Spotify tailoring for promoting effectiveness in cross-functional autonomous squads

Salameh, A and Bass, J

http://dx.doi.org/10.1007/978-3-030-30126-2_3

Spotify Tailoring for Promoting Effectiveness in Cross-Functional Autonomous Squads

Abdallah Salameh(✉) and Julian M. Bass

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

Abstract. Organisations tend to tailor agile methods to scale employed practices to have cross-functional autonomous teams while promoting sustainable creative and productive development at a constant pace. Thus, it is important to investigate how organisations tailor agile practices to get the balance right between teams' autonomy and alignment. Spotify model is originally introduced to facilitate the development of music streaming services in a very large-scale project with a Business-to-Consumer (B2C) model. However, developing a large-scale mission-critical project with a Business-to-Business (B2B) model is not essentially supported by the Spotify model. Thus, embracing Spotify model for such projects should be concerned about the question of *how Spotify practices are adjusted to promote effectiveness of cross-functional autonomous squads in a mission-critical project with B2B model?*

In this paper, we conduct a longitudinal embedded case study, which lasted 21 months during which 14 semi-structured interviews were conducted. The Grounded Theory (GT) is adopted to analyse the collected data. As a result, we identify practices and processes that promote effectiveness in cross-functional autonomous squads, which have never been discussed in terms of Spotify model before. We also present *"Spotify Tailoring"* by highlighting modified and newly introduced practices by the organisation in which the case study was conducted.

Keywords: Spotify · Tailoring · Autonomous teams · Cross-functional · Large-scale agile · Offshore · Outsource · Mission-critical · Case study

1 Introduction

The introduction of agile development has shifted the focus from the individual level into the team level by employing self-organising teams that are autonomous but aligned [5,8]. To succeed in complex environments, software organisations should find ways to build their teams' autonomy based on their environmental demands and limitations as there is no one size fits all autonomy approach [9]. In

R. Hoda (Ed.): XP 2019 Workshops, LNBIP 364, pp. 20–28, 2019.
https://doi.org/10.1007/978-3-030-30126-2_3

fact, the topic of autonomous teams is immature within software engineering as there are challenges and future research directions that need to be addressed [9].

Spotify model is created around autonomous yet aligned squads [4]. It has been introduced for a very large-scale project with hundreds of developers over 40 teams across 4 cities [4]. Due to the lack of scientific research on this model, there were no guidelines about how to build and maintain the alignment between the squads. In our previous work [8], we determined the influential factors on aligning Spotify squads in a large-scale project. In this paper, we aim to find *how Spotify practices are adjusted to promote effectiveness of cross-functional autonomous squads in a mission-critical project with B2B model?*

We carry out a longitudinal embedded case study in a very large-scale organisation which has a large-scale offshore outsourced mission-critical software project. We conduct direct observation over 21 months and 14 semi-structured practitioner interviews to find out how organisations are actually tailoring agile practices to get the balance right between teams' autonomy and alignment.

We identify utilised practices and processes that promote effectiveness in cross-functional autonomous squads. This effectiveness is presented in the ability of aligning the Spotify squads which in turn enables squads' autonomy. To the best of our knowledge, these practices and processes have not been identified before in terms of Spotify model. Due to the confidentiality agreement with the organisation, we do not provide a detailed description of the explored project.

2 Spotify Model

Spotify model, which is driven by creating cross-functional autonomous squads, is a result of tailoring Scrum and Lean methods to fit a very-large scale project [4,8]. Spotify squads are encouraged to use Lean Startup, which promote innovation [4]. The overall structure consists of Squads, Chapters, Guilds and Tribes [4]. Squads have access to agile coaches who are in charge of improving squads' ways of working [4]. Also, each squad has a Product Owner (PO) who is responsible for (1) prioritising the work, (2) matching the product backlog, and (3) maintaining a high-level roadmap, which shows where the organisation is heading [4].

