Influential factors of aligning Spotify squads in mission-critical and offshore projects – a longitudinal embedded case study

Salameh, A and Bass, J

http://dx.doi.org/10.1007/978-3-030-03673-7_15

Influential Factors of Aligning Spotify Squads in Mission-Critical and Offshore Projects – A longitudinal embedded case study

Abdallah Salameh[1][0000−0002−3012−9353] and Julian Bass[2][0000−0002−0570−7086]

University of Salford, 43 Crescent, Salford M5 4WT, UK
{a.salameh,j.bass}@salford.ac.uk

Abstract. Changing the development process of an organization is one of the toughest and riskiest decisions. This is particularly true if the known experiences and practices of the new considered ways of working are relative and subject to contextual assumptions. Spotify engineering culture is deemed as a new agile software development method which increasingly attracts large-scale organizations. The method relies on several small cross-functional self-organized teams (i.e., a squads). The squad autonomy is a key driver in Spotify method, where a squad decides what to do and how to do it. To enable effective squad autonomy, each squad shall be aligned with a mission, strategy, short-term goals and other squads. Since a little known about Spotify method, there is a need to answer the question of: *How can organizations work out and maintain the alignment to enable loosely coupled and tightly aligned squads?*
In this paper, we identify factors to support the alignment that are actually performed in practice but have never been discussed before in terms of Spotify method. We also present *Spotify Tailoring* by highlighting the modified and newly introduced processes to the method. Our work is based on a longitudinal embedded case study which was conducted in a real-world large-scale offshore software intensive organization that maintains mission-critical systems. According to the confidentiality agreement by the organization in question, we are not allowed to reveal detailed description of the features of the explored project.

Keywords: Spotify · Alignment · Large-scale agile development · Agile transformation · Mission-critical system · Longitudinal case study

1 Introduction

Since the introduction of the agile manifesto, several agile methods have been developed and introduced. The introduction of those methods shows a fundamental shift in how organizations try to cope with encountered complexity and volatility issues [11]. Thus, no wonder why agile methods are increasingly becoming attractive for many software development organizations of different sizes [4, 6, 28]. It is important to mention that agile methods have some common share techniques such as iterative and incremental development, communication, coordination,

collaborative development, acceptance of the uncertainty, etc. [35]. However, each method has its specific techniques [35], strengths and weaknesses [28].

Large-scale contexts show particular challenges as several teams should work closely together to release a single software product while large groups of people need to collaborate and coordinate [7, 16, 21]. Spotify engineering method, as an agile method, is increasingly attracting the practitioners. The substantial notion of the method is to create autonomous yet collaborative squads. However, an observed challenge of this method is how to get the balance right between squads autonomy and effective collaboration among them. Due to the lack of scientific research related to the Spotify method [19, 18, 17], there is no insightful data through which we can deduce *how organizations can work out and maintain the alignment in order to enable loosely coupled and tightly aligned squads.*

In this paper, we adopt a longitudinal embedded case study in large-scale offshore software intensive organization which maintains mission-critical systems. We conduct direct observation over 19 months and 9 semi-structured practitioner interviews to find out how the alignment is actually performed. We identify four influential factors to the alignment, which are actually performed in practice. To the best of our knowledge, these four factors have not been identified before in terms of Spotify method. We also present "Spotify Tailoring" by highlighting the modified and newly introduced processes to the method.

The rest of this paper is organized as follows: in the next section, we present the background. Sect. 3 describes our research method. Sect. 4 discusses threats to validity. In Sect. 5, we present our findings. In Sect. 6 we discuss those findings. Finally, we conclude and provide future directions in Sect. 7.

2 Background

2.1 Large-scale agile development

The suggested definitions of "large-scale project" are wide-ranging and reflect the various types and contexts of large-scale projects. Dingsøyr et al. [9] defined large-scale agile development as *"agile development efforts that involve a large number of actors, a large number of systems and interdependencies, which have more than two teams"*. This is in line with Rolland et al. definition [31]. Whereas, a very large-scale agile development is defined as *"the agile development efforts with more than ten teams"* [9]. Similarly, while Large Scale Scrum (LeSS) is applied to up to 10 Scrum teams (of seven people), LeSS Huge is applied to a few thousand people working on one product [22]. Some authors refer to very large-scale agile as *"Enterprise Agile"* [1, 13]. Also, Dingsøyr et al. [7] provided a standard taxonomy of agile development scale and associated this scale to the some employed coordination approaches. In this paper, "large-scale project" means a project that has at least two teams working on the same product.

Agile methods (such as LeSS [22], Scaled Agile Framework (SAFe) [23] and Spotify [19]) are increasingly used in large-scale software development. Dingsøyr et al. summarized some of the discussions on research challenges in large-scale

agile development [8]. Also, the challenges and the success factors in large-scale agile transformations have been reported [6].

2.2 Inter-team coordination

Coordination among teams helps transparent participation of all stakeholders, and enables rules and standards that would overcome possible difficulties which might hinder the large-scale projects. Scaled agile has been a hot research topic in the agile community for some years now, where Inter-team coordination is one of important topics that have been pointed out in large-scale agile [26].

