacturing (Manu) and iii) automotive (Auto). All are large companies developing products and systems that include software, hardware, and mechanical components. Even though all companies are implementing agile methods, they are in different stages of that implementation. Prior to the workshops, each company was asked to prepare: i) one or more projects to discuss, ii) a set of artifacts to show and discuss during the workshop, and iii) a group of practitioners, that we refer here to as a FG involved in RE and testing activities to participate in these interviews. Table I presents our case study's planning.

In summary, we conduct a multiple case study, with three participating companies, where each company is studied within their corresponding case. We decide to consider only one unit of analysis per case because the information collected from the FG is self-contained, i.e., cannot be split into distinct units of analysis as we did not interview groups from different projects within each company, separately. We summarize our findings by analyzing cross-cutting concerns identified in all companies from an REST-alignment perspective. The FG for each company included participants with the following expertise:

- *Tele:* system engineering (1), system testing (1);
- *Manu:* system engineering (4), system testing (2), requirements engineering (1);
- *Auto:* in-house software development (2), software architect (1), system (1), electrical and hardware design (1).

We used semi-structured interviews where, in a three-hour workshop session, we established a dialogue with the company FG. Each workshop was hosted by the companies in their respective workplaces. We used an instrument[3] with a series of forms and questions to guide the dialogue. Using a funneling

[1] http://www.scaledagileframework.com/

[2] https://less.works/

[3] Available at https://goo.gl/PrjZW5.

principle [14], we begin by questioning them about their requirements artifacts and then we narrow the questions down to specific aspects of test activities.

Nonetheless, during those dialogues, the subjects were allowed to diverge from the questions in order to explain specific aspects of their company's apparatus and *modus operandi*. This step was meaningful to our interviews to collect, particularly, data on the tacit knowledge from stakeholders.

After preparation and planning of the workshops, we began our data collection phase by iterating through one instance of the same instrument per company and recording the audio from the workshop. The workshops happened in different weeks, according to the practitioners' availability. After conclusion of data collection, we started to analyze the data using i) workshop audio files, ii) notes taken by the participants, iii) documents sent by practitioners that were presented by them as requirements or testing artifacts. We coded the data in order to classify and cluster the artifacts and challenges reported by the companies. Examples of codes which emerged include *V&V planning*, *levels and decompositions*, *test as requirements*.

A. Description of RE and ST in each case

Most of the artifacts collected are closer to the requirements/ system level. The goal, for each case, is to present the challenges for each company with respect to the collected artifacts or explanations from practitioners (presented as quotations below), helping us to answer RQ1.

B. Telecommunication

Tele is experienced with software development, and its teams have been using agile methods, such as Scrum, for years, such that it created a Scrum variant for their system development. However, long lead times due to dependencies on hardware components still hinders agility in their development and testing. The data was collected from a project with an agile software development team.

Most of the testing activities rely on an artifact named *test report analysis*, that includes guidelines on how developers should create test artifacts and the test infrastructure for a specific system feature. This artifact is created during the planning stage of a feature by test architects. Existing guidelines and templates to create this artifact are not always followed and updates on the features are not necessarily reflected on this artifact, hence reducing its REST alignment. The reason behind sporadic guideline compliance, with the test report analysis artifact, is that a team member is sometimes encouraged to take ownership of features leading to a trade-off, where on one hand he or she reduces their documentation effort (i.e., they no longer need to create a thorough test artifact), and on the other hand the team becomes dependent of his/her involvement.

> *". . . there we have somewhat a challenge, one is developing faster and having some functionality and for some requirements, we need to keep changing . . . and potentially we can also break this [test analysis report] for test for another product, when we are changing for other product."* — Tele

That ownership is less encouraged in their cross-function development teams, where roles are not explicitly assigned and, ideally, anyone can assume the role of, for instance, a tester and create test cases. However, in practice, some developers still take ownership of the tester role and focus their time on test activities.

The test report analysis is used by development teams to create test cases at, predominantly, the unit and integration level. Testers also do some exploratory testing at the system level, however, they comprise a smaller portion of their test set. Conversely, they are still hard to automate and require human interaction (i.e., manual execution). In turn, tests for non-functional requirements, such as regulatory requirements, are executed separately and independently of functional tests. That enables independent test reports of products that need to comply with similar non-functional requirements but, in fact, have distinct functional requirements.

