Success and Failure Factors for Adopting a Combined Approach: A Case Study of Two Software Development Teams

Ingrid Signoretti[1], Maximilian Zorzetti[1](✉), Larissa Salerno[1], Cassiano Moralles[1], Eliana Pereira[2], Cássio Trindade[1], Sabrina Marczak[1], and Ricardo Bastos[1]

[1] MunDDoS Research Group, School of Technology, PUCRS, Porto Alegre, RS, Brazil
{ingrid.manfrim,maximilian.zorzetti,larissa.salerno, cassiano.moralles}@acad.pucrs.br
{cassio.trindade,sabrina.marczak,ricardo.bastos}@pucrs.br

[2] Instituto Federal do Rio Grande do Sul (IFRS), Porto Alegre, RS, Brazil
eliana.pereira@restinga.ifrs.edu.br

Abstract. The combination of Agile, User-Centered Design and Lean Startup has emerged as a solution for teams that are struggling with lack of user involvement and delivering products that fulfill stakeholder needs. Adopting such a development approach involves several factors, some of which can assist or hinder the adoption process. Currently, the literature reports on studies on such factors, but only for agile-only methods. Motivated by this knowledge gap, our goal is to map the success and failure factors of a combined approach adoption. We conduct a case study with two software development teams from a large organization transitioning to the combined approach. We used semi-structured interviews and focus group sessions to collect data. Our findings show five success factors categories (e.g., team engagement, technical aspects) and one failure factor category (team autonomy at risk), along with several argumentation points suggested by the teams to argue against a company policy perceived to be a very impactful failure factor. This study contributes to academic literature by reporting on success and failure factors of a combined approach transformation, and could be used as a starting point in defining tools (e.g., maturity models) to aid organizations in transitioning to the combined approach.

Keywords: Agile · User-centered design · Lean startup · Success factors · Failure factors · Agile transformation

1 Introduction

Combining Agile Software Development with User-Centered Design (UCD) and Lean Startup into a novel development approach is a topic that is being widely explored in literature [7,9,23]. This triad approach helps teams in facing development gaps that agile by itself does not handle (such as stakeholders engagement in the development process) and in exploring and understanding user needs so as to build an assertive product [4]. Due to its many reported benefits in the literature, the combined approach has been the subject of interest in organizations ranging from startups [9] to multinational companies [21]. However, the transformation process to the combined approach implies the same challenges of a transformation to an agile-only approach, possibly even more.

Current literature reports a variety of success and challenge factors faced in an agile-only transformation [17], such as quality, continuous improvement, and waste elimination for success factors [18] and hierarchical issues, cultural aspects, and change resistance for challenge factors [13,17]. As mentioned, these factors were established for agile-only transformations, and therefore might not fully apply to a transformation to the combined approach.

Motivated by this gap in the literature, we conducted a case study with two software development teams that are undergoing a transformation to the combined approach. We observed both teams closely and gathered data to determine the success and challenge factors of the teams' transformation to the combined approach. Our study provides a starting point for teams to improve their continued use of the combined approach; and suggests the need of mechanisms to support and/or accelerate the transformation process, such as maturity models.

The remainder of the paper is organized as follows. Section 2 discourses on the combined approach and transformation processes. Section 3 presents our research method and details the case setting. Section 4 reports the success and failure factors of the teams' transformation. Section 5 discusses our findings. Section 6 wraps up the paper, discusses limitations, and proposes future work.

2 Background

2.1 Combined Approach

The combined use of Agile Software Development, User-Centered Design (UCD), and Lean Startup has been argued as a manner to tackle the limitations of agile, such as lack of customer involvement [1] and proper addressment of stakeholders needs [8]. While its UCD [16] character centers the development on the user, promoting creativity and empathy and helping developers to approach problems using a user-centric view [9], its Lean Startup [19] approach focuses on adding value to business stakeholders by looking for the best solution through experimentation, in which business hypotheses are constantly validated with real data, bringing about the constant pivoting of solutions until a fit resolution is achieved.

Several studies have been made on the combined approach. Fashion retailer Nordstrom report on the very successful case of their innovation team [9], in which they iteratively supplemented their development approach, eventually reaching a combination of Agile, Lean Startup, and Design Thinking. Other studies [4,7,23] propose a process model for the combined approach, while Signoretti et al. discuss the activity and mindset changes that the approach entails [22] and highlight general benefits that the approach brings about [21], along with reporting on the upcoming challenges that teams new to the approach think they will face as the transformation goes on [21].