In the last two decades, large-scale development has faced coordination challenges. The setup of team of teams has been used because of the increasing dependencies, complexity and uncertainties [2, 20, 34]. As projects increase in size and complexity, the number of inter-team dependencies tends to increase as well [6, 24]. Hence, more coordination effort is needed to deal with these dependencies so that the goals of the teams and the overall goals of the project are achieved [20, 29, 34]. Also, as the requirement specifications might change over time, large-scale projects come with more uncertainties that also affect the coordination due to the unpredictability of required tasks [20].

Eckstein [12] identifies self-organizing teams as a difficulty in a large-scale agile and suggested a model to help in resolving it. Although inter-team coordination is characterised as non-agile in nature where teams are supposed to be self-organized and be empowered, there is inevitable need for the teams to coordinate with each other. If inter-team coordination is weak, it can contribute to integration failure. Hence, scaled agile also need different practices and processes to become more efficient and to overcome the challenges of large-scale agile [3].

2.3 Spotify

The Spotify method has been developed to utilize agile development with hundreds of developers over 30 teams across 3 cities [19]. The overall structure consists, mainly, of Squads, Chapters, Guilds and Tribes [18, 17, 19]. Spotify has numerous squads which are loosely coupled and tightly aligned [18], self-organized and use their own preferable agile methods. Each squad is autonomous and has two types of missions 1) a long-term mission, which is based on the product strategy, and 2) a short-term mission that is quarterly renegotiated [18]. Squads autonomy is presented in the ability for bypassing layers of management and acting upon the decisions that the members have taken together. The Squads are encouraged to implement some Lean Startup principles such as Minimum Viable Product (MVP) where a feature is not finished until the impact is analysed [19]. While a squad leader is responsible for communicating what problem needs to be solved and why, Squads' job is to collaborate with each other to find the best solution [18, 17]. A Squad has access to a coach who is in charge of improving squads' ways of working [19]. Also, each squad has a Product Owner (PO) who is responsible for prioritising the work and for matching product backlog for each

squad [19]. Moreover, it is the responsibility of the POs as whole to maintain a high-level roadmap that shows where the organization is heading [19].

A Tribe is a collection of co-located squads that is designed to be smaller than 100 people and aims to promote collaboration and to minimize dependencies that can slow or obstruct a squad. A gathering is arranged regularly within a Tribe to show what have worked on and delivered so that others can learn [19]. Within the same tribe, there are small groups of people (i.e. Chapters) that have similar skills and working within the same competency area. People within a Chapter meet regularly to help in solving the problems. Chapters are considered the glue that sticks the company together without sacrificing too much autonomy [19]. While Chapters are always located within a Tribe, there are groups of people (i.e. Guilds) who are wide-reaching with a desire to share knowledge, tools, code, and practice across the whole organization [19].

The alignment of squads refers to the extent of which the organization strategy and goals are proudly understood and undertaken by having focused team interactions rather than tactics. Each Squad has autonomy to decide what to build, how to build it, and how to work together while building it. Though, squads need to be aligned with the squad's mission, product strategy, and short goals. Thus, squads should be autonomous, but don't sub-optimize and cross-pollination is better than standardization [18]. Also, due to the alignment on the product-level, the developers become experts in specific areas [19]. However, since *"squads share products instead of owning them"* [18], collective code ownership is adopted to encourage all squads to contribute [28].

Despite the inevitable dependencies between them, squads should have no blocking dependencies [19]. Hence, the Spotify method adopts some continuous discussions to find ways to eliminate the problematic dependencies [19]. In Scrum-of-scrums, a synchronization meeting is continuously conducted between ambassadors to report completions, next steps and impediments on behalf of the teams they represent [29]. However, it is considered beneficial to have smaller and focused inter-team meetings with only participants of similar interests [29]. In Spotify, conducting a meeting for the sake of synchronization is only allowed on demand to coordinate the squads involved [19]. Spotify tries to minimize handoff to scale without having dependencies and coordination. It achieves this by permitting the working on other squads' tasks when facing conflicting priorities [18]. Thereby, the other squad can review the code [18].This indicates the existence of possible contradicted situations that require inter-team coordination.

3 Research Design and Methodology

In order to understand how the alignment is maintained in a Spotify Engineering Culture, we conduct a qualitative research comprising a longitudinal embedded case study over two years period [32]. This qualitative research is based on a case study on a large-scale offshore mission-critical services provider, that last 19 months. During the case study, approximately 200 ceremonies were observed

from 5 project teams, and 9 semi-structured interviews have been conducted. A Grounded Theory (GT) approach is adopted to analyse the collected data.

3.1 Research Site

Our case study is carried out in a real-world company. The company employs approximately 650 staff members in its software development organization including support and management staff in 60 markets. It processes around 60 billion EUR per year. Our study focus only on managing mission-critical autonomous online services. These autonomous systems are collection of connected sub-systems which operate under the control of one administrative online system that presents a common and clearly defined management policy to the service.

