ASSOCIAÇÃO DE POLITÉCNICOS DO NORTE (APNOR)

INSTITUTO POLITÉCNICO DE VIANA DO CASTELO

AGILE METHODOLOGIES IN PROJECT MANAGEMENT: A STUDY IN SMALL COMPANIES DEVELOPING DIGITAL PRODUCTS

Marlene Manuela Ferreira Ribeiro

Dissertação apresentada ao Instituto Politécnico de Viana do Castelo para obtenção do Grau de Mestre em Gestão das Organizações, Ramo de Gestão de Empresas

Orientada por

Professora Doutora Ana Teresa Martins Ferreira de Oliveira

Professora Doutora Sara Maria da Cruz Maia de Oliveira Paiva

Viana do Castelo, Outubro de 2025.

ASSOCIAÇÃO DE POLITÉCNICOS DO NORTE (APNOR)

INSTITUTO POLITÉCNICO DE VIANA DO CASTELO

AGILE METHODOLOGIES IN PROJECT MANAGEMENT: A STUDY IN SMALL COMPANIES DEVELOPING DIGITAL PRODUCTS

Marlene Manuela Ferreira Ribeiro

Dissertação apresentada ao Instituto Politécnico de Viana do Castelo para obtenção do Grau de Mestre em Gestão das Organizações, Ramo de Gestão de Empresas

Orientada por

Professora Doutora Ana Teresa Martins Ferreira de Oliveira

Professora Doutora Sara Maria da Cruz Maia de Oliveira Paiva

Viana do Castelo, Outubro de 2025.

Abstract

The present study investigates the implementation of Agile project management methodologies in small companies developing digital products in the Minho region of Portugal. Through semi-structured interviews with 11 project managers analysed using template analysis, this research explores the methodologies employed, the rationale behind their choice, the practices involved, the software and tools that support them, how teams are organised, the performance metrics they use, and the main challenges they face.

Findings suggest that although Scrum and Kanban are the dominant methodologies, their application is highly contextual. Teams frequently alternate between them based on project characteristics, combine them into hybrid methodologies like Scrumban, or adopt unlabelled approaches that selectively incorporate practices from multiple others. Tailoring Agile methodologies and practices is the small companies' strategy to suit their unique constraints and needs.

Teams employ diverse software applications for overall project management, communication (especially pertinent for distributed teams and remote work), design collaboration, and code. Role overlap is frequent, reflecting cross-functionality as a way to overcome limited resources.

Time-based key performance indicators are common, accompanied by practices such as time tracking per task and story points converted directly in time. However, there was also a notable presence of indicators of different nature, addressing all three aspects of the Agile triangle: constraints, value, and quality. Despite this balanced approach, several challenges emerge, including inaccurate estimations and fluctuating client requirements.

Keywords: Agile methodologies, Agile tailoring, project management, small companies, digital products.

Resumo

O presente estudo visa investigar a implementação de metodologias Agile na gestão de projetos em pequenas empresas que desenvolvem produtos digitais na região Minho, em Portugal. Através de entrevistas semiestruturadas com 11 gestores de projeto, analisadas com recurso a template analysis, esta pesquisa procura compreender as metodologias adotadas, as razões subjacentes à sua escolha, as práticas associadas, o software e as ferramentas que as suportam, como as equipas se organizam, que métricas de desempenho são utilizadas e os principais desafios enfrentados.

Os resultados revelam que, embora Scrum e Kanban sejam as metodologias predominantes, a sua aplicação varia consideravelmente em função do contexto. Existem empresas que alternam entre as duas, consoante as características do projeto. Outras combinam-nas em metodologias híbridas como o Scrumban. E outras organizações há ainda que seguem abordagens próprias, selecionando práticas de várias metodologias e sem conseguirem encontrar um rótulo. A customização de metodologias e práticas ágeis é uma estratégia empregue pelas pequenas empresas para responder às necessidades, considerando os seus constrangimentos específicos.

Há uma diversidade de aplicações e software em uso para as finalidades de gestão global de projetos, comunicação (aspeto particularmente pertinente em caso de trabalho remoto e equipas distribuídas), e colaboração em design e em código.

Em termos de papéis, observou-se que a sobreposição de funções é recorrente, o que sugere que a transversalidade pode atuar como um mecanismo compensatório na limitação de recursos.

Indicadores de desempenho relacionados com o fator tempo são comuns, e geralmente acompanhados por práticas como registo de tempo por tarefa e conversão direta de story points em tempo de execução. Contudo, verifica-se igualmente a presença de indicadores de outras naturezas, abrangendo os três vértices do triângulo Agile: constrangimentos, valor e qualidade. Não obstante esta abordagem equilibrada, emergem vários desafios, entre os quais estimativas imprecisas e requisitos altamente variáveis.

Palavras-chave: Metodologias ágeis, customização de metodologias ágeis, gestão de projeto, pequenas empresas, produtos digitais.

Dedication

To all those who live by the mantra of continuous improvement.

Acknowledgments

This work would not have been possible without the pragmatic guidance and unwavering support of my mentors, Ana Teresa Oliveira and Sara Paiva. I am particularly grateful to Professor Ana Teresa for her infectious enthusiasm, which brought Professor Sara into this research.

I extend my sincere thanks to the companies and the individuals who generously shared their time and insights—thank you for believing in the value of this study.

My heartfelt appreciation goes to my family, for allowing me the space and time necessary to complete this project, even at the cost of shared moments and uneven housework. A special tribute to Joel for his infinite patience and understanding.

List of Abbreviations and/or Acronyms

AM – Agile Mindset

APNOR – Associação de Politécnicos do Norte

CECSVS-IPVC – Comissão de Ética para as Ciências Sociais, da Vida e da Saúde

IPVC – Instituto Politécnico de Viana do Castelo

IT – Information Technology

JIT – Just in Time

LSD – Lean Software Development

LS – Lean Startup

TA – Template Analysis

WIP – Work in Progress

XP - Extreme Programming

General Index

Figures Index

Tables Index

Introduction

The continuous evolution of technology and its ever-accelerating pace lead to uncertainty and risk. It presents a significant challenge for companies, requiring a rapid adaptation to remain competitive.

Handling the resources of a project in a way that can achieve the best possible outcome is the goal of every project manager. We understand project management as the “application of knowledge, skills, tools, and techniques to project activities” (Project Management Institute, 2021, p. 4) with the purpose of delivering quality and value to the customer while optimally addressing the project constraints of scope, time and budget (Highsmith, 2009). One must consider that there are a multitude of methodologies that could be employed.

Traditional methods, also known as plan-based approaches, such as Waterfall, Iterative or Spiral, while very useful in relatively stable contexts where requirements are known *a priori*, may be too heavy and not as efficient for digital product development (Al-Saqqa et al., 2020). In this context, lightweight Agile methodologies appear to be effective approaches to project management. The principles of prioritising delivering value to customers and involving them in the development process, giving teams equal autonomy and responsibility, delivering working software/digital products rather

than writing extensive documentation, and maintaining the ability to adapt to change rather than following a rigid and unchanging plan (Beck et al., 2001) have become part of the culture of many organisations. However, organisations substantially differ in the methods and tools they use to implement these principles.

Kanban, Scrum and Extreme Programming (XP) are among the most common Agile methodologies, and each of them can be used alone or in combination with others. Agile is not a one-size-fits-all approach. Its implementation is tailored to various factors, including the context and situational requirements (Jalali & Wohlin, 2010), the company, the organisational culture, or the team (Noteboom et al., 2021; Rasnacis & Berzisa, 2017).

The aim of this study is to investigate Agile practices in small companies that develop digital products in the Minho region. To provide a structured representation of reality, we have established the following specific objectives:

- verify what are the adopted Agile methodologies and the reasons for the choice;
- understand how Agile methodologies are reflected in the practices of the companies, and how the teams are organised in terms of roles and responsibilities;
- know which tools and software are used to support the project processes;
- understand how the teams evaluate their performance;
- identify challenges that organisations face in project management.

Although the available scientific literature on Agile methodologies is rich, the study of the concrete reality of micro and small enterprises has been little explored (Leiner, 2022).

In 2019, micro and small enterprises represented 98.9% of companies in the European Union and contributed to more than a third of value added (Eurostat, 2022). In 2022, in Portugal, micro and small enterprises (those that employ a maximum of 50 people and whose annual business volume does not exceed 10 million euros) represented 99.4% of the total number of companies and more than 57 billion euros of value added (Pordata, 2023). Considering their weight in the economy and the benefits of a good use of Agile methodologies in project management, it is important to fill this gap. The study presented here aims to contribute to this.

This study relied on both academic and practitioner publications, as the Agile movement has largely emerged and evolved in the context of organisations, and thus largely outside of academia. In fact, most of the knowledge about Agile has been generated within the professional community and has preceded academic research (Dingsøyr et al., 2008) From the practitioner literature, perhaps the most iconic example is the Agile Manifesto itself, which, although only a simple letter of principles and intentions, is the foundational document of the movement and is still a valuable point of reference for understanding what Agile is.

This dissertation has been structured in a simple manner. Subsequent to this introduction, the literature review section presents the foundations of Agile, its mindset, and describes the most

common methodologies. In the Methodology chapter, the research design is detailed. Semi-structured interviews were conducted with project managers from small companies in the Minho region and the data was analysed with Template Analysis (TA) (King et al., 2018; King & Brooks, 2017; Knott et al., 2022). Case summaries were written to avoid the loss of particularities (Brooks et al., 2015). The findings are presented in the Results section, followed by the Discussion section, where these results are interpreted in the light of the existing literature. In Conclusion, the key insights and limitations are summarised, and some future research directions are suggested.

1. Agile Project Management

To assess which methodologies, practices, and frameworks can be considered Agile, and to determine if a project is truly being managed in an Agile manner, we first need a clear definition and understanding of Agile itself as a philosophy.

1.1 Agile Foundations

The Agile Movement is often considered to have been born with the publication of the Manifesto for Agile Software Development, also known as Agile Manifesto (Beck et al., 2001). However, several Agile methodologies, such as Scrum and XP (Extreme Programming) emerged many years before (Ozkan et al., 2020). And it is even possible to go back even further. The roots of Agile's core principles, such as incremental and iterative project development, as well as of self-organised teams, can be traced back to the first decades of the 20th century, in industries as diverse as manufacturing, mining and space exploration (Ozkan et al., 2020). This fact underscores that Agile is more of a philosophy than a set of methodologies designed and limited to specific sectors of activity, as we will discuss later. First, we will dissect the foundational document that has served as a guide to many teams and organisations worldwide (Hohl et al., 2018; Ozkan, 2019), the Agile Manifesto.

The Agile Manifesto dates back to 2001 and, despite the rapid pace of information circulation and technological advances, it remains relevant today. More than 20 years later, this public declaration consisting of two sections – Values and Principles -, still can be found in the same web page where it was originally published. The Agile Manifesto was written over a single weekend, by a group of 17 developers and project managers who later formed *Agile Alliance*, a well-known non-profit organisation that sponsors conferences, studies, workshops, and other initiatives. Understanding the Agile Manifesto in detail is the starting point for identifying true Agile management.

1.1.1 Agile Manifesto Values and Principles

Starting with the values, the Agile Manifesto lists as follows: “individuals and interactions over processes and tools”, “working software over comprehensive documentation”, “customer collaboration over contract negotiation” and “responding to change over following a plan” (Beck et al., 2001, para. 2).

The authors emphasise that this is a matter of priorities. It is not that processes and tools, documentation, contract negotiation and plans are not important. They are and should not be ignored. But they cannot be the primary focus of effort (Highsmith, 2009).

The Principles section of the Agile Manifesto, the second section, expands on the values, providing more actionable orientations to development projects.

The first principle highlights that the main goal of Agile development is to satisfy the customer by delivering valuable software quickly and frequently. In a world marked by volatility, uncertainty, complexity and ambiguity - the famous concept of VUCA - this implies accepting changing requirements when this can positively impact the customer's competitiveness. This is the second principle. The third is to provide functional software in frequent deliveries. In Agile projects, software is developed iteratively and incrementally. The project is divided into iterations, i.e. short periods of time - usually two to four weeks - in which there are specific objectives or features to develop. This delivery of functional features in relatively short time windows forces teams, on the one hand, to consider the number of functionalities to be included and their complexity in relation to the objective of delivering value (Highsmith, 2009), prioritising work and even eliminating irrelevant aspects, and, on the other hand, allows them to look at smaller parts of the project in greater detail, recognize parameters for improvement and make adaptations more quickly. This also allows for more effective follow-up with the client, who sees their requirements materialised in functional software. However, the client is not expected to monitor the project only at the end of the iteration. The fourth principle of the Agile Manifesto is precisely that the client and the team work closely together on a daily basis. The fifth principle relates to the individuals on the team and states that they must be motivated. For this to happen, an enabling environment must be created, with good working conditions and autonomy. Another crucial aspect in the effectiveness of Agile projects is communication, which, according to the sixth principle of the Manifesto, must be personal and direct. For some years, this was considered to mean that individuals had to be in the same physical space, but today, with the multiplicity of synchronous and asynchronous communication tools available to companies, it is

possible to maintain personal and direct lines of communication between individuals even when working remotely. The seventh principle stated in the Manifesto explains how progress is measured: by the delivery of functional software. It's a straightforward and easy-to-understand form of measurement. But it is important that this progress has a sustainable basis, i.e. that it is possible to maintain a constant pace of evolution throughout the project - the eighth principle. In the ninth, there is a reminder about the importance of quality. This principle makes it clear that technical and design excellence increases agility. The tenth principle of the Agile Manifesto is simplicity, defined as "the art of maximising the amount of work not done" (Beck et al., 2001, para. 11), for example by setting aside activities that don't add value to the product. The 11th principle stresses the importance of allowing teams to self-organise in order to maintain the ability to adapt to change. Finally, the 12th principle follows on from the previous one by adding that self-organised teams should regularly reflect on how they can optimise their work, making adjustments and adaptations as necessary.

Therefore, values and principles are simple guidelines to provide direction. They help make decisions, prioritise work, filter information, define what to include in the product and what to leave out, as well as which development practices to use (Highsmith, 2009). The Agile Manifesto does not prescribe a methodology. In fact, the original contributors emphasise the diversity of agile methods and underlying principles, and do not label Agile as synonymous with any single methodology (Hohl et al., 2018).

1.1.2 Agile Declaration of Interdependence

The Agile Manifesto focused on software development projects. Management was not directly addressed and for this reason, in 2005, a new document was created, the Declaration of Interdependence for project leaders (Highsmith, 2009).

The Declaration of Interdependence emerged to underscore the critical nature of collaboration in Agile projects and to make noticed the diverse lines in which it occurs: between the members of the development team, as well as between the team and both the customer and stakeholders (Nee, 2012). All of them are interdependent.

The Declaration of Interdependence, dedicated to project leaders and project managers, states that uncertainty is an inherent aspect of any project. It must be managed through iterations, some planning, and adaptation above all. Iterations in the project and in the team practices allow to improve the reliability of both products and processes. The Declaration of Interdependence asserts that the return on investment increases by prioritising continuous delivery of quality and value to the customer over meeting time, budget and scope constraints, and that is only possible by engaging the customer in the processes. Additionally, the team and its members must have access to the necessary resources in an environment where creativity and innovation are recognized and encouraged. The leaders' role is to create these favourable conditions, manage the teams, and engage clients and stakeholders. However, it is the team that autonomously and responsibly manages its own tasks.

1.2 Agile as a Mindset and Agile Methodologies

Our research aims to get a picture of how Agile is implemented in organisations, and what methodologies, practices, and tools are being used in Agile project management. However, considering the values and principles of the Agile Manifesto and the Agile Declaration of Interdependence, we can understand that Agile requires an attitude, a specific way of thinking, and is more a mindset than a rigid set of practices (Denning, 2016; Highsmith, 2009; Larman, 2003). As the Agile Manifesto (Beck et al., 2001) emphasises, valuing individuals and interactions over processes and tools is at the core of the Agile philosophy. The same is to say that, while methodologies and practices play a significant role in Agile project management, defining Agile solely through them is insufficient. Agile requires a mindset, a predisposition to act and an effective behaviour from individuals, teams and organisations that goes beyond practices, procedures and methods (Miler & Gaida, 2019; Ozkan et al., 2020). For this reason, using Agile methodologies in project management does not mean that in fact a project is being managed in an Agile way, nor does it mean that the team or organisation within which it is being developed can be considered Agile (Highsmith, 2009). Before we go further on this study and see how Agile is adopted and practiced in small companies, we must first define Agile as a mindset.

We could define this attitude (this mindset) in a psychological sense as a hypothetical construct of the individual, a predisposition to act, a set of ideas, beliefs, opinions, feelings, that is not directly observable (only observable by behaviour), and that can change over time (Vala & Monteiro, 1999). It is a mindset that guides action, a system of values and principles that can be more or less shared within a team or within an organisation, that translates into a way of acting that can use different methodologies and practices. But what characterises the Agile Mindset? And do methodologies play a significant role? These interrogations are addressed in the following pages.

1.3 The Agile Mindset

As demonstrate by our literature research, in recent years, there have been several efforts to define what agility or the Agile mindset is and what its characteristics are. A survey of practitioners, conducted by Miler & Gaida (2019), suggests that the definition of agility among practitioners does not fully correspond with the principles outlined in the Manifesto. The authors did not investigate why, but hypothesised that this discrepancy could be due to insufficient understanding of Agile among the respondents, the specifics of the companies studied, or the evolution of the concept of agility and Agile practices, and called for further research. Klünder et al. (2022) state that it is becoming clear that organisations should in fact go beyond the values and principles in the Manifesto and Declaration of Interdependence as they do not clearly define what is the Agile Mindset.

Agility can be manifested at many levels, from the individual level to the team, process, tools, strategy, and organisational culture (Ozkan et al., 2023), and is largely defined as the ability to respond to change (Ozkan et al., 2020). For instance, Highsmith, one of the authors of the Agile Manifesto defines agility as the capacity “to both create and respond to change in order to profit in a turbulent business environment (...), the ability to balance flexibility and stability" (2009, p. 40).

Miler & Gaida's (2019) definition emphasise the individual level and how it can echo in the team level. For the authors, Agile mindset is as "a set of one's attitudes, behaviours and ways of thinking that enhance their and their team's effectiveness in working following the Agile values and principles to the benefit of the customers" (p. 841). It is important to note that what benefits the customers, including both the client for whom the solution is being developed and the end-user of the solution, may not always align with how they imagined it. For example, client requirements and their vision of a product don't always solve the real problem. That is why Ozkan et al. (2020) define agility as "the ability to move quickly and easily (…), to adapt to changes of the reality or to create changes becoming the reality, let us say in the domain of software solution development." (p. 722), emphasising the importance of considering the broader context rather than blindly following client requirements.

The definition of Agile mindset (AM) by Eilers et al. (2022) also highlights its individual dimension:

> The AM is the attitude of an individual within a dynamic work context that is expressed by positively evaluating how they: 1) continuously seek new insights to respond to changes, 2) transparently share and discuss methods and results of work with others, 3) decide for themselves how to proceed, and 4) are continuously customer oriented in a co-creation process at work. (pp. 8–9)

The people factor has an undeniable influence in the success of a project. Organisational transitions to Agile may fail when the actor's Agile mindset is not solid enough (Eilers et al., 2022).