2.2 Transformation Process

Facing an agile transformation process is an infrastructural project that requires an elaborated and detailed plan from organizations. In such a plan, attention must be paid to the ensuing management, structural, and technical changes while also addressing mindset and cultural issues [17]. Most transformations take place to better align future product development with corporate strategies, so as to better respond to market changes. From a development team perspective, however, motivations to change include a team's lack of engagement or dissatisfaction with the current development method and/or work culture [3].

Julian et al. [11] reports on two transitioning strategies: a "gradual" approach, in which practices are gradually integrated into the organization; or a "big bang" approach, in which all practices are adopted by-the-book. In any case, a transformation has a set of success and failure factors, which are decisive points for organizations to evaluate and scale their transformation strategy. In agile-only transformations, success factors include the use of a pilot transformation team, endorsement of mindset change towards agile values, and promotion of social events [17]; while failure factors include change resistance, intra-organization coordination and communication, and issues with hierarchical and organizational boundaries [3]. As previously stated, however, current literature only encompasses agile-only transformations, and while we assume that a transformation to the combined approach is similar (if not more complicated, given its three-pronged method), there is currently no evidence supporting this.

3 Research Method

In our previous case study [21], we reported on the early benefits (e.g., increased shared knowledge) brought by the combined approach and the current and upcoming challenges (e.g., changing work habits) that the transformation incurs, as perceived by two software development teams that had recently adopted the combined approach. Six months later, we call upon both teams again (now even more entrenched in the combined approach) seeking to understand what pushes the transformation to the combined approach towards success or failure.

3.1 Case Setting

We conducted a case study [20] with two software development teams from ORG (name omitted for confidentiality reasons), a multinational IT company. ORG has software product development sites in the USA (headquarters), India, and Brazil. With over 7,000 employees and responsible for about 1,200 software products. The company started an agile transformation in 2015, but in late 2017 the transformation strategy changed as the CEO understood that the company should improve their user experience by focusing on products. The organization then switched from a project road-map to focus on a product-oriented mindset. This change demanded of teams more in-depth knowledge of their users and business needs. For this reason, the company decided to invest in a combined approach of Agile, UCD, and Lean Startup. The adopted approach was inspired by the Pivotal Labs[1] methodology, which proposes principles and ceremonies based on the three aforementioned approaches. It also suggests the adoption of a cross-functional team composed of three main roles: Product Designer, Product Manager, and Software Engineer. Pivotal Labs' main goal is to help teams to build software products that deliver meaningful value for users and their business. It offers a framework and starting point for any team to discuss its needs and define its own way towards software development. As part of a big bang transformation approach [11], two teams were selected to train daily with consultants from Pivotal Labs. The teams were formed with highly skilled employees that could lead the transformation.

We observed *in-loco* those two teams from the financial area located in Brazil, and both teams develop services for company internal use. The teams are composed of 2 Product Managers, 1 Product Designer, and 4 Software Engineers each. *Team A* is responsible for a software product that manages, calculates, and generates data about company projects related to equipment (e.g, peripherals and computers for personal or server use) and service delivery (e.g., machine installation, support, and replacement). The product manages general project information, such as personnel assignment and time spent on tasks, and also calculates the associated costs of services offered by the products sold by ORG. Team A is tasked with integrating all existing operations of the product into a single application that fulfills user needs and business expectations. *Team B* is responsible for a software product that consumes data from multiple ORG applications (including Team A's) to calculate the average cost of equipment developed in Brazil. The application generates reports for internal accounting, such as inventory reports for tax purposes. Team B had to conduct research to understand current product processes and automate them into the application.

3.2 Data Collection and Methods

To confirm and deepen our previous findings [21], we applied a set of data collection methods that will be explored next. Also, Table 1 shows the profile of the study's participants.

Individual Semi-structured Interviews. We sought to confirm and expand upon our previous findings [21] in regards to the transformation process through individual semi-structured follow-up interviews. We asked the participants to confirm factors collected previously, such as the impact of the combined approach on team engagement, the relationship between team and stakeholders, technical

[1] https://pivotal.io/Labs.

Table 1. Participants' profile

ID	Team	Role	IT Exp. (yr.)	ORG Exp. (yr.)
P1	B	Software Engineer	10	4
P2	B	Product Manager	19	0.5
P3	A	Software Engineer	6	1
P4	B	Software Engineer	15	11
P5	A	Product Designer	27	10
P6	A	Software Engineer	21	8
P7	B	Software Engineer	7	7
P8	A	Product Manager	21	6
P9	B	Product Designer	5	4
P10	A	Product Manager	16	7.5
P11	B	Product Manager	23	10.5
P12	A	Software Engineer	5.5	4
P13	A	Software Engineer	20	11
P14	B	Software Engineer	5	5

aspects, team autonomy, and project-centered budget allocation. This led us to new factors such as team and stakeholders trust and communication, and team autonomy at risk. As these interviews unearthed several new impacting factors on the transformation, we decided to conduct a focus group session to discuss them in depth. The interviews were voice recorded and transcribed for analysis, lasting 30 min on average.

