Designing Scaled-agile Organizations: A Taxonomy of Design Criteria

Julian Frey
University of Bamberg, Bamberg, Germany, julian.frey@uni-bamberg.de

Axel Hund
Neu-Ulm University of Applied Sciences, Neu-Ulm, Germany, axel.hund@hnu.de

Daniel Beimborn
University of Bamberg, Bamberg, Germany, daniel.beimborn@uni-bamberg.de

Designing Scaled-agile Organizations: A Taxonomy of Design Criteria

Julian Frey[1], Axel Hund[2], and Daniel Beimborn[1]

[1] University of Bamberg, Bamberg, Germany
{julian.frey, daniel.beimborn}@uni-bamberg.de

[2] Neu-Ulm University of Applied Sciences, Neu-Ulm, Germany
axel.hund@hnu.de

Abstract. Scaled-agile organizations (SAOs) have emerged as a popular re-sponse to the rapid digital transformation of entire industries. However, we cur-rently lack a conceptual understanding of potential design choices of SAOs and calls for effective organizational structures remain only partially answered. Hence, we seek to answer the question of *how different designs affect the imple-mentation of SAOs in incumbent organizations.* We do this by developing a tax-onomy following the approach by Nickerson et al. and based on data from six cases studies. Our findings provide a taxonomy that identifies a set of eight de-sign criteria across two levels. The taxonomy advances our understanding of the different SAO designs and helps to increase the conceptual clarity of SAOs. We provide a valuable basis for further research and supply practical insights.

Keywords: Scaled-agile organizations, scaling agility, organizational design, taxonomy

1 Introduction

Scaled-agile organizations (SAOs), i.e., a novel form of organizational design that aims to improve the speed of product delivery by extensively implementing agile methods [1, 2], have emerged as a popular response to the rapid digital transformation of entire industries. By scaling agile approaches from the team to the organizational level, companies seek to achieve a higher degree of flexibility to take advantage of the high malleability of digital technologies [3]. Since the organizational structures of incumbent companies are often still from the industrial age [4], new approaches to organizational design [5, 6], product architecture [7, 8], and strategy [9] are needed. The necessity of organizational change is demonstrated by research on related concepts such as agile IT setups [10, 11] and Bimodal IT [12]. However, SAOs represent a different, far-reaching and strategically planned approach to implementing the necessary change and thereby differs from agile IT setups and Bimodal IT. While in the context of agile IT setups, there are insights that show how scaling agile methods to several teams within the IT function works and which different configurations companies pursue, research on Bimodal IT shows how agile methods can best be implemented side-by-side to traditional software delivery in incumbent organizations [12]. SAOs are related with Bimodal IT in the sense that they both aim to achieve a stronger coordination and communication in the delivery process of digital products or increments thereof. In contrast to agile IT setups and Bimodal IT – where agile methodologies and DevOps are used to set up an IT function that both delivers product increments in agile and non-agile developments [13] – SAOs include the business functions as well. This is for instance demonstrated by the installation of BizDevOps teams, referring to the fact that not only software developers and operators form a team but also people from the business unit [14].

SAOs can be seen as the so-far maximal merger of previous "IT departments" and "business departments" and potentially affect our understanding of Business-IT alignment (BITA). BITA still ranks high on the list of most important management concerns [15]. While research emphasizes the shift towards a more unified view on BITA (i.e., formulating a unified digital business strategy) [9], the latest research on BITA additionally departs from this "static" view of alignment and emphasizes the focus on alignment activities "that IT managers and business managers need to carry out jointly as to coordinate goals and operations within IT and across other organizational functions" [15, p. 36]. However, there remains the distinction between "IT managers" and "business managers". SAOs take this a step further by merging business and IT roles into unified teams, increasingly blurring this separation. The setup of SAOs therefore addresses the oftentimes theorized fusion of business and IT "into an overarching phenomenon" [9, p. 472]. Yet, while practitioners across industries are already implementing SAOs [2, 16], we currently lack a conceptual understanding of potential *designs* of SAOs, making it difficult to fully understand the phenomenon and its implications. Confirming this, Dingsøyr and Moe rank the question of "what are effective organizational structures and collaboration models in large projects?" [17] as second most relevant research challenge and call for answering this question. Additionally, Gerster et al. acknowledge that their findings are based on "snapshots of the current state" [1, p. 99] of these transformations towards SAOs, hence call for a further investigation of these transformations. Kalenda et al.'s review surfaces the particular challenges that arise in SAOs, including a too rapid roll-out and the integration of non-agile parts of the organization [18]. Hence, we seek to increase conceptual clarity by exploring the different *design choice*s that organizations have when implementing a SAO and aim to answer the following research question: *How do different designs affect the implementation of SAOs in incumbent organizations?*

To do so we build a taxonomy following Nickerson et al. [19] and based on empirical data from a multiple case study series with six companies and a total of 49 interviews to identify the *designs' criteria* (i.e., the characteristics of the taxonomy) and the respective *design choices* (i.e., the dimensions of the taxonomy) for incumbent organizations that implement SAOs. *Design criteria* refers to the distinct relevant factors organizations can decide upon for their SAO; *design choices* refers to the particular options that can potentially be chosen in the given *design criterion*. Based on these *design criteria* and *design choices* we build a taxonomy and show different combinations of *design choices* as observed in our sample. In doing so, our taxonomy provides a helpful approach to structure further research of SAOs and offers an overview of the various *design choices* that practitioners have when implementing them. In the following, we summarize the background , present our methodology , show our findings and conclude by discussing the implications and offering areas for future research.