However, although the Agile mindset has an important internal component, it requires a favourable context - the external component - in order to manifest itself and evolve. Mordi & Schoop's (2020) definition, which was developed through a research process that included a systematic literature review and contributions from semi-structured and unstructured interviews with Agile professionals, incorporates the organisational level:

> Agile Mindset is a mindset based on the values and principles of the Agile Manifesto, whose main characteristics are trust, responsibility and ownership, continuous improvement, a willingness to learn, openness and a willingness to continually adapt and grow. It is underpinned by specific personal attributes on the individual level and an enabling environment on the organisational level, which allows autonomy of people and teams, managing uncertainty and a focus on customer value, with the goal of achieving a state of being Agile instead of merely doing Agile. (p. 9)

We can hypothesise that on the required "enabling" environment methodologies plays a part.

Ozkan et al. (2023) argue that methodologies and tools have been prioritised over mindset because this last one is "abstract, vague, and latent", "difficult to measure, even to observe and demonstrate" (p. 206). Tools and methodologies, on the other hand, are more tangible, even they are "on the 'less valuable side' of the Agile Manifesto" (Ozkan et al., 2023, p. 206).

The authors of the Agile Manifesto, as well as many researchers and practitioners, emphasise that "doing Agile" is not the same as "being Agile" or having an Agile mindset. Failure to recognise this

results in practices and methodologies often implemented and executed incorrectly (Hohl et al., 2018). Applying a set of Agile practices and methods is not the same as having a proper Agile mindset (Miler & Gaida, 2019). In fact, to implement the most adequate Agile practices and methods, a mature Agile mindset as to be in place (Ozkan et al., 2020).

As Durbin & Niederman explain “Following Agile approaches requires the spirit of Agile as well as the mechanics of following its ‘rules’” (2021, p. 9). While we also recognize that "being Agile" is a prerequisite for "doing Agile" correctly, in this study we focus on the ways of implementation of Agile methodologies and practices in organisations. We've included a brief explanation of the Agile mindset because mindset and methodologies are interdependent. Mindset plays a critical role in selecting the most appropriate practices and methodologies for specific project, team, and organisational contexts. And methodologies and practices are the tools that put the Agile mindset into action. In practice, we can't separate them. That is the reason why it is relevant to know how companies are implementing Agile methodologies.

1.4 What are Agile Methodologies

The terms "Agile methodology" and "Agile framework" are often used interchangeably. However, in this study, we will distinguish between the two for the sake of clarity.

An “Agile methodology” is a consistent set of practices, tools, principles, and rules at team-level.

On the other hand, an 'Agile framework' is a skeletal structure designed to bring different teams together to scale Agile and implement a coherent Agile way of working across teams or the entire organisation (organisational level).

Naming all the Agile frameworks, methodologies and practices is challenging. There are a lot, each one has its own specific practices, and they can be used by themselves or combined, generating a large number of possibilities. Henny Portman (2020) attempted to map the Agile landscape by listing all the methodologies and frameworks in use. Although Portman was able to provide a wide-ranging list of approaches, he recognised that it was impossible to mention them all. Additionally, there may be some overlap because “commercial drivers play a role” (p. 1). Many project management consultants and other professionals try to adapt the existing frameworks to develop their own methodologies, often with minimal to no differentiation, essentially "repackaging" existing ideas with a new name.

The map created by Portman provides insight into the impact of Agile methodologies on an organisation. The author categorises Agile approaches into three levels: team, product/program, and portfolio, as shown in Figure 1.

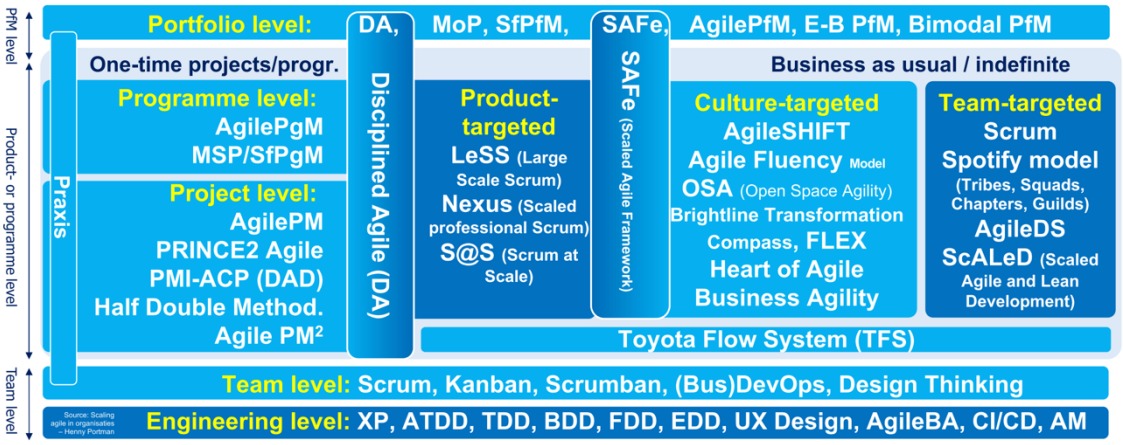

Figure 1. Overview Agile approaches, frameworks and methods
Source: Portman (2020, p. 1)

The dark blue boxes represent approaches that are only applicable in IT-focused organisations. The team level, which also covers the engineering level, corresponds to our methodology level. The product-programme and portfolio levels correspond to what we call the framework level. This study focuses on Agile methodologies and related practices, tools, and principles at the team level, not from an engineering perspective, but from a broader management perspective. As this study was carried out with companies with no more than 40 employees and therefore with a reduced number of teams, the frameworks for scaling across the organisation are not as relevant as methodologies at the team level.

To avoid an excessively long list of methodologies and to align with the study's focus, we will refer to and explain in detail the most commonly used by practitioners, rather than attempting an exhaustive compilation.

1.5 The Most Common Agile Methodologies

In 2020, Ozkan et al. conducted a literature review on Agile methodologies and identified a total of 28. Out of the 28 methodologies reviewed, the majority were classified as “obsolete” or “nearly obsolete” based on four parameters: obsolescence of the core principle or approach, being eclipsed by newer methods, lack of recent studies or absence of references in the VersionOne State of Agile Report series (now named “State of Agile Report”). Only seven methodologies (or methods) were referred to as “alive”: Scrum, Extreme Programming (XP), Lean Software Development, DevOps, Scrumban, Kanban, and Disciplined Agile Delivery (DAD). This is a good starting point for our list of Agile methodologies to analyse in detail. However, we will exclude DAD based on our definitions of methodology and framework. DAD is a scalable framework developed to address problems above the team level and work on agility at an organisational scale (Ambler, 2013), which does not fit our criteria.

To determine the current relevance of this list and identify any additional methodologies, we consulted the latest annual reports published by reputable Agile institutions that provide an overview of the global Agile landscape.

There is no doubt that Scrum is the most commonly used Agile methodology. The 17th State of Agile Report (2023), released by Digital.ai, shows that Scrum remains the most widely used methodology, with 63% of Agile users reporting its use. This has been the case since the first State of Agile Report (formerly VersionOne State of Agile Report) in 2006. The results of the 2023 State of Agile Report published by The Braintrust Consulting Group are consistent: nearly 70% of the respondents (both Agile and non-Agile users) said they were working with Scrum. State of Agile Culture 2020 Report, a study conducted by Agile Business Consortium, JCURV and Truthsayers, also revealed a significant prevalence of Scrum with 52% of Agile practitioners reporting its use.

Other Agile methodologies reported in these studies are Kanban, Extreme Programming (XP), DevOps, Scrumban, Lean Startup (in which we could include Lean Software Development as a more specific methodology under Lean Startup), and Scrum/XP Hybrid. Each of these is defined in detail in the following pages.

1.5.1 Kanban

Kanban is a methodology that emphasises the visualisation of the workflow and was developed at Toyota to improve their processes, being one of the main tools of Just In Time (JIT) and Lean Manufacturing. In 2004, the tool was adapted to software development by David J. Anderson (Zayat & Senvar, 2020). This methodology relies on a board – the Kanban board -, a visual tool that provides visibility of work progress to everyone on the team in order to maximise efficiency or, in other words, to optimize the flow. By visualising work stages and flow of the work among them, Kanban boards enable teams to identify roadblocks, avoid over-commitment, and enhance communication and cooperation as everything is visible to the whole team (Project Management Institute, 2021). The Japanese word “Kanban” means precisely “signboard” (Petricioli & Fertalj, 2022). The main objectives of Kanban are to protect the team members from task overload increasing productivity, adapt the work flow to change effectively, apply just-in-time and prioritisation to reduce unnecessary effort, and focus on delivering value (Alqudah & Razali, 2018).

1.5.1.1 The Kanban Board

The Kanban board is divided into columns, each of which represents a workstation or a part of the project, such as "development" or "test". The most basic version of a Kanban board consists of three columns: "to do", "doing", and "done". Under each column are cards representing the tasks, work items, or user stories that need to be completed as part of the project. Classified as "to do" are the tasks that have not yet been started, and they should be organised according to priority in order to be transferred or "pulled" into the column to the right, the "doing" column, which represents the work that is currently being done. Once the task is completed, it is moved to the "done" column. These three states are actually part of every Kanban board and can have as many columns below them as needed to accurately reflect the team's workflow.

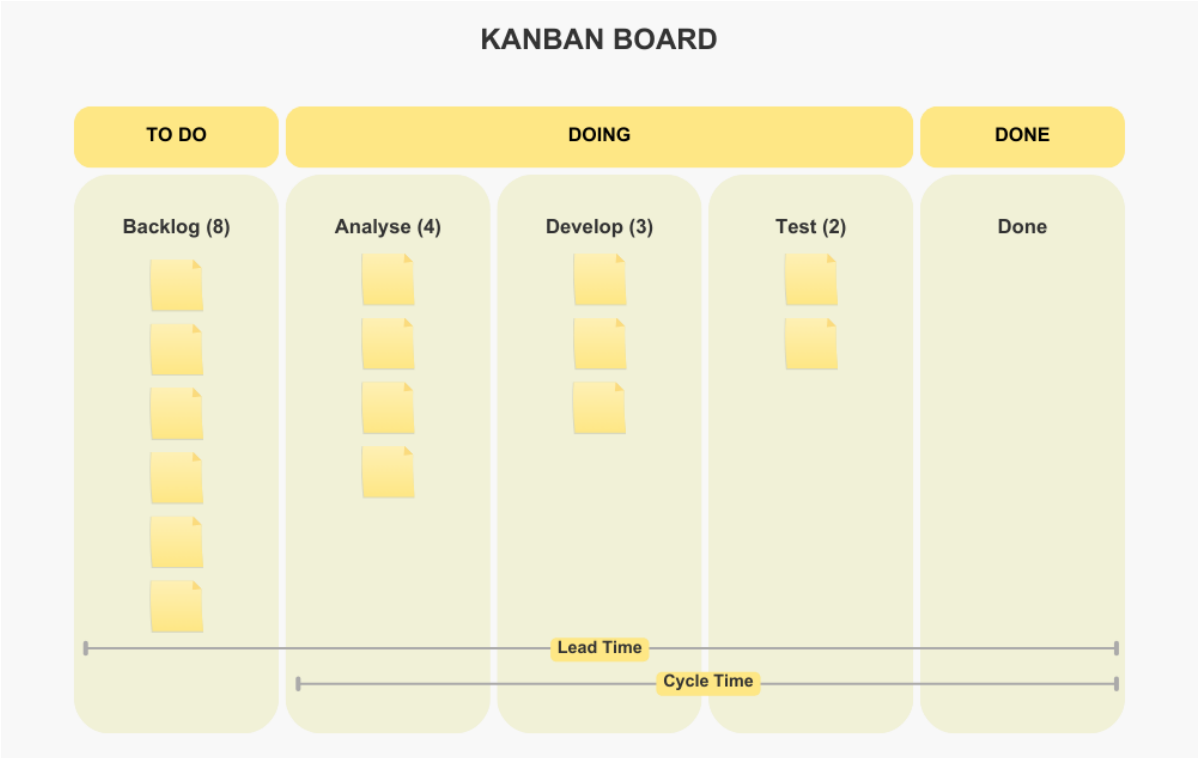

Figure 2. A simplified version of a Kanban board

Source: Adapted from Project Management Institute (2021)

Figure 2 is a representation of a Kanban board in which we have five stages: the "to do" list or "backlog", "analyse", "develop", "test", and "done". Forward may be the most logical workflow, but Kanban board's flexibility allows cards to move in the opposite direction when necessary. For example, a feature in the "test" column can be moved back to the "develop" column if issues are identified during testing.

1.5.1.2 Work In Progress Limits

Each column has a limit for the number of work items, tasks or user stories, that can be added in the form of cards, called the Work In Progress (WIP) limit. The WIP limit shall be adapted to the team's resources and skills. The idea is that limiting the amount of work that is being done simultaneously can actually speed the work done over time because of the cost of constant context switching (Kišš & Rossi, 2018; Ozkan et al., 2022). In the example of Figure 2, only four tasks can be worked on concurrently in the "Analyse" workstation, three in "Develop," and two in "Test." To test another feature, one of the cards in the "Test" column must first be moved to the "Done" column. In Kanban, as is characteristic of Lean approaches, when a task is moved to the next column, it clears space and pulls another task with it (Petricioli & Fertalj, 2022).

Each card must have, besides a brief description, an owner, meaning, a person to whom the card is assigned. Other data can be added such as a due date, or dependencies. This tool makes it easy to identify bottlenecks across workstations (Zayat & Senvar, 2020). If a column is near to its charge limit, team members can gather around those tasks that are slowing the progress.

1.5.1.3 Kanban Team Roles

Kanban, unlike other methodologies does not prescribe specific roles to the team members (Alqudah & Razali, 2018).

1.5.1.4 Performance Indicators in Kanban

Lead time and cycle time are utilized as performance indicators (Project Management Institute, 2021). The first corresponds to the delivery time or time elapsed between the addition of a card to the board (i.e., the "to do" or backlog stage) and the delivery of the associated work item to the customer. The second indicator, cycle time, is the time it takes for a work item to be moved from the “doing” stage to “done” stage, meaning the interval between the initiation of a task and its completion.

1.5.2 Scrum

Scrum is the most popular of the Agile methodologies and is often the one chosen when teams start working with Agile (Portman, 2020). It involves managing projects in short, time-limited iterations called sprints. Teams self-organise to deliver working pieces of the product incrementally in each sprint, with frequent feedback loops and continuous adaptation. It's well suited for projects with some degree of uncertainty, where requirements may evolve over the course of the project, and was developed by Ken Schwaber and Jeff Sutherland, two of the authors of the Agile Manifesto. In 2020, the authors published "The Scrum Guide," a 13-page document that elucidates the fundamental principles of Scrum. As defined in the guide, Scrum can be understood through the examination of its roles, events, artifacts, and the rules that tie all together (Schwaber & Sutherland, 2020).

1.5.2.1 Scrum Team Roles

The team is the “fundamental unit of Scrum” (Schwaber & Sutherland, 2020, p. 5). It shall be small, with less than 10 people, in order to assure the team is able to communicate easily and effectively. The recommendation for larger teams is to break them into smaller teams, even when they are working for the same product. Those cohesive teams working on the same product then share the Product Goal, the Product Backlog and the Product Owner. The team is self-managed, responsible for all the product-related activities and held accountable for creating value in each sprint, following the Agile principles.

Within the Scrum team there are three roles: the Developers, the Product Owner and the Scrum Master.

The Developers are the ones responsible for developing the product, delivering a working software or product increment at each interaction or sprint. They are entrusted with creating the plan for each Sprint (the Product Backlog) and the definition of “done”, as well as adjusting their daily plans to achieve the Sprint Goal. The definition of “done” is shared by everyone and is of great importance in ensuring the quality of the work delivered.

The Product Owner is the one responsible for defining the Product Goal and their decisions must be respected by all members of the team, both in the definition of the Product Backlog and during the Sprint Reviews. The Product Owner is then the person in charge of managing the Product Backlog

and must ensure its full comprehension by the team. In order to achieve this, the Product Owner must communicate clearly the Product Goal, as well as the Product Backlog items in an orderly manner, with the most pressing items at the top.

The Scrum Master is a multifaceted role, encompassing both the function of a guide and that of a facilitator for a multitude of stakeholders within the project ecosystem. The Scrum Master is a leader who “serves” the Developers, the Product Owner and the whole organisation by coaching the team and the organisation in the Scrum principles (Schwaber & Sutherland, 2020, p. 6). The Scrum Master coaches the team members towards self-management and cross-functionality to tackle diverse tasks, comply with the pre-established definition of “done”, be able to respond to changes, develop high-value increments at each sprint, and maintain focus on the Product Goal. It is the responsibility of the Scrum Master to guarantee that all Scrum ceremonies – sprint planning, daily stand-ups, sprint reviews and sprint retrospectives – are conducted as scheduled. The Scrum Master is also tasked with identifying and removing obstacles to the team’s progress, as well as creating an enabling and positive atmosphere throughout the project’s duration. Additionally, the Scrum Master also serves the Product Owner by assisting in the definition of the Product Goal, the management of the ever-evolving Product Backlog, the clear communication of information, and the cooperation with the team and other stakeholders. In the context of the organisation, the Scrum master is the individual responsible for all the training and guidance on the Scrum methodology, fostering a culture of collaboration and continuous improvement.

1.5.2.2 Scrum Events

The Scrum events are all the ceremonies that take place during the development of the project and they all occur within the main event, the Sprint. The Sprint is a time-boxed iteration, typically two to four weeks in length, during which functional features of the product are delivered according to the Sprint Goal. When it ends, a new one begins. Each Sprint brings the team closer to the Product Goal.

Sprint Planning is the very first event of a Sprint. It is the moment when the Product Backlog is discussed to plan the Sprint and the following questions are answered:

- What value will be added to the product during this Sprint and what is the Sprint Goal?
- What work items from the Product Backlog can be done during this Sprint?
- How will the work items get done, or in other words, how can the work be planned to increment value and meet the established definition of “done”?

Schwaber & Sutherland (2020) propose a ratio of no more than eight hours for Sprint Planning within a one-month Sprint. The Sprint Goal, the selected work items for the Sprint and the plan to get them done are then referred to as the Sprint Backlog.

To monitor progress towards the Product Goal and adapt the Sprint Backlog as needed, daily stand-up meetings are held – the so-called Daily Scrum. To streamline procedures and enhance predictability, the Daily Scrum happens every day at the same time in the same location. This 15-minute meeting is an opportunity for developers to reflect on the work completed the previous day

and plan for the upcoming day. It facilitates communication, identifies potential roadblocks, and encourages assistance. Discussions about the optimal solution to a problem are held after the Daily Scrum to maintain the meeting's brevity and respect for everyone's time. Developers are encouraged to communicate as needed throughout the day. However, the Daily Scrum is a scheduled meeting designed to facilitate reflection on progress without disrupting the work flow.

In order to ascertain the outcome of a sprint, another ceremony is held. A review of the sprint is conducted. The Sprint Review is an event during which the Scrum Team presents the work completed during the sprint to the stakeholders and gathers feedback. The Product Backlog may be modified in accordance with newly discovered information or new opportunities. For a one-month Sprint, Schwaber & Sutherland (2020) recommend limiting the duration of the Sprint Review to a maximum of a four hours.

The final event on the Sprint is the Sprint Retrospective, a moment to analyse how the work was done and identify potential avenues to increase quality and speed. Individuals, interactions, processes and tools are examined to facilitate continual improvement of methods, and the findings are integrated into subsequent sprints and/or projects. The duration of the meeting shall be slightly shorter than that of the Sprint Review, which is recommended by Schwaber & Sutherland (2020) to be no longer than three hours maximum for a one-month sprint.