Focus Group Session. We conducted a focus group session to discuss the success and failure factors we had mapped from the individual semi-structure interviews. The session was conducted in two time slots of 1.5 h each. First, in two separate rooms, each team freely discussed each of the factors we mapped. To guide their discussion, we organized these factors in the form of a questionnaire in which the team had to indicate its level of agreement to each of the factors. The factors were grouped by the emerged categories. We used a 5-points Likert scale. Table 2 lists these factors by category. We observed their discussions and took note of them. Afterwards, during the 30 min break that we offered to the participants, we briefly analyzed their answers and came up with talking points pertaining to the discrepancies between each team's answers and our notes as a means to prioritize the factors to be first discussed during the second time slot. In the second 1.5 h, we had both teams meet in the same room to discuss their answers. All factors were debated by the teams. The session was also voice recorded and transcribed for analysis.

3.3 Data Analysis

We conducted Krippendorff's [14] content analysis procedure using a qualitative approach to the ethnographic content analysis, where we focused on the narrative description of the situations, settings, and the perspective of the actors

Table 2. Questionnaire

Question	Factors
Q1. How relevant are the following factors to having a team engaged?	Shared knowledge, mutual feedback, co-responsibility for team activities and deliveries, team ownership, shared product vision
Q2. How relevant are the following factors to promoting trust between teams and stakeholders?	Frequent contact with the stakeholders, team empathy, code delivery in production environment, experiments to understand the problem and solution, mutual feedback, stakeholders and teams working together, mutual transparency, stakeholders see teams as problem solvers
Q3. How relevant are the following factors to promoting communication between teams and stakeholders?	Frequent communication, Face-to-face meetings, team empathy, team and stakeholder working together, team understanding about the problem, development considering UCD activities, stakeholders involvement in the whole process, mutual transparency, team proactivity, constant feedback
Q4. How relevant are the following factors regarding the technical aspects?	Behavior-driven development, pair programming, CI/CD pipeline, test-driven development, unit testing, concise stories writing, frequent deliveries
Q5. How relevant are the following factors to promoting team autonomy?	Middle management trust, solution ownership, middle management support, team decision-making autonomy, team autonomy to conduct small releases in production, autonomy to make decisions about the team scope
Q6. How much can the following factors influence and put the team autonomy at risk?	Budget definition, team resistance to change, stakeholders not understanding teams' work, deploys barriers, lack of middle management support, inter-team interlocks, lack of stakeholder support, team being physically close to the organization, excessive control, defined project schedule
Q7. How relevant are the following factors to the investment in the combined approach adoption for the whole organization?	Story cycle time, middle management satisfaction, application downtime, effectiveness in solving problems, return of investment, business satisfaction, user satisfaction, delivery frequency, problem life cycle, Number of defects

involved in the phenomena of our case study. As we use recording/coding units, we organized the analysis into the following steps: organization and pre-analysis, reading and categorization, and recording the results[2]. We first read the dataset, extracted text excerpts, and marked them as codes. These codes were revisited and grouped into larger codes, forming categories. We constantly reviewed our coding scheme with two seniors researchers (the last authors of this paper) aiming to mitigate any limitations or bias in our analysis. Both senior researchers also reviewed the questionnaire and interview scripts.

4 Results

Our results present the success and failure factors for adopting the combined approach, as perceived by both development teams.

4.1 Success Factors

The teams presented a set of success factors. We organized the factors into five major categories that emerged during our analysis: team engagement, team and stakeholder trust, team and stakeholder communication, technical aspects, and team autonomy. Table 3 consolidates all success factors identified per category.

Team Engagement. The teams emphasized the importance of team engagement aspects, such as a shared product vision, shared responsibilities, shared knowledge, team ownership, and feedback between team members.

One of the participants mentioned that a shared product vision promotes greater value for the team, since everyone gets to know the product—*"Everybody has the understanding about the product, not just the Product Designer or the Product Manager. So everybody knows the reason for working on a product and the importance of it."* (P5) Another participant stated the following on shared responsibilities—*"The whole team makes the decisions. Problems are discussed, as well as solutions. The difference is that before the combined approach we had one person deciding things, and now the whole team has this responsibility."* (P10) They also highlight the importance of having shared knowledge—*"When I miss the Daily Stand-up meeting, I start my day feeling out of the loop."* (P5).

