Ambidexterity in Agile software development: a conceptual paper

Carin Lindskog and Monika Magnusson
Information Systems, Karlstad University, Karlstad, Sweden

Abstract
Purpose – The purpose of this study is to apply the concept of organizational ambidexterity as a conceptual lens to increase the understanding of tensions between exploitation (continuity) and exploration (change) in Agile software development (ASD) project teams, and particularly the balancing (ambidextrous) strategies utilized.

Design/methodology/approach – A conceptual framework was constructed from interdisciplinary sources on ambidexterity. A literature review of publications on ambidexterity in ASD was then performed, and the results from the selected publications were classified according to an extension of the conceptual framework.

Findings – Contextual ambidexterity in ASD is affected by the four basic coherent concepts: time, task, team and transition. The study found that most ambidextrous factors and strategies were task and team-related. In addition, a mixture of hard (performance) strategies and soft (social) strategies is needed in order for people/teams to (be able to) become ambidextrous.

Practical implications – To provide a better understanding of ASD, it is important to identify a broader set of ambidextrous factors and strategies that can impact ASD project teams. The expanded conceptual framework can serve as a basis for future empirical research and provide insights to practitioners on how to strengthen ambidexterity in ASD projects.

Originality/value – The contribution is of great importance for ASD research and practice, as ASD methods are a popular method for managing projects within ASD and in other non-software organizations. In addition, as more and more organizations struggle to deal with rapidly changing environments, interest in the phenomena of paradoxical tensions and the strategy (ambidexterity) to deal with these tensions increase.

Keywords Agile software development, Plan-driven software development, Ambidexterity, Exploitation, Exploration, Paradoxical tensions

Paper type Conceptual paper

1. Introduction

In times of change, organizations both need to exploit their existing business field as well as explore new business fields (Pellegrinelli et al., 2015) in order to persist over time (O'Reilly and Tushman, 2011). Balancing stability and structure with agility and changeability is central to achieving efficiency in a changing environment. In many organizations, there are increasing calls for organizational agility, i.e. the ability to respond, adapt rapidly and thrive in a changing environment (Holbeche, 2018). Greater flexibility and changeability have historically been particularly sought after in software development (Abbas et al., 2008). Agile software development (ASD) is the most dominant approach to software development (VersionOne, 2019), and ASD is seen as a reaction to traditional or plan-driven software methods (Dyba and Dingsøyr, 2008). The main principle of ASD is about flexibility and endeavor to improve its project work constantly. It is accomplished by having short time stages, so-called sprints (Beedle et al., 1999), consisting of cycles of planning, performing, checking and then acting based on drawn conclusions (Schwaber and Sutherland, 2017). Even though the Agile way of working originates in the information technology (IT) industry and software development, this...
way of working is being adopted in a wide variety of businesses. Highsmith (2002) states that the Agile way of working is through good practices that should be carefully considered for any environment. The Agile way of working is also spreading to other industries and sectors such as banking (Johnston and Gill, 2017), manufacturing industries (Eliasson and Burden, 2013) as well as the public sector (Wisitpongphan and Khampachua, 2016). Combining the business-driven need for predictability and long-term planning with the flexibility of ASD methods is challenging and can be seen as a contradiction that may cause paradoxical tensions. Paradoxical tensions that exist are essential both to identify and recognize and have been used to grasp many complex organizational situations. According to Aubert et al. (2015), ignoring paradoxes can lead to a tunnel vision and can be detrimental to the organization. Paradoxical tensions can arise between exploitation and exploration that are discussed in different research fields and defined as, e.g. concepts, processes, orientations, set of activities and learning activities. Well-known examples of exploitation are refining and using existing knowledge. Furthermore, well-known examples of exploration are innovation, problem-solving and creating new knowledge (Gupta et al., 2006; March, 1991).

An organization’s ability to manage paradoxical tensions between exploitation and exploration is captured in the concept of organizational ambidexterity. Raisch and Birkinshaw (2008, p. 375) state: “To be ambidextrous, organizations have to reconcile internal tensions and conflicting demands in their task environments”.

Although ambidexterity has become a hot topic in organizational research, there is still a lack of understanding as regards “how” ambidexterity is concretely supported in practice. (Pellegrinelli et al., 2015; Sailer, 2019; Turner and Lee-Kelley, 2013). Even in the strongly emerging research field of ASD, there is a lack of knowledge of what these paradoxical tensions consist of and what strategies exist to deal with them (Wang et al., 2008). This study aims to increase the understanding of tensions between exploitation (continuity) and exploration (change) in ASD project teams, and particularly the balancing (ambidextrous) strategies utilized. The research questions are the following:

RQ1. What ambidextrous factors in ASD are reported in previous studies?

RQ2. What strategies for achieving ambidexterity in ASD project teams are reported in previous studies?

The overall structure of the study takes the form of six sections, including this introduction. Section 2 provides a theoretical background, whereas Section 3 gives an overview of the research method used. Section 4 presents findings from the literature on ambidexterity in ASD. In Section 5, a discussion regarding the findings, future work and the limitations of this study is provided. Finally, Section 6 concludes the article.

2. Theoretical background
First, a brief introduction of the concept of “project as a temporary organization”, the two different software development approaches and Agile practices are depicted below. Second, interdisciplinary sources of the literature were used to explain the concepts of exploitation, exploration, ambidexterity and the outcomes of ambidexterity. Third, a conceptual framework based on interdisciplinary sources of the literature is depicted. The concepts and the framework provide the initial framing necessary for developing an understanding of how ASD project teams balance conflicting demands.

2.1 Project as a temporary organization
Software development usually takes place in projects. The project becomes an extract of time and space, taking place in a context that exists before, alongside and after the project (Lundin
When a project is completed, something has been created that did not exist when the project was initiated. In the study by Lundin and Söderholm (1995), a framework was generated for contributing to the theoretical field of projects and temporary organizations. The concept of “temporary” refers to something that exists for a limited time and, usually, this time aspect is well-known from the beginning (Lundin and Söderholm, 1995).

According to Lundin and Söderholm (1995), temporary organizations can be understood by the four basic coherent concepts: time, task, team and transitions (see Figure 1), and can be used as a description or a classification of the temporary organization (Lundin and Söderholm, 1995). The definition of a task may put limits on time; likewise, time limits may disqualify certain tasks. Team members might be chosen due to their competencies applicable to the outlined tasks. Furthermore, team members and their competencies will influence what task or transition aspirations may be proposed. Task definition also implies aspirations about transition, and some of these may also select or define the task. Time is in the middle of the figure to underscore its role as the most important of the concepts (Lundin and Söderholm, 1995).

Unlike permanent organizations, temporary organizations (such as projects) are fundamentally action-based. Therefore, projects usually use project methods that describe the type of actions to be applied in what order (Sailer, 2019). In software development, there are project methods that can be categorized into two main approaches. These main approaches are described in the following section.

2.2 Plan-driven software development versus ASD

A software development process can be viewed as the actual way that a software product is developed in a real-world context. Traditional software development aspires to promote the usage of role-based teams and detailed plans of the development life cycle. Such a life cycle may, according to Abrahamsson et al. (2017), be seen as consisting of nine phases: (1) project inception, (2) requirements specification, (3) design, (4) coding, (5) unit test, (6) integration test, (7) system test, (8) acceptance test and (9) system in use. Scholars refer to the traditional software approaches as plan-driven, task-based or the waterfall model (Chau et al., 2003). In this paper, plan-driven software development is the term used as an alternative to ASD, even though planning also is an important activity in ASD (Vidgen and Wang, 2009). However, in ASD projects, the planning activities consist of shorter cycles or iterations (Chau et al., 2003).