The Spotify method is tailored and the developers are co-located in the headquarter. Besides the developers who are distributed into 6 squads, there are 1 architect, 5 POs, 2 agile coaches and 1 test lead.

3.2 Data Collection

The research draws on direct observation of over 200 ceremonies. The observed ceremonies include daily stand-ups, planning sessions, backlog grooming and retrospectives. The team meetings are usually conducted weekly during active periods. One advantage of direct observation is that it provided a deep understanding of the phenomenon that is studied. Another advantage is that it prevents any kind of suspected deviation between "semi-structured interviews" view of matters and the "real" case [30]. Nine semi-structured interviews are conducted for data collection [27]. Open-ended interview guide approach was used to provide interviewees with the opportunity to raise any other issues outside the scope of the semi-structured interview. The questions were revised after the second interview. Each interview was recorded, continued for around 45 minutes and transcribed verbatim for detailed analysis in a continuous basis.

3.3 Data Analysis

GT aims to generate a theory while having an absence of up-front clear research problem or hypothesis and by harnessing a constant comparison of data at increasing levels of abstraction [5]. Hence, the researcher tries to uncover the research problem as the main concern of the participants. In this paper we employ the classic approach (Glasserian approach).

These data have been analysed using the GT [5]. In essence, this is a process of continuous undertaking of data collection, coding and analysis, memoing, sorting and constant comparison of the collected data, and theoretical saturation. Initially, the audio interview and associated written transcript were carefully reviewed to ensure consistency. The interpretive process of open coding broke down the data analytically and generated categories and concepts. A few questions, suggested by Glaser [15], were asked when conducting open coding: "what

is this data a study of?", "what category does this incident indicate?", "what is actually happening in the data?", "what is the main concern being faced by the participants?" and "what accounts for the continual resolving of this concern?". Answering these questions allowed us to continue coding effectively without feeling overwhelmed by the data. Due to the limited ways in which we could organize and interact with the data by using NVivo 10, we settled for using print-outs of the transcripts and physically coded along the margins. A constant comparison was used to refine and sharpen the categories emerging from data, which are free from any prior assumptions held by the researchers. Furthermore, we analyzed the observations and compared them to the concepts derived from the interviews. We found our observations did not contradict but rather supported the data provided in interviews.

4 Threats to the Validity

Threats to the validity are discussed as described by Runeson and Hőst [32]:

Construct validity refers to the extent of which the employed research methodology captures studied concepts and what is investigated according to the research questions [32]. Therefore, the interview protocol was tested during two pilot interviews and minor revisions were made. Also, a GT research study produces a "mid-ranged" theory [14]. This implies that the theory is not claimed to be universally applicable, but it can be modified by having constant comparison to accommodate more data from new contexts [14]. However, a key contribution of a GT study is the focus on conceptualization and the production of modifiable concepts that interrelate to generate an abstract theory, which explains the main concerns of the participants in a substantive area [5].

Internal validity refers to the risk of interference in causal relations within the research [32]. Since this study is of empirical nature, incorrect data is a validity threat. However, this is a longitudinal study that involves repeated observations of the same variables over long periods of time which gave a chance to further validate the findings [32]. In case of the interviews, the written extensive notes, continuous observation and the recordings assured the correct data.

External validity refers to the extent of which the findings of the research can be generalized and of interest to other cases [32]. This case study includes interviews with different roles and squads to brace the external validity of this research. The generalizability is affected negatively because the interviews are performed at one company. However, the purpose of qualitative studies puts more emphasis on describing and recognizing a contemporary phenomenon and less emphasis on generalizing the findings. Nevertheless, results from this case study may benefit the investigation of phenomena within similar contexts.

Conclusion validity refers to the extent of which the data and the analysis are dependent on the specific researchers. In this case study, the conducted semi-structured interviews was tested through two pilot interviews and subsequently followed by a revised version. We adopts a longitudinal embedded case study in which the observations of the same variables were repeated to confirm the

findings, and GT is used to analyze the collected data. Also, we aim to conduct a comparative study to validate and generalize the results.

5 Findings

A synergy has been observed between the following categories, which are depicted in Fig. 1, and strengthening the alignment among the squads. In this section, we will ignore the *"adaptive structure with more focus on communities"* category since it has been covered already in the literature [17–19] yet it is discussed in the context of this case study.

5.1 Strengthening collective code ownership

Collective code ownership requires alignment over the product-level. The squads are provided the freedom to do the required software development on different associated systems due to the realization of collective code ownership. However, it has been realized that sharing products instead of owning them causes a waste of time and resources. This is because of either the lack of knowledge and expertise on the product-level or due to the insufficient ownership.