> *"For the functional tests the pass/fail is the primary measure . . . then for the non-functional tests there are more specific reports."* — Tele

Additionally, there is a i) variety of toolkits to assist developers with unit testing and ii) a test infrastructure for continuous testing during integration builds where test cases can be scheduled differently as well (e.g., daily, weekly runs). One of the main factors enabling this infrastructure is traceability on code level. However, on higher levels of abstraction, such as for features, traceability becomes more difficult, since updated requirements and many-to-many relationships between features and test cases need to be maintained.

> *"There are more specific tests that we call them multi-feature test . . . more typically relating to multiple features."* — Tele

C. Manufacturing

This company aims to become more agile throughout its different teams, and, supported by management's initiative to introduce tools and processes, stakeholders are still converging towards standardized usage of tools and templates for specification documents. Therefore, only some teams in the company use tools that support traceability. However, for the other teams, most of the information transfer across artifacts is done manually by consulting distinct files.

The existing template documents are designed to ensure some alignment between test specification and execution. For instance, they use a *design verification plan and report* (DVP&R) artifact that has fields so that engineers specify test configurations and group them in order to optimize the test setup during test execution, since test sessions involve physical components and are costly. The artifact enables stakeholders to verify technical feasibility and dependencies of the testing activities.

> *"That was the idea when I showed you the matrix in the beginning . . . So this is preventing you from doing the same test several times by accident."* — Manu

In addition to the DVP&R, they use test specification templates that require system engineers to manually fill out numerous fields, such as test responsibility, goal, resources, and test steps. In addition, the same document has fields to

report on corresponding test executions. As a consequence, it is unfeasible to align the test specification with the system requirements, rather the DVP&R should bridge them.

D. Automotive

Similar to Manu, Auto is also adopting agile throughout its many teams (hardware, mechanical, electrical, etc.), even though software development has been predominately using agile methods, such as Scrum variants. One of the main challenges stated by the stakeholders is the variety of usage of the requirements artifacts across teams for system requirements and how they relate to testing. As a consequence, Auto wants to improve process consistency to handle requirements and testing within its team throughout its department, due to the variety of disciplines involved in the system development (e.g., electrical, mechanical and software engineering).

> *"We do work in very many different ways of the requirements both in terms of level and what [system] areas we cover, and since the software very much covers what they need . . . we are not taking care so well of needs from other systems."* — Auto

As an example, one of the stakeholders belongs to a team that uses an intermediate documentation to bridge unit tests and design information, but was not able to effectively align those unit tests to the system requirements.

> *"We call it 'documentation' because it's not requirements, since they are not tested by our testing group. We refer to them when we unit test, so it's possible for us to trace back what we have done in our unit tests. . . . but to not confuse anyone else, we call it 'design documentation'."* — Auto

In order to foster consistent communication across teams, the practitioners suggest creation of interfaces between hardware and software teams, to provide both perspectives and enable cross-functional development across those two levels. In addition, they argue that their REST alignment is existing but limited (for example, towards regression testing), i.e., that it is effective, but more expensive/time-consuming than necessary. Currently, there is impact analysis when changing requirements, but the company wants to also improve efficiency in selecting and executing non-changed requirements as part of their regression test sessions.

> *"How to work with development verification together with regression testing? That is something that we should have; this change structure to steer up verification on new development. But we have to have a product structure to steer up regression testing, I think, that is one idea."* — Auto

Ultimately, Auto identifies that stakeholders want to leverage alignment between requirements and tests, especially to improve automation. The challenges for Auto are; however, to determine and distinguish which information separates tests from requirements and design.

> *"Test cases could serve as means to discover what the system should do, since we do design documentation more to document what it can do, then the tests are the description of what it should do, and I think that's how we can live with design and tests, rather than requirements and design as we do today."* — Auto

IV. Alignment challenges and agile RE practices

With respect to RQ2 and RQ3, Table II contains the alignment challenges as presented by Bjarnason et al. [4] and the corresponding company that stated facing such challenges. Challenges *Ch3*, *Ch4*, *Ch6* and *Ch7* comprise, respectively, requirements specification quality, ST quality, requirements abstraction levels and traceability. Note that the unchecked cells in Table II indicate that we cannot confirm, based on the collected data, whether that the corresponding company faces that particular challenge or not.