Feedback between team members was stated to be an essential factor to promote team engagement—*"We must be free to give and receive feedback. Sometimes we notice that a colleague is distracted and losing track during meetings. We must give them this kind of feedback, seeking to improve team engagement."* (P4) Another factor was team ownership, especially for the Software Engineers—*"Even as the Product Manager and Product Designer are closer to the users and business due to the nature of their work, the software engineers can not lose their sense of ownership. It is essential that they participate in ceremonies with the stakeholders, as a way to promote the feeling of ownership."* (P2)

[2] We used the Atlas.TI2 digital tool, available at https://atlasti.com/.

Team and Stakeholder Trust. They state that working in a problem-oriented perspective is great for stakeholder trust—*"The stakeholders see us as problem solvers and not only as requirement developers. They see that we are worried about their real needs and looking to deliver the best solution."* (P2) and that this is made possible due to experiments—*"We produce small things through experiments, and this makes our team more assertive on the understanding of the problem and the possible solution."* (P11) even though they might not be as important to the stakeholders themselves—*"The stakeholders do not know how we get to the product, they only see the final result. They do not understand that what we are doing is an experiment."* (P9)

A Product Designer mentioned that having frequent contact with stakeholders enhances their feelings of trust—*"We gain their trust when we talk with the users and understand their needs."* (P5) As such, the participants identify the importance of having team empathy with the users—*"A user saw that we were engaged to solving his problem, that we worry about his difficulties and are working to improve that. From that moment onward we knew that the user trusted us."* (P5) Mutual feedback was also mentioned as a way to increase stakeholder trust—*"We must consider the users' feedback constantly. Both sides feel more confident when what the stakeholders need is aligned with what the team is producing."* (P9) As a consequence, the team and stakeholders work together closely to guarantee that the product being developed is the right one.

A Product Manager mentioned the fact that code delivery in a production environment is of greater value to the stakeholders, which can then understand the effort and the concerns of the team with their needs—*"The stakeholders observe our efforts to deliver with added value. They are informed about everything."* (P11) Mutual transparency between stakeholders and team was also stated as a contributing success factor—*"We just need to develop this relationship, showing to the stakeholders what we are doing and the results. Always being transparent about delivery dates and the issues that we face during product development"* (P2) although members from team B mention that mutual transparency is more important to user stakeholders, as business stakeholders are more interested in general outcomes than the inner workings of the team.

Team and Stakeholder Communication. Frequent communication was reported as a success factor—*"Meetings are important, promoting stakeholder and team communication, but must be used only when necessary. Decision-making must not happen only in meetings: we communicate with the stakeholders as soon as a decision must be made"* (P2) as well as face-to-face meetings—*"Both team and stakeholders benefit from face-to-face meetings, creating intimacy and improving communication."* (P9) To foster communication, having the stakeholders involved since the product's conception seems to be the way to go—*"The team creates an empathetic view since the beginning, and not just when the delivery is made."* (P5) As a consequence, stakeholders and team work together—*"It is important to share decisions about problem prioritization, about what is the best solution... Have the stakeholder work with us."*

(P5) Given their accounts, the team having a proper problem understanding is of utmost importance.

Team empathy with users was also mentioned as an important aspect in communicating with stakeholders—*"The techniques used to gather user information, such as interviews, help us see the needs of the user and put us at their side, consequently getting us closer to them."* (P2) Team proactivity was mentioned as well—*"We must go and understand the problems that the user has on their application, and not just wait for them to say what we have to do."* (P12) although this might be negatively perceived by other teams—*"We act proactively but other teams do not like our attitude, as they get the idea that we are doing their jobs. We get misunderstood for being proactive."* (P13) Thus, considering UCD activities during development to actively engage users, mostly by the Product Designer, is a great practice—*"The Product Designer helps us on approaching the user. The way that the Product Designer communicates with stakeholders is different and brings benefits to us all."* (P9)

Finally, they mentioned mutual transparency as an important factor related to communication—*"Being clear and transparent with the stakeholders results in a lot of pluses to communication and consequently to our relationship."* (P11) However, this comes with a caveat: team B's current relationship with business people is not ideal—"If we had the same kind of relationship that team A has with their business people it would be great. However, today we do not have that, and having transparency now would reflect negatively on us." (P9)

Technical Aspects. A CI/CD pipeline brings about several benefits—*"A CI/CD pipeline is crucial. It promotes fast feedback and helps us validate stories in the production environment."* (P2) A Software Engineer says that delivering code in such an environment made the software engineers more satisfied with their work—*"If the software engineers see the deliverable going to production, they feel more accomplished, leading to more code quality later"* (P14) even if the pipeline itself does not add value—*"CI/CD helps a lot in improving quality aspects. However, it is not a key aspect in adding value to deliverables."* (P9) Frequent deliveries also helps the developers themselves—*"Having frequent deliveries allows us to be more effective"* (P8) and *"Continuous deliveries are essential for us to confirm if we are delivering the right thing"* (P9).