Squads' autonomy is presented in the ability for minimising dependencies among them, bypassing layers of management, and acting upon internal decisions without relying on other squads [4,8]. To enable effective autonomy, the squads shall be aligned together [4]. Spotify creates alignment by employing an adaptive structure, which is based on two dimensions, (1) vertical (i.e. Squads and Tribes) and (2) horizontal (i.e. Chapters and Guilds) [4]. Also, Spotify employs an *alignment on the product-level* to create expertise in specific areas [4]. In fact, previous research on Spotify model has identified influential factors on aligning Spotify squads by highlighting modified and newly introduced practices to the model [8]. This in turn indicates the necessity of expanding the alignment of the squads to cover further aspects based on the organisation's needs.

In Scrum-of-scrums, a synchronisation meeting is continuously conducted between the ambassadors of the teams to report completions, next steps and

impediments [6]. However, having inter-team meetings with only participants of similar interests is considered more beneficial [6]. In Spotify, a "squad-of-squads" meeting is conducted in which the leaders communicate what problems needs to be solved and why. The squads are expected to collaborate with each other to find the best solution. Since squads share products instead of owning them [4], *collective code ownership* is adopted implicitly [8]. A synchronisation meeting is conducted on demand to coordinate the involved squads [4]. Facing conflicted priorities between squads demands inter-team coordination [8]. Tackling such tasks, which have conflicted priorities, by other squads who lack expertise on the product-level demands a utilisation of peer code review between squads [4].

Spotify adopts a *fail-friendly environment*, which is not about who's fault it is, but rather about capturing failures in a fast pace to learn and improve quickly [4]. Also, Spotify adopts *Postmortem Documentation* process, which is performed at the end of projects to determine what were successful or unsuccessful, to mitigate future risks [4]. Thereby, the organisation tends to *improve the process and the product* [4]. Furthermore, Spotify values *cross-pollination* more than standardisation as it causes less resistance.

The employed release strategy in Spotify is based on enabling *decoupled releases* that simplifies the release process and encourages squads to provide small and frequent independent releases [4]. To achieve this strategy, Spotify employs a *decoupled architecture* [4]. To expose possible integration problems and to minimise the need for code branching, squads are allowed to *release unfinished work as hidden* by utilising toggle switch [4]. Each client application in Spotify has a *release train* that departs on its regular schedule [4]. A *limited blast radius* process is utilised through the delivery of small releases over a limited number of end-users to do small experiments, prevent possible ripple effect, and to learn quickly instead of wasting time controlling all risks in advance [4].

3 Research Design and Methodology

Our case study is carried out in a very large-scale organisation that employs 650 staff members in 60 markets and processes around 60 billion EUR per year. The project, which is the scope of the case study, is considered as offshore outsourced mission-critical software project that manages autonomous financial services that operate under a common defined management policy. In this project, the development programme is of large-scale size [1] since developers are distributed over 6 squads. There are also 1 architect, 3 key account managers (KAMs), 5 POs (2 POs are empowered with KAM role), 2 agile coaches, and 1 test lead.

Due to the lack of scientific research related to the Spotify model, this research draws on a *longitudinal embedded case study* [7] to investigate how organisations tailor agile practices and processes to get the balance right between squads' autonomy and effective alignment among them. This research is comprised of *direct observation* of around 225 ceremonies that last 21 months, and 14 *semi-structured interviews*, which continued for around 50 min. After the second interview the questions were revised. Each interview was recorded and transcribed verbatim for detailed analysis in a continuous basis.

In this paper we employ the *GT* (Glasserian approach) to analyse the data. In essence, this is a process of continuous memoing, sorting, data collection, coding, analysis and constant comparison, and theoretical saturation. Open coding process is used to break down the data analytically and generate categories and concepts. While conducting open coding process, a few questions, suggested by Glaser [2], were asked to facilitate the coding process. A constant comparison was used to refine the categories emerging from the identified concepts. Furthermore, the observations were analysed and compared to the derived concepts from the analysed interviews. In result, minor contradictions were identified, which were explored and accommodated in the resulting grounded theory.