2 Background

In the following, we will give a brief explanation of SAOs, show how they are a novel form of organizing and analyze previous literature. We define a SAO as an organizational design with the main goal to alter product delivery that has implemented the following design principles: (A) decentralized teams with a high degree of decision-making autonomy, (B) routine application of agile working methods in and between teams, (C) interdisciplinary teams with business experts as well as IT experts, so-called BizDevOps teams and (D) design principles (A) to (C) apply permanently, i.e, not only for project contexts. Typically, companies use frameworks derived from practical experience such as SAFe, LeSS or the Spotify Model [20]. Building on research by Gerster et al., we derived an abstracted, extended and generic representation of an agile unit through our case study series that helps to understand a SAO's scope (Figure 1).

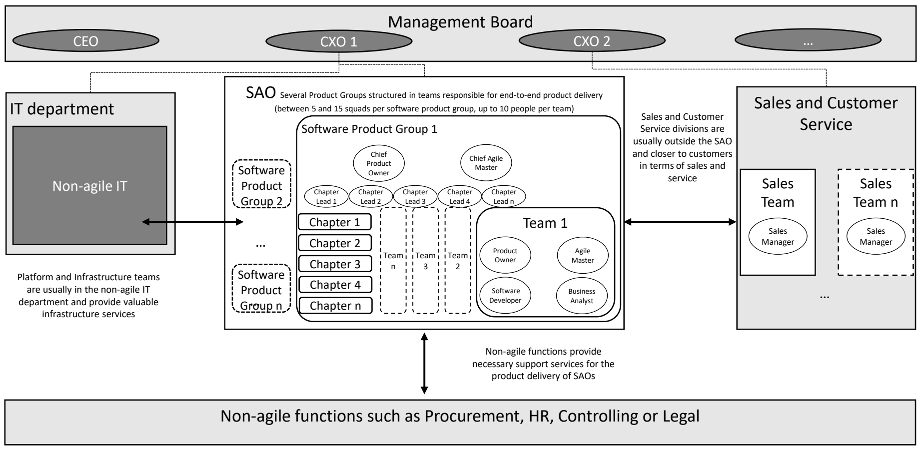

Figure 1. Abstracted, generic Scaled-agile Organization

In a SAO, IT roles and business roles work together in cross-functional teams (in practice often called squads or BizDevOps teams), which are teams that are responsible end-to-end for the delivery of a specific product (e.g. the student credit card offering of a bank) [21]. Roles within these squads comprise a product owner responsible for the product, several frontend software developers and backend software developers, business analysts and an agile coach. Various squads are aggregated in software product groups (in practice often called Tribe or Cluster). These software product groups are responsible for complete groups of products and their delivery (e.g. the complete set of credit card offerings of a bank). A chief product owner usually leads these groups, often but not always jointly with a chief agile master. In addition to this, there are chapters.

They consist out of experts of the same skill domain (e.g. all frontend developers, business analysts or agile coaches) and are usually the disciplinary home of the squad members. All chapters usually have a chapter lead that is the disciplinary responsible for the chapter members. All these different software product groups and chapters together are the integral part of a SAO, usually complemented by some coordinating roles, which are left out here for the sake of simplification. Additionally, three additional types of organizational units exists where the SAO has crucial interfaces: (1) a separate IT department for the operational backbone [22], (2) Sales and Customer Service units that often times link the SAO and the end user of the product/service and (3) all so-called non-agile or "non-development functions" [23] which refers to all organizational functions that do not follow agile ways of working and structuring (i.e., Procurement, HR, Controlling and Legal). Showing that and how the SAO interacts with other organizational units also clarifies that the SAO constitutes only a subset of the overall company.

SAOs can be seen as a new form of organizational design [24] as they promise to provide a new solution (merging IT and business units into small, agile, self-organized, product-oriented teams [1]) to an established organizational problem (responding flexibly and accurately to external changes (e.g., the new nature of digital innovations [7] , new technologies [3], competition, etc.) - referred to as organizational agility [25]. . In particular, by setting up BizDevOps teams that are responsible for the end-to-end delivery of a product a SAO aims to solve the problem of task division. Second, at the same time, these BizDevOps teams (also called cross-functional teams) constitute the approach to solve the problem of mapping the tasks to groups of agents (task allocation). Third, indirectly, SAOs also can be seen as an approach to solve the problem of reward provision as a core underlying assumption of design principle B (the routine application of agile working methods in and between teams) is that employees have a high intrinsic motivation in their respective roles [26]. An emergent stream of research on SAOs has looked at SAOs from different angles, mostly in a narrower definition by only considering a change within the IT function [27]. Important research has analyzed SAOs from a team perspective. For instance, Dikert et al. [23] look at the scaling process of agile product development in several teams and emphasize the strong need for increased coordination and communication. From an individual perspective, there has been research on individual roles in SAOs such as architects [28] or product owners [29] demonstrating the advantages and challenges associated with SAOs for individuals. On an organizational level, research has been limited so far. Gerster et al. analyze the transformation of organizations towards SAOs [1] and identify different "models" of SAOs. In doing so, they (a) shed light on the transformation process of organizations from a classical IT function towards a SAO and (b) identify four different types of SAOs, i.e., generic forms of SAOs. However, they focus on the general different structures that exist. An explicit taxonomy that surfaces the detailed *design criteria* and design *choices* is missing.