For each challenge listed, we present: i) a description of the challenge (for detailed description refer to Bjarnason et al. [4]), ii) the risk of not addressing the challenge, iii) the observations collected from the interviewed companies and iv) a brief discussion on how recommended agile RE practices can help to address the corresponding challenge in order to improve REST alignment in our context.

Ch1: Sometimes, practitioners feel that goals are missing or unclear, which could result in requirements misunderstanding, lack of insight into and awareness of different perspectives. **The risk:** Questionable decisions and costly requirements changes at a late stage in the development cycle. **Results:** For our cases, this is one of the main challenges to scale RE in agile because agile methods emphasize reactivity and informal communication [6, 4] that are hard to guarantee across multidisciplinary teams within a large organization. **Agile RE practice(s):** Suggested practices for this challenge should encourage communication and interaction across teams. The companies already use *cross-functional teams* with satisfactory communication. We suggest to also strike a balance between *face-to-face communication* and *requirements modeling* since focusing on them alone is a hindrance for, respectively, large-scale RE and agile RE.

Ch2: At the product level, weak cooperation negatively affects the alignment, specially at team and organizational boundaries where cooperation between people is required. **The risk:** Increased lead times, additional rework, and conflicts in resource allocation between projects. **Results:** In our data collection, Manu and Auto expressed concerns in finding isolated resistances across some teams when transitioning towards agile methods. Although teams do collaborate, the adoption of tools, templates and processes requires communication among stakeholders. **Agile RE practice(s):** Those companies have already started to use *cross-functional teams* to increase communication and interaction between stakeholders (especially in their development teams). By involving, as well, different roles and expertise in product development they will be able to leverage knowledge sharing to facilitate communication.

Ch3: Requirement specifications that are difficult to change and verify may indicate that information is missing, hindering developers and testers in writing and testing the software. **The risk:** Increased testing effort and the risk to misinterpret and fail to deliver customer value, which is essential for agile RE. **Results:** All case companies reported challenges in dealing with their requirements specification. The difficulty is to create testable requirements[4] that are easily updated and reflect changes to the test artifacts. **Agile RE practice(s):** A

[4]Our interpretation is to have requirements specifications that one can easily derive, and ideally execute, test cases from.

TABLE II
RESULTS FROM DATA COLLECTION RELATING THE CHALLENGES IN REST ALIGNMENT [4] WITH AGILE RE PRACTICES [9].

Id	Challenges in REST alignment	Tele	Manu	Auto	Agile RE Practices
Ch1	Aligning goals and perspectives within an organization		X	X	Face-to-face communication, req. modelling
Ch2	Cooperating successfully		X	X	Cross-functional teams
Ch3.1	Defining clear and verifiable requirements	X	X	X	Change and req. management, acceptance tests
Ch3.2	Keeping requirements documents updated	X	X	X	
Ch4.1	Full test coverage	X		X	Req. prioritization, management and continuous planning
Ch4.2	Defining a good verification process		X		
Ch4.3	Verifying quality requirements		X	X	
Ch5	Maintaining alignment when requirements change	X	X	X	Req. management, continuous planning, shared conceptualizations
Ch6.1	Defining requirements at different abstraction level	X			Iterative requirements, req. management
Ch6.2	Coordinating requirements at different abstraction levels	X	X	X	
Ch7.1	Tracing between requirements and test cases	X	X	X	Continuous planning and review meetings
Ch7.2	Tracing between requirements abstraction levels	X	X	X	
Ch8	Time and resource availability	X	X		Req. prioritization
Ch9	Managing a large document space	X	X	X	Req. management, prioritization and continuous planning

combination of practices apply, including *change management*, *requirements management* and *acceptance tests*. Requirements and change management help stakeholders to drop/add features while minimizing change impact. Along with traceability to acceptance tests, those practices enable team members to cope with the dynamic nature of agile RE, since the effort in maintaining links between tests and requirements information can be delegated to tools instead of people.