Regarding code quality (which is a factor in and of itself), the participants mentioned techniques that contribute to it, behavior-driven development (BDD), pair programming, unit testing and test-driven development (TDD)—*"Pair programming, BDD, unit testing, and TDD. Mainly TDD, which made the teams more confident about code quality"* (P6) Concise stories were also mentioned as a success factor—*"We quickly identified the added value to a story when it is written in a concise manner."* (P6)

Team Autonomy. For the teams, having the middle managers' trust and support is essential to their autonomy—*"We must build a relationship of trust*

Table 3. Success factors

Category	Success factor
Team engagement	Shared knowledge
	Shared product vision
	Shared responsibilities
	Feedback between team members
	Team ownership
Team and stakeholder trust	Frequent contact with stakeholders
	Code delivery in production environment
	Mutual transparency
	Working in a problem-Oriented mindset
	Experiments
	Mutual feedback
	Team and stakeholders working together
Team and stakeholder communication	Frequent communication
	Face-to-face meetings
	Team empathy with users
	Team and stakeholders working together
	Team understanding of the problem
	Development considering UCD activities
	Stakeholder involvement since product conception
	Mutual transparency
	Team proactivity
Technical aspects	Pair programming
	Unit testing
	Concise stories
	Test-driven development (TDD)
	CI/CD pipeline
	Behavior-driven development (BDD)
	Frequent deliveries
Team autonomy	Solution ownership
	High management support
	Middle management trust and support
	Team decision-making autonomy
	Small releases in production environment

with middle managers, because we need to have them on our side" (P9) and *"They must help us deliver our best. Their support is really important."* (P13) Higher management support is important as well—*"Higher management support helps middle management understand how they must work with the teams now."* (P6)

Small deliveries in production was considered a factor as it adds value to the product—*"There is a considerable effort on the process of having code delivered to production. However, it is important for autonomy, since only when deliverables are in production that we show the added value to the product. Small deliveries allows us to not break the current deployment, and we need this freedom."* (P5) This implies in having autonomy to make decisions—*"We must have this free pass to make our own decisions. Deciding the solution, what is the best for the user... within reason, of course. The point is that the team is the product owner and this decision is ours."* (P5)

Another aspect that was considered important to team autonomy was having solution ownership—*"The teams must have product ownership, especially the solution itself. It is not just about developing requirements."* (P6)

4.2 Failure Factors

The principal threats to the transformation are barriers to the use of the combined approach itself, or rather any kind of factor that interferes with the teams' autonomy. Table 4 consolidates the identified failure factors.

Team Autonomy at Risk. Teams resented the lack of middle management support—*"The managers are learning how to work with autonomous teams that do not depend much on their job"* (P7) and exemplified that it could cause barriers to the production environment—*"We had our code ready to be in production, but the managers told us to wait for two months because the deployment environment was not stable and had a lot of issues. So we faced these barriers and were not allowed to go to production."* (P2)

The team members are also concerned with the previous *modus operandi* of the organization, especially the practice of project schedules—*"Now we work looking to solve problems, not just 'work on a project'. But the stakeholders do not understand this way of working yet. They still ask for documents and a defined schedule"* (P5) and *"We are worried about this need of a defined schedule because it directly affects our decision-making power."* (P10) Excessive control is also part of old policy—*"We have the challenge of dealing with excessive control on ORG. They have a process to all things, security-level, program-level... and this generates bureaucracy, which impacts our autonomy."* (P9) Budget al.location policy being centered around projects and not for development capacity can put autonomy at risk as well—*"The budget al.location policy is project-based, while we are working in solving problems. The managers are worried about that because they do not know who will give financial support to us."* (P6) Another participant says—*"We highly depend on the business, which* *provides money to the products. We are worried that they will act as the product owners and will want to control everything, taking away our autonomy."* (P7)