3 Methodology

To answer the research question we develop a taxonomy based on a case study series [30] with six cases and 49 interviews to identify the relevant *design criteria* and *design-choices* that exist when setting up a SAO. By conducting a case study series, each case serves to test findings independently and allows for flexible, opportunistic data collection and cross-case pattern search [31–33]. We conducted at least six interviews in each organization to be able to define and refine the *design criteria* and *design choice*s by each organization. The digital setting of the data collection helped us to distribute interviews within the organizations over several weeks with the advantage of being able to analyze identified *design choice*s after a set of interviews and fine tune them into the following interviews. After a first set of three cases, we were able to select additional cases that were on other ends of the respective *design criteria*. This helps us to cover a large space of potential *design choice*s. We cluster the *design criteria* and the *design choices* on two different analysis levels: the organizational level and the team level.

For building the taxonomy we follow the 7-step-approach by Nickerson et al. [19]. Based on the approach we developed the appropriate meta-characteristic that serves the purpose of our taxonomy (i.e., to answer the research question of identifying designs in SAOs). Hence, we defined our meta-characteristic as *design criteria of SAOs*. Due to the iterative nature of the method distinct ending condition(s) need to be defined. Our process is limited by the following objective ending conditions: (1) Every dimension (in this case the *design criterion)* is unique and not repeated; (2) every characteristic (in this case the *design choice*) is unique within its dimension. In addition, we carefully considered the subjective ending conditions (i.e., conciseness, robustness, comprehensibility, extensibility and explainability) during the taxonomy process [19]. We started with a conceptual-to-empirical iteration by drawing on concepts from previous literature in the field of SAOs. We screened the literature for potential *design criteria* to come up with our first iteration. For our second iteration, we did an empirical-to-conceptual approach by using the data available through the case studies and screening it for relevant new dimensions and characteristics. To do so, we inductively coded the set of interview data available at hand and added new dimensions (i.e., *design criteria* and new characteristics (i.e., *design choices*) to come up with a revised taxonomy. We only accepted *design criteria* that were mentioned across cases. At the same time, only criteria that were mentioned at least twice per case study (i.e., by at least two interviewees) have been included in our list of *design criteria*. For a certain set of *design criteria* (i.e., those criteria that had been derived in the first iteration), we explicitly asked closed question where interview partners were asked to assess the respective category for their organization. In addition to the interviews, we used secondary company material (both publicly available data and internal documents) as well as logs from the discussions we had after each interview. Table 1 depicts our case selection. To allow for a maximum variety of characteristics we tried to analyze a diverse set of organizations. The analyzed organizations vary greatly in their absolute size, and are active in different industries. Moreover, the relative size of employees in the SAO compared to the overall organization differs from 5% up to 50%. Lastly, also the type of market varies (B2B, B2C or both). All these factors potentially affect the companies' *design choice*s.

Table 1. Analyzed Organizations

Company	Size (#FTEs)	Industry	Interviews	Roles of interviewees
Alpha	~15.000	Manufacturing	7	TL, CPO, POs, CLs, RTMs
Beta	~1.000	Financial Services	6	CXO, PO, CLs, RTMs
Gamma	~150.000	Telecommunications	10	TL, CPOs, POs, CLs, RTMs
Delta	~50.000	Financial Services	7	TL, CPOs, PO, CAM, CR, RTMs
Epsilon	~10.000	Software	9	CPOs, CAMs, POs, CLs, RTMs
Zeta	~10.000	Software	10	TL, CPOs, CAMs, POs, CLs, RTMs
TL – Transformation Lead (Person responsible for Organizational Transformation), (C)PO – (Chief) Product Owner, CL – Chapter Lead, RTMs – Regular team members (e.g., frontend developer, backend developer, agile coaches, business analysts, other expert roles), CXO – Management Board member				

4 Findings

Based on our analysis we identify eight different *design criteria*, i.e., distinct relevant factors organizations can decide upon for their SAO, along the organizational level and the team level. Within these *design criteria*, two to three different *design choices exist*. In the following, we will present each *design criterion* with its respective *design choices*, show its potential impact and lastly apply the taxonomy on the studied cases. Table 2 shows an aggregation of all the identified *criteria* and the respective *choices*.