> *"Handling maintenance or improvement tasks associated to the product-line by other squads is considered as time consuming and will likely require relearning to be able to take the right action"*–P1, Agile Coach and Architect

> *"There should be someone or a team in charge of the bigger picture... A bit more structure around the ownership based on the missions and verticals, and by considering a long-term road map"*–P6, PO and Key Account Manager

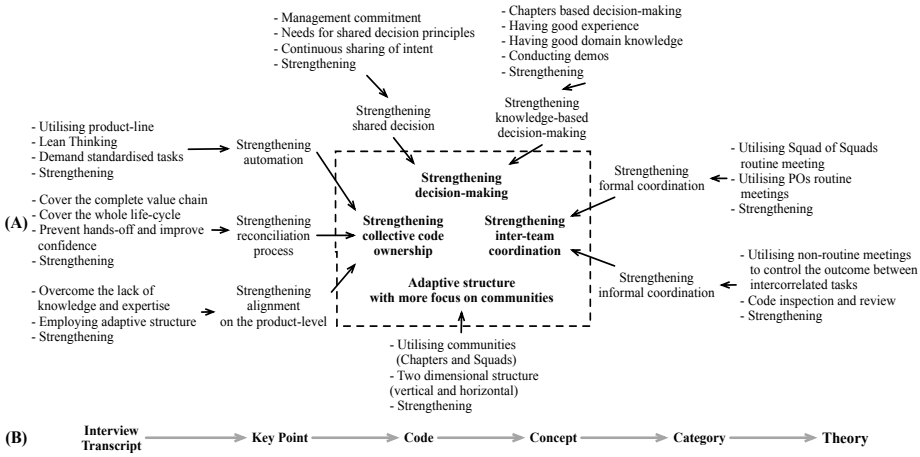

Fig. 1. (A) Emergence of the categories, (B) Levels of data abstraction (GT)

Having many discrete tasks affects knowledge sharing and causes a waste of time and resources and this will raise the need of employing a relearning process. *"The nature of maintenance tasks requires having discrete tasks where each developer works alone"*–P1, Agile Coach and Architect. This case is mostly for those user stories with maintenance nature. Thus, *"code review and inspection are considered crucial to ensure the continuity of deploying successful releases and for the sake of knowledge sharing"*–P1, Agile Coach and Architect. Thereby, the squads should be aligned with the product-level to support collective code ownership and to facilitate the process of knowledge sharing and mastering.

> *"People might abuse the freedom that agile provides. Sometimes, they do not take care of things that they don't like. The responsibility and commitment should be a central part of the work."*–P7, PO and Key Account Manager

Collective code ownership demands a reconciliation process. Handling shared products requires a reconciliation process between the key associated parties, and before adopting the desired change at the system level. Thus, *"the development team may need, sometimes, to discuss their proposed solution with their peers, management team or architecture"*–P3, Senior Developer. This is because of either a lack of expertise at the product-level or the realization of product as a service where the complete value chain and the whole lifecycle need to be taken into account. Without having a reconciliation process, *"...the task might get blocked or a waste of resources might be a result of implementing inefficient solution or shortage in the commitment of management or third parties"*–P3, Senior Developer.

The developers tend to be, sometimes, unconfident when working on a shard product that is not within their expertise regardless of having communities (Chapters and Squads). For instance, in case of having an incident where a hot-fix is requested, the reviewers, who are either from other squads and possess the expertise or from the same squad, were hesitant to handle the situation. This is mostly because of either the complexity of such tasks or being uninformed about the low-level details which requires knowledge transformation and relearning.

> *"A Hot-Fix is requested in case of facing incident after having a new release. If not, the release is rolled-back. Mostly, the developer who owns this task is requested to solve it with some support..."*–P6, PO and Key Account Manager

Constantly adjusting the plan by utilising a Kanban board for each customer satisfies their needs and guarantees having aligned strategy while progressing toward their goals. Providing a visual control through the Kanban board results in an effective mechanism and low coordination overhead. As a result, better management decisions are taken at the right time in a continuous basis. Thus, all squads follow a standardized operational process that is controlled by Kanban board in a routine base because of having a strong culture of cross-pollination. This routine is depicted through the definition of "Done" which indicates the

completion of the implementation of tasks and rules the overall process flow (includes: backlog management, task selection, implementation, peer review, testing, integration, packaging, deployment and feedback). Thus, POs will be able to insure the alignment of the organizational strategy.

> *"Continuously maintaining the practices, such as the definition of 'DONE', helps the customers in gaining a strong competitive advantage due to the rapid response in a highly disciplined manner... Constantly, adjusting the plan through Kanban board satisfies them and guarantees the alignment of the strategy while progressing toward their goals"*–P6, PO and Key Account Manager

The automation of standardized tasks strengthens collective code ownership. Adopting a product-line (PL) requires task standardization to streamline the process of working within the same squad and aids other squads when working on related aspects. Since the organization is providing a service that manages autonomous systems, it adopts a PL way of working in order to integrate the system into the external sub-systems. Fig. 2 illustrates the lifecycle of the PL. A task standardization, which is a key principle in Lean Thinking (eliminating the waste), is depicted through predefined checklists. These predefined checklists are utilized to facilitate planning, estimation, documentation, technical details such as security check-list, code review, knowledge sharing, etc. This process helps the organization to speed up the process and to eliminate possible issues. However, these checklists need to be enhanced further and other potential areas need to be covered.