Table 4. Failure factors

Category	Failure factor
Team autonomy at risk	Lack of middle management support
	Code deployment barriers
	Project schedules
	Excessive control
	Project-centered budget al.location
	Lack of stakeholder support
	Stakeholders not understanding the teams' work
	"Interlocks" with other teams
	Resistance to change
	Team physically close to the organization

Regarding their daily work, they were concerned with lack of stakeholder support and that stakeholders do not understand how the team works— *"The stakeholders have a habit to give finished requirements. Now, they are concerned that we are 'taking' their jobs. We are helping them understand how they must act now."* (P5) Dependencies with other ORG teams, or interlocks as they call them, were mentioned as well— *"ORG has a lot of teams, and as such there are interlocks. We try to remain focused on the problem, but sometimes we can be looking to accomplish the needs of other teams, stopping the delivery of added value to our products."* (P12) Another daily factor is resistance to change— *"The team members must get used to this new way of work. There are some people that do not accept the change and are resisting it. This takes away some of the team's autonomy"* (P9) and *"If we have resistance from our manager or someone on the team, we are sure that it will cause issues to our autonomy."* (P9) It was also stated that the team not being in an environment cut off from the organization, or rather the team being physically close to the organization, could be risky— *"I am concerned in how the team will behave when we return to our real offices. The distance to that site is helping us stay autonomous. We will probably be pressured to work the old way again."* (P12)

The participants were especially particular about how ORG has their funding policy set up to be project-based instead of based on pure development capacity, or on a smaller product-based basis; a practice that causes extreme concerns to both teams, as the combined approach moves them away from big projects and into constant problem solving. Unprompted, both teams started to discuss possible indicators that could be used to argue against this policy, and in favor of a combined approach-friendly one. Table 5 presents their argumentative points.

Table 5. Points for arguing against the project-centered budget allocation policy.

Failure factor	Argumentative point
Project-centered budget allocation	User satisfaction
	Business satisfaction
	Frequent product deliveries
	Middle management satisfaction
	Product necessity understanding
	Problem understanding
	Solution effectiveness
	Cycle time
	Application downtime
	Number of defects

The teams discussed how to convince the business people that funding the combined approach is worthwhile—*"We have to work to make the business people happy. If we give the business feedback of what and how the team is doing, it could be and an indicator for this funding thing, to justify their investments."* (P12) Factors such as user satisfaction and business satisfaction were considered as good indicators—*"These are related to better communication with stakeholders. If we prove that we are working to solve their problems, they will see us a return of their investment. Consequently, both users and business will be convinced."* (P14)

Having frequent product deliveries was mentioned as well—*"We deliver products with added value. If we deliver sooner, we make the users more happy and engaged. And this could be a factor to change how the business allocates their money."* (P5) A Product Designer also considered middle management satisfaction—*"We must show to our managers that we are adding value to the product, and to the organization as a consequence. Once we have their support, they could tell the same story to higher levels of management."* (P9)

They stated that they can use their increased product necessity understanding to convince the adoption of a product-focused mindset, and that their increased problem understanding is great for arguing for product investments—*"Understanding the problem allows us to discuss it with the stakeholders and explain exactly why we need money to improve our product."* (P5) Overall solution effectiveness was reported as an argument as well—*"We focus on identifying problems and not only on developing pre-defined requirements. This helps our effectiveness, and the users seem to be more confident with this way of working on their needs."* (P7) Story cycle time was also pointed out by a Product Designer—*"One of the things that we can show is the time that a story takes between arriving and going to production. The time spent prioritizing and working on a problem, and making it available to users can be a good indicator."* (P5)

Lastly, they mentioned simple metrics, application downtime and number of defects—*"There are metrics that are easy to prove. These are indicators that help not so technical people understand the gains of using this approach."* (P7)

5 Discussion

The success and failure factors of the transformation are especially useful as they were gathered from a team-level perspective, which is essential to consider when conducting an agile adoption process, since its main focus is on team-level development activities [12].

The success factors promote the encouragement for teams to continue believing in the transition to the combined approach, creating an engaging feeling of teamwork through shared knowledge, shared product vision, shared responsibilities, and team ownership. These factors were all reported as extremely important aspects of the adoption, and that obtaining them requires a strong sense of responsibility and belonging, along with mutual feedback and trust, as corroborated by Mchugh, Conboy, and Lang [15] in their study.

The combined approach also demanded stakeholders to adopt a new perspective and to be more engaged with product development. Having the stakeholders' trust and respect is crucial for agile teams [15], and having them involved with development is also relevant to the combined approach due to its heavy emphasis on UCD and Lean Startup activities [21], which paints an ill omen for team B when analyzing their struggling relationship with the business.