Table 2. Taxonomy of SAOs

Design criteria		Design choices		
Organizational level	Implementation strategy	Strict adherence to distinct framework(s)	Loosely following distinct framework(s)	Own framework development or no framework
	Structural implementation	Virtual organization	Virtual organization with real pilot teams	Real organization
	Alignment choice	Aligned	Autonomy with guardrails	Autonomous
Team level	Geographical distribution	Congregated	Partly congregated	Distributed
	Leadership, governance style	Single leader	Duo	Trio
	Team modularity	Cross-functional		Semi cross-functional
	Hierarchy levels	Two	Three	Four
	Agile coach allocation	Per team		Rotating

4.1 Organizational-level design criteria

On the organizational level, we identify three *design criteria*. First, our data shows that organizations differ in their implementation strategy approach [34]. Based on our interviews, we identify that organizations either implement a SAO by (a) strictly adhering to distinct framework(s) such as SAFe or LeSS (e.g. as in the cases of Gamma and

Delta), (b) loosely following distinct framework(s) (e.g. as in the cases of Beta and Epsilon) or (c) not following a previously established framework at all but developing one from scratch (e.g. as in the cases of Alpha and Zeta).

Second, the *structural implementation* can be designed in two ways. Organizations can select between (a) the setup of the SAO as a virtual, second organizational layer where the organizational chart and formal disciplinary hierarchies remain (e.g. as in the cases of Beta, Gamma and Epsilon) or (b) the implementation of the SAO as their primary, real organization where not only operatively the organizational composition changes but also the organizational chart (including the dissolution of previous formal hierarchies). This can be observed in the cases of Alpha, Delta and Zeta. While the first choice certainly is the easier one to implement (amongst other things only little regulatory and labor relations challenges), the latter choice is the more profound setup of a SAO with a higher potential to completely reap the desired benefits. In reality, we observe that there is an in-between *design choice* where organizations implement some parts of the SAO as a real organization but set up most of the organization virtually. This can be seen as an initial step to move from choice (a) to choice (b).

Third, organizations design SAOs with different *alignment choice*s. Managers can decide how much operational autonomy they want to give single teams within the SAO. Operational autonomy refers to the degree to which single teams decide on what they work on and how they do it [35]. Here, organizations pick between three *choices* – (a) high autonomy for all teams (e.g. as in the case Beta), (b) strong alignment with relatively little autonomy (e.g. as in the cases of Alpha and Epsilon) and (c) autonomy with implemented guardrails (as in the cases of Gamma, Delta and Zeta). According to our data, these choices come with distinct advantages and disadvantages. Choice (a) is most in line with the core principles of agile methodologies [36] and provides single teams with large amounts of flexibility and independence (e.g. in terms of task prioritization, choice of technology and tools, budgeting (i.e., autonomous decision on what the allocated financial resources are used for). However, this autonomy comes at a cost as it complicates the coordination and communication mechanisms implemented in a SAO. As one interviewee states: "we have an extremely hard time as an organization with what is laid out in SAFE as portfolio management". As indicated by our interviews, for smaller organizations (such as Beta) choice (a) can still work, however larger organizations prefer other options. Choice (b) is on the other end of the spectrum as it allows for a minimum amount of flexibility for teams by letting them decide only about how to solve their assigned task. Prioritization, choice of technology and tools and budgeting for instance, are not up to the teams themselves but are decided centrally. Choice (c) represents a middle ground. While budgeting usually is still not up to the teams, decisions about task prioritization are more inclusive. Concerning technology and tools, SAOs here provide a set of predefined technology and tools teams then are allowed to freely choose from.

4.2 Team-level design criteria

On the team level, we identify five *design criteria*. First, *geographical distribution* plays a role. Teams can either be geographically distributed or congregated (at one location only). Depending on this choice, the effectiveness and efficiency of the product delivery within the SAO varies. For instance, distributed teams can lead to efficiency losses as one interviewee underlines by saying "the whole thing is just remote, this growing together and doing, the short official channels and so on - is worse than ever before". In addition, congregated teams usually implement agile practices easier (SAO design principle B). According to our data collection, the reasoning behind distributed teams is the larger pool of employees and roles available that helps to ensure that teams can be responsible from end-to-end for their product or service (SAO design principle C). We can observe distributed teams in the cases of Gamma and Delta and congregated teams in the four other cases. A third potential choice could be to have a small set of geographical locations for the teams ("Partly congregated").

Second, the *leadership and governance style* is a salient factor. Here organizations select between three different choices: a single leader, a duo or a trio. A single leader design comprises a single leader who is equipped with the same duties and rights as a classical line manager but a different, more servant management style. We can observe this choice in the case of Beta. The second choice, a duo, refers to a clear separation into one role responsible for the product (e.g. the product owner) and one role responsible for the disciplinary management responsibility (e.g. the Chapter Lead). We can observe this choice in cases Alpha and Gamma to Epsilon. Lastly, a trio refers to a setup with an additional leadership role that is responsible for the processes (e.g. an Agile Coach with specific process responsibility). We can observe this choice only in case Zeta. All choices come with distinct reasoning. While a single leader design has the advantage of (1) a higher decision power and faster decision execution and (b) suggests a better BITA as this one person is end-to-end responsible for the product delivery process, it can be disadvantageous compared to the other two choices as the person carrying out this role is limited in its potential to understand all roles within the unit with depth and can only assert limited attention to each employee. A trio has the advantage that each leadership role can focus on their respective responsibility area and therefore carry out their duties in more depth and detail. At the same time, one observed reasoning that speaks against a trio is the fact that different responsibilities can translate to divergent opinions and decisions that lead to dissent and thus lower performance. The duo, as the in-between choice where the disciplinary lead is also the process lead, shares the advantages and disadvantages of the two other *design choice*s in a weaker form.