1.5.2.3 Scrum Artifacts

Scrum Artifacts are essentially information to help the Scrum team and the stakeholders align understandings, goals and efforts, providing transparency into what's being built and how the project is progressing. There are three primary types of Scrum artifacts:

- The Product Backlog
- The Sprint Backlog
- The Increment

As Schwaber & Sutherland (2020) explain, each artifact has a commitment attached, that drives focus and clarity.

The Product Backlog is a prioritised list of all potential elements that could be incorporated into the final product and are referred to as the Product Backlog items. The Product Backlog is a dynamic entity that is continually refined through the addition of new items or adjustments to existing priorities, always in accordance with the Product Goal. The Product Goal is the commitment of the Product Backlog and remains constant throughout the project, serving as the final objective for the Scrum Team.

The Sprint Backlog has its own specific commitment, the Sprint Goal, which is, of course, aligned with the overall Product Goal. The Sprint Backlog is the outcome of the Sprint Planning meeting, a list of work items selected by the Scrum team from the Product Backlog, for work during a Sprint.

An Increment is concrete progress towards the Product Goal, a deliverable functional piece of the final product, meaning that it could be released to users if needed. At the end of a Sprint, one or

more Increments can be done. Each increment builds on the previous increments, so with each sprint the product becomes more and more feature-rich. However, the work done within a Sprint cannot be considered an Increment unless it meets the definition of “Done”, which is the commitment for the Increment. According to Schwaber & Sutherland (2020), the definition of “Done”, also known by the acronym DoD, “is a formal description of the state of the Increment when it meets the quality measures required for the product” (p. 12). The DoD is shared by all the team or teams working on the same product. If a product backlog item cannot be marked as “done”, it cannot be present at the Sprint Review and must be returned to the Product Backlog.

Figure 3 depicts the Scrum product development cycle, commencing with the creation of the Product Backlog. In the Sprint Planning Meeting, tasks are selected from the Product Backlog to create the Sprint Backlog and begin the Implementation phase. During the Sprint, the team works on the selected items to move towards the Sprint Goal and the Product Goal, meeting daily on the Daily Scrum Meetings to discuss what was done the previous day, what will be done that day, and identify obstacles that have arisen. At the end of the sprint, there are Increments, which are functional additions to the evolving product or, in other words, developed features that meet the pre-established definition of “done”. At the Sprint Review, the Increments are presented to the stakeholders, feedback is gathered, and, after that meeting, the team holds a retrospective meeting to discuss the sprint and how processes could be improved in the next one. When a sprint ends, a new one begins. The cycle of planning, implementation, review and retrospective adjustment is repeated until the Product Goal is reached, reflecting Scrum’s iterative nature (Petricioli & Fertalj, 2022).

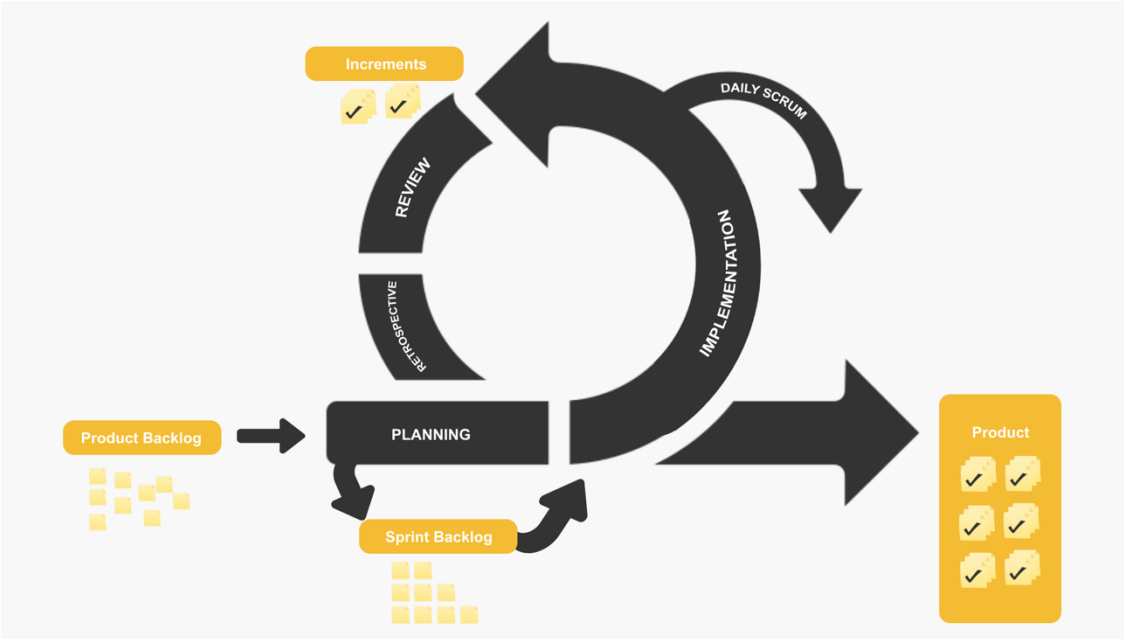

Figure 3. Scrum Cycle

1.5.2.4 Performance Indicators in Scrum

Performance in Scrum is usually measured recurring to burndown charts and team velocity (Zayat & Senvar, 2020). During Sprint Planning, the team estimates the effort required for each task, also

referred to as user story or work item, and tries to evaluate how many can be done in the time-frame of one Sprint considering the effort required. This estimation is done in what is called “story points”. A burndown chart is a visual representation of the team’s progress along the sprint, indicating the story points, or number of tasks or work items completed at each day. At the end of the sprint, the team's velocity for that sprint is calculated by summing the story points of all the completed user stories (Highsmith, 2009).

1.5.3 Kanban versus Scrum: Advantages and Disadvantages

Ozkan et al. (2022) defend that sprints may not be the optimal approach in situations where constant change is a significant factor and where adjustments must be made swiftly, such as in the case of software maintenance teams dealing with unforeseen issues that have great difficulty to plan and adhere to sprint schedules. That is one of the reasons teams transition to Kanban and to Scrumban.

A study by Ozkan et al. (2022) indicates that Kanban has an overall advantage over Scrum and found that, in recent years there were a lot of transitions from Scrum to Kanban and hybrid methodologies. The authors recommend to blend Scrum and Kanban according to the company’s and project’s needs. Alqudah & Razali’s research (2018) suggests that Scrumban can be more effective in time savings, quality improvement, and waste minimization.

1.5.4 Scrumban

As Schwaber & Sutherland (2020) explain, “while implementing only parts of Scrum is possible, the result is not Scrum”, but hybrid models. Although methodologies that use time-boxed iterations and incorporates some or all of the roles of Scrum may be sometimes mistakenly referred to as "Scrum" in everyday practice due to its widespread popularity, the blending of different methodologies actually results in new ones. Hybrid models have been developed over the years and Scrumban is the most popular Agile-Lean combination (Petricioli & Fertalj, 2022), combining principles of Scrum and Kanban. However, there is not a single rigid way to do Scrumban. Rather than adhering to a single prescriptive approach, teams select and adapt the practices from Scrum and Kanban that best fit their context and the project requirements (Alqudah & Razali, 2018).

1.5.4.1 Practices of Scrumban from Kanban and from Scrum

In Scrumban, work is always visualised on a Kanban board, and the tasks, or units of work, are moved following the pull system based on the priorities and team capacity. As in Kanban, there are WIP limits that can be imposed both to a person’s work in progress as well as to a column of the board or workstation. Using the example on Figure 2, if one of the elements of the team wants to pull a ticket from the “Develop” work station to “Test” but that column is at its WIP limit of 2, the rest of team shall work together to complete one task on “Test” and clear space. It can also be added a buffer list between “Develop” and “Test” columns: a “Ready to Test” column (Ladas, 2009, 2019).

In Scrum, tickets are attributed to people in what is commonly referred to as “early binding”. But this can cause some problems. For instance, if a team member gets two tickets that are less important. This means that a less important ticket can be worked before a more important one. Scrumban does

it differently. It does not assign people to tickets. It orders the tickets by the level of importance. So when it is time to pull a card to the “doing” stage, the one that should be pulled is the one on the top of “to do” list.

The team may work within sprints, but applies it the Kanban thinking to optimize the flow of the units of work (Portman, 2020). Corey Ladas, one of the most prominent contributors to the development of Scrumban, actually claims that “in a well-regulated pull system, iterations add no value at all” (2009, p. 47). Instead, he holds that the ideal is a single flow of work, continuous integration, a kind of a single sprint driven by an optimized pull system.

When used in Scrumban, iterations, or sprints, are flexible in several aspects, from the length that may vary based on the project’s needs, to the way work enters the workflow (Ladas, 2009). The product backlog is created with a just in time philosophy with the pull system and WIP limits, minimizing wasted effort and ensuring a steady flow of value delivered. This means that not all the user stories or work units are identified from the beginning of the sprint. Instead, new work items may be added to the sprint whenever needed. Changes to the sprint backlog can be done during the sprint as new needs emerge and a task can be moved back to the starting column of the Kanban board (Petricioli & Fertalj, 2022). Prioritisation is done dynamically based on customer needs and feedback, with a focus on delivering value early and often. While on Scrum the sprint planning is done on a certain schedule (often every two or three weeks), in Scrumban the planning of each sprint is done with a trigger point: when the number of items on the “to do” list is below the defined trigger point, it means there is need for a ceremonial planning session (Ladas, 2009, 2019).

A further distinction between Scrum and Scrumban is that there are typically no story points or effort estimation for each work item in the sprint planning. Instead of estimating each item as Scrum does, Scrumban sets a fixed size limit for the sprint backlog, which is the number of tasks or work items to be included. This reduces the time required for planning. The limit is based on the team’s average performance. To calculate the limit, the team must consider the average number of items completed per iteration and the average cycle time. For instance, if it takes five days to complete five items, a two-week iteration with ten work days should have a backlog limit of ten tasks (Ladas, 2009, 2019).

Stand-up meetings may be held on a daily basis to synchronize the team members’ activities, discuss progress, identify any obstacles or bottlenecks, and make adjustments to keep a smooth flow of work (Ladas, 2009) or done only when a need is identified (Alqudah & Razali, 2018).

Flexibility also reflects on the team structure. There are no fixed roles for team members on Scrumban (Petricioli & Fertalj, 2022). Self-organisation is encouraged, and each team decides what roles are necessary in each project (Alqudah & Razali, 2018).

1.5.4.2 Performance Indicators in Scrumban

Performance indicators in Scrumban include metrics like lead time, cycle time, and throughput. Lead time tracks the time taken from task initiation to completion, cycle time focuses on the time taken to complete a single task, and throughput measures the amount of work completed over a specific

period. These indicators help the team assess their efficiency, identify bottlenecks, and continuously improve their process (Ladas, 2009, 2019).

Scrum's burndown charts are not the preferred metrics in Scrumban because, although they indicate the number of tasks completed, they do not provide clues about why tasks are or are not completed (Ladas, 2009, 2019). Scrumban proposes a Cumulative Flow Diagram (CFD), that provides additional information about lead times and work-in-progress inventories by breaking down the workflow into the various states or workstations.

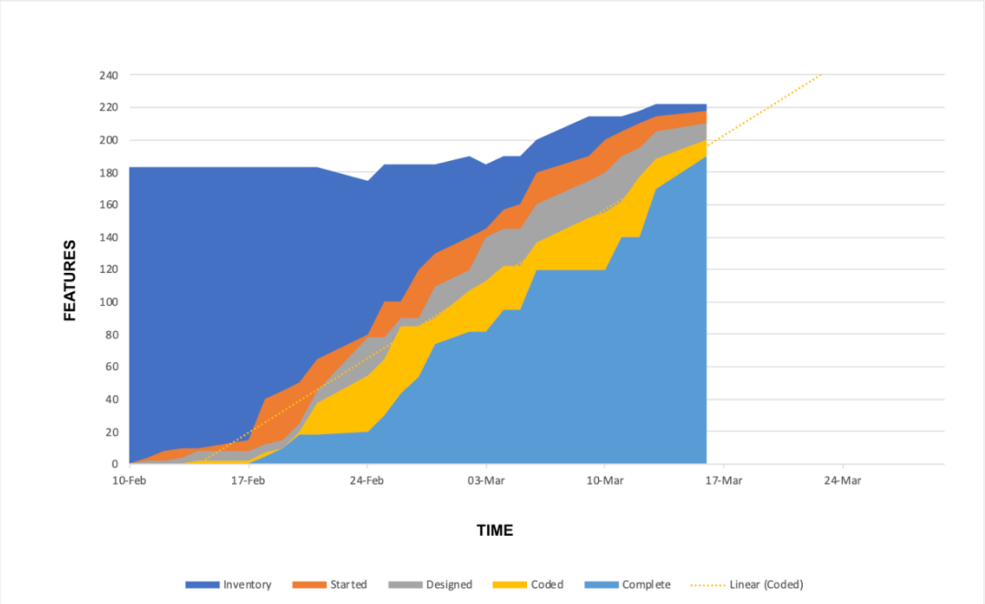

Figure 4. "Cumulative Flow Diagram"

Source: Ladas (2019)

1.5.5 Lean Startup or Lean Software Development

For the purposes of this work, which focuses solely on project-level methodologies, we use "Lean Software Development" (LSD) and "Lean Startup" (LS) interchangeably to refer to the same methodology for developing projects, but it is necessary to point out that the terms are not exactly the same. LSD is a methodology for developing software and digital products, and, while LS can refer to software development as well, it also encompasses methodologies for managing businesses, at the inter-teams and portfolio levels as Portman (2020) calls it. Nonetheless, LSD does not prescribe team roles.

1.5.5.1 Lean Startup Development Cycles

Lean Startup has been popularized in recent years by the work of Eric Ries. It is a lightweight project management methodology that enables the iterative development of product prototypes in a sustainable manner. This is achieved through "Build-Measure-Learn" (Bortolini et al., 2021) It starts with the construction of a product vision and ideation. Subsequently, the experiment is built: a prototype or a minimum viable product (MVP) with enough features to be tested by early customers to understand if it meets their needs. The next stage is the measurement which can be conducted

using a variety of feedback tools, including direct observation, interviews, or surveys. Finally, the most crucial stage: learning. This is the stage at which it is possible to ascertain what aspects of the product are well-received by customers, and which are not, and can lead to pivoting, iterating, escalating or abandoning the product. Each cycle of BML, or iteration, brings the product closer to market needs in a way that reduces waste and increases efficiency, as it's a Lean rule.

1.5.5.2 Performance Indicators in Lean Software Development

LSD stresses the relevance of measuring performance to improve processes and quality, and of using data to make informed decisions rather than relying on speculation (Poppendieck & Poppendieck, 2003). Velocity measurements, such as tracking the amount of work done in each iteration, are a reliable way to forecast performance and make predictions about future projects. Burn-down charts are also used to visualise progress. Nevertheless, the overarching objective of LSD is not merely the tracking of metrics like cost and schedule, but to focus on delivering business value.

1.5.5.3 Lean Startup: Agile, Lean or both?

The literature review done by Ozkan et al. (2020) points LSD as an Agile method, although its main focus is not on agility but on reducing waste. There is no absolute consensus about if Lean Software Development and Lean Startup methodologies fall under the Agile umbrella, are pure lean, or a complementary approach to Agile (Petricioli & Fertalj, 2022).

Mary and Tom Poppendieck, the pioneers of Lean Software Development are into this perspective that lean thinking can guide the implementation of Agile development practices. The authors identified seven principles of Lean to apply to software development.

Table 1. The Seven Principles of Lean to Apply to Software Development

Principle	Description
Eliminate waste	Anything activity that consumes resources (time or money) but does not add actual value to the final product in the point of view of the customer is considered waste. This can include for instance unused features or excess documentation.
Cultivate a culture of learning	Software development is a discovery process. To enhance that discover, the team must be open to continuous learning and adaptation, an attitude encouraged by Lean.
Decide as late as possible	In uncertain situations, delaying decisions to when the information is clearer can lead to better choices. Lean development promotes keeping options open as long as possible to be able to correctly identify and respond to requirements.
Deliver as fast as possible	Rapid development allows for quicker feedback from customers and learning faster.
Empower the team	The ones that work the product are the ones more well-aware of all the particularities and details, so they are the best equipped to make decisions, much better than any centralized entity. They must be given the autonomy and the responsibility of making their own decisions on how to do the work.

Build integrity and quality	A high-quality product is the one that embodies integrity and meets user needs. A constant focus on providing quality and value to the user/customer is promoted by Lean, and reduces the changes of bugs and errors being delivered to the customer.
See the whole	Focusing solely on individual areas of expertise can lead to suboptimal system performance. Lean practices encourage a holistic view, considering the entire system's functionality.

Source: Poppendieck & Poppendieck (2003)

It is clear that Agile and Lean have to many points of intersection to be easily separated.

“The aim of LSD is to approach the zero (waste) point. Agility leans more on the expansion of perspectives; learning (fail fast), reworks (creating features only to understand customers better at the earliest) and so on. This "haste” to respond quickly in Agile may “make waste", implying that Lean and Agile approaches can be contrast serving in two different directions. However, there is a Lean perspective in the manifesto by advocating just enough documentation, reducing “ineffective communication” occurring in the hierarchy, tools and processes.” (Ozkan , 2020. p. 728)

1.5.6 Extreme Programming (XP)

Extreme Programming (XP) was proposed by Kent Beck even before the Agile Manifesto was published. The objectives of the methodology were to increase quality, accelerate the response to changing customer requirements and simultaneously reduce development costs, therefore making the process more efficient. It is “an integration of well-known software engineering practices”(Al-Saqqa et al., 2020, p. 261) on a set of values, principles and practices.

1.5.6.1 Extreme Programming Values, Principles and Practices

In their book “Extreme Programming Explained: Embrace Change”, Kent Beck and Cynthia Andres (2004) outlined the values, principles and practices of the methodology – a structure that was later used to present Agile to the world in the Manifesto. XP offers recommendations on the practices for development, including the focus on incremental progress, the recognition of the importance of redundancies on software development, and the incentives to maintain consistency in practices and processes in order to reduce errors. However, it is described as a mindset on its essence for the importance it gives to the attitude. The core values of XP are communication, simplicity, feedback, courage and respect. It places significant emphasis on the creation of a supportive environment for all team members, with the objective of fostering full cooperation and transparency within the team. Having a multitude of perspectives and abilities on the team is encouraged. Each team member is expected to take ownership of their work and be accountable for their actions. Improvement is a continuous process, and all failures are viewed as opportunities for learning (Beck & Andres, 2004).

The table below lists some of the primary practices of XP, as described by Beck & Andres (2004), along with a brief explanation.