Ch4: Similarly to Ch3, poor quality in test artifacts and activities (e.g., unclear or incomplete test artifacts or process definition) affects alignment to achieve full test coverage, verification of quality requirements, as well as compliance to a verification process. **The risk:** Increased cost and effort when dealing with late requirements changes, since testers may not know how to promptly verify the changes and assess full test coverage. **Results:** Test coverage and non-functional tests become particularly challenging, since the information can be scattered between artifacts and teams in the organization. **Agile RE practice(s):** Ideally, *testing before coding* should be feasible for most large-scale agile projects. However, dependencies of modules and sub-systems across teams can be a hindrance for test case maintenance, such that *requirements prioritization, management and continuous planning* can, instead, leverage V&V planning and reduce test effort, especially for regression test sessions.

Ch5: This challenge is related to consistent updating of RE and ST artifacts, and being able to assess the impact of changes. This effort to keep consistency is important when requirements are actively used. Otherwise developers and testers may need to search for artifact information from other sources. **The risk:** Wasted effort in development or testing activities along with inconsistent deliverables, either because the developers are not aware of change or because the tests no longer validate what is actually described in the updated requirements. **Results:** Companies do not yet use techniques for REST alignment [4, 8], even though they want to, particularly, improve and automate regression and NFR tests. **Agile RE practice(s):** The suggestions here overlap, to some extent, with **Ch3** and **Ch4**, since stakeholders need diligence to adopt *requirements management*, *continuous planning* and *shared conceptualizations* [9] to bridge both ends of the V model across the system's levels.

Ch6: While Ch5 addresses consistency between deliverable and artifacts, Ch6 includes the consistency of the information being represented in the requirements and test artifacts at different abstraction levels, in order to enhance downstream coverage in the V model. **The risk:** Increased test effort and costs to validate specific requirements, and the difficulty in selecting tests for subsystems or components being developed by different teams. **Results:** As mentioned in Section III-A, all companies report on challenges about levels and decomposition of their requirements from customer requirements to system requirements and downstream through the V model. **Agile RE practice(s):** *Iterative requirements* along with *requirements management* can help team members alleviate the cluttering and gauge large-scale agile requirements engineering. For example, having requirements emerging over time via frequent interactions with stakeholders allows test prioritization aligned with a prioritized backlog.

Ch7: Creation and maintenance of trace links between requirements and test cases is costly. This cost involved in both introducing and maintaining traceability often compensates for additional costs introduced by lacking traceability. **The risk:** Increased effort in assessing change impact, as developers and testers will be tricked into guessing connections between artifacts, again propagating inconsistencies in the process. **Results:** Traceability is a challenge for all interviewed companies, even though they have guidelines and tool support to provide trace links to their artifacts. The main part of the problem is adopting the corresponding tools and the habit of creating and maintaining the trace links in the tools. **Agile RE practice(s):** *Continuous planning* and *review meetings* can encourage diligence towards maintenance of trace links, since developers and testers would need to refer to their status during each retrospective [9].

Ch8: The alignment between the demanded functionality and quality levels expected of the products versus the amount of testing that can be performed. **The risk:** Unnecessary

increase in costs or, in case resources for satisfactory test coverage are unavailable, the inability to properly verify and/or validate the product. **Results:** This was explicitly discussed by Tele and Manu, particularly regarding costs for test execution, as testing sessions usually require allocation of expensive competences or hardware/mechanical components, thus increasing the testing costs. **Agile RE practice(s):** In order to apply existing test case selection techniques to cope with resource constraints, stakeholders need to do *requirements prioritization*, so that test cases can be automatically selected. Most automatic techniques assume, however, traceability between test and requirements since manual selection can be costly and error prone [15].

Ch9: If not managed properly, the information in requirements or test databases will become redundant and unnecessary. **The risk:** Inconsistent (or even conflicting) information that accumulates when that information is updated, hindering the creation of baseline versions (and ultimately affecting regression tests). For instance, outdated test cases may trigger failures related to obsolete requirements rather than actual faults in the SUT. **Results:** This was reported by all companies: agile methods discourage comprehensive documentation [6, 4], while regulation and legacy requirements seem to make such documentation necessary. **Agile RE practice(s):** Bjarnason et al. [4] recommend *user stories* to address the challenge of traditional requirements documentation along with *requirements modeling and management* to address the need for shared information across teams in large-scale companies. There is some connection to **Ch7** as well, since traceability tools can facilitate maintenance of this document space, even if smaller.