Third, *team modularity*, which we define as the degree to which teams are independent from each other, is a criterion that is relevant for the SAO design. Teams can be designed either modular or coordinative. With modular, we refer to the fact that these teams can work completely autonomously to achieve their product delivery without the need for interaction with other teams or being dependent from other teams doing their groundwork. These teams are completely cross-functional in the sense that they are completely able to cover a product delivery process from end to end. We can observe modular teams in the cases of Beta, Delta, Epsilon and Zeta. Coordinative teams necessarily require such coordination or groundwork to be able to deliver their product. These teams are called semi cross-functional as they only cover part of the delivery process. We can observe coordinative teams in the cases of Alpha and Gamma. What we can observe very often in SAOs is that both delivery requires either the coordination with (1) upstream functions such as Sales and Customer Service or (2) supporting functions such as Legal (e.g. when a product release requires legal approval). This also is often times very much restricted by the IT architecture of the organizations. Only if an organization has implemented a modular architecture ("global standards with loosely coupled applications, data, and technology components to preserve the global standards while enabling local differences" [37], it can pick from all choices.

Fourth, the amount of *hierarchy levels* is a *design criterion*. Here SAOs vary between two to four hierarchy levels. Figure 1 depicts a SAO with three hierarchy levels (squad, software product group and complete SAO). We can observe this choice for the cases Alpha and Delta to Zeta. By contrast, smaller SAOs usually do only have squads that are subsumed in the SAO (e.g. in our sample set this can be observed in the case of Beta). On the other end of the spectrum, very large SAOs opt for an additional fourth hierarchical level as the number of products and software product groups is so vast that an additional layer is required that each comprises a set of different software product groups (e.g. in our sample set this can be observed in the case of Gamma). We can observe that the more hierarchical levels are required the higher the effective amount to conduct alignment activities and the potential for a loss in autonomy and thus agility.

Fifth, we identify the *agile coach allocation* as an additional *design criterion*. Organizations can choose to implement an agile coach in each team or let agile coaches be distributed over several teams (usually between two to five). Our interviews indicate that the first choice better helps to execute SAO design principles A and B. We can observe this choice in the cases of Alpha, Delta and Zeta. Based on our collected data, the second choice allows for better knowledge sharing (in particular with regards to processes and rituals of the agile frameworks) as the agile coaches are active across teams. We can observe this choice in the cases of Beta, Gamma and Epsilon.

4.3 Taxonomy Application

Table 3 depicts the application of the taxonomy on the six cases. Our application reveals a variety of different *choices* in each *design criterion* with various different ways of combining them (i.e., configurations). While we have touched upon the reasoning behind the various choices in the above section, we want to emphasize general observations that can be drawn from the taxonomy application. First, it can be noted that there are no clear patterns between the analyzed cases, so we see six different configurations. This supports our assumption of a vast diversity of SAO designs. Second, there are isolated *choices* that are not applied in our cases. These design choices are nevertheless coming from the collected data as interviewees point out additional potential choices which then nonetheless have not been taken. These are for instance the choice of v*irtual organization with real pilot teams* in the *design criteria virtual or real organization* on the one hand and *partly congregated* in the *design criteria geographical distribution* on the other. Although we do not see these *choices* directly in the cases, we can observe that these choices are often used as transition stages for companies to move from one choice to another. For instance, Delta used a virtual organization with real teams as an in-between setup. As stated by an interviewee: “The right people were then put in place as [chief product owner] at the time, and they in turn were able to recruit people from the side on a volunteer basis, so that means that only those who wanted to come to the new organization, and that was a huge difference.”

Table 3 Case application of the SAO *taxonomy*

Design criteria		**Alpha**	**Beta**	**Gamma**	**Delta**	**Epsilon**	**Zeta**
Organizational level	Implementation strategy	Own development or none	Loosely following	Strict adherence	Strict adherence	Loosely following	Own development or none
	Virtual or real organization	Real	Virtual	Virtual	Real	Virtual	Real
	Alignment choice	Aligned	Autonomous	Partly autonomous	Partly autonomous	Aligned	Partly autonomous
Team level	Geographical distribution	Congregated	Congregated	Distributed	Distributed	Congregated	Congregated
	Leadership, governance style	Duo	Single leader	Duo	Duo	Duo	Trio
	Team modularity	Semi cross-functional	Cross-functional	Semi cross-functional	Cross-functional	Cross-functional	Cross-functional
	Hierarchy levels	Three	Two	Four	Three	Three	Three
	Agile coach allocation	Per team	Rotating	Rotating	Per team	Rotating	Per team

Third, we want to single out certain design criteria. The *leadership and governance style* criterion shows that only the smallest organization currently has implemented a single leader design so far. All other studied cases adopt different designs. Although this choice would come with distinct advantages such as faster decision making with less coordination efforts, our analysis suggests that companies prefer other designs choices. This might be because organizations want to distinguish the new structures from classic leadership structures and avoid the perception of returning to leadership structures that SAOs aim to overcome (a single leader that delegates tasks and takes decisions on his own – which inferiors than have to execute). Concerning the *implementation strategy* criterion, only one organization opts for a self-developed framework apart from existing templates. This seems puzzling as at the same time, many interviewees emphasize that they feel standard templates do not suffice for their own organizations’ SAO as it is too special or complex.