Table 2. Primary Practices of XP

Practice	Description
Sit Together	Team members are expected to sit together, in close proximity. Open space reduces barriers and facilitates communication.
Whole Team	Each team is part of the whole and shares the same objective. The teams are small and cross-functional teams (up to 12 elements so that everyone can communicate easily).
Informative Workspace	The workspace must have information about the ongoing work, such as Kanban boards, progress charts, or any other visual tools, that are easily understandable by anyone.
Energized Work	People are not always productive and working for long hours can in fact do more harm to a project than help it move forward. Team members are encouraged to find a balance between hard work and rest.
Pair Programming	Two developers working together on the same code. This allows to discuss ideas and brainstorm in a more frequently, contributing to reduce errors and bugs, and improve quality. This doesn't mean that the developers must always do pair programming, and in fact the solo programming is sometimes required, but the frequent use of this practice is considered an optimal approach.
User Stories	User stories must be written down. A user story is a general explanation of a goal written from the perspective of the end user that can be translated into a software feature. For example: "As a user, I want to upgrade my subscription so that I can access additional functionalities."
Weekly Development Cycles	XP breaks down development into short iterations, typically lasting 1 to 2 weeks. This guarantees faster feedback to adapt to changing requirements sooner.
Quarterly Cycle	Plan the work for each quarter. Longer-term planning and strategy must also be considered.
Slack	Identify low-priority tasks on weekly and quarterly cycles that can be dropped or delayed if the team is held up by more critical tasks.
Ten-Minute Build	Do short builds to raise more opportunities for feedback. This requires automating as much work as possible.
Continuous Integration	XP demands integrate and test at regular intervals, with the aim of reducing the feedback cycle to a minimum. The greater the number of changes integrated at once, the greater the cost of analysis and resolution of issues.
Test-First Programming	Writing tests before writing the code.
Incremental Design	Investing in the design of the system on a daily basis instead of planning it all ahead. The design must grow as the implementation moves forward and the system's needs arise.

Source: Beck & Andres (2004)

1.5.6.2 Extreme Programming Team Roles

XP may have some key roles for the members of the team, as explained in Table 3. Having a more structured team can be beneficial in the early stages of adopting XP (Beck & Andres, 2004). However, it is noteworthy that "on a mature XP team [the roles] aren't fixed and rigid" (Beck & Andres,

2004), as the goal is to build a team where everyone feels empowered to contribute their best, and in some case the same individual may hold two roles (e.g. developer and tester).

Table 3. Roles and Responsibilities on XP teams

Role	Responsibilities
Programmers / Developers	Write the code and tests.
Customers	Identify the requirements and give feedback throughout the development process.
Testers	Create closer relationships with users by helping them learn about the product and listening to their feedback.
Tracker / Project Manager	Manage the project scope, schedule, and resources. Keep track of the progress in each iteration.
Coach	Guide the team members in following XP process.
Consultant	When the team needs deep technical knowledge or an outside vision to unblock, a consultant may be required to guide the team on solving a specific problem.
Product Manager / “Big Boss”	Define the product vision. Act as a coordinator, providing resources, guidance, and support, while maintaining team’s autonomy and trusting their decisions and processes.

Source: Al-Saqqa et al. (2020) and Beck & Andres (2004)

1.5.6.3 Extreme Programming Life Cycle

XP life cycle is comprised of five iterative and continuous stages: Exploration, Planning, Iterations to Release, Productionizing and Maintenance, and Death (Al-Saqqa et al., 2020).

1) Exploration Phase: the requirements are expressed in the form of user stories and the technologies, tools and other resources are explored; it can take one to four weeks.

2) Planning Phase: the user stories are prioritised and the a first story estimation is done to schedule the first release; it can take one to two days.

3) Iterations to Release Phase: each iteration shall take one to four weeks; at the end of one iteration, the features are tested and feedback is incorporated in the next iteration; development practices such as pair programming and refactoring (the improvement of code) take place in this phase.

4) Productionizing and Maintenance Phase: extra tests are implemented, including performance tests, before release to the customer; ideas are documented for later implementation in the maintenance stage; some iterations may occur after the initial release to production, which may imply extra effort in the customer support.

5) Death Phase: the product is in the death phase when the user has no more stories to add, which means that has completely satisfied the needs of the customer; death phase can also happen when the product totally fails to meet the customer expectations and there is no budget or interest in continuing to develop a solution.

1.5.7 DevOps

Sousa et al. (2019) explain that DevOps is not a totally clear concept. For some authors, it is a methodology for software development, to others a project management methodology, for others a type of organisational culture, and for others a movement emerging on IT. DevOps can be defined as a set of practices, tools, and cultural philosophies to enhance collaboration between software development (Dev) and IT operations (Ops) teams in order to achieve a continuous stream of planning, integration, delivery and feedback, to develop and release to the market high-quality products quickly (Faustino et al., 2022; Ozkan et al., 2020).

1.5.7.1 DevOps Life Cycle and Key Principles

DevOps is based on Scrum and XP, lying on the principles of short development cycles that incorporate constant feedback and whose purpose is to deliver value to the customer, but goes beyond by including the operations team since the beginning (Kim et al., 2021).

The key principles of DevOps include Continuous Integration (CI), Continuous Delivery (CD), automation, measuring and monitoring, and sharing (Kim et al., 2021). Developers regularly merge their code changes into a central repository, after which automated builds and tests are run (CI), and software can be released to production at any time (CD), often using automation tools and automated deployment pipelines (Azad & Hyrynsalmi, 2023). Automation plays a central role in DevOps, with the aim of having all repetitive tasks of build, test, deploy and monitor automated, to reduce human errors, save time for other non-repetitive tasks, and have shorter cycles of development and feedback (Sousa et al., 2019). Comprehensive monitoring is put in place to gain full visibility into the health of the system and the user experience, and measurement of progress and efficiency is done consistently to identify areas for improvement, in a clear Lean mindset.

Last but not less important, we have to emphasise the culture of sharing that is crucial for DevOps to succeed (Sousa et al., 2019). When Devs and Ops are separated, Devs are more concerned with the delivery, and Ops with the stability and smoothness of IT service and infrastructure. In DevOps, although each team keeps their own responsibilities, cooperation is fostered to facilitate each other's work. Learning is shared among teams, avoiding silos. Goals are common and the measurement of the development and operations teams are done simultaneously. The production of documentation must be shared among Development and Operations teams to create a holistic view of the product and maximise both process efficiency and quality for the customer (Sousa et al., 2019).

DevOps emphasises many of the people-related principles on the Agile Manifesto (Ozkan et al., 2020) and considers the team key element to success (Faustino et al., 2022).

1.5.7.2 DevOps Performance Indicators

DevOps measures performance and progress using a variety of metrics, such as deployment number (the number of deployments per day, week, or month), lead time (the time it takes for code to go from

"code committed" to "successfully running in production") and Mean Time to Restore Service (MTTR, the time it takes to resolve a service incident).

1.6 Agile Methodologies Hybridisation and Tailoring

Agile offers a broad spectrum of methodologies to managing projects. Moreover, these methodologies can be hybridised, allowing them to be combined with each other or with other project management approaches.

Scrumban is an example of a hybrid Agile methodology, combining aspects of two Agile methodologies - Scrum and of Kanban - as it best suits the teams using it. This means that not all teams employ Scrumban in the same manner. The possibilities are multiplied even further by the fact that Agile methodologies can be mixed with other approaches. In fact, 42% of the respondents to the 17th State of Agile Report (2023) reported that their organisation used (not necessarily exclusively) hybrid approaches that combine Agile with another family of methodologies, such as Waterfall, Lean or Spiral. Brown & Tuxworth (2020) found that, while it was commonly pointed that combining two approaches mitigates the limitations and disadvantages of each, there was "no universal agreement on the factor influencing the choice of the hybrid approach" (p. 4).

Even when teams adopt a "pure" methodology, they may not follow exactly the same rules and practices, have the same roles or use the same tools and performance indicators. And it is also common to implement a combination of Agile practices rather than adopting a methodology in its pure form (Cao & Ramesh, 2008; Sandstø & Reme-Ness, 2021). This is referred to as "methodology tailoring". It can be defined "as the adaptation of the method to the aspects, culture, objectives, environment and reality of the organisation adopting it." (Campanelli & Parreiras, 2015, p. 87)

As Sandstø & Reme-Ness explain, when selecting the right practices to apply, many factors have to be considered, "such as team size, culture and project complexity" (2021, p. 2). A study from Campanelli & Parreiras (2015) concluded that contextual criteria, such as project type, communication, culture and management support, and the established objectives, such as business goals, innovation level and complexity, were the most decisive in defining the way the methodologies should be tailored. Studies on critical factors for project success indicate that "dabbling in Agile by adopting a single practice is not likely to lead to observable or sustainable increases in overall performance" (F. Tripp & Armstrong, 2018, p. 177). The optimal strategy is "to blend the methods according to the needs instead of blindly adhering to one method" (Ozkan et al., 2022, p. 892) and thus Agile practices "should be tailored for each project's requirements" (Sandstø & Reme-Ness, 2021, p. 261).

1.7 Success in Agile

To be considered successful, a project must generate value, whether in the short or long-term (Project Management Institute, 2021). Project management in product development extends beyond delivering functionalities or features. It must also assess its own efficiency and ability to create value. Methodologies, while setting out some rules and providing specific tools, do not prescribe exactly

how success should be measured. Thus, the key performance indicators chosen by a team are based on a specific definition of project's constrains and purpose.

The Iron Triangle is a well-known concept in the field of Project Management whose purpose is to assist in the definition of a successful project. It represents "the relationship between key performance criteria" (Pollack et al., 2018, p. 527). Although success is the ultimate goal of a project, its definition is not straightforward and there is no consensus on what each vertex of the Iron Triangle represents, as Pollack et al. (2018) explain. In their research, they sought to identify the most commonly accepted version of it. Time, cost and quality appear to be the most frequently utilized criteria for evaluating project success. Time is defined as the schedule or duration of the project, cost refers to the budget or financial resources required, and quality is the desired attributes of the final product.

Highsmith (2009) proposes a new triangle: the Agile Triangle, composed of the vertices value, quality and constraints. It divides the quality into two clearly defined aspects - the value generated for the customer (the value vertex) and the technical quality or reliability of the product (the quality vertex). -, and combines the constraints (time and cost) in one single vertex.

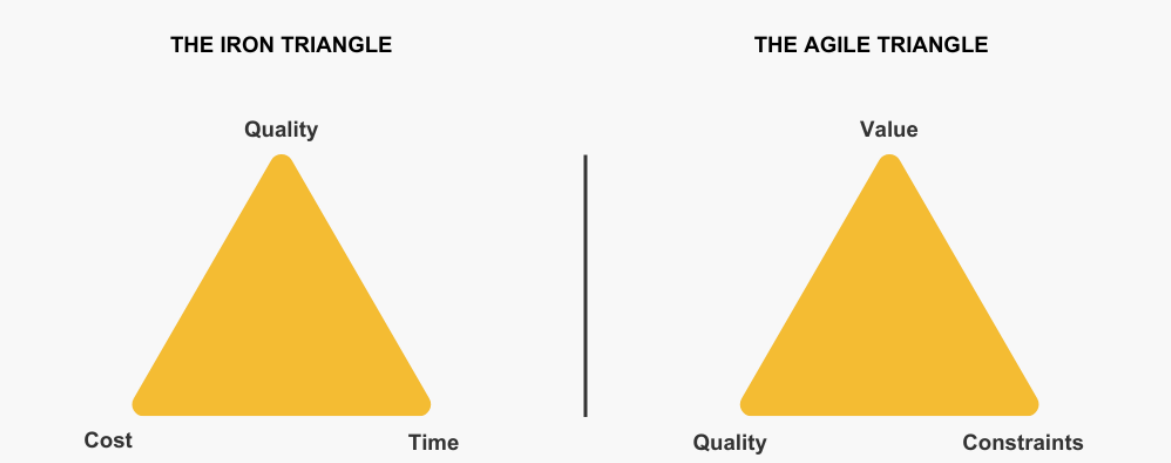

Figure 5. The Iron Triangle versus The Agile Triangle

In the Agile Triangle, the goal is to achieve value and quality within acceptable constraints, which are adjusted as needed and possible. This encourages adaptability and rewards teams for being Agile, rather than for strictly conforming to the original scope, schedule and budget. Agile teams must then develop performance indicators that align with the Agile Triangle, and go beyond time and cost metrics such as velocity, cycle time or lead time, using value metrics such as customer satisfaction or return on investment, and quality metrics such as deployment frequency, defect rates or lead time for changes. Hence, it is likely that the key performance indicators utilized by the companies in our study will go far beyond those traditionally associated with each methodology as previously exposed.

1.8 Why this study is relevant

The Agile mindset is the foundation, but the methodologies and practices are the instruments that put this mindset into action. Understanding how these tools are used in a real-world scenario can

provide a practical roadmap with valuable insights for organisations adopting Agile or adapting their Agile processes.

While there is no one-size-fits-all approach to Agile, studies like this with concrete examples of Agile in action, while not a silver bullet, can provide some guidance. Seeing a successful implementation can serve as a practical starting point for an organisation's Agile journey, inspiring creative adaptations and helping to improve processes. In addition, knowing the struggles and challenges others face creates more realistic expectations and encourages proactive rather than reactive problem solving. These are the reasons why this research into the practices, tools, team roles, key performance indicators and challenges of Agile project management is relevant.

2. Methodology

The participants interviewed in this study were selected by convenience, recurring to our contact network, from companies who respected the following criteria:

- Micro and small enterprises (employing a maximum of 50 people and whose annual business volume does not exceed 10 million euros);
- Developing digital products;
- Based in the Minho region;
- With at least three years of experience in Agile methodologies.

Potential participants were screened via email and LinkedIn messages to confirm eligibility.

The participant group is well diversified across several categories. There is a balanced representation of companies building their own products and those developing products for other organisations. Furthermore, the companies span a range of sizes, and vary in terms of years of experience in business and in Agile. It is worth noting that none of the companies in the study operate within a fully presential working model. Some have adopted hybrid models with mandatory office

days, but the majority works on a flexible remote basis and do not require the employees to attend the office at all. The precise annual business volume of each company was not disclosed, but it was ascertained that it met the defined criteria.

In each company, a Project Manager was identified as the person best equipped to answer the questions posed in the study. In this context, Project Manager is defined as “the person assigned by the performing organisation to lead the project team that is responsible for achieving the project objectives. Project managers perform a variety of functions, such as facilitating the project team work to achieve the outcomes and managing the processes to deliver intended outcomes.” (Project Management Institute, 2021, p. 4). Considering this broad definition, we also interviewed individuals whose formal company role was Product Owner, Product Manager, Head of Projects or Delivery Manager, and even C-level roles, namely Chief Technology Officer (CTO) and Chief Executive Officer (CEO).

Table 4 provides a detailed breakdown of company characteristics, including business model, size, time in market, work model, and Agile experience, and participant profiles, covering roles, academic backgrounds, and Agile training.

Table 4. Company and Participant Characteristics

Business Model	Market proprietary products	5
	Build products for other companies	6
Number of employees	0-10	1
	11-20	5
	21-30	2
	More than 30	3
Time in market	Less than 5 years	2
	5-10 years	6
	11 to 15 years	1
	More than 15 years	2
Experience in Agile	3-5 years	3
	More than 5 years	8
Work model	Flexible remote	7
	Hybrid	4
Participants’ Role	CTO or CEO	4
	Project Manager	2
	Product Manager	1
	Head of Projects	1
	Delivery Manager	1
	Product Owner	2
Participants’ Academic Background	Software Engineering	9
	Other	2
Participants’ Training on Agile	Self-taught / Informal	6
	Certified	5
Total number of companies		**11**

A total of 12 interviews were conducted, but the data from one company had to be excluded because, although the company director had described the company as "Agile", the interview with the project manager revealed that the company actually worked most of the time in Waterfall and had little experience of Agile, and therefore did not meet the criteria of having at least three years' experience of this type of methodologies. Thus, this study's findings are based on 11 interviews.

2.1 Method

The relevant information was sent to each participant via email, including an overview of the study, its purpose, a discussion of the ethical considerations involved, and a document to be signed acknowledging consent to participate in the study. This aligns with the standard procedure of the Comissão de Ética para as Ciências Sociais, da Vida e da Saúde (CECSVS-IPVC), the consultative, collegial, multidisciplinary, and independent committee for ethical issues related to research activities at ESTG-IPVC, which previously had issued a favourable statement on the research protocol proposed for this study.

A semi-structured interview script (Appendix A) was developed based on the literature review. This approach aimed to gather relevant information in a logical and flexible manner. The interview was structured around eight key themes, with the first two being contextual information:

1. Participant profile: professional experience, academic background and specific training in Agile methodologies
2. Company profile: company age, size, core business, and Agile project management maturity evaluated in years of experience.
3. Project management methodologies and implementation approach: Agile and hybrid methodologies in use.
4. Reasons for choosing those methodologies and practices: rationale behind the choice of methodologies and practices.
5. Core practices: key practices associated with the chosen Agile methodologies.
6. Team roles: defined roles and responsibilities within project teams.
7. Software and tools: employed software and tools for project management.
8. Performance indicators: key performance indicators to measure project management success.

The interviews were conducted via video calls between the months of November and December of 2024, sequentially until no new, substantial information emerged, and saturation (Knott et al., 2022) was reached. Saturation was reached at 11 interviews, when the existing themes and codes were sufficiently developed to provide a comprehensive understanding of the research questions:

- What methodologies are being used by companies and what are the reasons for their choice?

- What practices are associated with these methodologies?
- How are the teams organised in terms of roles?
- What software applications support their project management?
- How do companies measure the success of project management?

This iterative process involved analysing the data from each interview by recurring to Template Analysis (TA) technique (King et al., 2018; King & Brooks, 2017; Knott et al., 2022) and using the insights gained to guide subsequent interviews. As is the standard in TA, the procedure commenced with the identification of *a priori* themes (King et all, 2022). Key themes were derived both from academic research and practitioner literature, initially associated with a small number of codes and with an open perspective "to avoid overlooking new and important aspects of the data that did not explicitly relate to those themes elicited from earlier work" (Brooks et al., 2015, p. 215). These themes were clustered and organised into a structure - an initial template. The initial template was then implemented to code the data collected, and improved to fit the study's needs in an iterative process in which themes could be added, deleted or moved between clusters. When the template, with its themes, sub-themes and codes, was deemed to meet the adequacy criteria and no further refinements were needed, it was used to analyse and interpret all data.

For this qualitative analysis, TA was the elected technique because it is effective in organizing themes and at the same time flexible to adapt to emerging subjects. However, some limitations must be acknowledged. "The focus in Template Analysis is typically on across case rather than within case analysis, the result of which is unavoidably some loss of holistic understanding in relation to individual accounts."(Brooks et al., 2015, p. 218). Because the implementation of Agile methodologies is shaped by the specific organisational context, we wrote case summaries, as recommended by Brooks et al. (2015), to mitigate the loss of particularities. Rather than looking for a single interpretation or a universally generalizable conclusion, this analysis embraces the multitude of participants' perspectives, offering a nuanced understanding of Agile practices in small companies.

To guarantee anonymity, the 11 companies were assigned letter codes from A to K. Each interviewee was then identified using the same letter code as their respective company. For instance, the participant from Company A was designated "Participant A," the participant from Company B was labelled "Participant B," and so on.

3. Results

The initial version of the template analysis was developed considering the structure of the interview script, designed to align with the study's objectives and informed by the literature review. This structure allowed us to identify *a priori* themes (e.g., project management methodologies, reasons for the choice, core practices) while remaining open to new themes that could arise during the interviews. The data from themes 1 and 2 of the interview script was utilised solely as contextual background for describing the sample.

The tables in Appendix B provide a structured summary of each participant's responses using template analysis, including key excerpts from the interviews. A sequential review of those tables discloses the manner in which the template analysis has evolved based on participant input until the template reached its final version, which can be seen in Table 5.

As the interviews were subjected to analysis, a number of sub-themes and codes emerged, as well as a new theme – Challenges (7). Although not asked directly, all the participants have mentioned, directly or indirectly, the challenges they face and that can contribute to understanding the companies' project management choices and thus help respond to our research questions.