Table 1. Spotify Tailoring for promoting effectiveness

Category	Adopted practices or processes	Spotify	Case Study
Adaptive structure	Two dimensional structure	Yes	Yes
	Utilizing communities (Chapters and Squads)	Yes	Yes
	Utilizing communities (Guilds and Tribes)	Yes	No
Collective code ownership	Alignment over the product-level	Yes	Yes
	Adopting a reconciliation process	Unknown	Yes
Decision-making	Shared understanding of business objectives	≈Yes	Yes
	Emphasising on shared decision-making	≈Yes	Yes
	Utilising knowledge-based decision-making	Unknown	Yes
Inter-team coordination	Formal inter-team coordination	No	Yes
	On-demand inter-team coordination	Yes	Yes
	Informal inter-team coordination	Unknown	Yes
Knowledge sharing	Peer code review between two squads	Yes	Yes
	Limited Fail-friendly environment	No	Yes
	Routine-meetings for squad-of-squads and demos	Yes	Yes
	Chapter based knowledge sharing	Unknown	Yes
	Informal and on-demand knowledge sharing	No	Yes
	Postmortem Documentation	Yes	Yes
	Cross-pollination results in standardisation	Yes	Yes
Mission based planning	Innovation based missions embrace Lean Startup	≈Yes	Yes
	PL based missions embrace standardisation	No	Yes
Release strategy	Decoupled releases via decoupled architecture	Yes	Yes
	Unfinished work shall not be released as hidden	No	Yes
	Backward compatible releases	No	Yes
	Release trains (features with toggle switch)	Yes	Yes
	On-demand releases	Unknown	Yes
	Limited Blast Radius	Yes	Yes

≈**Yes**: partially covered, **Yes**: covered, **No**: not covered, **Unknown**: no evidence

4 Findings

In this section, we describe 3 emerged categories, which support the theory of balancing squads' autonomy and alignment to promote the effectiveness of autonomous squads, by describing only newly introduced and scaled practices. These practices, which are the main focus of this paper, are presented in grey in Table 1. The rest of practices and processes are highlighted on in Sect. 5.

4.1 Knowledge Sharing

Limited Fail-Friendly Environment. Spotify adopts a fail-friendly environment, which embraces fast failures to learn and improve quickly. However, the organisation in question utilises a limited fail-friendly environment as it provides mission-critical software service. *"As we provide software financial services, failure is not tolerated"*–P1, Agile Coach and Architect. However, failures are inevitable during the pilot launch of new features, which aims to improve and verify the behaviour of new features. When a failure is encountered, *"the responsible squad decides whether to switch off the targeted feature or to roll-back the release to overcome encountered issues"*–P7, PO and KAM. This in turn preserves squads' autonomy as only the responsible squad is involved in investigating the problem. Also, The organisation employs these introduced failures to embrace the learning and improvement for both of the process and the product. *"We share the reasons behind encountered release issues in our squad-of-squads weekly meeting to improve the product and the process if needed"*–P12, PO.

Chapter Based Knowledge Sharing. Spotify employs the communities of chapters, which represent the glue that sticks the whole organisation together, to establish cross-functional autonomous squads that are aligned together. In these chapters, members meet to help in solving problems within their competency areas. However, it is unknown if Spotify emphasises on the continuous sharing of knowledge within chapters. In the organisation in question, *"sessions are conducted to share knowledge and expertise within our chapters... At the end of each session, we plan for the next one"*–P8, Senior Developer. This in turn improves squads' abilities and strengthens their autonomy.

Informal and On-Demand Knowledge Sharing. While the software development programme in Spotify is of very-large scale (>300), the development programme in this organisation is of large-scale (<100). *"We do not benefit from Guilds and Tribes as the development programme size is smaller than Spotify's"*–P6, PO and KAM. Thus, guilds and tribes are not applicable for this project. Therefore, *"we call for meetings through Slack or email to discuss subjects of interest... Those who are interested can join"*–P10, PO. This in turn strengthens knowledge sharing while preserving squads' autonomy.