> *"Code review results, sometimes, missing important aspects due to the weakness of coverage in the current predefined checklist of the PL. Enhancing the existing ones and creating new ones to cover other important task types could be beneficial"*–P2, Senior Developer

> *"We should have efficient check-lists to provide wider coverage in the PL and to have things go smoothly"*–P6, PO and Key Account Manager

Despite the fact that Spotify way of working results in a very little standardization because of having loosely coupled squads that are tightly aligned, it was realized that whole organization tends to be leaner due to the strong culture of cross-pollination besides being transformed from Lean. The organization implements a number of Lean principles such as; Kanban board, continuous process

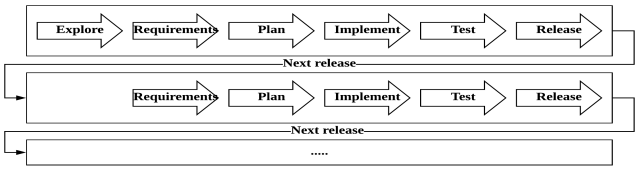

Fig. 2. Product-line lifecycle

(Lean Thinking), documentation for the software development process (value stream mapping), and automation whenever possible. The organization *"tries to automate the processes whenever is possible by utilising DevOps through continuous integration, delivery, deployment, testing and release"*–P4, Senior Developer. Also, Lean thinking is depicted through the adoption of continuous processes. This includes continuous backlog monitoring, prioritization and feedback, integration, delivery, deployment, release, testing, refactoring, verification and inspection.

> *"We have instructions to 1) setup the system for developers, 2) build unit tests, 3) access and deploy, 4) fix release configuration, 5) follow the processes, and 6) do a code implementation and validation using a predefined checklist for the PL"*–P9, Senior Developer

5.2 Strengthening decision-making

Having autonomous squads demands an ability of taking right decisions without relying on others. Having a continuously communicated and shared intent facilitates decision-making. *"We should be familiar with how the GOOD should be look like in all areas associated to the provided service!"*–P6, PO and Key Account Manager. Also, P8, Senior Developer, confirms this need. Obstacles to decision making emerged from lacking shared decision principles. Thus, embracing the principles of having good shared decisions requires having policies and principles that are explicitly and regularly communicated.

Tackling business associated decisions and developing a functionality that can be utilized by all portfolios are considered important aspects when providing a service that manages autonomous systems. In fact, decision-making is, mostly, shifted from a domain independent approach, which relies on routine decisions, to a knowledge-based approach, which relies on the experience and knowledge.

> *"Taking a decision about which solution to adopt is time consuming for the developers with no good knowledge in this domain and makes it hard to work independently"*–P1, Agile Coach and Architect

Since the company provides a product as a service, the complete value chain needs to be taken into consideration. Thereby, the proposed solutions might be tackled with different stakeholders from different squads. This implies that non-routine decisions are adopted to control the outcome of the tasks rather than to control the behavior of the process where the uncertainty is high. However, a more conventional point of view is depicted by highlighting the needs of having a routine meeting between the POs. This meeting is needed to ensure the alignment between squad's missions and the overall road map of the organization. Also, it ensures the ownership of the product itself while the customers, in B2B contexts, are mainly enthusiastic about how to use the service in their own way.

> *"We do not have a product management team. POs should ensure the continuous alignment between them in respect to the road map and be in charge of the bigger picture"*–P6, PO and Key Account Manager

5.3 Strengthening inter-team coordination

The Engineering Culture of Spotify does not directly cover inter-team coordination practices and processes since it tries to speed up the process by having autonomous teams instead of relying on managers. However, the existence of possible contradicted situations requires inter-team coordination to resolve conflicted priorities between the teams and to take into consideration the complete value chain and the whole lifecycle. Thus, *"we need to have a team spirit in order to have the work done and not by only working independently. The harmony of the used processes should be high"*–P2, Senior Developer.

Inter-team coordination is essentially depicted in two routine meetings; 1) Squad of Squads meeting and 2) POs meeting. In Squad of Squads meeting, POs provide brief information about what their teams are determined to do by sharing their intents and obstacles (if any) instead of reporting work status. In POs meeting, they arrange between themselves to ensure 1) the alignment between squads' missions, 2) the ownership of the product itself, and 3) the alignment for the overall road map of the organization. Moreover, non-routine inter-team coordination process is utilized between the associated squads' members to control the outcome of the tasks and to overcome the high-level of uncertainty (aforementioned in Sect.5.2).

> *"We start our week by having a meeting for all squads. In this meeting the POs highlight on what obstacles they are facing and the work they determine to do"*–P6, PO, Key Account Manager

> *"We have regular meetings between the POs to tackle important tasks in the upcoming iteration(s). Also, POs meet up to discuss and prioritize intersected tasks to aligned between the squads"*–P4, Senior Developer

Exposing the adopted coordination processes and providing rules would streamline inter-team coordination. This exposure will level up the alignment and thereby it will become a matter of nature and people will start to behave naturally in different circumstances to achieve the desired goals.