The initial and final templates are depicted in Table 5.

Table 5. Initial and Final Templates

Initial Template	Final Template
1. Project management methodologies 1.1. Project management methodologies in use 1.2. Hybridisation and tailoring	**1. Project management methodologies** 1.1. Scrum 1.2. Kanban 1.3. Scrum and Kanban 1.4. Scrum and Scrumban 1.5. Unidentified hybrid approaches
2. Reasons for the choice	**2. Reasons for the choice** 2.1. Internal drivers 2.1.1. Disciplined approach 2.1.2. Simplicity 2.1.3. Flexibility / adaptability 2.1.4. Predictability for the team 2.2. External influences 2.2.1. Client requirement
3. Core practices	**3. Core practices** 3.1. Scrum-related practices 3.1.1. Sprints and sprint planning 3.1.2. Sprint review 3.1.3. Sprint retrospective 3.2. Cross-methodology Agile practices 3.2.1. Daily syncs 3.2.2. Refinement meetings 3.2.3. Pair programming 3.2.4. Milestone-based planning 3.2.5. Time tracking 3.2.6. Knowledge sharing sessions 3.2.7. Weekly planning 3.2.8. Post-mortem 3.2.9. Decision logs 3.3. Agile methodologies tailoring 3.3.1. Alternatives to retrospective per sprint 3.3.2. No predefined periodicity for syncs 3.3.3. Allow stories to be added during the sprint 3.3.4. Estimation in time, not in story points 3.3.5. Sprint review, retrospective and planning in one single session 3.3.6. Mid-sprint review 3.3.7. No estimation per story 3.4. Kanban-related practices 3.4.1. Task pulling 3.4.2. WIP limits
4. Collaborative software and tools	**4. Collaborative software and tools** 4.1. Project tracking, documentation and backlog management 4.1.1. Zoho Sprints 4.1.2. Confluence 4.1.3. Google Workspace 4.1.4. Clickup 4.1.5. Jira 4.1.6 Trello 4.1.7. Notion 4.1.8. Odoo 4.1.9. GitHub 4.1.10. Linear 4.1.11. PHC 4.1.12. Power BI 4.1.13. Software developed in-house 4.1.14. GitLab 4.1.15. Coda 4.1.16. Microsoft Teams 4.2. Communication 4.2.1. Google Chat

	4.2.2. Google Meet 4.2.3. Slack 4.2.4. Discord 4.2.5. Microsoft Teams 4.3. Design collaboration 4.3.1. Figma 4.4. Code, testing and version control 4.4.1. Bitbucket 4.4.2. Github 4.4.3. Insomnia
5. Core team roles 5.1. Product Owner 5.2. Scrum Master 5.3. Developers 5.4. Project Manager 5.5. Product Manager	**5. Core team roles** 5.1. Roles 5.1.1. Product Owner 5.1.2. Scrum Master 5.1.3. Developers 5.1.4. QA 5.1.5. Designer 5.1.6. Project Manager 5.1.7. Architect 5.1.8. Team Lead 5.1.9. Product Manager 5.1.10. Business Manager 5.1.11. DevOps 5.1.12. Release Manager 5.1.13. Delivery Manager 5.1.14. Data Analyst 5..15. Tech Lead 5.1.16. Business Expert 5.2. Role overlaps
6. Key performance indicators	**6. Key performance indicators** 6.1. Efficiency and productivity 6.1.1. Velocity 6.1.2. Lead time 6.1.3. Timely delivery of projects 6.1.4. Earned value management 6.1.5. Delivery by assignee 6.1.6. Delivery by task type 6.1.7. Delivery by priority 6.2. Quality and continuous improvement 6.2.1. Team feedback 6.2.2. Bug tracking 6.3. Value 6.3.1. Client satisfaction 6.3.2. Contribution to OKRs (Objectives and Key Results)
	7. Challenges 7.1. Organisational and structural challenges 7.1.1. Scaling Agile 7.1.2. Managing cross-team collaboration 7.1.3. Lack of structured KPIs 7.1.4. Distributed teams 7.2. Operational and process challenges 7.2.1. Time estimation 7.2.2. Balancing flexibility and process rigor 7.2.3. Balance quality and constrains 7.2.4. Balance unexpected issues and planning 7.2.5. Meeting lengths 7.2.6. Workload balance 7.3. Client interaction challenges 7.3.1. Delays in client deliverables 7.3.2. Changing requirements and priorities 7.3.3. Overwhelming communication processes

3.1 Template Analysis

3.1.1 Project Management Methodologies

The answers related to the first theme “Project management methodologies” generated less codes than expected. Only two pure methodologies were identified (1.1. Scrum and 1.2. Kanban). Some participants answered their companies used more than one methodology depending on the nature of the projects, namely Scrum and Kanban (1.3) (“Scrum is our main Agile methodology. (…) For R&D or internal projects, we prefer milestone-based Kanban” – Participant A; “Kanban is our default when Scrum’s rigid structure isn’t suitable.” – Participant B; “Kanban (...) it's our default methodology (…). When we work with outsourcing clients, we follow the full Scrum process” – Participant D; “We use Scrum for development and Kanban for project management." – Participant C), and Scrum and Scrumban (1.4) (“When it’s a product or software project that’s going to take time, it has to be Scrum (…) [and] when we do Kanban, it's a mix of Kanban and Scrum" – Participant E). A few participants reported not following a specific methodology, opting instead to mix different Agile methodologies as needed, not specifying which (1.5. Unidentified hybrid approaches) ("We say that our approach is hybrid... always adapted and inspired by various methodologies depending on the situation." – Participant F; “We don’t follow a strict methodology but rather take ideas from multiple ones and adapt them to what works for us." – Participant G).

3.1.2 Reasons for the Selection of Agile Methodologies

Under the second theme, "Reasons for the choice", the responses were divided into two sub-themes: "Internal drivers" (2.1), or factors that come from within the organisation, such as workflow needs, team preferences and culture, and "External influences" (2.2), or factors that come from outside the organisation and dictate the methodology it will follow.

The benefits of a disciplined approach (2.1.2) (“When things need to be managed very well and because there are very concrete deliverables that cannot fail, that is typically Scrum." – Participant E), simplicity (2.1.2) (“It’s simpler (…) and flows more naturally." – Participant B), flexibility and adaptability (2.1.3) (“It allows us to adapt as we go” – Participant D; “(…) give us the flexibility to maintain structure where it’s needed while adapting to unexpected changes without losing momentum.” – Participant F) and predictability for the team (2.1.4) (“Scrum helps us create a routine and predictability" – Participant C; “Our methodology allows us the flexibility to make adaptations along the way, while giving us the stability of a routine." – Participant K) were the internal drivers mentioned for the choice of project management methodologies.

In terms of external influences on the definition of the methodology, client requirements (2.2.1) were mentioned by some participants from companies developing products for other businesses (“It is often imposed by the client” – Participant E; “Clients request it” – Participant B).

3.1.3 Core Practices

The responses to the theme “Core practices” resulted in the identification of four sub-themes: Scrum-related practices (3.1), cross-methodology Agile practices (3.2), Agile methodologies tailoring (3.3) and Kanban-related practices (3.4).

The Scrum-related practices mentioned by the participants included sprints and sprint planning (3.1.1), sprint review (3.1.2) ("Sprint planning and review are non-negotiable—they connect the team’s work to stakeholders and provide clear updates." – Participant A), and sprint retrospective (3.1.3) ("At the end of the day, the retrospective meeting" – Participant B).

The participants answers yielded several cross-methodology practices: daily syncs (3.2.1) (“On dailies, every team member talks about their tasks, regardless of the project they're on, which fosters collaboration and problem-solving” – Participant D), refinement meetings to clarify requirement and tasks (3.2.2.) ("Backlog refinement ensures we know what’s important and adjusts priorities as needed." – Participant A), pair programming (3.2.3) (“Pair programming happens organically” – Participant A), milestone-based planning (3.2.4) (“We plan milestones and review progress at each stage” – Participant G), time tracking of task or project execution (3.2.5) (“We ensure that hours are logged” – Participant B; “Each employee records the time they work on the project with date and time and description." – Participant E), knowledge sharing sessions (3.2.6) (“There might be a KT [knowledge transfer] session where a developer demonstrates a specific dashboard to the team, sharing their expertise” - Participant B), weekly planning (3.2.7) (“We plan weekly” – Participant G), post-mortem review which is a meeting that takes place after an incident or a significant event to find opportunities for improvement (3.2.8) (), and decision logs (3.2.9) (“Documenting decisions helps us evolve consistently and ensures that future decisions are made thoughtfully” – Participant K).

Tailoring Agile methodologies (3.3.) was reported by almost all participants, first in a generic sense (“We use Scrum but adapted to our realities, strengths, and weaknesses—it’s not the book version.” – Participant H; “No one does it by the book” – Participant J; “If we follow all Scrum rules, we wouldn't be as Agile as we are” – Participant K), but also with some specific examples from which codes emerged. Some alternatives to retrospective per sprint (3.3.1) were mentioned, ranging from quarterly (Participants A and F) and annual retrospectives (Participant G), to retrospectives at the end of a project (Participant I) and even the encouragement of retrospectives by each individual on an ongoing basis (“We found that we don't need a ceremony for reflection. If someone thinks that something is not right, they communicate. If they find a better way to do something, they will share with the team to improve. We encourage this attitude.” – Participant K). Team syncs, the short meetings in which a team aligns the work and shares the progress, usually called “dailies” or “daily stand-ups”, may be adjusted too and not held daily or even with a regular periodicity (3.3.2). Participant A revealed their “teams have autonomy to decide how often they hold dailies” and Participant B explained that “not all projects need daily stand-ups” and they may happen “every other day or weekly”. For some companies, sprint backlogs are often with stories or tasks added or removed in the middle of a sprint (3.3.3) (“If something critical comes up, we prioritise it immediately, even if it disrupts the sprint." – Participant C; “We aim to avoid changes during a sprint, but

commercial needs or legal obligations can force adjustments." – Participant H; "Tasks can be re-evaluated and swapped if something more important appears" – Participant I; "It is not uncommon to add multiple tasks to the sprint since they are related to the sprint goal. We actually leave some space for them when planning" – Participant K). Many companies reported estimate task effort in time (3.3.4), not in the more abstract story points, part of Scrum and Kanban ("Each point is a day" – Participant F; "We have the estimated time for the task"- Participant G; "Story points, we don't use them. (...) We abandoned it, because you realized and a sprint planning meeting, which should have lasted half an hour or an hour, lasted all afternoon, with endless discussions." – Participant E). Company J has even abolished estimation per story (3.3.7), relying solely on milestone-based planning ("We also took the estimation part out of the equation because in most cases it was counterproductive, it caused burnout in the team, frustration and we were distracting ourselves from the essential"). Another interesting tailoring of practices is related to Scrum ceremonies. Although Scrum specifies that the sprint review, retrospective and planning are separate ceremonies, many companies combine them into a single session to save time (3.3.5) ("We combine planning, review, and retrospective into one session to save time, especially for team members in hybrid or remote work." – Participant H). Mid-sprint reviews (3.3.6) are another practice reported by the companies that is not on Scrum rules. Participant K explains that it "is a check-up moment in the middle of the sprint (…), helps reorganise priorities, adjust the plan without waiting until the end of the Sprint." Participant I further elaborates that it allows to "analyse what can actually be done by the end [of the sprint] and if something more important has come up" and if there is need "to remove tasks" to let new ones to enter.

Under the sub-theme "Kanban-related practices", two topics emerged: task pulling (3.4.1) ("If they already have a little more knowledge of what they are developing, they can say 'look, I'll take this [task]'" – Participant B) and WIP limits (3.4.2) ("Yes, we have work in progress limits." – Participant D).

3.1.4 Collaborative Software and Tools

The responses to the theme "Collaborative software and tools" generated four sub-themes: Project tracking, documentation and backlog management (4.1), Communication (4.2), Design collaboration (4.3) and Code, testing and version control (4.4). Certain software applications mentioned serve multiple purposes and are used by the companies in multiple ways. Some companies leverage the full capabilities of these software applications, others utilise only specific features for specific needs, as deemed necessary. Consequently, they appear under more than one sub-theme.

For planning and tracking the progress of projects, as well as to organise all the documentation, the participants revealed their companies use Zoho Sprints (4.1.1) ("We use Zoho Sprints for backlog management" – Participant A), Confluence (4.1.2) ("...Confluence for documentation to keep everything organised." – Participant C), Google Workspace (4.1.3) ("Collaborative documents and spreadsheets, more complex things that require formulas or formatting that Notion can't handle are in Google Drive" – Participant K), Clickup (4.1.4) ("ClickUp manages to be in a single tool both the documentation part and the task part" – Participant B), Jira (4.1.5) ("Jira covers everything we need,

from project management to development workflows." – Participant I), Trello (4.1.6) ("Trello helps us quickly visualise what we're working on within the product team." – Participant C), Notion (4.1.7) ("The technical team uses Jira, the Content & Marketing team uses Notion. And then we have links from one to the other" – Participant K), Odoo (4.1.8) ("This ODOO... each employee records the time they work on the project with date and time and description" – Participant E), Github (4.1.9) ("GitHub allows us to tag issues, set priorities (P1 to P5), and keep everything tracked."- Participant E), Linear (4.1.10) ("Linear organises tasks by state (e.g., to do, in progress, QA) with clear ownership and expected durations." – Participant F), PHC (4.1.11) ("Trello gives us a Kanban-style view of the overall project, while PHC is used to track task hours and progress." – Participant G), Power BI (4.1.12) ("Since ClickUp has APIs, we will fetch the data and work in Power BI to make our own charts." – Participant H), Software developed in-house (4.1.13) ("We use ClickUp and we also have some part that is internally developed software."), GitLab (4.1.14), Coda (4.1.15) and Microsoft Teams (4.1.16) ("We have GitLab and we define all the tasks and milestones there. So as auxiliary tools we have Teams for project documentation, and we also have Coda, we put the technical documentation there." – Participant J).

Given the flexible remote and hybrid working models of these companies, applications that facilitate internal communication, either in real time or asynchronously, are of considerable relevance. The participants made reference to the use of Google Chat (4.2.1) ("We use Google Chat for internal company communication which is one of the most important tools we use because almost no one uses email" – Participant A), Google Meet (4.2.2) ("Meetings, dailies when we are not all at the office, all done with Google Meet" – Participant K), Slack (4.2.3) ("Slack for internal communication with channels by department and project" – Participant F), Discord (4.2.4) ("We also use Discord for dailies, and we usually spend all day on Discord, in one room, we are always connected to each other." – Participant D), and the already mentioned in the sub-theme 4.1 Microsoft Teams (4.2.5) ("We use Microsoft, Microsof Teams" – Participant I).

Only Figma (4.3.1) was mentioned as enabler of design collaboration (4.3) ("Figma helps us create layouts and gather client feedback efficiently." – Participant D).

Under the theme "Code, testing and version control", three software applications were reported by the participants: Bitbucket (4.4.1) ("We use Bitbucket for code management" – Participant A), Github (4.4.2), which was already mentioned as a software used for project planning and management ("GitHub, which is the platform we use to manage code, and we use, for example, the Scrum and Kanban tools that GitHub has, those boards" – Participant E), and Insomnia (4.4.3) ("Insomnia to test APIs"- Participant F).

3.1.5 Core Team Roles

From the answers to the questions related to the theme "5. Core Team Roles" two sub-themes have emerged: Roles (5.1) and Role overlap (5.2).

In total, 16 roles were identified: Product Owner (5.1), Scrum Master (5.2), Developers (5.3), QA (5.4) ("We usually have the Product Owner, the developers and the QA. The Product Owner is the

often the Scrum Master" – Participant A), Designer (5.5) (“a Product Designer or UI and UX Designer” – Participant F), Project Manager (5.6), Architect (5.7), Team Lead (5.8) (“In Kanban or in a more hybrid situation, there is the Project Manager, there is the architect, there is a team lead, and then there are the Developers” – Participant B), Product Manager (5.9) ("As a Product Manager, what I do is, together with the team, the entire team, (...) I try to understand how the product, how we, what we can improve, what user problems we can solve to achieve the company's objectives." – Participant C), Business Manager (5.10) ("Business managers are the ones who have direct contact with the client" – Participant D), DevOps (5.11) ("DevOps provides support at the infrastructure and CI level" – Participant F), Release Manager (5.12), Delivery Manager (5.13) ("The Delivery Manager is the facilitator in carrying out tasks/projects and Release Manager coordinates and manages the product launch process" – Participant F), Data Analyst (5.14) ("The Data Analyst is the one who analyses data to provide insights that inform business and development decisions." – Participant F), Tech Lead (5.15) (“We have a Tech Lead who leads technical requirements and the implementation of projects/tasks." – Participant F) and Business Expert (5.16) ("Every team has at least one business expert, sometimes more, depending on the area… a person who has more knowledge of the business area" – Participant H).

Role overlaps (5.2) was frequently related in different forms ("I'm CTO, I act as Product Owner as well" – Participant A; “We do a little bit of everything. I just don't do development, because that's no longer possible, it's no longer compatible. And I also do a bit of Team Lead." – Participant E; "Despite my Delivery Manager title, I still oversee QA and release management responsibilities." – Participant F; "The Product Manager, the CEO, or the Tech Lead can act as Scrum Master. The Tech Lead is also a developer. The Product Owner is the Content Team Lead. Having many hats in a small company is the norm." – Participant K).

3.1.6 Key Performance Indicators

The responses to the questions related to the Key performance indicators (6) in use by the companies generated three sub-themes: Efficiency and productivity (6.1), Quality and continuous improvement (6.2) and Value (6.3).

The indicators employed by the companies to measure efficiency and productivity include velocity (6.1.1) ("Velocity is not just for us to plan sprints. In a recent project, improving velocity over sprints was also an indicator that we were becoming more familiar with a technology. We realized we could deliver value faster.” – Participant K), lead time (6.1.2) (“Lead time to measure feature delivery time” – Participant A), timely delivery of projects (6.1.3) (“The KPIs are the delivery of the project on time” – Participant J), earned value management (6.1.4) ("EVM helps us link the work delivered to the budget and time spent, providing a clearer picture of productivity (…) We analyse earned value to ensure that the delivered output aligns with the resources used and the planned timeline.” – Participant B), delivery by assignee (6.1.5) ("Total points delivered by assignee. Each point equals one day; monthly reviews highlight deviations from expectations." – Participant F), delivery by task type (6.1.6) (“This allows us to check, on a monthly basis, what the team is most focused on. If it's projects, if it's improvements, bugs... the distribution of work." – Participant F), and delivery by priority

(6.1.7) ("Urgent tasks are reviewed to determine if they were proactive or reactive and whether they could've been avoided." – Participant F).

Quality and improvement are analysed with two indicators: team feedback (6.2.1) ("We try to find out if (...) the team was satisfied with that project or how the project was going, whether the team liked it or not, whether we should continue in that direction or not (...) What problems did we have in this project, which we were able to resolve in advance in future ones?" – Participant D) and bug tracking (6.2.2) ("for quality bug tracking is important" – Participant H).

As value creation indicators, the participants reported their companies measure client satisfaction (6.3.1) ("Client satisfaction is a continuous metric, assessed through feedback during the project, not just at the end." – Participant B) and contribution to the company's objectives and key results (6.3.2) ("What we defined for this quarter's OKRS, was it achieved or not…" – Participant C).

The challenges faced by the companies (7) were divided into three sub-themes: high-level organisational and structural challenges, related to the organisation's characteristics and culture (7.1), team-level operational and process challenges (7.2) and client interaction challenges (7.3).