As we have described above there are different rationales behind the organizations *design choices* depending on explicit advantages and disadvantages. In addition, as briefly introduced in our methodology, there are also implicit environmental factors that have been identified both from the interviews itself and the secondary data available of the organizations. These include, amongst other, the size of the organization, the industry or industries they are active in, the type of digital product they are developing. In particular, the relative size of the SAO compared to the total size of the organization might be an additional factor that affects the SAO's design. Depending on this share of the SAO compared to the whole organization, challenges concerning implementation success and organizational acceptance arise. Related to that and not strictly a "*choice*" organizations can pick is their absolute size; we identify that smaller organizations (such as Beta) have it easier to achieve acceptance and encounter less challenges with scaling the SAO (the larger the organization the higher the complexity) as the absolute size of the SAO is relatively small. In addition, the type of software that the organization delivers (disruptive vs. incremental improvements) and the type of customer (whether the SAO's customer are either external customers (end users of the product or internal customers (users of a software product within the organization) affect SAOs. While excluded from the taxonomy, these factors can affect SAO designs as well.

5 Discussion and Conclusion

We started with the question of h*ow different designs affect the implementation of SAOs in incumbent organizations*. By (1) deriving *design criteria* and *design choice*s and (2) analyzing respective pros and cons associated with these we answer this question.
Our findings allow us to make four contributions: First, by analyzing SAOs from a diverse set of organizations we have identified existing *design criteria* and the respective choices organizations can take. In doing so, we have created a classificatory framework of SAOs that allows for structuring this new phenomenon. The identified *design criteria* provide structure and improve conceptual clarity concerning SAOs. The granular identification of design criteria can potentially alleviate the challenges identified by Kalenda et al. [18] by allowing managers to gently "play" with individual criteria to reduce resistance to change by providing stakeholders with increased transparency in change as well as conceptual clarity of the "target organization" to better manage a too rapid roll-out through a phased transition as well as mitigating quality assurance issues. By structuring them along different levels, we can shine light on the absolute and relative importance of certain criteria that are relevant when designing SAOs. This helps to advance research by Dingsøyr and Moe that rank the topic of "effective organizational structures and collaboration models" second in the context of research challenges related to SAOs [17]. Second, in elaborating this taxonomy based on the organizations studied, we contribute to a deeper understanding of what forms of SAOs exist. While Gerster et al. [1] identify a four variations of Generic Agile Units, our taxonomy reveals a larger diversity of SAO designs by distilling a variety of underlying *design criteria*. We show that the six studied organizations have different designs and there is potential for more combinations of different *design choices* (i.e., the non-observable configurations as shown in Table 3). Also, we extend on the research by Gerster et al. as the taxonomy allows us to track design changes over time which is crucial as there is "a high likelihood that adopted agile forms of organizational design will be further modified and enhanced over time." [1, p. 99] Third, as touched upon in the introduction SAOs – viewed through a BITA lens – potentially affect our understanding of how organizations can better organize coordination and communication of what has been known as business and IT functions. Extending on previous research in this context,

our taxonomy underscores the relevance of an increased "focus on a larger collection of activities that IT managers and business managers need to carry out jointly as to coordinate goals and operations within IT and across other organizational functions" [15, p. 36]. For instance, the *design criteria alignment choice* and *leadership and governance style* show this need for a more intense coordination vividly as they require IT people and business people within the SAOs to decide jointly on these *design criteria*. In this context, as a "theory for analyzing" [38] our contribution can be taken as a basis to help further theorizing in the context of BITA in SAOs. Fourth, our analysis reveals that organizations intensify the use of SAOs rather than scale it down confirming previous findings by Gerster et al. and recent practitioner surveys [16] that indicate a preference for integrated and collaborative models of organizational designs over disintegrated approaches such as innovation labs or Bimodal IT. All six cases in our data set have indicated that they increased their SAOs in terms of employees and budgets over the analyzed period. In addition, we can make some recommendations for practice. Our taxonomy provides a useful, structuring guide that can be helpful both in implementing new SAOs and in optimizing existing SAOs and thereby advances calls for further investigations of how scaling agility in organizations can work [23]. Our data shows that companies are continuously testing the design of an SAO to optimize it. Our taxonomy helps by showing the range of *design choices* and enabling structured experimentation.