A. Summary and discussion

Based on the collected data (Section III-A) and our analysis on its REST alignment challenges, we answer our research questions. Regarding RQ1, we confirm the findings of existing work by identifying the main *RE-related challenge* from companies as the downstream alignment between requirements and tests. In addition, we summarize the main RE-related challenge identified in each company.

Tele has stated that its challenge is to better align three elements: requirements specification, test cases and the platform responsible for test execution and management. The outcome is to benefit from updates in requirements and automatic selection of test cases to optimize requirements coverage. In turn, Manu's challenges are mainly related to consistent usage of their tools and document templates, since the resulting improvement on traceability can address their issues with the amount of manual work involved in creating and managing test specifications. Auto's challenges are to bridge agile software development and requirements management, by, e.g., representing higher level information as interfaces to improve communication among different experts (e.g., software and mechanical engineers), and build infrastructures with baselines that should be constantly tested.

When analyzing the *REST alignment challenges* to answer RQ2, we saw that *the main challenges, for the companies, are* (Table II): requirements quality (Ch3.1, Ch3.2), alignment maintenance (Ch5), abstract levels (Ch6.2), traceability (Ch7.1, Ch7.2) and management of requirements documents (Ch9). Surprisingly, the major challenge is related to RE rather than test artifacts (Ch3.1, Ch3.2 and Ch 3.3), since most companies have to first move towards large-scale and improve their RE processes so that, then, they can benefit from that in their testing activities.

Answering RQ3, *the most recommended practices* for the companies are emphasizing requirements management, prioritization, and updates, along with continuous planning. Requirements management can be a rather general practice, but here we emphasize that using tools with traceability and versioning support yielding two main benefits: the allow stakeholders to automate the maintenance of requirements information, and to reuse that information to leverage automated and agile testing [5].

V. Threats to validity

We relate our choice of theory and case (see Table I in Section III) towards threats to construct validity. In order to comply with existing constructs we refer to existing practices and challenges that have been validated in previous studies on large-scale agile system development, i.e., [4, 8]. There is still a risk that the constructs are unsuitable due to the reliability of their original studies. However, we observe consistence in the identified challenges throughout our and the original studies.

Similarly, external validity depends on the extent that we can generalize our findings, given that the constructs come, also, from case studies that might have low external validity, since the cases' conclusions are limited and still specific to a company or project within that company where data was collected. In addition, all three companies are from Sweden. Nonetheless, our objective is exploratory at this stage, and we intend to expand towards generalization by including other frameworks (e.g., LeSS and SAFe) and, possibly, more practitioners in future studies.

The main internal validity threat relates to the interviews performed during the workshop, since we allow the FGs to diverge slightly from the schedule and the semi-structured interview, in order to keep an open dialogue and allow description of tacit knowledge for practitioners in each company. We mitigate this threat by using consistent instruments and methods throughout the different workshops and analyses. Ultimately, we cannot emphasize significantly control of data collection, since this is a qualitative study and we assume that each company has its own peculiarities, as a consequence of their disparate domains.

Regarding reliability, there are several limitations in our study, specially since data sharing is limited due to non-disclosure agreements with the participating companies. Conversely, two points improve reliability of our study. First, all authors in this paper were involved in this study, particularly during data collection. In addition, we planned so that not all

authors would attend all workshops, hence reducing the risks of bias when conducting the interviews, or coding the results. Second, we re-use existing frameworks from related work, thus other researchers aiming to reproduce our study can refer to those additional sources for any extra information needed on our constructs.

VI. Conclusions

We investigate REST alignment challenges [4] in three large companies currently moving towards large-scale agile system development. Based on data collected in FGs with each company, we discuss how existing agile RE practices [9] can help stakeholders to overcome their challenges to improve REST alignment. Our results show that the companies need to first address many RE related challenges of large-scale agile system development, including consolidating creation and maintenance of requirements artifacts, in order to enable REST alignment, eventually leading to better management of test artifact and automated test activities [5]. In addition, the feasibility of REST alignment is influenced by the maturity of the company's test processes and activities. Ultimately, REST alignment and automated tests allows team members and stakeholders to welcome changes, improve communication and minimize verification costs [4, 8].

A limitation of our study is that we used one set of REST challenges and agile RE practices, when several frameworks could have been included in our study. Nonetheless, our conclusions enable future work to include and compare different frameworks. Even though the REST alignment challenges refer to alignment between RE and software tests, our contexts includes system testing and still several challenges were observed and reported by the companies.