Abbas et al. (2008, p. 3) argue: “Software development is an unpredictable activity; therefore, we need an adaptive process to control this unpredictability. Iterative and incremental development will be the best controller for this process. In addition, it needs creative and talented people”.

The first time the concept of Agile methods or agility appear in the main-stream business literature was, according to Conboy (2009), as early as 1991. Today’s Agile methods originate from a set of values and principles outlined in the so-called Agile Manifesto (Beck et al., 2001).
Conboy (2009, p. 340) argues that Agile methods should “Rapidly or inherently create change, proactively or reactively embrace change, and learn from change while contributing to perceived customer value (economy, quality and simplicity), through its collective components and relationships with its environment.”

Cram and Marabelli (2018) distinguish between knowledge viewed as an “object” that can be exchanged in the process of documentation in compliance with plan-driven software development approaches. This can be compared to viewing knowledge as a “relationship” exchanged in face-to-face meetings, in compliance with an Agile development approach. Hence, plan-driven software methods extensively use documentation. In contrast, Agile methods advocate that the written documentation should be “lean and mean” (Chau et al., 2003, p. 1). To compensate for the reduction in the Agile documentation, Agile methods strongly encourage direct and frequent communication and collaboration (Chau et al., 2003).

Furthermore, in plan-driven software development methods, the project manager plays a central role in the project lifecycle with responsibilities to plan, to lead, build the teams, motivate, communicate and make the decisions (Shastri et al., 2016). Hoda and Noble (2017) claim that the plan-driven manager’s role needs to evolve to suit the Agile way of work. Hence, in ASD projects, the role of a manager is transitioning from driving to empowering the team. In short, the meaning of management in ASD projects is shifting from leading and controlling to participating and mediating (Vidgen and Wang, 2009). In Table 1, noticeable differences between plan-driven software development and ASD are presented.

In the literature review conducted by Vinekar et al. (2006), it is highlighted that plan-driven software development is appropriate when the requirements are stable and predictable and when the project is large, critical and complex. ASD, on the other hand, is appropriate when there is a high degree of uncertainty and risk in the project arising from frequently changing requirements and/or the novelty of technology used (Vinekar et al., 2006).

<table>
<thead>
<tr>
<th>Category</th>
<th>Plan-driven development</th>
<th>Agile development</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development Model</td>
<td>Life-cycle, stage-gate</td>
<td>Iterative, incremental</td>
</tr>
<tr>
<td>Focus</td>
<td>Process</td>
<td>People</td>
</tr>
<tr>
<td>Management</td>
<td>Driving</td>
<td>Empowering</td>
</tr>
<tr>
<td>Customer involvement and</td>
<td>Formalized requirements captured before initiation of design and development as needed</td>
<td>Active customer and extensive user participation throughout the project, including a high degree of readiness for change. Requirements are estimated for workload, prioritized and contextualized as stories or test cases</td>
</tr>
<tr>
<td>requirements</td>
<td>interaction between the development team and customers. Requirements gathering and delivery phases</td>
<td></td>
</tr>
<tr>
<td>Team composition</td>
<td>Clearly defined, role-based teams, such as business analysts, developers and testers. Developers work individually within teams</td>
<td>Cross-functional teams, with team members playing multiple roles throughout the project. Standard 40-hour work weeks are employed to preserve work-life balance</td>
</tr>
<tr>
<td>Product Features</td>
<td>All included</td>
<td>Most important first</td>
</tr>
<tr>
<td>Testing documentation</td>
<td>End of the development cycle</td>
<td>Iterative and/or drives code</td>
</tr>
<tr>
<td>Documentation</td>
<td>Extensive documentation, consisting of requirements, design specifications and development plans. Heavily and rigorously use of documentation</td>
<td>Lean and mean. “Just enough” documentation</td>
</tr>
<tr>
<td>Training</td>
<td>Formal, facilitated training sessions. Often conducted in classrooms using static training materials</td>
<td>Informal training practices to enhance knowledge sharing, such as pair programming and daily stand-up meetings</td>
</tr>
</tbody>
</table>

**Note(s):** The table is inspired by Cram and Marabelli (2018), Hoda et al. (2008)
The way to embrace Agile practices can be unique, as each organization takes into account the specific needs of each area of their particular development process. Organizations, therefore, need help to understand how to choose the right combination of Agile practices (Campanelli et al., 2018). As a support for this choice, the next subsection will be a brief description of Agile practices.

### 2.3 Agile practices

The basis for this subsection is that recent research has found that project management methods can facilitate ambidexterity by prescribing patterns of alternating exploitation and exploration actions and by assigning specific roles (Sailer, 2019).

Scholars typically assume that Agile methods are the same as Agile practices (Collins and de Lucena, 2012). The Agile way of working can, therefore, be seen as a collection of methods or practices that are incremental, cooperative, straightforward and adaptive (Abrahamsson et al., 2017) with a core of common principles or values. It is important to point out that the Agile way of working is so much more than a way of working. It is a whole new way of thinking, i.e. a new mindset, which people absorb to varying degrees; as already mentioned, the difference between “being Agile” and “doing Agile” (Denning, 2016). “Doing Agile” explains adopting an Agile methodology or a set of Agile techniques, and simply following them. “Being Agile”, on the other hand, is embracing or “living with” the Agile values, principles, and the special Agile mindset (Denning, 2016). Therefore, flexible methods should be fully implemented instead of

<table>
<thead>
<tr>
<th>Practice</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Daily standup</td>
<td>Is a short timeboxed (15-minute) every-day meeting for the development team</td>
<td>Schwaber and Sutherland (2017)</td>
</tr>
<tr>
<td>Sprint/iteration planning</td>
<td>Sprint is a short time-boxed (interval of time) and the work to be performed in the sprint is planned at the sprint planning. This plan is created by the collaborative work of the entire Scrum Team</td>
<td>Beedle et al. (1999), Schwaber and Sutherland (2017)</td>
</tr>
<tr>
<td>Sprint retrospective</td>
<td>The sprint retrospective happens after the sprint review and before the next sprint planning. The sprint retrospective is an opportunity for the Scrum Team to review itself and create a plan for corrections to be enacted during the next sprint</td>
<td>Schwaber and Sutherland (2017)</td>
</tr>
<tr>
<td>Sprint/iteration review</td>
<td>This is an informal meeting at the end of the sprint to inspect the increment and adapt the product backlog if needed. During the sprint review, the scrum team and stakeholders collaborate about what was done in the sprint</td>
<td>Schwaber and Sutherland (2017)</td>
</tr>
<tr>
<td>Short iterations</td>
<td>Short iterations are needed for an effective product delivery to be able to handle the rapid change in, among other things, requirements management</td>
<td>Ahmed et al. (2010)</td>
</tr>
<tr>
<td>Continuous testing</td>
<td>Continuous testing is used to reduce the time and energy required to keep the code well tested and to prevent regression errors from remaining undetected long periods</td>
<td>Saff and Ernst (2004)</td>
</tr>
<tr>
<td>Self-organizing teams</td>
<td>Teams that have a common focus, mutual trust, and respect. Team members are collaborating, in a speedy, decision-making process and have the ability to deal with ambiguity</td>
<td>Cockburn and Highsmith (2001)</td>
</tr>
<tr>
<td>Pair programming</td>
<td>Two developers working together on the development and refinement of a piece of code at one computer</td>
<td>Abrahamsson et al. (2017)</td>
</tr>
</tbody>
</table>

Table 2. Agile practices

Note(s): The table is inspired by Highsmith (2002), VersionOne (2019)
using “cherry-picking” of practice without due regard for how they unite each other (Schwaber and Sutherland, 2017). Agile practices are, in other words, the visible, observable things you can see an Agile team doing. Table 2 is partly based on the annual survey on State of Agile from 2019 (VersionOne, 2019), where the five most popular Agile methods were reported (the first five in Table 2) and partly based on ASD practices by Highsmith (2002). According to Chau et al. (2003), plan-driven software methods and ASD methods use different training mechanisms (see Table 1). Formal training sessions are commonly used in plan-driven software methods. In addition to formal training sessions, much of the learning in Agile methods is done by informal practices, for example, pair programming (i.e. two developers working together on the development and refinement of a piece of code) and daily stand-up meetings (i.e. short meetings with project team members) (Chau et al., 2003).