This research is subject to certain limitations. First, the findings acknowledge certain environmental factors, i.e., the initial context in which the analyzed organizations operate, that affect the *design choice*s; however this research does not further elaborate on how these factors affect the *design criteria* and *choices* organizations have when implementing SAOs. An analysis of these environmental factors and their impact could allow to demonstrate that depending on these environmental factors (from factors as wide ranging as from company size, type of software product to individuals' previous experience with agile methods) organizations emphasize certain *design criteria* over others. When thinking about these *design choice*s, a consideration of these environmental factors is useful, as they can affect (a) the amount of choices an organization can take (in terms of which *design criteria* they can even actively decide upon) and (b) whether the organization faces limitations concerning the *design choice*s they can decide upon with a certain criteria. Second, as depicted in Table 3 some *design choices* are not included in our data set. Thus, we cannot analyze these design choices in a practical surrounding. For both limitations, an amplification of the data set should be considered. Third, the *design criteria* we identify apply to structural *design choices* of SAOs; an analysis of *design choices* concerning capabilities, skills or routines seems useful. Fourth, SAOs are often times under constant flux, changes in the *design choices* appear both from a strategic point of view over time as well as on the team level on a frequent, in some cases even weekly basis. This can lead to difficulties in assessing differences between cases. However, there is also an upside to this; further research could make use of this by identifying different trajectories that these SAOs undertake to improve our understanding of how SAOs evolve over time.

Nonetheless, with the presented findings we provide scholars with a solid foundation for hopefully fruitful further research endeavors that help us to understand SAOs and their impact on incumbent organizations' quest to thrive in a digital world.

References

1. Gerster, D., Dremel, C., Brenner, W., Kelker, P.: How Enterprises Adopt Agile Forms of Organizational Design. SIGMIS Database, vol. 51, 84–103 (2020). doi: 10.1145/3380799.3380807
2. Rigby, D.K., Sutherland, J., Noble, A.: Agile at scale. How to go from a few teams to hundreds. Harvard business review : HBR, vol. 96, 88–96 (2018)
3. Kallinikos, J., Aaltonen, A., Marton, A.: The Ambivalent Ontology of Digital Artifacts. MISQ, vol. 37, 357–370 (2013). doi: 10.25300/MISQ/2013/37.2.02
4. Svahn, F., Mathiassen, L., Lindgren, R.: Embracing Digital Innovation in Incumbent Firms: How Volvo Cars Managed Competing Concerns. MIS Quarterly, vol. 41, 239–253 (2017). doi: 10.25300/MISQ/2017/41.1.12
5. Yoo, Y., Boland, R.J., Lyytinen, K., Majchrzak, A.: Organizing for Innovation in the Digitized World. Organization Science, vol. 23, 1398–1408 (2012). doi: 10.1287/orsc.1120.0771
6. Majchrzak, A., Griffith, T.L.: The new wave of digital innovation: the need for a theory of sociotechnical self-orchestration. In: Nambisan, S., Lyytinen, K., Yoo, Y. (eds.) Handbook of Digital Innovation, pp. 17–40. Edward Elgar Publishing (2020). doi: 10.4337/9781788119986.00011
7. Yoo, Y., Henfridsson, O., Lyytinen, K.: Research Commentary —The New Organizing Logic of Digital Innovation: An Agenda for Information Systems Research. Information Systems Research, vol. 21, 724–735 (2010). doi: 10.1287/isre.1100.0322
8. Hylving, L., Schultze, U.: Accomplishing the layered modular architecture in digital innovation: The case of the car's driver information module. The Journal of Strategic Information Systems, vol. 29, 101621 (2020). doi: 10.1016/j.jsis.2020.101621
9. Bharadwaj, A., El Sawy, O.A., Pavlou, P.A., Venkatraman, N.: Digital Business Strategy: Toward a Next Generation of Insights. MIS Quarterly, vol. 37, 471–482 (2013). doi: 10.25300/MISQ/2013/37:2.3
10. Jöhnk, J., Röglinger, M., Thimmel, M., Urbach, N.: How To Implement Agile It Setups: A Taxonomy Of Design Options. Proceedings of the 25th European Conference on Information Systems (ECIS), Guimarâes, Portugal, pp. 1521–1535
11. Paasivaara, M., Behm, B., Lassenius, C., Hallikainen, M.: Large-scale agile transformation at Ericsson: a case study. Empir Software Eng, vol. 23, 2550–2596 (2018). doi: 10.1007/s10664-017-9555-8
12. Haffke, I., Kalgovas, B., Benlian, A.: Options for Transforming the IT Function Using Bimodal IT. MIS Quarterly Executive, vol. 16, 101–120 (2017)
13. Horlach, B., Drews, P., Drechsler, A., Schirmer, I., Böhmann, T.: Reconceptualising Business-IT Alignment For Enabling Organisational Agility. Twenty-Eighth European Conference on Information Systems (ECIS2020), Marrakesh, Morocco.
14. Horlach, B., Drews, P., Schirmer, I., Boehmann, T.: Increasing the Agility of IT Delivery: Five Types of Bimodal IT Organization. Proceedings of the 50th Hawaii International Conference on System Sciences (2017). doi: 10.24251/HICSS.2017.656