> *"Explaining the values behind the followed processes besides making them obvious are considered crucial. By making them explicit for the teams, we will know what to do and when to do it."*–P5, PO

> *"I do know whom to contact internally and why for each part of the system... This makes my life easier to coordinate, extract the requirements, and to take the right decisions quickly"*–P7, PO and Key Account Manager

Since external parties are involved in the PL, it is crucial to have close collaboration and coordination to carry on the software development. Otherwise, the process of integrating the system into external sub-systems might get blocked and consequently the organization will be behind the anticipated due dates.

> *"Some tickets get blocked for some time in the PL due to the needs of having feedback from third parties. This affects the planning, thereby the customers get informed about the situation by the POs"*–P4, Senior Developer

6 Discussion

The Spotify method is increasingly attracting the industry as a result of tailoring several agile and lean practices together. The key principle the Spotify method is that it is driven by creating autonomous squads. Autonomy is important to have motivated and innovative teams that can deliver quickly while reducing possible mistakes caused by handoffs. However, autonomy can not be precisely defined since the squads are still involved in the same project and there is a common share interest among them. Hence, there should be alignment between the squads and the overall product goals and plans. In spite of Spotify awareness of alignment and their significant roles for the method, there is no adequate guidelines about how to build and maintain the alignment.

Building the alignment is subject to several contextual factors, where the size of the organization is one of them [4]. In this section, we discuss the outcomes of our case study which reveal the factors that influence the way by which the alignment is maintained in a real-world large-scale organization. Also, we present the "Spotify Tailoring".

One of the important reasons for the organization under discussion to transform to the Spotify method is the need of having loosely coupled, yet tightly aligned teams while adopting different agile methods. Since the organization adopts LSD whilst the Spotify method encourages the implementation of Lean Startup principles such as MVP, there was a common ground through honouring product perfection (i.e., maximizing customer value). Moreover, the organization continues to employ Lean Thinking with more emphasis on PL. Hence, the divergence from LSD to Spotify did not impose a considerable challenge.

Based on our findings, we identify four main influential factors of aligning the Squads. Table 1 presents them and shows the applicability of the underlying concepts of these factors within the Spotify method and the case study.

Spotify introduces an adaptive structure based on a matrix of two dimensions, 1) vertical (i.e. Squads and Tribes) and 2) horizontal (i.e. Chapters and

Table 1. The influential factors to the alignment in Spotify

Factor	Concept	Spotify	Case Study
Adaptive structure with more focus on communities	Utilizing vertical and horizontal dimensional structure	Yes	Yes
	Utilizing communities (Chapters and Squads)	Yes	Yes
	Utilizing communities (Guilds and Tribes)	Yes	No
Collective Code Ownership	Alignment on the product-level	Yes	Yes
	Reconciliation process	N/A	Yes
	Automation process	N/A	Yes
Facilitating decision-making	Shared decision-making	N/A	Yes
	Knowledge-based decision-making	N/A	Yes
Inter-team coordination	Formal coordination	No	Yes
	On demand coordination	Yes	Yes
	Informal coordination	Unknown	Yes

Guilds) [19], to build communities around them. This adaptive structure empowers the squads and inspires their innovation. The organization, utilizes the communities of squads and chapters to establish the alignment. However, it does not benefit from the Guilds and Tribes due to the small size of software development teams compared to the size of the teams in Spotify organization itself (P6, P9). Furthermore, the case study indicates that the Guilds and their meetings pose unnecessary time consuming efforts. Consequently, the organization replaces the Guild communities by arranging demo sessions and non-routine meetings between the members who share the same interest.

Since the Spotify method is more about sharing than owning the product, collective code ownership is adopted implicitly. However, there is a link between the degree of autonomy and the shift from code ownership towards collective code ownership, where less autonomy leads to the latter. This link is perceived, in this case study, through the ability of making decisions in an entrepreneurial climate without too much interference from the management. In fact, the Spotify method makes an alignment over the product level, which is beneficial in long-term missions. For the organization in question, however, product level alignment impacts the performance of the squads when it encounters conflicting priorities. This impact is caused due to the lack of knowledge of other product levels. To avoid wasting time and resources, the organization realized the importance of having a reconciliation process between the involved squads. The importance of such a process is because of: 1) not considering the complete value chain and the whole lifecycle in terms of a product as a service, 2) the lack of knowledge of other product levels, or 3) not providing common solutions. Another way that the organization has adopted to streamline the process of working on shared products is the automation of standardized tasks, which is mostly related to the PL by employing Lean Thinking.

The Spotify method does not prescribe how the alignment is maintained when it comes to decision-making. In fact, Kniberg [18] highlights the importance of getting things into production easily rather than knowing who is making the decision. The autonomy of the squads, in our case study, is influenced by decision-making related obstacles (after the transformation). The two determined obstacles are 1) unwillingness to commit to decisions due to a poor alignment on the product-level, and 2) facing conflicted priorities that results in tackling the tasks by another squad who lacks the expertise on the product-level. These two obstacles are in line with what Drury et al., found [10] where they cause a lack of ownership. As a result, the organization adopts shared decision principles and continuous sharing of intent at different strategic, tactical, and operational levels that are explicitly and regularly communicated. This adoption facilitates the process of shared decision-making and strengthens the alignment between the squads. In turn, the alignment of shared decisions strengthens the commitment and ownership, and builds a culture of providing quick help to other squads, which eventually supports the notion of *"we are all in this boat all together"*. The principle of shared decisions becomes a matter of culture where their policies are implicitly followed and adapted by time. Sharing of intent, on

the other hand, clarifies how the GOOD should look like and strengthens squads' autonomy. By continuously sharing the intent, decision-making is shifted from a domain independent approach to a knowledge-based approach that facilitates decision-making and removes the dependencies.