Under the organisational and structural challenges, the participants identified scaling Agile (7.1.1) "As the team grows, maintaining efficient workflows and avoiding process overload is a priority." – Participant H), managing cross-team collaboration (7.1.2) ("We always have elements from other companies on the client's team, and we need to align with their processes and practices." – Participant B), lack of structured KPIs (7.1.3) ("We need to think properly about this subject of KPIs. We are very focused on producing and as a small team with few resources, things like these, less urgent, are postponed." – Participant K), and distributed teams (7.1.4) ("Feedback must be organised efficiently to keep remote engineers productive." – Participant F).

The operational and process challenges identified were time estimation (7.2.1) ("Our biggest challenge is undoubtedly time estimation because our work is not repetitive. We are always developing new features with different technical challenges and it is very difficult to understand the level of effort and how long it will take us." – Participant K), balancing flexibility and process rigor (7.2.2) ("We don't follow Scrum or Kanban rigidly; it's a blend that works for us, but explaining it to clients can be tricky (…) a more structured approach might be better for larger projects." – Participant E), balance quality and constrains ("If meeting the deadline risks quality, we either extend time or allocate more resources, depending on the client's flexibility, it's challenging" – Participant B), balance unexpected issues and planning ("We allocate 20-30% of sprint time to support, but emergencies can still disrupt plans." – Participant H), meeting lengths ("Our refinement meetings are long because we aim to convince the engineering team of the strategy, it's a pain point." – Participant C), and workload balance (7.2.6) ("Developers often work across projects, so we try to balance workloads to avoid burnout." – Participant D).

In the client interaction level, the challenges included delays in client deliverables that can delay the whole project (7.3.1) ("Clients don't always deliver required materials on time, which delays development and integration." – Participant D), changing requirements and priorities (7.3.2) ("Midway

through the project, clients sometimes request additional features or changes that weren't initially defined.” – Participant I) and overwhelming communication processes (7.3.3) ("A client who wants, for example, daily, practically daily, to do retrospectives, to see what is happening, wants to see results. It is very difficult to manage such a client." – Participant E).

3.1.7 Case summaries

Each company has its own set of methodologies, practices, tools, roles and performance indicators. Not presenting the companies singularities would be oversimplifying the context-dependent ways in which small companies implement Agile. To preserve the specifics of each company, we wrote case summaries as recommended by Brooks et al. (2015). As data collection was limited to a single interview per company, these case summaries cannot be considered an exhaustive representation of each company. However, even with their partial nature, they still provide valuable insights into each case and can help complement our analysis. More information on each Company case can be found in Appendix B.

3.1.7.1 Company A

Company A is a software company specializing in security solutions for the gaming industry, that has been operating for nine years and currently has between 15 and 20 employees.

Scrum is its main Agile methodology, with adaptations based on past learnings and project needs. They also use Kanban for R&D projects or smaller, discovery initiatives. The choice between Scrum and Kanban is made according to project complexity, team size, and the necessity for structured planning versus flexibility.

The company keeps Scrum events and artifacts with a few adaptations. For instance, stand-up meetings may not occur on a daily-basis, depending on the team's preference, and sprint retrospectives are held quarterly instead of after every sprint to avoid excessive meetings. Sprint reviews and sprint planning are considered the most important ceremonies. Backlog refinement meetings occur every two weeks to reassess priorities and define new tasks.

The Product Owner (also acting as Scrum Master) works closely with developers and designers, validates designs, refines backlogs, and aligns projects with company goals. Although Product Owner provides guidance, decisions are made collaboratively. The developers are assigned tasks (mostly by the Product Owner) based on their technical expertise, but they manage their own workload and execution. Designers work is executed outside sprints in a parallel process.

Zoho Sprints is used for backlog management and progress tracking. For code management, the company uses Bitbucket, and Confluence for technical documentation – both from Atlassian Suite. Communication is facilitated by Google Chat and Google Meet, and for file sharing they use Google Drive. For design collaboration, the company has Figma.

The company analyses burn-down charts for tracking sprint progress, lead time for measuring task completion efficiency. Timely delivery is important but not more than product quality. Qualitative

assessments are done by collecting feedback from the team members through one-on-one meetings and quarterly retrospectives.

The company has adapted Scrum to reduce excessive meetings, such as holding retrospectives only quarterly, but is still struggling to make Scrum events efficient. As the team grows, the company may introduce performance-based rewards and more quantitative tracking methods to improve accountability and efficiency. However, past attempts to track task hours suffered from a lack of adherence.

3.1.7.2 Company B

Company B is a company specialized in Data Engineering, Data Science, and Data Strategy services, with six years of activity and ten to 15 employees.

The company has a preference for Kanban due to its flexibility. However, Scrum is implemented often when working with clients who require a more structured management. The choice between Scrum and Kanban depends on the nature of the project, client requirements, and the maturity of the team. When working with Scrum, the company follows the methodology strictly, implementing all standard ceremonies and practices.

Although the company follows all Scrum rules when working in Scrum, the process is more flexible when working in Kanban - stand-up meetings don't have a daily rule, they are done as needed. In Kanban, the only scheduled meeting is a weekly meeting to refine tasks, adjust priorities and distribute work. Because of the nature of their business, logging hours per task is a common practice. Team members often participate in multiple projects simultaneously, and some may involve integrating Company B employees into external teams.

For projects using Scrum, standard roles such as Scrum Master, Product Owner, and Developers are clearly defined. For Kanban-based projects, teams include a Project Manager, Team Leader, and Developers, with distribution of responsibilities based on expertise.

ClickUp is their primary tool for both task management and documentation. They complement this with Google Drive to support documentation, and Google Meet for virtual communication. The company may also use other software by client demand,

The company uses indicators such as burn-down/burn-up charts and Earned Value Management (EVM), and finds timely delivery essential to attend client’s expectations and guarantee satisfaction. They conduct qualitative assessments with regular check-ins and feedback loops with clients, as well as team retrospectives and feedback sessions.

Ensuring high-quality deliverables within budget, time, and resource constraints is an ongoing challenge. On the other hand, Many employees work on multiple projects, which requires careful scheduling of meetings and workload distribution.

3.1.7.3 Company C

Company C is a health-focused company that develops their own products (B2B and B2C). The company has now more than 30 employees.

It follows Scrum, with some tailoring. The methodology provides predictability and routine with structured sprints and planning, while allowing for flexibility to accommodate urgent issues. Because the teams that support customers are the same teams that develop the products, it is common for support-related tasks to be included in a sprint even though they do not contribute to the sprint goal.

Engineering and product teams work in separate but coordinated cycles, aligning efforts through refinement and planning sessions.

Product managers define strategy, align stakeholders, and oversee development, while the team lead oversees technical implementation, and developers execute tasks based on sprint planning. Designers have a parallel process to feed each sprint. Role overlaps are common.

Company C relies mostly on Atlassian Suite - Jira is their backbone for backlog management and sprint tracking, Trello is used for high-level planning, Confluence handles documentation and Bitbucket is used for code management and version control. Communication is done mostly through Slack.

The team estimates effort using story points but does not rigidly track velocity, instead focusing on product impact and goal achievement. Success is measured more by user adoption and OKR aligned than by raw sprint completion metrics.

While Scrum provides structure, the company frequently adapts it to allow for urgent fixes and evolving priorities – for instance, a designated support engineer prioritises bug fixes based on impact, allowing urgent issues to be handled within or outside the sprint cycle.

The company encourages critical thinking and discussion, but meetings often run too long. The team is looking for ways to streamline discussions while keeping everyone engaged and in alignment.

3.1.7.4 Company D

Company D is a software development company specializing in custom software solutions, including mobile and web applications. It was founded about three years ago and currently has a team of 10 to 15 people.

Kanban with Lean principles is the default methodology of Company D to improve efficiency and reduce waste. The company might also work in Scrum when collaborating on projects with external teams. The choice of Kanban aligns with the company’s need for flexibility, given the projects durations (often short-term) and scopes.

Daily stand-ups (in both Kanban and Scrum) ensure alignment and knowledge sharing across projects. As a team that works regularly in Kanban, they perform task pulls and impose work in progress limits. Retrospectives are held informally throughout the project rather than as structured sprint reviews. Due to the nature of their business, time tracking per task is a regular practice. However, there is no fixed time for each task-the company follows a broader roadmap with feature-based planning.

Project manager oversees internal workflows and supports development processes, while the CTO guides technical strategy, defines project technologies, and ensures engineering best practices, and

developers handle implementation. Testing responsibilities are shared among developers, the project manager, and business managers

ClickUp is their main tool for task management. Communication is primarily through Slack, which is integrated with ClickUp for real-time task updates. For instant communication, they also use Discord. Google Drive is their repository for documentation, client assets, and project resources. Figma is used for design collaboration.

Timely delivery of projects is the main KPI. The team assesses deviations from initial project estimates to refine future planning. Each team member logs hours worked on tasks to assess progress and workload balance. Project success is measured by client satisfaction and post-delivery feedback, as well as team feedback.

The biggest challenge is time estimation. Clients are sometimes late in providing needed resources or feedback, and often expect changes late in development, adding pressure to a workload already strained by team members working on multiple projects simultaneously. Improved documentation and prototyping help mitigate this but are not enough.

3.1.7.5 Company E

Company E is a software development company specialising in blockchain, fintech, and digital product development, established 16 years ago, and with a team of approximately 30 employees.

The company has a hybrid Agile approach, primarily using Scrumban because the rigidity of Scrum does not match the evolving nature of their projects. Nonetheless, they may also use by client demand.

They do retrospectives in an informal way and more to assess if the sprint goals were met. Due to the nature of the business, team members track the time spent on each task.

Most of the projects have a product owner, but some may have a project manager instead. Project managers act as client proxies and oversee project timelines and planning, while product owners focus on long-term projects, product vision and strategy to maximise product value. The tech leads are the ones who often assign tasks within their respective teams (front-end, back-end, DevOps) and provide technical oversight. Developers and designers work within their specialization areas, and the QA team conducts tests. There are role overlaps.

For backlog management, the team uses GitHub, with occasional use of Jira when client requirements dictate. Internal communication is handled through Slack and Google Meet. Google Sheets is used for resource allocation and team planning. Odoo is employed for time tracking.

Timely delivery is the key performance indicator. The team compares initial effort estimates with actual hours logged to track project profitability.

The relationship with the client presents many challenges. Some clients expect daily involvement, requiring the company to set boundaries for Agile ceremonies. Late-stage changes from clients can impact sprint planning. While effort tracking is well-established, success is largely measured by client

satisfaction and budget adherence, which are insufficient for objective measurement. The company recognizes the need for more structured metrics.

3.1.7.6 Company F

Company F is a fintech company specialising in blockchain and digital payment solutions. Established in 2018, the company has grown to approximately 30 employees.

The company follows a hybrid Agile methodology, blending Scrum, Kanban, and Lean principles to maintain flexibility while ensuring structured project management.

Tasks are estimated in days, not story points. The company conducts daily syncs, weekly planning and frequent backlog refinement meetings to ensure the distributed team is aligned, working as one, supporting each other, and that the goals and priorities are well defined. They keep track of decisions with decision logs – a document for each project. To refine processes and learn, the team organises retrospectives in the end of big projects, and post-mortems after major incidents.

A Product Manager is responsible for defining the requirements and priorities with the stakeholders. Each project as a Tech Lead, developers, a designer, a QA, a Release Manager, a Delivery Manager, a DevOps Engineer, and a Data Analyst. The same person assumes the roles of QA, responsible for tests and quality, Release Manager, overseeing the deployment process, and Delivery Manager facilitating the Agile ceremonies and the project execution.

The company uses Linear for task and project management. Documentation, decision logs, and project records are maintained using Notion and Google Workspace, and GitHub handles code management and version control. Slack is their main communication platform, organised by department and project channels. Google Meet is used for virtual meetings and Figma for design collaboration.

The company measures lead time from multiple perspectives, such as delivery by assignee, delivery by task type, and delivery by priority. Other relevant KPIs include team feedback gathered through retrospectives and post-mortem meetings, monitoring quality by tracking bugs.

A distributed team presents many hurdles in keeping the team organised and motivated. The participant also noted that scaling Agile and fostering cross-functional collaboration in a growing distributed team are significant challenges that are currently being addressed with constant check-ins to validate processes and promote knowledge sharing. In addition, ensuring that workloads remain balanced in a fast-paced environment adds another layer of complexity.

3.1.7.7 Company G

Company G is a software development company specialized in integrated solutions for business process automation. Established in 2016, has a team of 12 employees.

The company does not follow an identifiable methodology, choosing to incorporate elements of Scrum and Kanban as they found this approach much more flexible and adequate to their needs.

The team plans every Friday the work for the following week. Tasks arising after this meeting are generally deferred to the next planning session, unless they are really urgent. Tasks are assigned by the Project Manager and added to Kanban boards in Trello for macro-view and in PHC for time tracking, and then synced with Google Calendar. Dailies can be done by department or with just a few team members, depending on what projects are in progress and the scope of the projects. At the end of the year, a retrospective is conducted to assess opportunities for improvement.

The team consists of a technical director, project managers, developers (divided into front-end, back-end and full-stack), and designer. In addition to being the technical director, the CEO can also take on some of the work of the developers and the role of project manager. QA is done by an external consultant to ensure independent quality testing.

Company G uses PHC as their primary software for project management, task tracking, and logging support requests, which they complement with Trello for a macro view of projects and progress. Google Calendar is utilized for scheduling tasks.

Timely and on budget delivery of projects is a core KPI. The team tracks discrepancies between estimated and actual completion times to improve planning accuracy. In a year-end retrospective the team analyses all projects to identify patterns and process improvements.

Company G points out time estimation as a big challenge because of unexpected support tasks on one hand and client dependencies on the other.

3.1.7.8 Company H

Company H is a software company specialising in management solutions for schools and training centres. Established 17 years ago, the company has grown to a team of 35 employees.

It follows Scrum, adapted to the company's workflow and specific needs. The team started with Scrum because it was suggested by a colleague who had worked at a large technology company, and they found it to be a great methodology for delivering value consistently while adapting to change.

Company H runs 15-day sprints with daily syncs to review progress and obstacles, combining sprint planning, review and retrospective in the same session - always in person. The planning always leaves room for unexpected urgent tasks.

Each team has a tech lead, plus several developers, and, at least, one senior business specialist. The team may include a designer as well, depending on the project needs. A product owner defines priorities and oversees project execution and a QA is responsible for the testing.

Company H combines ClickUp with an internally developed software for task management and documentation. They use Power BI to track KPIs and analyse project performance, providing valuable insights for decision-making. Communication is facilitated through Microsoft Teams.

Timely delivery of projects and timely execution of tasks are key performance indicators. The company also analyses patterns in bug reports, feature requests, and client and team feedback.

While the sprint backlog is generally respected, changes and unexpected issues can impact planning. Scaling Agile in a growing team without process overload was another challenge identified.

3.1.7.9 Company I

Company I, officially established three years ago, is a data-oriented organisation that operates as a collaborative laboratory, with almost 40 employees.

Company I adopts a hybrid Agile approach, integrating elements of Scrum and Kanban to fit its project dynamics. Scrum is applied most of software development projects, ensuring structured sprint cycles and backlog management. Kanban, on the other hand, is used for project management and task prioritisation, particularly in innovation-related initiatives and funding competitions.

The company has two distinct teams, one responsible for project acquisition, working in Kanban, and another focused on execution, working in Scrum but with some tailoring. Stand-up meetings are part of their daily schedules. Tasks can be re-prioritised within ongoing sprints, ensuring responsiveness to urgent client demands or newly identified priorities. Retrospectives are typically held at the end of projects rather than after each sprint.

Scrum Master works also as Project Manager and oversee Agile processes, ensuring alignment with business goals. Tech leads coordinate development efforts and provide technical support to project teams. Developers and Data Analysts work on implementation and designers engage at the start of projects for initial designs and return at the final testing stages to ensure usability and refinement. There are role overlaps, e.g. Scrum Master can be done by a Project Manager or by a Product Owner. QA is also done by project managers and designers.

Company I uses Jira for organizing tasks and workflows, and GitHub for source code repository and version control. Internal communication is conducted through Microsoft Teams, while Figma is used for design., ensuring code integrity and collaboration.

Instead of using story points, the company evaluates project completion against estimated time allocations. Rather than tracking team velocity, Company I assesses performance per project module and client validation milestones. Client involvement is predefined in the project communication plan, ensuring periodic feedback and iterative refinements. That feedback is used as a key performance indicator, as well as team feedback.

Defining requirements with clients to create a proper roadmap is a big challenge. Sprint cycles are defined, but mid-sprint task adjustments are permitted to accommodate changes and critical issues.

3.1.7.10 Company J

Company J is a software development company with a team of approximately 20 employees, established over eight years ago, that specialises in building custom digital products tailored to client needs.

Clients may require the company to work under other methodologies, but usually the company generally follows a Scrum-based approach, adapted to practical needs.

The company organises two types of daily syncs: a company-wide daily of 10 minutes and a project specific daily with 15 minutes. They don’t do task estimation as they found it “counterproductive” in most of cases. Reviews, retrospectives and planning are done in the same session (though these are distinct segments), and a rotating 'meeting owner' facilitates the process. Refinement is done after the sprint planning.

The team always has a technical lead and developers. The product owner sets priorities and interacts with the customer. Sometimes the same person can combine the roles of Tech Lead and Product Owner.

Company J relies on GitLab for backlog organisation and sprint planning, and Coda for technical documentation. Communication is handled through Microsoft Teams.

The main KPI is the timely delivery of projects, tracked by sprint delivery.

3.1.7.11 Company K

A software development company specialising in entertainment & lifestyle mobile apps. While their primary focus is B2C, they have also developed solutions for other businesses. The company has been in the market for 10 years and has operated with Scrum for the past 5 years. MB has a team of 8 employees working in a flexible remote environment with an office space available for collaboration.

The company follows Scrum, with significant adaptations to ensure flexibility. The adoption of Scrum came as the team started to grow, requiring more organisation without losing agility. Since Scrum is a well-known methodology, they decided it was the right fit if tailored to its needs.

The company conducts daily syncs with the whole team, not just the development teams, to ensure alignment across departments. As refinement happens during the sprints, it is common to have new stories entering the sprint after it has started and the team actually leaves space for that to happen. Mid-sprint reviews help them reorganise priorities, adjust the plan without waiting until the end of the sprint. Sprint review and sprint planning are done in the same session. The company does not have retrospective meetings as the team is encouraged to continuously share their feedback and suggestions for improvement.

The team has a Product Manager who defines the product strategy and high-level goals, and a Product Owner who translates that strategy into actionable plans with the development team. A Tech Lead oversees the technical implementation and the developers handle both front-end and back-end development. Given the product's strong focus on content, UX/UI designers and content creators are also an integral part of the team. There are several role overlaps.

While the technical team uses Jira for backlog management and sprint tracking, and Confluence for technical documentation, the Content & Marketing team, uses Notion for backlog management, and p documentation. Everything is linked manually. They also use Google Docs and Spreadsheets alongside Notion for product documentation. Slack and Google Meet facilitate communication and Figma supports design collaboration. GitHub is their software for code repository and version control.

Meeting deadlines is a primary KPI, but delivery is only considered successful if quality aligns with project goals. Client satisfaction is assessed through user engagement metrics and occasional surveys.