Hoda, Noble, and Marshall [10] state that the lack of customer involvement is a tremendous challenge for agile teams. Without stakeholder engagement and support, teams have a hard time delivering the right product and fulfilling stakeholder needs. Dorairaj and Noble [5] mention that a benefit of having good communication between team and stakeholders is that it forms a strong bond—making both parties very effective when collaborating.

Diebold and Mayer [2] report that the most adopted agile practices originate from XP, even when the adopted agile method is not specifically XP, as is the case with the combined approach. Agile practices such as pair programming, BDD, TDD, and concise user stories were reported as great achievements for the teams. Diebold and Mayer [2] also emphasize that using agile practices reduces project risk and increases team productivity and motivation.

As for team autonomy, the support of higher and middle management were extremely important factors on the transformation in the teams' perspective, seeing as their autonomy is directly impacted by decisions such as project funding. These views are shared by Dikert et al. [3], who state that management support must be ensured during agile adoptions.

Regarding the failure factors, both teams reported that the lack of middle management and stakeholder support is a great challenge for the successful transformation of ORG, corroborating with the work of Dikert et al. [3], which mentions that resistance to change and skepticism towards a new way of working are challenges for transformations.

The project-centered budgeting policy was one of the most interesting factors reported by the teams. Upon reflecting on the transformation by our study's prompt, they seemed enthusiastic to look for solutions to this particular problem, as their way of working seems to be most impacted by it. The current hierarchical structure of ORG (and its decision-making ramifications) is not optimized for the combined approach, but the teams brought up a series of indicators (e.g., story cycle time, business satisfaction, and user satisfaction) that could be used to convince higher staff that the approach is worth investing in.

The study results also highlighted that most of the success and also failure factors are related to human aspects.

As a final consideration, we emphasize how the success and failure factors of the combined approach adoption range from technical-level to hierarchical-level concerns, implying that the development team alone is not the only party that needs adaptation—higher-level staff also need to get involved, which can be difficult due to their lack of knowledge on the workings and needs of the development front. We note how most of the factors are related to human aspects. This is not surprising, given that Agile, UCD, and Lean Startup are people-oriented methodologies. However, it is a surprise that even with several studies on this issue, companies are still struggling with it. Human aspects are crucial issues in an agile-only transformation [6], and just as much in a combined approach one. The difference is in the teams' maturity in understanding that and being able to suggest and make modifications that could decrease these transformational barriers. These issues highlights the need of a tool (e.g., a maturity model) to guide the transformation process—a tool capable of conducting the adoption in a way that facilitates the involvement of teams, stakeholders and higher-level staff, by presenting indicators that such staff could understand, for instance.

6 Conclusions, Limitations, and Future Work

We reported the success and challenge factors of adopting a combined approach of Agile Software Development, UCD and Lean Startup through a case study with two software development teams from a multinational company. Ours findings revealed five major categories of success factors (team engagement, team and stakeholder trust, team and stakeholder communication, technical aspects, and team autonomy), and the ultimate challenge factor type being of risks to team autonomy. We also report possible solutions for the distinct challenge factor of "product-focus instead of project-focus", as teams thought it to be most of utmost importance for the transformation.

The findings contribute to the literature by reporting on success and challenge factors for the transformation to the combined approach, as current literature only comprehends similar studies regarding agile-only methods. Industry practitioners can make use of our findings to understand what types of scenarios they could face when dealing with a similar transformation in large organizations.

As inherent to any empirical study, our study has limitations. To mitigate construct validity concerns, we used multiple data sources to triangulate findings and had senior researchers accompany each step of the study. We also observed teams working in a real setting that were composed of members playing distinct roles, each with unique IT experiences. These actions aimed to mitigate such concerns. In regards to generalization, we can not claim that our results apply to distinct scenarios, since the teams' maturity, organizational vision, and their instance of the combined approach are factors that need to be well-considered during a large-scale adoption.

As future work, we suggest the replication of the study in other organizations of similar configuration, so as to compare findings. The findings could be used as a starting point to building a tool that helps organizations in conducting and scaling up the transformation to the combined approach.

Acknowledgments. We thank the study participants and acknowledge that this research is sponsored by Dell Brazil using incentives of the Brazilian Informatics Law (Law no. 8.2.48, year 1991).