The organization encounters other challenges in terms of shared decision-making which also match the findings of Moe et al., [25]. These challenges are: 1) insufficient alignment of strategic product plans, 2) insufficient allocation of development resources, and 3) performing development and maintenance tasks within the teams. As for the first challenge, conducting non-routine meetings between the involved squads' members are useful to control the outcome of the tasks that have high level of uncertainty. Thereby, non-routine decisions are conducted (bounded rational decision-making [25]). Also, utilizing a routine decision-making (aka rational decision-making [25]) by having meeting between the POs is deemed as crucial for the sake of ensuring 1) the alignment between squads' missions, 2) the ownership of the product itself, and 3) the alignment of the overall road map of the organization. However, decision-making is mostly shifted from a domain independent approach, which relies on routine decisions, to a knowledge-based approach (naturalistic decision-making [25]), which relies on the experience of the team members. This knowledge-based approach is oriented by having "Chapters" besides the alignment at the product-level.

Since the Spotify method intends to speed up the process by having autonomous teams, it does not formally support inter-team coordination processes but only on demands. As projects in larger scale software development increase in size, complexity, dependencies and uncertainties [2, 20, 24, 34], more coordination effort is needed to achieve teams' goals and project's overall goal [20, 21, 29, 34]. Hence, large-scale software development needs to use standards and structures [33] by embracing and exposing inter-team coordination process [3, 12]. For instance, the teams are synchronized and coordinated in Scrum-of-Scrums through 1) inter-team Sprint Planning meetings, 2) inter-team Daily Scrums, 3) inter-team Product Refinements, and 4) inter-team Sprint Reviews. In our case study, two essential routine coordination meetings are determined: 1) Squad of Squads meeting and 2) POs meeting. Also, non-routine inter-team coordination process is used between the involved squads' members to 1) control the outcome of tasks, which is inline with the findings of Paasivaara et al., [29], 2) overcome uncertainties, and 3) share interesting expertise.

7 Conclusion and future research

In this paper, we have explored the alignment among the squads in the Spotify method to uncover the empirical evidence of how the alignment is maintained among the squads. Our work is based on a longitudinal embedded case study which was conducted in a real-world large-scale offshore software intensive organization that maintains mission-critical systems. The analysis revealed the influential factors to the alignment among the squads that are actually performed in

practice. We also present "Spotify Tailoring" by highlighting the modified and newly introduced processes to the Spotify method.

Our work in this paper contributes to identify influential factors for aligning Spotify squads and presents "Spotify Tailoring" with respect to a specific case study. In order to gain more confidence in the presented results of our work, we strongly believe that future work should focus on conducting more comparative studies to further investigate the influential factors. We also encourage further exploration of "Spotify Tailoring" through studying the observed divergence between the authentic Spotify method by the Spotify organization and the tailored Spotify method by other organizations and practitioners.

References

1. Bass, J.M.: How product owner teams scale agile methods to large distributed enterprises. Empirical Softw. Engg. **20**(6), 1525–1557 (Dec 2015)
2. Bick, S., Scheerer, A., Spohrer, K.: Inter-team coordination in large agile software development settings: Five ways of practicing agile at scale. In: Proceedings of the Scientific Workshop Proceedings of XP2016. XP '16 Workshops, USA (2016)
3. Bjørnson, F.O., Vestues, K.: Knowledge sharing and process improvement in large-scale agile development. In: Proceedings of the Scientific Workshop Proceedings of XP2016. pp. 7:1–7:5. XP '16 Workshops, ACM, New York, NY, USA (2016)
4. Campanelli, A.S., Parreiras, F.S.: Agile methods tailoring a systematic literature review. Journal of Systems and Software **110**, 85 – 100 (2015)
5. Corbin, J., Strauss, A., Strauss, A.L.: Basics of qualitative research. Sage, 4 edn. (2014)
6. Dikert, K., Paasivaara, M., Lassenius, C.: Challenges and success factors for large-scale agile transformations: A systematic literature review. Journal of Systems and Software **119**, 87 – 108 (2016)
7. Dingsøyr, T., Fægri, T.E., Itkonen, J.: What is large in large-scale? a taxonomy of scale for agile software development. In: Jedlitschka, A., Kuvaja, P., Kuhrmann, M., Männistö, T., Münch, J., Raatikainen, M. (eds.) Product-Focused Software Process Improvement. pp. 273–276. Springer International Publishing, Cham (2014)
8. Dingsøyr, T., Moe, N.B.: Research challenges in large-scale agile software development. ACM SIGSOFT Software Engineering Notes **38**(5), 38–39 (2013)
9. Dingsøyr, T., Moe, N.B., Fægri, T.E., Seim, E.A.: Exploring software development at the very large-scale: A revelatory case study and research agenda for agile method adaptation. Empirical Softw. Engg. **23**(1), 490–520 (Feb 2018)
10. Drury, M., Conboy, K., Power, K.: Obstacles to decision making in agile software development teams. Journal of Systems and Software **85**(6), 1239 – 1254 (2012), special Issue: Agile Development
11. Dybå, T., Dingsøyr, T.: Empirical studies of agile software development: A systematic review. Information and Software Technology **50**(9), 833 – 859 (2008)
12. Eckstein, J.: Sociocracy: An organization model for large-scale agile development. In: Proceedings of the Scientific Workshop Proceedings of XP2016. pp. 6:1–6:5. XP '16 Workshops, ACM, New York, NY, USA (2016)
13. Fitzgerald, B., Stol, K.J.: Continuous software engineering: A roadmap and agenda. The Journal of Systems & Software **123**, 176–189 (2017)
14. Glaser, B.: "naturalist inquiry" and grounded theory. Forum : Qualitative Social Research **5**(1) (2004)