Despite relying on historical velocity, estimations remain an issue for the company, with timely delivery of projects identified as a key challenge. The participant has stated the team is prioritising execution over tracking and recognizes the lack of structured KPIs to measure project management success.

4. Discussion

The aim of this study was to understand how small companies developing digital products in the Minho region are using Agile for project management. The first research question was what methodologies are being employed and the underlying rationales for their selection. The participant responses identified Scrum, Kanban and Scrumban. Some participants reported not following a single methodology and could not identify the methodologies they used, and were therefore labelled in this study as using “unidentifiable Agile methodologies”.

Scrum has been highlighted as one of the most prevalent methodologies, with nine out of the 11 participants citing it. This aligns with the global trends document in the 2023 State of Agile Report and the State of Agile Culture 2020 Report. However, there may be a divergence in the way these small companies are actually implementing Scrum. They tend to customise it and utilise it in conjunction with other methodologies. Almost all the companies interviewed tailor Agile practices rather than strictly adhering to a prescribed methodology, even when they claim to follow a specific methodology, as we will analyse in detail later.

Some companies favour Scrum as their primary methodology for product development, yet using it in conjunction with Kanban for simpler projects (Participant A) or for overseeing projects at a high-level (Participant I). This employment of Kanban for internal projects aligns with its origins in Lean manufacturing and its focus on workflow optimisation (Zayat & Senvar, 2020). Its simplicity and adaptability make it well-suited for small companies with limited resources.

Scrum's structured approach and predictability, with defined ceremonies and roles, are the internal drivers for its adoption, and are particularly valuable for small companies managing projects for other businesses. It allows teams to deliver value periodically and incrementally while keeping the focus on the product goal (Schwaber and Sutherland, 2020). This is also one of the external factors for choosing Scrum. Clients also trust this methodology deliverability and often request its use, as reported by Participants B and E.

Other companies prefer Kanban for its simplicity and lightweight approach. As Participant B mentioned "It's simpler and flows more naturally". This supports the literature on Kanban's focus on workflow optimisation and ease of implementation (Petricioli & Fertalj, 2022).

Flexibility and adaptability, core principles of Agile, as outlined in the Agile Manifesto (Beck et al., 2001), were associated with both Scrum and Kanban. Both Scrum and Kanban enable teams to respond to change, but in different ways: Scrum through iterative cycles and Kanban through a continuous flow.

Scrumban, on its side, combines the iterative structure of Scrum with the flow-based approach of Kanban and was mentioned by one of the participants as their main methodology. The company does not use the term "Scrumban", but the participant described using a blend of Scrum and Kanban based on context and project needs, as defined by Ozkan et al. (2022) and by Alqudah & Razali (2018), and, despite operating within sprints, the team applies it the Kanban prioritisation thinking with the objective of optimising the flow of work (Portman, 2020).

There are also cases where companies do not follow a specific methodology, but instead mix different Agile methodologies as needed, reflecting a pragmatic approach from small companies, as noted by Campanelli and Parreiras (2015).

The second research question was "What are the core Agile practices of the teams?". The Scrum-related practices used by the companies are, as expected, all the Scrum events: Sprint, Sprint Planning, Sprint Review and Sprint Retrospective. But some of these were also found with some customisation, such as retrospectives conducted at the end of a project, quarterly, annually, or completely eliminated as a formal ceremony and done instead on an ongoing basis. Retrospectives can also be held in the middle of a sprint to understand "what can actually be done by the end or if something more important has come up", as Participant I explains. There are also cases where sprint review, retrospective and planning take place in a single meeting (Participant H). Adaptation is recurrent.

Numerous cross-methodology Agile practices were identified. For exemple, syncs, also known as stand-ups, are done by all the companies, reflecting the value of "individuals and interactions over

processes and tools” and the principle of communication from the Agile Manifesto (Beck et al., 2001). This is consistent with the finding of the systematic literature review by Ozkan et al. that this is indeed a “common communication manner in the hybrid models” (2022, p. 892). Periodicity, nonetheless, may vary. While daily syncs are the most common, some companies allow their teams to determine this meeting frequency according to project’s demands.

Another important meeting that helps the team stay aligned and delve into the project’s details and priorities to define the work to be done is the refinement meeting. As Participant A explained “Backlog refinement ensures we know what’s important and adjusts priorities as needed”. The timing of this meeting varies from company to company. Some hold it before sprint planning, some after planning but before the sprint starts, and some during the sprint.

Decision logs, documents that record key decisions, promote transparency and ensure everyone is on the same page. As Participant K emphasised, they have a long-term importance as keeping a history of decisions provides sustainability to the progress and future decisions.

Pair programming, a core practice from Extreme Programming (XP), was found in the companies analysed, although none of the participants mentioned this methodology. Pair programming reflects the principle of collaborative development and improves code quality (Beck & Andres, 2004), besides stimulating knowledge sharing. But there are also specific knowledge sharing sessions. This and post-mortem reviews are moments to learn and optimise processes, in a perspective of continuous improvement.

Weekly planning is often used by teams working in Kanban to stay aligned and respond to changing priorities. It may be combined with milestone-based planning (also used by companies working with another methodologies), keeping the team focused on key deliverables. These two practices align with Agile’s Manifesto (Beck et al., 2001) emphasis on delivering value incrementally.

Time tracking by task is a common practice among companies that develop products for other businesses. While not a core Agile practice, it is critical for companies to support resource management and ensure that time is allocated to high priority tasks, and to maintain transparency with the client, especially for hour-based contracts.

Time tracking leads to time estimates rather than more abstract effort measures such as story points. Interestingly, even teams that don't track time per task often estimate time by making approximations or converting story points directly into time. As Highsmith (2009) states, the goal of Agile is to deliver value and quality within acceptable constraints. This focus on the constraint of time (and therefore budget), although not fundamentally opposed to Agile, can potentially overshadow the focus on delivering value and maintaining quality. Nevertheless, it was the strategy these small companies found to respond to their context of hourly contracts and/or limited resources, having already tried story point estimation, as participant J and participant E reported. There is also the case of companies not estimating story size at all, relying solely on milestone planning.

Regarding adaptation and the Agile value of "responding to change over following a plan" (Beck et al., 2001, para. 2), it's noteworthy that several companies are open to letting new tasks enter during

the sprint or iteration cycle, whether they are linked to the goal of that cycle (Participant J) or to priorities that have emerged in the meantime (Participant C). This flexibility may be a consequence of non-exhaustive refinement meetings, but further research is needed to verify the validity of this assertion.

Our third research question was “what software applications support the small companies project management?”. Hybrid and flexible remote work models, as the ones found in these small companies in the Minho region, are pretty dependent on efficient software not just for communication, but also for project tracking, backlog management, work visibility, documentation, design collaboration, and code, testing and version control.

In the field of software applications, the uses are diversified as well and no conjunction of tools was repeated among companies.

Jira is widely used for backlog management and sprint tracking (Companies C, I, K), but there are also some less common options such as Linear (Company F), Zoho Sprints (Company A), GitHub (Company E) or PHC (Company G). Teams that reported using Jira use Confluence for documentation, a software from the same ecosystem (Atlassian), but there are also companies that rely on Confluence for documentation (Participant C) and Trello, also from Atlassian, for high-level project management (Participant G) although their main project management software is from other proprietary software houses.

Companies B and H, for example, prefer ClickUp, a tool that can do both project management and documentation. For documentation and collaboration, Google Workspace (especially Google Spreadsheets and Google Docs) and Notion are popular options, although there are others. For design collaboration, companies prefer Figma, and for code collaboration and version control GitLab, GitHub, BitBucket and Insomnia.

The companies doing time tracking per tasks can do it on ClickUp, PHC, or in a separate tool like Odoo.

For real-time communication, companies are using both synchronous (Google Meet, Microsoft Teams and Discord) and asynchronous (Slack and Google Chat) tools, challenging the assumption that Agile communication must be face-to-face. Modern lifestyles and technological advances have proven that restricting communication to face-to-face interactions would, in fact, be contrary to Agile (Ozkan, 2019).

As can be observed from the case summaries, companies developing products for other businesses may have their clients deciding on the primary software for project management, regardless of whether the team is mixed, or is the provider’s team solely.

When we look at the case summaries we can also understand that some companies use multiple tools (e.g., Jira and Notion, Notion and Google Workplace) to ensure they cover all needs. This could lead to fragmentation and inefficiency, but they applied some integrations to guarantee everything is connected and information flows even with distributed teams.

Another research question was “how are the teams organised in terms of roles?”. Many roles were listed, yet two functions were consistently present in all teams: the developers and a mediator or facilitator that will henceforth be referred to as the "Project Vision Holder". The project vision holder can take many forms such as Product Owner (as prescribed by Scrum), Product Manager, Project Manager or Delivery Manager. These individuals are responsible for communicating and preserving the project or product vision, for bridging the team and the client, and ensuring alignment with the business goals. In essence, these individuals serve as the connectors, the crucial link between strategy and execution. Beyond these key roles, the companies may have other specialised roles such as Designer, QA, DevOps, Data Analyst, among others, depending on the nature of the projects being developed and the teams’ specific needs.

Furthermore, it is a common practice for team members to assume multiple roles such as QA and Delivery Manager (Participant F) or Teach Lead and Developer (Company K). This approach adopted by small companies with small teams enables them to be cross-functional, allocating their available personnel in a manner consistent with Agile principles of collaboration and adaptability.

Our final question was “How do companies measure the success of project management?”. The companies combine qualitative and quantitative metrics and it looks like there is some correspondence with the Agile Triangle (Highsmith, 2009). The companies are primarily concerned with productivity and efficiency and use metrics related to constraints, particularly to the time constraints, which all companies have reported using. Metrics such as velocity, lead time, and timely deliveries help teams to ensure that projects stay on track. For example, Company B uses timely delivery and Earned Value Management (EVM), a more complex metric, to monitor project progress and guarantee that deadlines are met. However, they do not neglect the importance of delivering value to the customer and ensuring the product meets good quality standards. Companies K and B, for instance, mention user engagement and only admit a product meets its goals when the client is satisfied. Company C emphasises the need for alignment between products and company’s goals. Bug tracking and team feedback are employed to maintain high quality standards and continuous improvement, as explain by Participant F.

Nevertheless, the small companies face numerous difficulties that reflect the complexities of balancing value, quality, and constraints in their project management practices. This theme of challenges emerged naturally from participants' responses and helps to explain their options in Agile.

Team members often perform multiple roles, work across multiple projects, and while this cross-functionality complies with Agile principles and values, it can lead to workload imbalances.

Besides this, organisations face many other challenges at team level, from scaling their Agile practices as the team grows, to managing distributed teams and cross-team collaboration. This reflects the real world of Agile as an evolving system that constantly adapts but also contradicts the Agile Manifesto's assumptions about self-management. As (Ozkan, 2019, p. 5) argues, the idea of Agile teams self-managing is a "naive idealistic assumption". Team mergers and even client integration force reorganisation. There are always co-dependencies, concurrent forces that inevitably restrict individual team autonomy.

The limited resources these companies have make it difficult to balance quality and constraints, unexpected issues and planning and even to maintain rigorous processes. Company G, for example, often struggle to balance multiple priorities, leading to trade-offs between value, quality, and constraints. Company K's orientation is predominantly towards the day-to-day operations, which has resulted in the disregard of structured KPIs for measuring value and quality, making it difficult to fully align with the Agile Triangle’s balance.

Time estimation is another persistent challenge, particularly for companies that rely on time tracking and have hourly contracts. Companies like Company D and Company E use time-based estimations to manage client expectations and ensure transparency, but this approach can sometimes overshadow the focus on delivering value and quality.

At the client level, client dependencies often disrupt project and cross-project timelines. For example, Company D and Company E reported that late client feedback and changing requirements can introduce mid-sprint adjustments, resulting in delaying deliverables and straining resources. Excessive client involvement can also be counterproductive. If the client demands constant reporting and wants to micromanage the project, as in the case reported by Company E, in addition to taking up too much time in meetings, this does not give the team the autonomy it needs and limits its ability to find creative approaches, thus affecting the project results. This difficulty to work closely with the client can reveal some misalignment with Agile culture.

5. Conclusions, Limitations and Future Lines of Research

This study aimed to understand how small companies developing digital products in the Minho region implement Agile in their projects. It found Agile tailoring and context-driven diversity, but also some shared patterns.

When examining what methodologies are employed and why, it was found that Scrum was the most prevalent. But it is usually not the only methodology being followed, as it depends on the project nature (for example, Scrum for development and Kanban for R&D projects), and it is almost always not “done by the book”. While standard Scrum ceremonies (sprints, retrospectives) were commonplace, they were often tailored.

The same is valid for Kanban and, of course, for Scrumban. Some companies are even reluctant to adopt a rigid categorisation, pragmatically combining Agile practices without being bound to a single methodology. Tailoring methodologies and adapting practices is the norm.

Reasons for the choice of methodology can be related to its inherent value (e.g. flexibility, adaptability) and how it fits the company's culture and project needs, being “consistent with the

values and principles of Agile", "responding to a frequently changing environment", and "dynamic product requirements that prevail in small-to-medium scale enterprises (SME)" (Noteboom et al., 2021, p. 6779). But they can also be driven by customer demand.

Whether opting for Scrum's predictability, Kanban's fluidity, or inventive hybrids, teams prioritise practical outcomes over theoretical purity. The methods of the Minho companies corroborate the proposal of Ozkan et al. to "use Scrum and Kanban with the AND operator, not OR" as "they complement each other" and "produce better solutions together" (2022, p. 892).

Several Scrum, Kanban and cross-methodology practices are used for planning, reviewing and improve work and processes, but one is particularly noteworthy and may demand further study. Time tracking per task and estimations in time instead are common. Does this impact agility? Is this happening because the teams do not have the right mindset and thus "methods are often adapted in a wrong way and lose their purpose" (Klünder et al., 2022, p. 18)? Or is this indeed the best strategy for small companies to keep track of their progress and increase productivity, as have been claimed by participants from companies that have actually changed from story points to this time focus? These are interesting questions for further research.

The software tools supporting Agile were varied, with no standard combination across organisations. Jira and Clickup dominate for backlog and progress management, but many alternatives emerged such as Linear, GitHub or PHC. Teams mix tools for overall project management with other specialised tools for communication, design, and code, often integrating multiple platforms despite potential fragmentation. Also in tools, client requirements sometimes dictate the choice.

Team roles also reflect adaptability. While developers and product vision holders (e.g. Product Owners) were universal, other roles (e.g. QA, DevOps) emerged based on project needs. Small teams often combine roles (e.g. Participant G is CEO, Tech Lead and Developer) to maximise cross-functionality, although this sometimes strained workloads.

To measure success, companies mix quantitative metrics (velocity, lead time) with qualitative feedback (customer satisfaction, team retrospectives). While time-based KPIs dominated, reflecting contractual realities and resource constraints, value-based indicators, as OKR alignment, and quality measurements, such as bug tracking, are also in use, focusing all the three vertices of the Agile triangle. However, challenges such as estimation inaccuracies, customer scope changes and role overload revealed the fragility of this balance, particularly where Agile ideals collide with business constraints (e.g., hourly billing).

We must acknowledge that this study's methodology has some relevant limitations. The use of single 45 to 60 minutes interviews per company may have not have provided a comprehensive understanding of each company, as the participants may have omitted or been unable to fully articulate all relevant information. Therefore, although we have prepared case summaries to report on the particularities of each company's methods, practices, tools, roles and performance indicators, they do not represent a complete picture of each company's reality and practices and should not be interpreted as such. However, this approach also offers key strengths. The semi-structured interview

format allowed participants to share rich, contextualised perspectives that not only helped to answer our research questions, but may also provide interesting suggestions for further research. Additionally, by situating Agile methodologies within the broader context of the organisational conditions that enable or constrain its application, this study provides a grounded understanding of how agility is operationalized in practice. This approach not only aligns with the principles of Agile itself (adaptability, responsiveness, and a focus on people), but also ensures that the findings are relevant and meaningful to the specific realities of small companies in the Minho region.

Taken together, these findings paint a picture of Agile as a negotiated process - where methodologies, practices, tools, roles and performance indicators are dynamically configured around the imperatives of delivering value and quality, while surviving as a small business. Agile methodologies are not always used in their totally, are mixed, reinvented, adapted to specific needs and constraints, because the objective is to reach the best results (Campanelli & Parreiras, 2015) and combining different methodologies “can lead to greater improvements than concentrating on just one” (Petricioli & Fertalj, 2022, p. 1098).

The Minho companies' approaches may defy textbook Agile but exemplify its core principle of responding to change over following a plan (Beck et al., 2001) .

References

Alqudah, M., & Razali, R. (2018). An Empirical Study of Scrumban Formation based on the Selection of Scrum and Kanban Practices. *International Journal on Advanced Science Engineering Information Technology*, *8*(6), 2315–2322.

Al-Saqqa, S., Sawalha, S., & Abdelnabi, H. (2020). Agile software development: Methodologies and trends. *International Journal of Interactive Mobile Technologies*, *14*(11), 246–270. https://doi.org/10.3991/ijim.v14i11.13269

Ambler, S. W. (2013). *Going Beyond Scrum Disciplined Agile Delivery*. https://www.classes.cs.uchicago.edu/archive/2016/fall/51205-1/required.reading/BeyondScrum.pdf

Azad, N., & Hyrynsalmi, S. (2023). DevOps critical success factors — A systematic literature review. *Information and Software Technology*, *157*. https://doi.org/10.1016/j.infsof.2023.107150

Beck, K., & Andres, C. (2004). *Extreme Programming Explained: Embrace Change* (Second). Addison Wesley Professional.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Greening, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., & Thomas, D. (2001). *Manifesto for Agile Software Development*. https://agilemanifesto.org/

Bortolini, R. F., Cortimiglia, M. N., Danilevicz, A. de M. F., & Ghezzi, A. (2021). Lean Startup: A Comprehensive Historical Review. *Management Decision*, *59*(8), 1765–1783. https://doi.org/https://doi.org/10.1108/MD-07-2017-0663

Brooks, J., McCluskey, S., Turley, E., & King, N. (2015). The Utility of Template Analysis in Qualitative Psychology Research. *Qualitative Research in Psychology*, *12*(2), 202–222. https://doi.org/10.1080/14780887.2014.955224

Brown, M., & Tuxworth, G. (2020). Selection Factors Determining the Hybrid Approach: A Preliminary Study. *Americas Conference on Information Systems*. https://www.researchgate.net/publication/363207929

Campanelli, A. S., & Parreiras, F. S. (2015). Agile methods tailoring - A systematic literature review. *Journal of Systems and Software*, *110*, 85–100. https://doi.org/10.1016/j.jss.2015.08.035

Cao, L., & Ramesh, B. (2008). Agile Requirements Engineering Practices: An Empirical Study. *IEEE Software*, *25*(1), 60–67. https://doi.org/10.1109/MS.2008.1.