4.2 Mission Based Planning

The squads respond to customers' needs at different velocities based on their missions and scaled agile methods. While some missions value innovation more than plan fulfilment, others value plan fulfilment more than innovation.

Innovation Based Missions Embrace Lean Startup. Spotify encourages the utilisation of Lean Startup to promote innovation, likewise the organisation in question. Tasks of maintenance nature (i.e., adaptive or perfective) and/or of newly requested features are characterised with high degree of uncertainty. *"Developing new features and adapting or improving already existed ones impose challenges due to the high-level of uncertainty... providing dynamic and generic solutions increase the complexity"*–P9, Senior Developer. Such tasks require innovation to provide customers with business values. Hence, those squads tackling such tasks have missions that embrace Lean Startup principles. *"We have hybrid process based on Lean Startup and Kanban"*–P10, PO.

Software development estimation is considered as a waste. *"We sacrifice the predictability of delivery to provide valuable features"*–P6, PO and KAM. However, *"customers request sometimes an estimation before starting the development... We provide a rough estimation and keep the them involved"*–P12, PO.

Product-Line (PL) Based Missions Embrace Automation and Standardisation (Plan Fulfilment). Since the project under study manages autonomous financial services, a PL architecture is utilised to streamline the process of integrating the project into external sub-systems. PL based missions embrace a "waste repellent culture" (aka Eliminating Waste in Lean Thinking). This is depicted through the utilisation of predefined checklists to automate software development. *"We employ checklists in our PL to speed up the process and to cover the activities of planning, estimation, documentation, code review, and knowledge sharing"*–P5, PO. This automation in turn strengthens squads' autonomy and alignment. PL related tasks are characterised with low degree of uncertainty since *"sufficient documentations are received to integrate to the targeted APIs"*–P2, Senior Developer. Since the uncertainty is low and the requirements are matured, a planning process is employed to predict the delivery. *"up-front planning and estimation processes are employed in our PL by utilising predefined checklists, bucket size, on-demand planning techniques, and Lead/cycle time"*–P5, PO. Hence, POs can communicate the delivery deadlines with the customers.

4.3 Release Strategy

Unfinished Work Shall Not Be Released as Hidden. Spotify releases hidden features that are not 100% done. However, the organisation in question does not release unfinished features despite providing all new features with toggle on/off switch, which allows either hiding or exposing new features. This is to make sure (1) having clean code base to prevent possible inconsistencies between the squads while collective code ownership is adopted, and (2) having stable

features at production as the organisation provides a mission-critical services. *"It is crucial to have clean code base that only has stable working features as the code is shared by all squads"*–P1, Agile Coach and Architect.

Backward Compatible Releases. The organisation utilises configuration-driven development to control the behaviour of the software application, modules, or features at the execution time through configuration files. These files are used to (1) force certain business rules, (2) increase the software processing speed, (3) define interconnections between software components to make a compatibility, and (4) ease the development of correct distributed applications. Thus, the organisation provides backward compatible releases to prevent any deviation in the behaviour of the software service from the intended one in the old releases and to strengthen squads' autonomy. *"We always make sure that old features, components, and integrated APIs as well as their old configuration files working as expected... This is to satisfy customers' needs and to prevent possible conflicts of interests between the squads"*–P4, Senior Developer. Also, having backward compatible releases is powerful to facilitate the process of rolling back a release in case of encountering issues. *"New deployed releases shall be always backward compatible to be able to rollback in case of encountering issues"*–P8, Senior Developer.

On-Demand Releases. Since a decoupled architecture is employed in Spotify, a release train is established for each part of the software. Likewise for the organisation in question. *"We utilise a decoupled architecture to (1) facilitate the alignment of squads on the product-level, (2) mitigate possible dependencies between squads, and (3) prevent impacting the whole system when a mistake is introduced"*–P9, Senior Developer. However, it is unknown if Spotify facilitates providing on-demand releases in case of missing a release train. The organisation in question *"employs DevOps to automate the process of release delivery"*–P4, Agile Coach and Architect. Also, the organisation employs DevOps to facilitate on-demand releases in case of encountering a situation where a squad missed a release train. *"If we missed a release train this week, we can either wait for the next train or we can deliver the finished work whenever is demanded by a customer"*–P7, PO and KAM. This in turn increases the autonomy of the squads.