References

1. Bastarrica, M., Espinoza, G., Sánchez, J.: Implementing agile practices: the experience of TSol. In: International Symposium on Empirical Software Engineering and Measurement, pp. 1–10. Oulu, Finland, October 2018
2. Diebold, P., Mayer, U.: On the usage and benefits of agile methods & practices. In: Baumeister, H., Lichter, H., Riebisch, M. (eds.) XP 2017. LNBIP, vol. 283, pp. 243–250. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-57633-6_16
3. Dikert, K., Paasivaara, M., Lassenius, C.: Challenges and success factors for large-scale agile transformations: a systematic literature review. J. Syst. Softw. **119**, 87–108 (2016)
4. Dobrigkeit, F., de Paula, D., et al.: The best of three worlds-the creation of InnoDev a software development approach that integrates design thinking, scrum and lean startup. In: Proceedings of the International Conference on Engineering Design, Vancouver, Canada, pp. 319–328 (2017)
5. Dorairaj, S., Noble, J.: Agile software development with distributed teams: agility, distribution and trust. In: Agile Conference, pp. 1–10 (2013)
6. Gandomani, T.J., Zulzalil, H., Ghani, A., Sultan, A.B.M., Sharif, K.Y.: How human aspects impress agile software development transition and adoption. Int. J. Softw. Eng. Appl. **8**(1), 129–148 (2014)
7. Gothelf, J.: Lean UX: Applying Lean Principles to Improve User Experience. O'Reilly, Newton (2013)
8. Gregory, P., Barroca, L., Sharp, H., Deshpande, A., Taylor, K.: The challenges that challenge: engaging with agile practitioners' concerns. Inf. Sotw. Technol. **77**, 92–104 (2016)
9. Grossman-Kahn, B., Rosensweig, R.: Skip the silver bullet: driving innovation through small bets and diverse practices. Lead. Through Des. **14**, 815–830 (2012)
10. Hoda, R., Noble, J., Marshall, S.: The impact of inadequate customer collaboration on self-organizing agile teams. Inf. Softw. Technol. **53**(5), 521–534 (2011)
11. Julian, B., Noble, J., Anslow, C.: Agile practices in practice: towards a theory of agile adoption and process evolution. In: International Conference on Agile Software Development, Montreal, CA, Montreal, CA, pp. 3–18, May 2019

12. Karvonen, T., Rodriguez, P., Kuvaja, P., Mikkonen, K., Oivo, M.: Adapting the lean enterprise self-assessment tool for the software development domain. In: Euromicro Conference on Software Engineering and Advanced Applications, pp. 266–273. IEEE (2012)
13. Karvonen, T., Sharp, H., Barroca, L.: Enterprise agility: why is transformation so hard? In: Garbajosa, J., Wang, X., Aguiar, A. (eds.) XP 2018. LNBIP, vol. 314, pp. 131–145. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-91602-6_9
14. Krippendorff, K.: Content Analysis: An Introduction to Its Methodology. SAGE, Thousand Oaks (2018)
15. McHugh, O., Conboy, K., Lang, M.: Agile practices: the impact on trust in software project teams. IEEE Softw. **29**(3), 71–76 (2012)
16. Norman, D., Draper, S.: User Centered System Design: New Perspectives on Human-Computer Interaction. CRC Press, Boca Raton (1986)
17. Paasivaara, M., Behm, B., Lassenius, C., Hallikainen, M.: Large-scale agile transformation at Ericsson: a case study. Empirical Softw. Eng. **23**, 2550–2596 (2018)
18. Putta, A., Paasivaara, M., Lassenius, C.: Benefits and challenges of adopting the scaled agile framework (SAFe): preliminary results from a multivocal literature review. In: Kuhrmann, M., et al. (eds.) PROFES 2018. LNCS, vol. 11271, pp. 334–351. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03673-7_24
19. Ries, E.: The lean startup: how today's entrepreneurs use continuous innovation to create radically successful businesses. Currency (2011)
20. Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empirical Softw. Eng. **14**(2), 131 (2008)
21. Signoretti, I., et al.: Boosting agile by using user-centered design and lean startup: a case study of the adoption of the combined approach in software development. In: Proceedings of the Int'l Symposium on Empirical Software Engineering and Measurement, pp. 1–6. IEEE (2019)
22. Signoretti, I., Salerno, L., Marczak, S., Bastos, R.: Combining user-centered design and lean startup with agile software development: a case study of two agile teams. In: Stray, V., Hoda, R., Paasivaara, M., Kruchten, P. (eds.) XP 2020. LNBIP, vol. 383, pp. 39–55. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-49392-9_3
23. Ximenes, B.H., Alves, I.N., Araújo, C.C.: Software project management combining agile, lean startup and design thinking. In: Marcus, A. (ed.) DUXU 2015. LNCS, vol. 9186, pp. 356–367. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-20886-2_34