Besides expanding the theory framework in our case study by including other lists of challenges and practices, we plan to analyze the data from an agile testing perspective focusing on agile testing challenges and practices reported by Crispin and Gregory [5]. In addition, we are currently working to collect more data from the companies to elicit further concrete RE practices that can be implemented and evaluated for use in large-scale system development.

Acknowledgments

We thank all participants in this study for their great support, deep discussions, and clarifications. This work was supported by the Chalmers Software Center Project 27 on RE for Large-Scale Agile System Dev. and the SIDA BRIGHT project.

References

[1] K. Dikert, M. Paasivaara, and C. Lassenius, "Challenges and success factors for large-scale agile transformations," *Journal of Systems and Software*, vol. 119, no. C, pp. 87–108, Sep. 2016.

[2] L. Lagerberg, T. Skude, P. Emanuelsson, K. Sandahl, and D. Ståhl, "The impact of agile principles and practices on large-scale software development projects: A multiple-case study of two projects at ericsson," in *ACM / IEEE Int. Symposium on Empirical Software Engineering and Measurement*, 2013, pp. 348–356.

[3] T. Dingsøyr and N. B. Moe, "Research challenges in large-scale agile software development," *SIGSOFT Software Engineering Notes*, vol. 38, no. 5, pp. 38–39, 2013.

[4] E. Bjarnason, P. Runeson, M. Borg, M. Unterkalmsteiner, E. Engström, B. Regnell, G. Sabaliauskaite, A. Loconsole, T. Gorschek, and R. Feldt, "Challenges and practices in aligning requirements with verification and validation: A case study of six companies," *Empirical Software Engineering*, vol. 19, no. 6, pp. 1809–1855, Dec. 2014.

[5] L. Crispin and J. Gregory, *Agile Testing: A Practical Guide for Testers and Agile Teams*, 1st ed. Addison-Wesley Professional, 2009.

[6] V. T. Heikkilä, M. Paasivaara, C. Lasssenius, D. Damian, and C. Engblom, "Managing the requirements flow from strategy to release in large-scale agile development: a case study at ericsson," *Empirical Software Engineering*, pp. 1–45, 2017.

[7] M. Felderer and A. Herrmann, "Manual test case derivation from UML activity diagrams and state machines: A controlled experiment," *Information and Software Technology*, vol. 61, pp. 1–15, 2015.

[8] M. Unterkalmsteiner, R. Feldt, and T. Gorschek, "A taxonomy for requirements engineering and software test alignment," *ACM Trans. on Softw. Engin. and Methodology*, vol. 23, no. 2, pp. 16:1–16:38, Apr. 2014.

[9] I. Inayat, S. S. Salim, S. Marczak, M. Daneva, and S. Shamshirband, "A systematic literature review on agile requirements engineering practices and challenges," *Computers in Human Behavior*, vol. 51, Part B, pp. 915 – 929, 2015.

[10] B. Robinson, M. D. Ernst, J. H. Perkins, V. Augustine, and N. Li, "Scaling up automated test generation: Automatically generating maintainable regression unit tests for programs," in *2011 26th IEEE/ACM Int. Conf. on Autom. SW Eng. (ASE 2011)*, Nov 2011, pp. 23–32.

[11] T. Gorschek and A. M. Davis, "Requirements engineering: In search of the dependent variables," *Information and Softw. Technology*, vol. 50, no. 1-2, pp. 67–75, 2008.

[12] J. Kukkanen, K. Väkeväinen, M. Kauppinen, and E. Uusitalo, "Applying a systematic approach to link requirements and testing: A case study," in *16th Asia-Pacific Software Eng. Conf.*, Dec 2009, pp. 482–488.

[13] E. Uusitalo, M. M. Komssi, M. Kauppinen, and A. M. Davis, "Linking requirements and testing in practice," in *2008 16th Int. Reqts Eng. Conf.*, Sept 2008, pp. 265–270.

[14] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical Software Engineering*, vol. 14, no. 2, pp. 131–164, 2008.

[15] F. G. de Oliveira Neto, R. Torkar, and P. D. Machado, "Full modification coverage through automatic similarity-based test case selection," *Information and Software Technology*, vol. 80, pp. 124 – 137, 2016.