The main principle of ASD is what we mentioned before about flexibility and the endeavor to improve project work constantly. It is accomplished by structuring work in short time stages, so-called sprints (Beedle et al., 1999), consisting of cycles or iteration of planning, performing, checking and then acting based on drawn conclusions (Schwaber and Sutherland, 2017). Continuous testing is a process that usually involves some prioritization of test cases and/or automation of the testing process (Fitzgerald and Stol, 2014). Prioritization and automation help decrease the time between the introduction of errors and their detection (Fitzgerald and Stol, 2014). Self-organizational teams and intense collaboration within and across organizational boundaries are crucial parts of Agile approaches (Cockburn and Highsmith, 2001). According to the Agile Manifesto, Beck et al. (2001) claim that the best architectures, requirements and designs arise from self-organizing teams. Agile approaches emphasize, furthermore, the empowerment of, and trust in, ASD project teams since the team is responsible as a unit for the development of the product (McAvoy and Butler, 2006).

Scrum is the most commonly used ASD method (e.g., Papadakis and Tsironis, 2018; Papatheocharous and Andreou, 2014) and has been adopted widely by various industries (Dingsøyr et al., 2012). Just as with the Agile way of working in general, the essence of Scrum is a small team of people, which is extremely flexible and adaptive (Schwaber and Sutherland, 2017). Scrum represents a new approach to planning and managing software projects, bringing the decision-making authority to the operational level, i.e. the self-managing team (Moe et al., 2010). Scrum takes the stage to solve the tension between the inherent complexity of software development and the outside world’s quest for novelty (Annosi et al., 2016). Self-management is a defining characteristic in Scrum, and the Scrum Team is given both authority and responsibility for several aspects of their work, such as planning, assigning tasks to team members and making decisions (Moe et al., 2012; Moe et al., 2010). The team is supposed to be cross-functional, i.e. a team of individuals who represent different domains of knowledge and work together to obtain a specific goal (Daspit et al., 2013).

Three key roles are defined in a Scrum Team: (1) a product owner, (2) the development team and (3) a scrum master (Schwaber and Sutherland, 2017). A product owner’s role is to be responsible for strategic decisions since, he or she that possesses the business perspective. However, in the study by Sverrisdottir et al. (2014), understandings of the role and responsibility of the product owner were shown to differ quite a lot between the studied organizations and were rarely similar to the official Scrum method. Members of the development team are jointly responsible for the end product and must develop shared mental models by assigning shared understandings of both the teamwork and the task (Levesque et al., 2001). The role of the Scrum Master is to be responsible for promoting and supporting Scrum by helping everyone to understand Scrum theory, practices, rules and values. The Scrum Master is, furthermore, a servant-leader of the development team (Noll et al., 2017; Schwaber and Sutherland, 2017). A Scrum Master has also a significant influence on Scrum’s practical results; for example, if there is a paradoxical tension that there is a great

Ambidexterity in Agile software development
A deal of business pressure to deliver results. As a result of this pressure, a Scrum Master may be inclined to become ambidextrous and shorten or skip more exploratory measures to give the team more time to do (rather exploitative) project work (Sailer, 2019).

A more in-depth explanation of the concepts of ambidexterity, exploitation and exploration will follow in the next subsection.

2.4 Exploitation, exploration and ambidexterity

The purpose of the study is to apply the concept of organizational ambidexterity as a conceptual lens to increase the understanding of tensions between exploitation and exploration in ASD project teams, and particularly the balancing (ambidextrous) strategies utilized. Therefore, the phenomenon of organizational ambidexterity with associated concepts needs to be explained by reference to interdisciplinary sources of the literature on organizational ambidexterity.

He and Wong (2004) state that exploitation and exploration require substantially different structures, processes, strategies, capabilities and cultures to pursue and may have different impacts on organizational adaptation and performance. A primary concern with the current literature, according to Laureiro-Martínez et al. (2010), is the lack of agreement about critical elements concerning the definitions of exploitation and exploration. Table 3 summarizes seven different definitions of the two terms that appear in key articles.

Although there is no single definition of exploitation and exploration, several researchers agree that a significant challenge for organizations is the balance between exploitation and exploration (March, 1991; Rosemann, 2014; vom Brocke et al., 2016). According to Raisch and Birkinshaw (2008), long-term success demands an organizational balance between continuity and change (see Figure 2). When managers resolve or handle paradoxical tensions, they contribute to an organization’s ability to pursue conflicting goals simultaneously, which is the core of ambidexterity (Gibson and Birkinshaw, 2004; Gregory et al., 2015). The concept of ambidexterity was first mentioned in the organizational literature by Duncan (1976), and then the term has been commonly used across many fields and disciplines. Ambidexterity has

<table>
<thead>
<tr>
<th>References</th>
<th>Exploitation</th>
<th>Exploration</th>
</tr>
</thead>
<tbody>
<tr>
<td>March (1991)</td>
<td>Includes production, efficiency, selection, implementation, refinement and execution</td>
<td>Includes elements captured by terms such as search, variation, risk-taking, experimentation, flexibility, discovery and innovation</td>
</tr>
<tr>
<td>Crossan et al. (1999)</td>
<td>Exploitation is about creating reliability inexperience and thrives on productivity and refinement.</td>
<td>Exploration is concerned with creating a variety in experience and thrives on experimentation and free association</td>
</tr>
<tr>
<td>Holmqvist (2004)</td>
<td>Learning from a top-down process</td>
<td>A bottom-up learning process</td>
</tr>
<tr>
<td>Lubatkin et al. (2006)</td>
<td>Refining and using existing knowledge</td>
<td>Innovation, problem-solving and creating new knowledge</td>
</tr>
<tr>
<td>Gupta et al. (2006)</td>
<td>Improvements in productivity, improvements in processes and product extensions and enhancements</td>
<td>Innovation and knowledge creation</td>
</tr>
<tr>
<td>Vidgen and Wang (2009)</td>
<td>Available knowledge within an organization is accessed and the same mistakes are not repeated</td>
<td>Using and creating new knowledge and producing new products, services, organizational arrangements or business models</td>
</tr>
</tbody>
</table>

Table 3. Definitions of exploitation and exploration
become a hot topic in organizational research, and Birkinshaw and Gupta (2013) argue that the reason is versatility. As a result, the number of studies using ambidexterity as a central concept has grown exponentially. In organizational ambidexterity research, it has been discussed that exploitation and exploration are interrelated and can enable each other, a so-called duality view (Farjoun, 2010; Sailer, 2019). Furthermore, Turner et al. (2018) state that the combination of exploitation and exploration is especially appreciated when considering how complexities are to be approached. One organizational paradox of exploration and exploitation is that exploitation generates the income needed to supply future exploration and exploration, which generates opportunities for future exploitation (Hughes, 2018; Lavie et al., 2010). Therefore, an organizational solution to the balancing of exploitation and exploration is ambidexterity.