15. Luftman, J., Lyytinen, K., Zvi, T.b.: Enhancing the measurement of information technology (IT) business alignment and its influence on company performance. Journal of Information Technology, vol. 32, 26–46 (2017). doi: 10.1057/jit.2015.23
16. Römer, M., Eistert, T., Weiß, C., Schmidl, J., Venus, M., Röglinger, M., Linhart, A. and Utz, L.: Designing IT Setups in the Digital Age (2017), https://www.izb.fraunhofer.de/content/dam/izb/de/documents/Presse/2017/Designing_IT_Setups_in_the_Digital_Age.pdf
17. Dingsøyr, T., Moe, N.B.: Research challenges in large-scale agile software development. SIGSOFT Softw. Eng. Notes, vol. 38, 38–39 (2013). doi: 10.1145/2507288.2507322
18. Kalenda, M., Hyna, P., Rossi, B.: Scaling agile in large organizations: Practices, challenges, and success factors. J Softw Evol Proc, vol. 30, e1954 (2018). doi: 10.1002/smr.1954
19. Nickerson, R.C., Varshney, U., Muntermann, J.: A method for taxonomy development and its application in information systems. European Journal of Information Systems, vol. 22, 336–359 (2013). doi: 10.1057/ejis.2012.26
20. Conboy, K., Carroll, N.: Implementing Large-Scale Agile Frameworks: Challenges and Recommendations. IEEE Softw., vol. 36, 44–50 (2019). doi: 10.1109/MS.2018.2884865
21. Lohrasbinasab, I., Acharya, P.B., Colomo-Palacios, R.: BizDevOps: A Multivocal Literature Review. In: Gervasi, O., Murgante, B., Misra, S., Garau, C., Blečić, I., Taniar, D., Apduhan, B.O., Rocha, A.M.A.C., Tarantino, E., Torre, C.M. et al. (eds.) Computational Science and Its Applications – ICCSA 2020. Lecture Notes in Computer Science, vol. 12254, pp. 698–713. Springer International Publishing, Cham (2020). doi: 10.1007/978-3-030-58817-5_50
22. Winkler, T.J., Kettunen, P.: Five Principles of Industrialized Transformation for Successfully Building an Operational Backbone. MISQE, vol. 17, 121–138 (2018)
23. Dikert, K., Paasivaara, M., Lassenius, C.: Challenges and success factors for large-scale agile transformations: A systematic literature review. Journal of Systems and Software, vol. 119, 87–108 (2016). doi: 10.1016/j.jss.2016.06.013
24. Puranam, P., Alexy, O., Reitzig, M.: What's "New" About New Forms of Organizing? AMR, vol. 39, 162–180 (2014). doi: 10.5465/amr.2011.0436
25. Weber, Y., Tarba, S.Y.: Strategic Agility: A State of the Art Introduction to the Special Section on Strategic Agility. California Management Review, vol. 56, 5–12 (2014). doi: 10.1525/cmr.2014.56.3.5
26. Malik, M., Sarwar, S., Orr, S.: Agile practices and performance: Examining the role of psychological empowerment. International Journal of Project Management, vol. 39, 10–20 (2021). doi: 10.1016/j.ijproman.2020.09.002
27. Fitzgerald, B., Stol, K.-J.: Continuous software engineering: A roadmap and agenda. Journal of Systems and Software, vol. 123, 176–189 (2017). doi: 10.1016/j.jss.2015.06.063

28. Uludag, O., Kleehaus, M., Xu, X., Matthes, F.: Investigating the Role of Architects in Scaling Agile Frameworks. In: 2017 IEEE 21st International Enterprise Distributed Object Computing Conference (EDOC), pp. 123–132. IEEE (2017). doi: 10.1109/EDOC.2017.25
29. Bishop, D., Rowland, P., Noteboom, C.B.: Antecedents of Product Owner Preference for Agile Software Development. 2020 Americas Conference on Information Systems Proceedings
30. Myers, M.D., Newman, M.: The qualitative interview in IS research: Examining the craft. Information and Organization, vol. 17, 2–26 (2007). doi: 10.1016/j.infoandorg.2006.11.001
31. Eisenhardt, K.M.: What is the Eisenhardt Method, really? Strategic Organization, vol. 19, 147–160 (2021). doi: 10.1177/1476127020982866
32. Eisenhardt, K.M., Graebner, M.E.: Theory Building From Cases: Opportunities And Challenges. AMJ, vol. 50, 25–32 (2007). doi: 10.5465/amj.2007.24160888
33. Yin, R.K.: Case study research. Design and methods. Sage, Thousand Oaks (2009)
34. Hoda, R. (ed.): Agile Processes in Software Engineering and Extreme Programming – Workshops. Lecture Notes in Business Information Processing, vol. . Springer International Publishing, Cham (2019). doi: 10.1007/978-3-030-30126-2
35. Moe, N.B., Dahl, B., Stray, V., Karlsen, L.S., Schjødt-Osmo, S.: Team Autonomy in Large-Scale Agile. Proceedings of the 52nd Hawaii International Conference on System Sciences. doi: 10.24251/HICSS.2019.839
36. Beck, K., Beedle, M., van Bennekum, A., Cokburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A. and Jeffries, R., et al.: Manifesto for agile software development (2001), https://agilemanifesto.org
37. Ross, J.W.: Creating a Strategic IT Architecture Competency: Learning in Stages. MIS Quarterly Executive, vol. 2, 31–43 (2008)
38. Gregor, S.: The Nature of Theory in Information Systems. MIS Quarterly, vol. 30, 611 (2006). doi: 10.2307/25148742