5 Discussion and Conclusion

To maximise success in software development, organisations tend to tailor agile methods to best fit their needs. One of the important reasons for the organisation under discussion to get transformed from Lean into the Spotify model is the need for loosely coupled, yet aligned squads while adopting different agile methods. Spotify has scaled agile software development to attain better performance, productivity and innovation [4]. Since squads' autonomy is a key driver to enable the aforementioned attributes [9], Spotify focuses on enabling autonomous squads [4]. In fact, a common ground through maximising customer value was found since the organisation was adopting Lean whilst the Spotify model encourages the implementation of Lean Startup, which promote innovation.

To investigate how organisations tailor agile practices to get the balance right between squads' autonomy and alignment, we conducted a longitudinal embedded case study in an offshore outsourced mission-critical project of large-scale. The case study lasted, so far, 21 months during which 14 semi-structured interviews were conducted. The GT was adopted to analyse the collected data.

Based on the analysis of the collected data, a synergy has been discovered between (1) the identified practices by this case study, and (2) promoting effectiveness in autonomous squads. This effectiveness is presented in the ability of establishing the right balance between squads' autonomy and alignment. Squads' autonomy and alignment are interdependent and can have an inverse relationship. Too much alignment might hinder squads' autonomy, but at the same time without alignment the squads can be autonomous yet not effective. Since self-organising teams are at the heart of Agile software development, teams must have common focus, mutual trust and respect, as well as accountability to organise themselves to meet new challenges [3].

Table 1 presents these practices and processes, which are classified into 7 categories. The first 4 categories in the table have been highlighted as influential factors on the alignment of Spotify squads [8]. These influential factors are only a subset of practices and processes that contributes to the effectiveness of autonomous squads. The last 3 categories in the table present the new emerged practices and processes, which are the main focus of this paper. Modified and newly introduced practices and processes are presented in grey in Table 1, which are discussed in Sect. 4, whereas the rest are already covered in the literature [4]. The table also indicates the coverage of the adopted practices and processes by the Spotify model and the organisation. Moreover, the table clarifies the extent of which Spotify model has been scaled in the organisation (i.e., *Spotify Tailoring*).

As for future work, we intend to determine employed product development practices in the context of scaled Spotify model for a global B2B model and investigate how these product development practices are correlated with the presented practices in Table 1.

References

1. 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.) PROFES 2014. LNCS, vol. 8892, pp. 273–276. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-13835-0_20
2. Glaser, B.G.: Doing Grounded Theory: Issues and Discussions. Sociology Press, Mill Valley (1998)
3. Hoda, R., Noble, J., Marshall, S.: Organizing self-organizing teams. In: Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering, ICSE 2010, vol. 1, pp. 285–294. ACM, New York (2010)
4. Linders, B.: Don't copy the spotify model, October 2016. https://www.infoq.com/news/2016/10/no-spotify-model
5. Moe, N.B., Dingsøyr, T., Dybå, T.: Overcoming barriers to self-management in software teams. IEEE Softw. **26**(6), 20–26 (2009)

6. 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, September 2012
7. Runeson, P., Hőst, M.: Guidelines for conducting and reporting case study research in software engineering. Int. J. **14**(2), 131–164 (2009)
8. Salameh, A., Bass, J.: Influential factors of aligning spotify squads in mission-critical and offshore projects - a longitudinal embedded case study. In: Kuhrmann, M., et al. (eds.) PROFES 2018. LNCS, vol. 11271, pp. 199–215. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03673-7_15
9. Stray, V., Moe, N.B., Hoda, R.: Autonomous agile teams: challenges and future directions for research. In: Proceedings of the 19th International Conference on Agile Software Development: Companion, XP 2018, pp. 16:1–16:5. ACM, New York (2018)