2.5 Types of ambidexterity
Ambidexterity is about balancing equally important but seemingly paradoxical properties of an organization or a project. Prior research has recognized four fundamental approaches for managing the conflicting demands of exploitation and exploration: structural ambidexterity (organizational separation), sequential ambidexterity (temporal separation), domain separation and contextual ambidexterity (Hughes, 2018; Lavie et al., 2010). The notion of domain separation has recently been introduced and explained by Lavie et al. (2010), as that organization specializes in either exploitation or exploration in particular organizational domains while balancing these activities across domains. Different groups may serve to simultaneously explore and exploit in different domains. According to Hughes (2018), to achieve domain ambidexterity, an organization may specialize in either exploitation or exploration and then join with partners capable of offering the missing approach. Nevertheless, research on the domain-separation approach has been scarce and mostly limited. In our understanding, domain separation exists across organizational boundaries, which is why the domain-separation approach is not included in the current study. In Figure 2, the three different types of ambidexterity that are most recognized in the ambidexterity literature are depicted. The different approaches are also discussed in more detail below.

2.5.1 Structural ambidexterity. Structural ambidexterity refers to the conception of “dual structures” to separate conflicting demands into the responsibilities of different organizational units. In other words, structural ambidexterity or organizational separation involves distinct organizational units that either exploit or explore (Adler et al., 2009; Lavie...
Structural ambidexterity is composed of multiple subunits that are internally tightly coupled but loosely coupled to each other. A solution is to institute separate business units with different designs, tasks, cultures and processes focused on exploitation or exploration (Tushman and O'Reilly, 1996). Adler et al. (2009) state that the senior team must strategically integrate these sub-units. Markides (2013) calls this form of ambidexterity for spatial separation. In spatial separation, top- or corporate level managers should, according to Mom et al. (2007), be engaged in both exploitation and exploration activities, whereas business unit managers should focus on either exploitation or exploration activities, depending on the focus of their business unit. Hughes (2018) states that such a structural solution to balance exploitation and exploration suits large organizations better than small ones.

2.5.2 Sequential ambidexterity. Sequential ambidexterity or temporal separation proposes an alternative approach of organizational separation, whereby exploitation and exploration are separated over time rather than across organizational units. Lavie et al. (2010), Markides (2013), Papachroni et al. (2015) call this form of ambidexterity temporal separation. One example of sequential ambidexterity is when one focuses more on exploration in the early stages of a project (by exploring different types of technical solutions) and on exploitation at the end of the project during production/implementation (using existing knowledge). According to Gupta et al. (2006), Lavie et al. (2010), Papachroni et al. (2015), sequential ambidexterity is rooted in the notion of punctuated equilibrium. It means that temporal driving between long periods of exploitation and short bursts of exploration within the same organizational unit can be an alternative balancing mechanism. In this form of temporal separation, the managers need to shift their focus over time from pursuing incremental innovations or stability to pursuing radical innovations or strategic renewal, or vice versa (Mom et al., 2007).

2.5.3 Contextual ambidexterity. Contextual ambidexterity is the behavioral capacity to simultaneously pursue conflicting demands, such as exploitation and exploration across a business unit (Gibson and Birkinshaw, 2004; Pellegrinelli et al., 2015; Ramesh et al., 2012). These conflicting demands or paradoxical tensions are affected by how targets are set, by staff recruitment, by incentive systems and/or by organizational culture (Eriksson, 2013; Gibson and Birkinshaw, 2004; O'Reilly and Tushman, 2004). Contextual ambidexterity is realized by building a set of processes or systems that enable and encourage individuals to make their judgments about how to divide their time between conflicting demands for alignment (i.e. exploitation) and adaptability (i.e. exploration), rather than by creating dual structures (Tushman and O'Reilly, 1996). In the literature review made by Eriksson (2013), it is clear that most scholars focus on one or two of these different types of ambidexterity solutions. Recent research has, however, found that in reality, a combination of different solutions may be most practical, and Eriksson (2013) states that it is critical to discuss how all three types of ambidexterity can be utilized at different organizational levels.

2.6 Levels of ambidexterity

Achieving ambidexterity involves how ambidexterity is conceptualized, and if exploitation and exploration are perceived as competing with or complementing each other (Papachroni et al., 2015). Raisch et al. (2009) emphasize three levels of ambidexterity: organizational, group/team and individual (see Figure 1).

Ambidexterity can also be seen as “nested”, which means available on several levels simultaneously within the same organization (Birkinshaw and Gupta, 2013). However, research spanning several analysis levels is scarce (Raisch et al., 2009). Most studies of ambidexterity are focused on the organizational level (Laureiro-Martínez et al., 2010; Turner and Lee-Kelley, 2013). According to Raisch and Birkinshaw (2008), the level of analysis is
vitaly important because the choices of how to resolve the tension at one level of analysis are often resolved at the next level down. In addition, a single team may become ambidextrous by allocating different roles to each team member. O’Reilly and Tushman (2004) argue that one of the most important lessons is that ambidextrous organizations need ambidextrous senior teams and leaders. Laureiro-Martinez et al. (2010) are in the same vein when they argue for a better understanding of the individuals, their underlying nature, choices, abilities, purposes, expectations and motivations. According to Gibson and Birkinshaw (2004); Papachroni et al. (2015), the notion of contextual ambidexterity can be handled as a form of temporal separation at the individual level. It is accomplished when individuals are enabled to select either exploitation or exploration activities at different times, depending on the situation. In particular, individuals can maintain a balance between creativity, attention to detail and quality so that innovative performance does not necessarily undermine quality and efficiency (Lavie et al., 2010). In the scrutiny of contextual ambidexterity by Lavie et al. (2010), it reveals a micro level focus on either exploitation or exploration at a given time or location. In this approach, cultural values stimulate innovation to coexist with values of quality and efficiency. If we want to change behaviors in a system, we must first change the system’s underlying structure (Gibson and Birkinshaw, 2004; Markides, 2013).

2.7 Outcomes of ambidexterity
Successful organizations in dynamic environments have been viewed as ambidextrous, with a unique capability to balance current business and market needs and adapt to a changing environment for the future (Gibson and Birkinshaw, 2004). Ambidexterity has long been viewed as an essential driver of long-term performance (Raisch and Birkinshaw, 2008) leading to an organization’s survival (March, 1991).

Furthermore, ambidextrous organizations have a major opportunity to create vital strategic alliances with partners and effectively integrate their knowledge in a shared project (Rialti et al., 2018). Ambidexterity is also positively associated with performance in terms of capacity utilization and employee motivation (Lavie et al., 2010). In addition, Mom et al. (2009) found that a manager’s decision-making authority positively relates to their ambidexterity since their membership in cross-functional interfaces improves their ambidexterity, and the connectedness of the manager to other organization members positively relates to their ambidexterity.

3. Research method
To better understand tensions between exploitation and exploration in ASD project teams, a conceptual framework was constructed from interdisciplinary sources of the literature on ambidexterity. A literature review of publications on ambidexterity in the context of ASD was then conducted. To guide our literature review and the development of our conceptual framework, we followed two of the needed elements of theories: “what” and “how” (Chan and Thong, 2009; Whetten, 1989). The literature review helped us to identify the “what” factors and “how” strategies. All database queries were made in July 2020. In the review process, search terms (or keywords) were defined, and synonyms for the search terms were derived on the basis of the research questions. Specifically, we searched the titles, abstracts and keywords of publications contained in the databases using search terms. The search terms were constructed by joining the synonyms with the OR operator and AND operator for each element. Wildcards (“*”) have also been used in the search terms.

The search string used was: (“software development*” AND project*) AND (agile OR scrum OR “extreme programming” OR kanban OR practice* OR activit* OR procedur*) AND (ambidext* OR “exploitation and exploration” OR “exploitation versus exploration” OR paradox*). The search was performed in the following online research databases to locate relevant journal publications:
Scopus (18 publications), IEEEExplore Digital Library (2 publications), ACM Digital library (23 publications) and Business Source Premier (10 publications). Our initial search yielded 53 publications, and the number of unique publications after duplicates were removed was 49. The first author performed the searches in order to identify potentially relevant publications. Publications found were evaluated by reading the title, the keywords and the abstract. The evaluation resulted in 37 publications left for full-text screening. Publications were excluded if their focus was not on the context of ASD projects and ambidexterity or paradoxical tensions between exploitation and exploration. Publications were also excluded if they did not present empirical data. Also, it is to be noted that several publications were excluded because they addressed specific paradoxes, such as “the accuracy paradox” (Zimmermann et al., 2009), and “the ERP development paradox” (Johansson and de Carvalho, 2009), but did not explicitly mention paradoxical tensions between exploitation and exploration.