15. Glaser, B.G.: Doing grounded theory: Issues and discussions. Sociology Press (1998)
16. Hildenbrand, T., Rothlauf, F., Geisser, M., Heinzl, A., Kude, T.: Approaches to collaborative software development. In: 2008 International Conference on Complex, Intelligent and Software Intensive Systems. pp. 523–528 (March 2008)
17. Kniberg, H.: Spotify squad framework - part ii (April, 2014), https://medium.com/project-management-learnings/spotify-squad-framework-part-ii-c5d4b9398c30
18. Kniberg, H.: Spotify squad framework - part i (January, 2014), https://medium.com/project-management-learnings/spotify-squad-framework-part-i-8f74bcfcd761
19. Kniberg, H., Ivarsson, A.: Scaling agile spotify with tribes, squads, chapters & guilds (October, 2012), https://blog.crisp.se/wp-content/uploads/2012/11/SpotifyScaling.pdf
20. Kraut, R.E., Streeter, L.A.: Coordination in software development. Commun. ACM **38**(3), 69–81 (Mar 1995)
21. Larman, C., Vodde, B.: Practices for Scaling Lean & Agile Development: Large, Multisite, and Offshore Product Development with Large-Scale Scrum. Addison-Wesley Professional, 1st edn. (2010)
22. Larman, C., Vodde, B.: Scaling agile development. CrossTalk **9**, 8–12 (2013)
23. Leffingwell, D.: SAFe 4.0 Reference Guide: Scaled Agile Framework for Lean Software and Systems Engineering. Addison-Wesley Professional, 1st edn. (2016)
24. Melo, C., Cruzes, D., Kon, F., Conradi, R.: Interpretative case studies on agile team productivity and management. Information and Software Technology (2013)
25. Moe, N.B., Aurum, A., Dybå, T.: Challenges of shared decision-making: A multiple case study of agile software development. Inf. Softw. Technol. **54**(8) (Aug 2012)
26. Moe, N.B., Olsson, H.H., Dingsøyr, T.: Trends in large-scale agile development: A summary of the 4th workshop at xp2016. In: Proceedings of the Scientific Workshop Proceedings of XP2016. p. 1. ACM (2016)
27. Myers, M., Newman, M.: The qualitative interview in is research: Examining the craft. Information and Organization **17**(1), 2 – 26 (2007)
28. Nerur, S.: Acceptance of software process innovations the case of extreme programming. European Journal of Information Systems **18**(4), 344–354 (2009)
29. Paasivaara, M., Lassenius, C., Heikkilä, V.T.: Inter-team coordination in large-scale globally distributed scrum: Do scrum-of-scrums really work? In: Proceedings of the 2012 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement. pp. 235–238 (Sept 2012)
30. Robinson, H., Segal, J., Sharp, H.: Ethnographically-informed empirical studies of software practice. Information and Software Technology **49**(6), 540 – 551 (2007)
31. Rolland, K.H., Fitzgerald, B., Dingsoyr, T., Stol, K.J.: Problematizing agile in the large: alternative assumptions for large-scale agile development. In: ICIS 2016 PROCEEDINGS : 37 International Conference on Information Systems (2016)
32. Runeson, P., Hőst, M.: Guidelines for conducting and reporting case study research in software engineering. An International Journal **14**(2), 131–164 (2009)
33. Saeeda, H., Arif, F., Minhas, N.M., Humayun, M.: Agile scalability for large scale projects: Lessons learned.(report). Journal of Software **10**(7), 893 (2015)
34. Scheerer, A., Hildenbrand, T., Kude, T.: Coordination in large-scale agile software development: A multiteam systems perspective. In: 2014 47th Hawaii International Conference on System Sciences. pp. 4780–4788 (Jan 2014)
35. Sutharshan, A., Maj, S.: An evaluation of agile software methodology techniques. International Journal of Computer Science and Network Security **10**(12) (2010)