Eleven publications were finally included in the list of relevant publications documented in a spreadsheet program. We identified five publications that empirically examined the software development organizations as the unit of analysis, three publications that empirically examined the projects as the unit of analysis and three publications that empirically examined the team as the unit of analysis.

The included publications were coded by reading each publication thoroughly. NVivo[1] was used for the data extraction process as well as for the coding.

As the main essence of Agile methodologies involves interaction between time, task, team and transition, we used the conceptual framework in Lundin and Söderholm (1995, p. 451) (see Figure 1) as a “conceptual template” to categorize the ambidextrous factors and strategies from our literature review into the concepts of time, task, team and transition. A thematic analysis was conducted of the findings in the publications where a theme captures something significant about the data concerning the research question(s) (Braun and Clarke, 2006). In this study, a theme is the same as a factor and a strategy. In order not to confuse the reader, concepts are used for already established theoretical concepts from interdisciplinary theories. Examples of concepts in this study are: exploitation, exploration, ambidexterity, paradoxical tensions, time, task, team and transition.

In the current study, the first author coded based on the research questions:

RQ1: What ambidextrous factors in ASD are reported in previous studies?

RQ2: What strategies for achieving ambidexterity in ASD project teams are reported in previous studies?

The first author engaged in an evaluation of the ambidextrous factors and strategies (what and how) in the empirical findings in these publications. This evaluation included detailed readings of raw data to derive factors through interpretations made from the data by the researcher (Wood and McKelvie, 2015). The first step in the coding activity aims to break down data (quotes from the publications) into manageable pieces, factors and strategies. These factors and strategies are then categorized into the four concepts from Lundin and Söderholm (1995): time-related, work-related, team-related and transition-related. To gain a holistic understanding, we have also linked the concepts with Figure 2 and the contextual type of ambidexterity. We illustrate this interconnection in Figure 3.

RQ1 is a “what” question, and the ambidextrous factors from the publications were listed in the column called “Ambidextrous factor (what?)” in a spreadsheet. RQ2 is, furthermore, a “how” question, and ambidextrous strategies from the publications were listed in the column called an “Ambidextrous strategy (how?)”. Table 4 presents a summary of included publications on ambidexterity in ASD and their corresponding identified important factors. The identified important ambidextrous factors are explained in more detail in the next section.
A suitable pace strikes a balance such that the iteration cycle is long enough to get some meaningful work done but short enough not to lose momentum and responsiveness to change.

(Vidgen and Xiaofeng, 2009)

The team generally builds in a given slack time in estimates to allow for unknowns so that the team is not under the pressure to get story done as quickly as possible, rather, they can work to get quality code.

(Wang et al., 2008)

"A suitable pace strikes a balance such that the iteration cycle is long enough to get some meaningful work done but short enough not to lose momentum and responsiveness to change.

(Vidgen and Xiaofeng, 2009)

Allocated study time allows the team members to learn new things within working hours and helps to plan new things.

(Wang et al., 2008)
<table>
<thead>
<tr>
<th>Included article</th>
<th>Research design</th>
<th>Unit of analysis</th>
<th>Research question</th>
<th>Data collection method</th>
<th>Ambidextrous factor (what?)</th>
<th>Ambidextrous strategy (how?)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cram and Marabelli (2018)</td>
<td>A longitudinal case study</td>
<td>Organization</td>
<td>How do knowledge-sharing processes associated with ISD projects change when agile techniques are increasingly used alongside traditional techniques?</td>
<td>32 interviews and 48 company documents</td>
<td>Knowledge creating and sharing</td>
<td>Mixing explicit and tacit knowledge</td>
</tr>
<tr>
<td>Fontana et al. (2015a, b)</td>
<td>A multiple-case study</td>
<td>The agile software development team</td>
<td>How do teams mature in agile software development?</td>
<td>14 interviews and two types of questionnaire</td>
<td>Dual behaviors</td>
<td>Strategies that foster experimenting and learning</td>
</tr>
<tr>
<td>Lee et al. (2006)</td>
<td>Not specified</td>
<td>Globally distributed software projects</td>
<td>Not specified</td>
<td>22 semi-structured, face-to-face/telephone interviews</td>
<td>Communication</td>
<td>Have coping strategies, Frequent planning, Varying communications options</td>
</tr>
<tr>
<td>Napier et al. (2008)</td>
<td>A two-year action research study</td>
<td>Project portfolio management</td>
<td>Not specified</td>
<td>32 interviews, two workshops, 18 meetings, diagnosing report</td>
<td>Balanced practices</td>
<td>Actions and interactions related to building alignment and adaptability, Creating an organizational context</td>
</tr>
</tbody>
</table>

(continued)
<table>
<thead>
<tr>
<th>Included article</th>
<th>Research design</th>
<th>Unit of analysis</th>
<th>Research question</th>
<th>Data collection method</th>
<th>Ambidextrous factor (what?)</th>
<th>Ambidextrous strategy (how?)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Ramesh et al. (2012)</strong></td>
<td>A multi-site case study</td>
<td>Agile distributed teams</td>
<td><strong>RQ1</strong>: What is the exact nature of the flexibility-stability conflicts that arise in agile distributed product development settings? <strong>RQ2</strong>: What mechanisms do software teams devise to achieve a balance between flexibility-stability in agile distributed product development settings?</td>
<td>25 on-site and phone interviews and the review of documentation</td>
<td>Performance management, Balanced practices, Processes and systems as enabler</td>
<td>Fast-cycle feedback, development of collective identity, formal structures but with flexibility, process assimilation before delivering, mixing explicit and tacit knowledge, using productivity tools, creating formal structures but with flexibility, process assimilation before delivering, giving support and building trust and using meta-routines</td>
</tr>
<tr>
<td><strong>Temizkan and Kumar (2015)</strong></td>
<td>Tested hypotheses by using a combined dataset</td>
<td>Open source software (OSS) development projects</td>
<td><strong>RQ1</strong>: Does the network structure of artifact teams have a differential impact on OSS artifact development performance depending on the type of artifact? <strong>RQ2</strong>: Does heterogeneity exist in the network structure of developers depending on the type of artifact?</td>
<td>Dataset included 3,059 observations (projects)</td>
<td>Ambidextrous developers</td>
<td>Integrating and controlling information between exploitation and exploration teams</td>
</tr>
</tbody>
</table>

(continued)
<table>
<thead>
<tr>
<th>Included article</th>
<th>Research design</th>
<th>Unit of analysis</th>
<th>Research question</th>
<th>Data collection method</th>
<th>Ambidextrous factor (what?)</th>
<th>Ambidextrous strategy (how?)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Saxena et al. (2016)</td>
<td>2 case studies</td>
<td>Agile Distributed Development project</td>
<td>RQ1: What is the exact nature of the flexibility-stability conflicts that arise in agile distributed product development settings? RQ2: What mechanisms do software teams devise to achieve a balance between flexibility-stability in agile distributed product development settings?</td>
<td>Interviews</td>
<td>Balanced practices</td>
<td>Mixing hard and soft elements, role specification and giving support and building trust</td>
</tr>
<tr>
<td>Wang et al. (2008)</td>
<td>2 case studies</td>
<td>Team</td>
<td>Not specified</td>
<td>Interviews</td>
<td>Slack time</td>
<td>Planning for slack time, acquiring new skills and reviewing process</td>
</tr>
<tr>
<td>Vejseli et al. (2018)</td>
<td>Interview-study</td>
<td>Organization</td>
<td>How is the concept of an effective ITG framework changing in response to the demand for agility in organizations?</td>
<td>33 interviews and an questionnaire</td>
<td>Governance structures</td>
<td>Mixing governance structures</td>
</tr>
<tr>
<td>Vidgen and Xiaofeng (2009)</td>
<td>2 case studies</td>
<td>Organization</td>
<td>Not specified</td>
<td>14 interviews</td>
<td>Time-pacing, Balanced practices</td>
<td>Using fixed-length sprints, reserving time for study and daily feedback sessions</td>
</tr>
</tbody>
</table>
4. Ambidextrous factors and strategies in ASD

This section provides an overview of the findings from the analysis of the existing ASD literature in order to answer the research questions:

RQ1: What ambidextrous factors in ASD are reported in previous studies?

RQ2: What strategies for achieving ambidexterity in ASD project teams are reported in previous studies?

We extended the conceptual framework in Figure 2 with ambidextrous factors and strategies identified in the studied (empirical) publications, drawing principally from the included publications that we have discussed in Section 3 and from our interpretation of these publications (Boxall and Gilbert, 2007). We used the conceptual framework in Lundin and Söderholm (1995, p. 451) (see Figure 1) as a “conceptual template” to categorize the ambidextrous factors and strategies from our literature review into the concepts of time, task, team and transition (see Figure 4).

In the subsections below, the ambidextrous factors and strategies found through the study are marked in boldface, and the same factors and strategies are found in Figure 4.

4.1 Time-related factors and strategies

Our study identifies several Agile practices related to the time dimension, thus indicating that time management is an important strategy in the efforts toward ambidexterity. Time-related factors include time-pacing and slack time. Time-pacing or temporal pacing is suggested in the study of Vidgen and Wang (2009), as a way of combining flexibility and control in turbulent environments (Vidgen and Wang, 2009). Time-pacing affords a basis for correct planning. Using fixed-length sprints is one strategy to reach time-pacing (Vidgen and Wang, 2009). A challenge for a team is to decide on a pace that can be persistent over time. As Vidgen and Wang (2009, p. 369) emphasize:

The iteration cycle is long enough to get some meaningful work done but short enough not to lose momentum and responsiveness to change. Once a pace is set it is important to stick to it as the regular pacing. This brings stability to a team and small, frequent closures at the end of each boxed time period help keep team members satisfied and motivated.

The ambidextrous factor, slack time, means that the team makes a time schedule that allows for contingencies so that the team is not under pressure to get a user story (an informal, natural language description of one or more features of a software system) done as quickly as possible. Rather, they can focus on getting quality code (Wang et al., 2008). The strategy is therefore to plan for slack time. According to Vidgen and Wang (2009), exploration needs to be organized, and slack time is needed to foster the emergence of new ideas. Further, Vidgen and Wang (2009) state that the strategy of reserving daily time for one hour of project-related and non-project-related study respectively will facilitate synchronizing between exploitation and exploration.

4.2 Task-related factors and strategies

The task-related concept (Lundin and Söderholm, 1995) highlights that the purpose of temporary organizations is action because the focus is a shift from the decision (choosing the right goal) to the action (completing the task). A majority of the findings in this study are related to tasks and show that a combination of both exploitation and exploration is needed.

Furthermore, innovations in the shape of new products and services are one task where a combination of exploitation and exploration is found favorable. Chan et al. (2019) argue that small- and medium-sized enterprises (SMEs) need a good balance between exploitative innovation and explorative innovation. Exploitative innovation focuses on capitalizing
An extended conceptual framework of ambidextrous factors and strategies in ASD.

**Figure 4.**

An extended conceptual framework of ambidextrous factors and strategies in ASD.
existing resources and capabilities, often extending them incrementally to serve existing markets and consumers. As such, exploitative innovation usually results in a near-term gain. In contrast, explorative innovation focuses on developing new products or services to address the emerging market conditions and emerging consumer expectations.

Cram and Marabelli (2018, p. 324) define exploitation and exploration as follows:

Exploitation refers to the use of existing knowledge, such as at a firm with expertise pertaining to a traditional development approach, while exploration is increasingly focused on exploring new knowledge, such as that created through iterative, collaborative interactions between agile team members.

Fontana et al. (2015a) propose strategies that foster experimenting and learning in order to combine exploitation and exploration. The team should be aligned with specific outcomes—exploitation—but free to adapt practices as they please—exploration. Fostering alignment with clear expected outcomes (exploitation), but leaving space for the appearance of variety (exploration) by not defining the practices the team should implement (Fontana et al., 2015a).

Another strategy for software organizations to become ambidextrous is through the processes, i.e. actions and interactions, between interested parties as they attempt to transform practices related to building alignment and adaptability Napier et al. (2008). On the other hand, Ramesh et al. (2012) refer to a ambidextrous factor called performance management, a factor that includes discipline and stretch and is empowered by the strategy of fast-cycle feedback and the development of collective identity.

Project managers should also recognize the importance of balanced practices (Ramesh et al., 2012) as factors and strategies in order to achieve ambidexterity. Balanced practices on the strategic level might be divided into operationally focused balanced strategies such as creating formal structures but with flexibility and process assimilation before delivering quick value, empower performance management (discipline and stretch). Information systems must also support the needs of the balanced practice, for example by means of a strategy for using productivity tools (e.g. communication, collaboration and modeling tools) (Ramesh et al., 2012).

Another classification is relationally focused balanced practices, for example, in the form of trusting but verifying and cohesive but distributed project teams empowering social context (support and trust) (Ramesh et al., 2012). In the same vein, the balancing between flexibility and stability through the strategy of mixing hard (performance) and soft (social) elements was investigated by Saxena et al. (2016). The performance elements are characterized by discipline and stretch and manifested in the form of the extent of role specification within software teams. O’Reilly and Tushman (2004) are in the same vein when they argue that a single team may become ambidextrous by allocating different roles to each team member. Social elements are characterized by the strategy of giving support and building trust (Saxena et al., 2016). This strategy is of utmost importance because flexible approaches emphasize the empowerment and trust of ASD project teams, since the team is responsible as a unit for the development of the product (McAvoy and Butler, 2006).

Balanced practices on the project level, according to Vidgen and Wang (2009), are the development of the product through the iterative delivery of user stories balanced with routinized exploration, in which team members can search new ideas and new areas. Another practice and ambidextrous strategy is the daily feedback session with emphasis on the progress of the previous day, focusing on positive aspects as well as issues arising. The daily feedback session helps the team exploit and retain what the team is doing well. In these sessions, teams have the opportunity to reflect if processes need continuous adjustment and adaption to avoid inflexibility and deterioration (Vidgen and Wang, 2009). This practice is also an ability to test the robustness and is also an ability to experiment (Vidgen and Wang, 2009).
Having coping strategies for flexibility and rigor in ASD is of utmost importance, especially in global projects, as globalization increases the complexity of coordination (Lee et al., 2006). They found seven categories of coping strategies: common platform, labor organization, education/understanding, technology readiness, doing more, awareness/teamwork and adaptive use of technology (Lee et al., 2006). These coping strategies often played different roles in increasing flexibility or rigor; some strategies primarily contributed to flexibility and others mainly contributed to rigor (Lee et al., 2006). One example of a coping strategy for balancing the paradoxical tensions between rigor and flexibility is frequent planning (Lee et al., 2006). This strategy or practice is a response to the evolving user requirements and other uncertainties that the teams using the Agile processes are confronted with all the time. They do follow the plan they come up with, but also frequently adjust their plans through task prioritization (Lee et al., 2006). Repetitive and frequent planning is the natural consequence of frequent feedback loops in ASD projects due to the close relationships between a team and their customers as well as among team members (Lee et al., 2006).

Knowledge creation and sharing are central parts in ASD projects. Chan et al. (2019) because ASD strongly emphasizes constant communication among team members and customers, mainly through face-to-face interaction (Cockburn and Highsmith, 2001). In an ASD context, knowledge sharing refers to the transfer of both tacit and explicit knowledge among project stakeholders (Cram and Marabelli, 2018). They define knowledge viewed as an “object” that can be exchanged in the process of documentation in compliance with plan-driven software development approaches. This can be compared to the knowledge that is seen as a “relationship” and is exchanged in face-to-face meetings in compliance with an Agile development approach (Cram and Marabelli, 2018).

4.3 Team-related factors and strategies
In the Agile way of working, the team is given a central role, and ASD strongly emphasizes constant communication among team members and customers (Cockburn and Highsmith, 2001). In ASD project teams, an ambidextrous strategy is to varying communication options from which the team members can choose based on what works or does not work for a particular stage of the project. Formal communication is important because informal communication (e.g. face-to-face meetings) is often less effective due to cultural differences, language barriers and organizational boundaries (Lee et al., 2006). Ambidexterity can be achieved in the context of ASD projects by using a mix of formal and informal communication. Instead of focusing on using formal channels for some tasks and informal channels for others, Ramesh et al. (2012) advise developers to adopt tools that enable a suitable mix by improving informal communication while supplementing it with “just enough” documentation (Ramesh et al., 2012). Lee et al. (2006), however, state that detailed, comprehensive documentation and codified explicit knowledge are critical in global contexts because communication is problematic, and tacit knowledge is difficult to share.

To ensure alignment and adaptability within a project team, the top management team is responsible for creating an organizational context that enables ambidextrous balanced practices (Napier et al., 2008; Ramesh et al., 2012). In the same vein, Chan et al. (2019) argue that owner-managers and top management teams ought to adopt an open and flexible mindset toward innovation, business strategies and the organizing of products and structures. Furthermore, regarding the management level, Fontana et al. (2015b) state that ambidexterity is a critical ability to maturity, and the development is guided by outcomes that Agile teams pursue, instead of recommended practices. In ASD projects, contextual ambidexterity can be achieved by using processes and systems that enable the simultaneous pursuit of alignment and adaptability within a project team (Fontana et al., 2015b). Their suggestion is, therefore, to foster alignment (exploitation) with clear, predictable outcomes while at the same time leaving space for the emergence of diversity (exploration) without recommending the
concrete practices the team should implement. Fontana et al. (2015b, p. 93) conclude that ASD project teams are complex adaptive systems on the team level:

They are driven by a self-organized behavior, which challenges the management to use strategies that fosters experimenting and learning. The evolution—and the maturing—of this system is a discontinuous process of combining exploitation and exploration. If this combination is successful, it will lead to a higher performance through ambidextrous abilities.

Their study, also found that the behavior of Agile practitioners is a duality: the practitioners are serious professionals with a free-spirit and “joking” behavior (Fontana et al., 2015b). In order to reach ambidexterity, the team members need to become multidisciplinary skilled. Generally, in ASD project teams, the developers choose tasks they feel confident in completing, but they also pick up tasks that they are not good at, and then look for help at the daily planning time and work with a more skilled member through pair programming, in order to acquire new skills (Wang et al., 2008). Ramesh et al. (2012) highlight the importance of maintaining and fostering working relationships. In fact, a strategy by explicit and tacit knowledge is needed to be combined in ambidextrous ASD project teams (Ramesh et al., 2012). Organized exploration explicitly acknowledges and inspires team members’ desire to learn, but at the same time, it decouples learning from development activities so that team members can focus and separate exploitation from exploration (Vidgen and Wang, 2009).

Exploitation and exploration activities are described in different ways, depending on the software being developed. In Open Source Software (OSS) development, Temizkan and Kumar (2015) indicate that the patch development activities can be considered as exploitation activities. On the other hand, feature-request activities can be considered as exploration activities. Furthermore, Temizkan and Kumar (2015) propose that developers who are part of both exploitation and exploration networks can be called as ambidextrous developers. Ambidextrous developers integrate and control information between exploitation and exploration teams. In the interdisciplinary sources on ambidexterity, O’Reilly and Tushman (2004) argue that one of the most important lessons is that ambidextrous organizations need ambidextrous senior teams and leaders. Therefore, a better understanding of the individuals, their underlying nature, choices, abilities, purposes, expectations and motivations is needed (Laureiro-Martinez et al., 2010). It is important to reiterate that ambidexterity affects the different organizational levels (Birkinshaw and Gupta, 2013) because exploitation and exploration activities need specific network structures, based on the characteristics of each activity (patch development or feature request) (Temizkan and Kumar, 2015).

4.4 Transition-related factors and strategies

A paradoxical view on the people vs. processes contradiction manifested in ASD advocates that both people and processes are important factors, and both should be taken care of simultaneously in the same development process (Wang et al., 2008). The findings from Ramesh et al. (2012) indicate the need to develop mixed capacities in order to simultaneously pursue plan-driven software development and ASD, rather than generating dual structures that disperse the two types of development. Especially organizations with distributed ASD projects can achieve contextual ambidexterity by the strategy of using processes and systems that enable the simultaneous pursuit of alignment and adaptability within project teams (Ramesh et al., 2012). In an ASD process, spontaneous interactions happen all the time, but team members are not left to their team to interact. There are supporting structures in the process, derived from the interconnections of practices, which create a favorable managerial and cultural environment for interactions to happen (Wang et al., 2008). The Agile process needs continuous adjustment and adaptation to avoid rigidity and deterioration. A regularly reviewing process of ASD projects can help the team take gradual steps to change and improve the ASD process (Wang et al., 2008).
Furthermore, process reviewing is seeking opportunities to change and responding to change. Process reviewing lets people use their common sense, but at the same reflect on and criticize common senses that are accepted without questioning (Wang et al., 2008). One of the balanced practices, meta-routines, can be used to systematize nonroutine processes so that a balance between innovative and routine processes is achieved (Ramesh et al., 2012). On an organizational level, IT governance both ensures that tasks are completed with efficiency and reliability, constantly and incrementally improving itself and handles today’s increasingly strategic challenges with speed and agility. Thus, traditional hierarchical governance structures should be complemented by network-like governance structures that can react quickly to changes in the organization’s environment (Vejseli et al., 2018).

Chan et al. (2019), who studied software development in SMEs, describe that the mechanism of organizational ambidexterity is involved in both developing innovative capability and mitigating organizational rigidity. The strategy to overcome the organizational rigidity is to balance the resources. An example is to bring in interns and assign them to explorative innovation projects (Chan et al., 2019). These interns can bring fresh perspectives on existing trends and technology and help the organization to explore new technology and product opportunities (Chan et al., 2019). According to Chan et al. (2019), SMEs need to harness flat structures to be organizationally flexible to achieve both exploitative innovation and explorative innovation. SMEs should furthermore proactively manage organizational ambidexterity by appropriating and leveraging external resources where feasible to deal with internal gaps and constraints.

5. Discussion and future works
This study builds on previous organizational and ASD research to provide a clarification of the dynamic capacity of ambidexterity, and how to manage paradoxical tensions between exploitation and exploration successfully. Our study is in line with previous research indicating that the relationship between exploitation and exploration can best be thought of as a duality rather than a dualism, each mode enabling and supporting the other (Farjoun, 2010). Therefore, organizations must change the perspective on paradoxes and accept that it is possible to use paradoxes as a source of creative tension and fusion (Papachroni et al., 2015; Poole and Van de Ven, 1989). As the main essence of Agile methodologies involves interaction between the concepts of time, task, team and transition, we used the conceptual framework in Lundin and Söderholm (1995, p. 451) (see Figure 1) as a “conceptual template” to categorize the ambidextrous factors and strategies from our literature review. It should be noted that time, task, team and transition concepts are related to each other. For example, “reserving daily time for studies” can be seen as a time-related strategy, a team-related strategy, and a task-related strategy to facilitate ambidexterity.

Previous research on organizational ambidexterity has focused on “what” ambidexterity is (especially at the organizational level) instead of “how” it can be expressed concretely (especially at the team level). The new perspective that this study contributes is that we first structured and categorized how previous empirical research has explained the ambidextrous factors in the context of ASD. Besides, we identified examples of how the paradoxical tensions between exploitation and exploration are balanced (ambidextrous strategies), in previous empirical studies. Not surprisingly, we found in the results that most of the factors and strategies were task and team-related. This may be due to the fact that software development has a high degree of interchangeability and can also be seen as a complex activity (Boehm and Turner, 2004). In addition, ASD methods are strongly focused on the team and its interactions (Beck et al., 2001).

It is well known that ASD is seen as an opportunity to meet the requirements for changeability (exploration) (Abbas et al., 2008). However, excessive exploration can put the
organization in an endless cycle of change (March, 1991). Therefore, it should be noted that there are also exploitative functions and ambidextrous strategies in ASD methods. For example, short iteration is one of the most central practices for ASD (Ahmed et al., 2010). An ambidextrous strategy is to determine that the iteration cycle is long enough to get some meaningful work and stability, but short enough not to lose speed and sensitivity to change (Vidgen and Wang, 2009). One of the misconceptions regarding ASD is that it is not allowed to document or plan. However, in ASD projects, unlike plan-driven software development, planning activities instead consist of shorter cycles or iterations (Chau et al., 2003). In this study, we found that the strategy of frequent planning is an example of a management strategy for balancing the paradoxical tensions between rigor and flexibility is (Lee et al., 2006).

Another example of an ASD practice is pair programming (see Table 2). Pair programming is an opportunity to gain new skills (exploration) by working with a more skilled team member (Wang et al., 2008) and at the same time ensure that you create quality code (exploitation) (Spohrer et al., 2013).

Self-management is also a crucial element in ASD methods, where ASD teams are given both authority and responsibility, specifically in relation to making decisions (John McAvoy and Butler, 2009; Moe et al., 2010, 2012). In the small self-organizing teams, members must handle all tasks. Therefore, it is necessary to trust and help each other and share knowledge. A prerequisite for being as creative and flexible (exploration) as possible is that the team is stable. Team members must also have control over their own work and be able to plan and solve their own tasks (exploitation) (Dingsøyr et al., 2012). In this study, we found that strategy of giving support and building trust is an important strategy for ambidexterity in AS (Saxena et al., 2016).

One pattern we can see from the results in this study is that ASD project teams need a mix of hard (performance) strategies and soft (social) strategies to be ambidextrous (Ramesh et al., 2012). Gustavsson and Hallin (2014, p. 570) point out that “hard” is related to the rational and technical side of projects and project management and “soft” is related to the human side of the same. Examples of hard strategies in this study are frequent planning (Lee et al., 2006), proposing strategies that foster experimenting and learning (Chan et al., 2019), creating formal structures but with flexibility (Ramesh et al., 2012) and role specification (Saxena et al., 2016). On the other hand, soft strategies in this study are, for example, giving support and building trust (Ramesh et al., 2012), daily feedback sessions (Vidgen and Wang, 2009) and varying communication alternatives (Lee et al., 2006).

The expanded conceptual framework (Figure 4) offers both theoretical and practical contributions. In terms of theoretical contributions, the framework is based on an ambidextrous perspective to provide a new picture of factors and strategies for achieving ambidexterity on a concrete level. The theoretical contribution is, therefore, an identification of a broader set of ambidextrous factors and strategies that can affect ASD project teams. The conceptual framework may furthermore be a basis for future empirical research.

We further believe that the transition from plan-driven software development to ASD can lead to paradoxical tensions. The significant differences in philosophy and thinking between plan-driven software development and ASD can lead to paradoxical tensions in the transition to Agile methods. An example of such tension is that it can be difficult to unlearn old, traditional practices and to move toward new ones (e.g., Thangasamy, 2012). Abrahamsson et al. (2017) state that the transition to ASD can be seen as a paradigm shift and it is therefore important that the new, flexible way of working is understood, accepted and supported by the entire organization, i.e., all stakeholders. The importance of this paradigm shift is of the utmost importance to know when companies should begin their journey to “being Agile”, not only “doing Agile”.

Generalizations from the study should be made with caution in light of its limitations. First, only 11 publications were used. Searching keywords in full publications, not only in the title, keywords and abstract, could potentially extend the search. Future work can thus be
more inclusive concerning search terms and selection of databases. Second, there is certainly a great deal written about Agile practices used to balance exploitation and exploration, but where the concepts or keywords of this study have not been used in the title, keywords and abstract. Thus, these publications have not been included in the results. This is because theoretical insights about ambidexterity are novel to the research field of ASD, and we believe they deserve further investigation with complementary research methods and techniques, which can enhance the generalizability of our findings. More empirical studies are furthermore needed to examine the critical trade-offs resulting from the balance efforts and how to learn more about the driving forces and techniques used by organizations to balance exploitation and exploration.

6. Concluding remarks
The purpose of this study is to apply the concept of organizational ambidexterity as a conceptual lens to increase the understanding of tensions between exploitation and exploration in ASD project teams, and particularly the balancing (ambidextrous) strategies utilized. The study addresses the concept of ambidexterity by identifying concepts and proposing a conceptual framework, which is shown in Figure 2. Also, an expanded conceptual framework (Figure 4) provides insight into what constitutes ambidexterity in ASD project teams.

Our study uses concepts from the theory of organizational ambidexterity and from project management, to provide the initial framework required to develop an understanding of how ASD project teams balance conflicting demands. Many studies on ASD projects have either been inspired by industry practices or by practical considerations in implementing them. This has resulted in a lack of theoretically grounded research in the area of ASD. A literature review was conducted to answer the research questions in this study. Search terms were defined, and synonyms for the search terms were derived. Then a search was performed in four databases. The number of unique publications after duplicates had been removed was 49. All in all, 11 publications were included in the study.

This study brought together concepts from academic domains of knowledge and factors and strategies from practitioner domains of knowledge to better understand a complex, multifaceted problem in a new way. The study addresses the notion of ambidexterity by identifying ambidextrous factors and strategies and proposing an extended conceptual framework. The conceptual framework provides insights into what constitutes ambidexterity in ASD project teams. Interest in the phenomena of paradoxical tensions and the strategy (ambidexterity) to deal with these tensions is growing (Hughes, 2018) since more and more organizations struggle to address rapidly changing environments (Cooper and Sommer, 2016; Farjoun, 2016; Pellegrinelli et al., 2015).

The contribution is of great importance for ASD research and practice since ASD methods are a popular method for managing projects not only within ASD but also in other industries and sectors. The contribution of this study can, therefore, be applied outside the ASD area.

Note
1. https://www.qsrinternational.com

References


Thangasamy, S. (2012), Lessons Learned in Transforming from Traditional to Agile Development.


Zimmermann, T., Nagappan, N., Gall, H., Giger, E. and Murphy, B. (2009), Cross-Project Defect Prediction: A Large Scale Experiment on Data vs. Domain vs. Process.

Corresponding author
Carin Lindskog can be contacted at: carin.lindskog@kau.se

For instructions on how to order reprints of this article, please visit our website: www.emeraldgrouppublishing.com/licensing/reprints.htm
Or contact us for further details: permissions@emeraldinsight.com