Denning, S. (2016, June 7). *What's Missing In The Agile Manifesto: Mindset*. https://www.forbes.com/sites/stevedenning/2016/06/07/the-key-missing-ingredient-in-the-agile-manifesto-mindset/?sh=1f91ca5767ff

Dingsøyr, T., Dybå, T., & Abrahamsson, P. (2008). A preliminary roadmap for empirical research on agile software development. *Proceedings - Agile 2008 Conference*, 83–94. https://doi.org/10.1109/Agile.2008.50

Durbin, M., & Niederman, F. (2021). Bringing templates to life: Overcoming obstacles to the organizational implementation of Agile methods. *International Journal of*

Information Systems and Project Management, *9*(3), 5–18. https://doi.org/10.12821/ijispm090301

Eilers, K., Peters, C., & Leimeister, J. M. (2022). Why the agile mindset matters. *Technological Forecasting and Social Change*, *179*. https://doi.org/10.1016/j.techfore.2022.121650

Eurostat. (2022). *EU small and medium-sized enterprises: an overview*. https://ec.europa.eu/eurostat/web/products-eurostat-news/-/edn-20220627-1

F. Tripp, J., & Armstrong, D. J. (2018). Agile Methodologies: Organizational Adoption Motives, Tailoring, and Performance. *Journal of Computer Information Systems*, *58*(2), 170–179. https://doi.org/10.1080/08874417.2016.1220240

Faustino, J., Adriano, D., Amaro, R., Pereira, R., & da Silva, M. M. (2022). DevOps benefits: A systematic literature review. *Software - Practice and Experience*, *52*(9), 1905–1926. https://doi.org/10.1002/spe.3096

Highsmith, J. (2009). *Agile Project Management: Creating Innovative Products* (Second Edition). Addison-Wesley Professional.

Hohl, P., Klünder, J., van Bennekum, A., Lockard, R., Gifford, J., Münch, J., Stupperich, M., & Schneider, K. (2018). Back to the future: origins and directions of the “Agile Manifesto” – views of the originators. *Journal of Software Engineering Research and Development*, *6*(1). https://doi.org/10.1186/s40411-018-0059-z

Jalali, S., & Wohlin, C. (2010). Agile practices in global software engineering - A systematic map. *Proceedings - 5th International Conference on Global Software Engineering, ICGSE 2010*, 45–54. https://doi.org/10.1109/ICGSE.2010.14

Kim, G., Humble, J., Debois, P., Willis, J., & Forsgren, N. (2021). *The DevOps Handbook: How to Create World-Class Agility, Reliability, & Security in Technology Organizations* (2nd ed.). IT Revolution Press.

King, N., Brooks, J., & Tabari, S. (2018). Template analysis in business and management research. In *Qualitative Methodologies in Organization Studies* (Vol. 2, pp. 179–206). Springer International Publishing. https://doi.org/10.1007/978-3-319-65442-3_8

King, Nigel., & Brooks, J. M. . (2017). *Template analysis for business and management students*. Sage.

Kišš, F., & Rossi, B. (2018). Agile to lean software development transformation: A systematic literature review. *Proceedings of the 2018 Federated Conference on Computer Science and Information Systems, FedCSIS 2018*, 969–973. https://doi.org/10.15439/2018F53

Klünder, J., Trommer, F., & Prenner, N. (2022). How agile coaches create an agile mindset in development teams: Insights from an interview study. *Journal of Software: Evolution and Process*, *34*(12). https://doi.org/10.1002/smr.2491

Knott, E., Rao, A. H., Summers, K., & Teeger, C. (2022). Interviews in the social sciences. *Nature Reviews Methods Primers*, *2*(1). https://doi.org/10.1038/s43586-022-00150-6

Ladas, C. (2009). *Scrumban - And Other Essays on Kanban Systems for Lean Software Development*. Modus Cooperandi Press.

Ladas, C. (2019). *What is Scrumban?* Agile Alliance.

Larman, C. (2003). *Agile and Iterative Development: A Manager's Guide* (First edition). Addison-Wesley Professional.

Leiner, D. F. (2022). *Developing a framework for the implementation and utilization of agile methods in the project management of small and medium sized enterprises* [University of Applied Sciences Landshut]. https://opus4.kobv.de/opus4-haw-landshut/frontdoor/index/index/docId/345

Miler, J., & Gaida, P. (2019). On the agile mindset of an effective team - An industrial opinion survey. *Proceedings of the 2019 Federated Conference on Computer Science and Information Systems, FedCSIS 2019*, 841–849. https://doi.org/10.15439/2019F198

Mordi, A., & Schoop, M. (2020). *Making it Tangible - Creating a Definition of Agile Mindset*. https://www.researchgate.net/publication/342010154

Nee, N. Y. (2012). Leading agile teams through collaboration. *2012 PMI Global Congress Proceedings — Vancouver, Canada*. https://www.pmi.org/learning/library/leading-agile-teams-through-collaboration-5983

Noteboom, C., Ofori, M., Sutrave, K., & El-Gayar, O. (2021). Agile Project Management: A Systematic Literature Review of Adoption Drivers and Critical Success Factors. *Proceedings of the Annual Hawaii International Conference on System Sciences*, *2020-January*, 6775–6784. https://doi.org/10.24251/HICSS.2021.813

Ozkan, N. (2019, November 1). Imperfections Underlying the Manifesto for Agile Software Development. *1st International Informatics and Software Engineering Conference: Innovative Technologies for Digital Transformation, IISEC 2019 - Proceedings*. https://doi.org/10.1109/UBMYK48245.2019.8965504

Ozkan, N., Bal, S., Erdogan, T. G., & Gok, M. S. (2022). Scrum, Kanban or a Mix of Both? A Systematic Literature Review. *Proceedings of the 17th Conference on Computer Science and Intelligence Systems, FedCSIS 2022*, 883–893. https://doi.org/10.15439/2022F143

Ozkan, N., Eilers, K., & Gok, M. S. (2023). Back to the Essential: A Literature-Based Review on Agile Mindset. *Proceedings of the 18th Conference on Computer Science and Intelligence Systems, FedCSIS 2023*, 201–211. https://doi.org/10.15439/2023F6360

Ozkan, N., Gok, M. S., & Kose, B. O. (2020). Towards a Better Understanding of Agile Mindset by Using Principles of Agile Methods. *Proceedings of the 2020 Federated Conference on Computer Science and Information Systems, FedCSIS 2020*, 721–730. https://doi.org/10.15439/2020F46

Petricioli, L., & Fertalj, K. (2022). Agile Software Development Methods and Hybridisation Possibilities Beyond Scrumban. *45th Jubilee International Convention on Information, Communication and Electronic Technology (MIPRO)*, 1093–1098. https://doi.org/10.23919/MIPRO55190.2022.9803402

Pollack, J., Helm, J., & Adler, D. (2018). What is the Iron Triangle, and how has it changed? *International Journal of Managing Projects in Business*, *11*(2), 527–547. https://doi.org/10.1108/IJMPB-09-2017-0107

Poppendieck, M., & Poppendieck, T. (2003). *Lean Software Development: An Agile Toolkit*. Addison Wesley.

Pordata. (2023). *Empresas em Portugal*. https://www.pordata.pt/subtema/portugal/empresas-374

Portman, H. (2020). A new bird's eye view on the agile forest. *PM World Journal*, *IX*(X).

Project Management Institute. (2021). *A Guide to the project management body of knowledge (PMBOK® guide)* (7th ed.).

Rasnacis, A., & Berzisa, S. (2017). Method for Adaptation and Implementation of Agile Project Management Methodology. *Procedia Computer Science*, *104*, 43–50. https://doi.org/10.1016/j.procs.2017.01.055

Sandstø, R., & Reme-Ness, C. (2021). Agile Practices and Impacts on Project Success. *Journal of Engineering, Project, and Production Management*, *11*(3), 255–262. https://doi.org/10.2478/jeppm-2021-0024

Schwaber, K., & Sutherland, J. (2020). *The Scrum Guide. The Definitive Guide to Scrum: The Rules of the Game*. https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf

Sousa, L., Trigo, A., & Varajão, J. (2019). DevOps - foundations and perspectives. In Association for Information Systems (Ed.), *19.ª Conferência da Associação Portuguesa de Sistemas de Informação (CAPSI'2019)*. https://aisel.aisnet.org/capsi2019

Vala, J., & Monteiro, M. B. (1999). *Psicologia Social* (Sexta edição). Fundação Calouste Gulbenkian.

Zayat, W., & Senvar, O. (2020). Framework Study for Agile Software Development Via Scrum and Kanban. In *International Journal of Innovation and Technology Management* (Vol. 17, Issue 4). World Scientific. https://doi.org/10.1142/S0219877020300025

Appendices

Appendix A Interview Script

Interview Script

Subject: Project management Agile methodologies and practices

Theme	Objective	Guiding questions
1. Participant profile	To know the professional background and experience of the participant, especially in Agile methodologies	How many years of project management experience do you have? And in Agile methodologies? How many years have you been a project manager in this company? What is your university degree? Have you taken any specific training in Agile or project management? Is any of it certified?
2. Company profile	To better understand the context of the company	How old is the company? In a few lines, how would you describe the company's core business? How many years of Agile experience does the company have?
3. Project management methodologies and approaches	To assess the company's overall project management philosophy and approach	What project management methodologies do you use? Are they the same for all projects?
4. Reasons for choosing those methodologies and practices	To understand the rationale behind the choice of methodologies and practices.	Why did the company choose these methodologies and practices?
5. Core practices	To identify the core practices associated with the methodology(ies)	What are the key practices associated with the chosen Agile methodologies? Can you describe a typical day for you or the project team?
6. Team roles	To understand how the team is organized	What are the defined roles and responsibilities within the project team?
7. Software and tools	To know the employed software and tools for project management	What software and tools do you use?
8. Performance indicators	To identify the key performance indicators to measure project management success	What key performance indicators (KPIs) do you use to understand if the project was well managed?

Appendix B Coded Participants Responses and Template Analysis Evolution

COMPANY A		
Themes and codes	**Reported**	**Notes and quotations**
1. Project management methodologies in use		
1.1. Scrum		
1.2. Kanban		
1.3. Scrum and Kanban	x	"Scrum is our main Agile methodology"; Kanban is used for simpler projects, with smaller teams, especially on R&D and kick-off phases. "Kanban works better for smaller teams or simpler projects where Scrum's overhead is unnecessary."; "For R&D or internal projects, we prefer milestone-based Kanban, which gives flexibility for complex, evolving tasks."
2. Reasons for the choice		
2.1. Internal drivers		
2.1.1. Disciplined approach	x	"Scrum ensures we stay connected to stakeholders and maintain a clear plan with frequent feedback."; "Scrum provides structure for projects needing clear deliverables."
2.1.2. Simplicity	x	"Kanban gives freedom to R&D teams, where deliverables aren't tied to a consumer-facing product."
2.1.3. Flexibility / adaptability	x	"Kanban gives freedom to R&D teams, where deliverables aren't tied to a consumer-facing product." "Kanban is more flexible for R&D projects where exploration and iteration are the focus"
3. Core practices		
3.1. Scrum-related practices		
3.1.1. Sprint planning and sprints	x	Sprint planning is highlighted as one of the most important ceremonies for team alignment. "Sprint planning and review are non-negotiable—they connect the team's work to stakeholders and provide clear updates."
3.1.2. Sprint review	x	Sprint reviews, as well as planning, are critical for product updates and stakeholder alignment. "Sprint planning and review are non-negotiable—they connect the team's work to stakeholders and provide clear updates."
3.1.3. Sprint retrospective		
3.2. Cross-methodology Agile practices		
3.2.1. Daily syncs		
3.2.2. Refinement meetings	x	"Backlog refinement ensures we know what's important and adjusts priorities as needed."
3.2.3. Pair programming	x	"Pair programming happens organically, especially because some of our developers have been working here for 4-5 years."
3.2.4. Milestone-based planning	x	Milestones are used in Kanban projects, often aligned with quarterly goals.
3.3. Agile methodologies tailoring		
3.3.1. Alternatives to retrospective per sprint	x	"We skip retrospectives for every sprint; instead, we evaluate quarterly to reduce the number of meetings."
3.3.1 No predefined periodicity for syncs	x	"Teams have autonomy to decide how often they hold dailies—some do them daily, others every two days."
4. Collaborative software and tools		

4.1. Project documentation and backlog management		
4.1.1. Zoho Sprints	x	“We use Zoho Sprints for backlog management, and ticketing and everything scrum-related”
4.1.2. Confluence	x	"Confluence is an Atlassian product more focused on documentation, especially technical documentation, and is widely used."
4.1.3. Google Workspace	x	"everything that is Google's environment, the Google Drive... many things are also done there"
4.2. Communication		
4.2.1. Google Chat	x	"we use Google Chat for internal company communication which is one of the most important tools we use because almost no one uses email"
4.2.2. Google Meet	x	"Google Meet for meetings"
4.3. Design collaboration		
4.3.1. Figma	x	"Figma is a design tool but helps the developers as well"
4.4. Code, testing and version control		
4.4.1. Bitbucket	x	"We use Bitbucket for code management"
5. Core Team Roles		
5.1. Roles		
5.1.1. Product Owner	x	"We usually have the Product Owner, the developers and the QA. The Product Owner is the often the Scrum Master"
5.1.2. Scrum Master	x	
5.1.3. Developers	x	
5.1.4. QA	x	
5.1.5. Designer	x	Designer is only part of the team when working in Kanban
5.2. Role overlaps	x	"Product Owners are often Scrum Masters, but as teams grow, team leads might take over that responsibility." "I'm CTO, I act as Product Owner as well"
6. Key performance indicators		
6.1. Efficiency and productivity		
6.1.1. Velocity	x	"The typical ones, i.e. team velocity indicators, burn charts to measure sprint progress, lead time to measure feature delivery time. And some kind of KPIs to consider, say, if the project or the sprint was successful. And obviously afterwards we always have the roadmap and the deadline here."
6.1.2. Lead time	x	
6.1.3. Timely delivery of projects	x	
6.2. Quality and continuous improvement		
6.2.1.Team feedback	x	"We hold one-on-ones every three months to evaluate progress, identify blockers, and gather feedback." "Quarterly retrospectives help us assess team performance and make meaningful changes."
7. Challenges		
7.1. Organisational and structural challenges		
7.1.1. Scaling Agile	x	"We need to find a way to scale Scrum and Kanban effectively as the team expands."
7.2. Operational and process challenges		
7.2.1. Time estimation		

7.2.2. Balancing flexibility and process rigor	x	"We realized that fortnightly retrospectives were too frequent and not adding much value, so we moved to quarterly reviews."

COMPANY B		
Themes and codes	**Reported**	**Notes and quotations**
1. Project management methodologies in use		
1.1. Scrum		
1.2. Kanban		
1.3. Scrum and Kanban	x	"Kanban is our default when Scrum's rigid structure isn't suitable. It's simpler, more flexible, and flows more naturally."
2. Reasons for the choice		
2.1. Internal drivers		
2.1.1. Disciplined approach	x	"Scrum is rigid and requires commitment"
2.1.2. Simplicity	x	"Kanban is our default when Scrum's rigid structure isn't suitable. It's simpler, more flexible, and flows more naturally."; Kanban, with its visual board and flexibility, is better suited for mixed or junior teams.". "Kanban is more natural and less resistant—it flows better."
2.1.3. Flexibility / adaptability	x	"Kanban is our default when Scrum's rigid structure isn't suitable. It's simpler, more flexible, and flows more naturally."
2.2. External influences		
2.2.1. Client requirement	x	"We use Scrum in projects where it makes sense and clients request it."
3. Core practices		
3.1. Scrum-related practices		
3.1.1. Sprint planning and sprints	x	"With Scrum, we follow the textbook: 15-day sprints, daily meetings, and a retrospective at the end."; "In Kanban, we hold a weekly meeting to refine tasks, adjust priorities, and allocate work."
3.1.2. Sprint review	x	
3.1.3. Sprint retrospective	x	"At the end of the day, the retrospective meeting"; "For Kanban, there's no need for a dedicated retrospective because we're constantly discussing progress in weekly meetings."
3.2. Cross-methodology Agile practices		
3.2.1. Daily syncs	x	
3.2.2. Refinement meetings	x	"In Kanban, we hold a weekly meeting to refine tasks, adjust priorities, and allocate work."
3.2.3. Pair programming		
3.2.4. Milestone-based planning		
3.2.5. Time tracking	x	"We ensure that hours are logged, and tasks are documented properly."; ""ClickUp allows us to track time and extract detailed reports to evaluate progress."
3.2.6. Knowledge sharing sessions	x	"There might be a KT session where a developer demonstrates a specific dashboard to the team, sharing their expertise."; "We see this as an opportunity to not only spread knowledge but also create alignment across projects."
3.2.7. Weekly planning	x	"A weekly session to refine and adjust priorities"
3.3. Agile methodologies tailoring		
3.3.1. Alternatives to retrospective per sprint		

3.3.1 No predefined periodicity for syncs	x	"We also adapt meeting frequency; not all projects need daily stand-ups—sometimes we have them every other day or weekly."
3.4. Kanban-related practices		
3.4.1. Task pulling	x	
4. Collaborative software and tools		
4.1. Project documentation and backlog management		
4.1.1. Zoho Sprints		
4.1.2. Confluence		
4.1.3. Google Workspace	x	"... everything is stored in Google Drive."
4.1.4. ClickUp	x	"We moved from Jira and Confluence to ClickUp because it consolidates everything we need into one tool and is more cost-effective."; "ClickUp manages to be in a single tool, both the documentation part and the task part"
4.2. Communication		
4.2.1. Google Chat		
4.2.2. Google Meet	x	"Google Meet is our primary tool for remote meetings..."
4.3. Design collaboration		
4.3.1. Figma		
4.4. Code, testing and version control		
4.4.1. Bitbucket		
5. Core Team Roles		"In Scrum projects, the structure is clear: Scrum Master, PO, and Developers. In Kanban, we add flexibility with Team Leaders overseeing progress."
5.1. Roles		
5.1.1. Product Owner	x	
5.1.2. Scrum Master	x	
5.1.3. Developers	x	
5.1.4. QA		
5.1.5. Designer		
5.1.6. Project Manager		
5.1.7. Architect	x	
5.1.8. Team Lead	x	in Kanban or in a more hybrid situation, there is the project manager, there may be, and of course there is the architect, there is a team leader, and then there are the Developers.
5.2. Role overlaps	x	The interviewee is Scrum Master as well.
6. Key performance indicators		
6.1. Efficiency and productivity		
6.1.1. Velocity	x	"Burn-down charts help us track whether we’re on schedule for completing the sprint’s tasks and meeting deadlines." "ClickUp supports tracking metrics related to earned value, helping us measure productivity and velocity."
6.1.2. Lead time		
6.1.3. Timely delivery of projects	x	"We measure project success by whether it was delivered on time, meeting the client's requirements and expectations."
6.1.4. Earned value management	x	"EVM helps us link the work delivered to the budget and time spent, providing a clearer picture of productivity." "We analyse earned value to ensure that the delivered output aligns with the resources used and the planned timeline." "ClickUp

		supports tracking metrics related to earned value, helping us measure productivity and velocity."
6.2. Quality and continuous improvement		
6.2.1 Team feedback	x	"Retrospectives highlight areas for improvement"
6.2.2. Bug tracking	x	ClickUp has indicators such as "number of adjustments, numbers of bugs"(..)
6.3. Value		
6.3.1. Client satisfaction	x	"Client satisfaction is a continuous metric, assessed through feedback during the project, not just at the end."
7. Challenges		
7.1. Organisational and structural challenges		
7.1.1. Scaling Agile		
7.1.2. Managing cross-team collaboration	x	"We always have elements from other companies on the client's team, and we need to align with their processes and practices."; "It's common for a developer to be partially allocated to two projects, needing to attend multiple dailies and coordinate tasks."
7.2. Operational and process challenges		
7.2.1. Time estimation		
7.2.2. Balancing flexibility and process rigor		
7.2.3. Balance quality and constraints	x	"It's tricky to manage priorities when a developer is split between two or more projects with overlapping deadlines."; "If meeting the deadline risks quality, we either extend time or allocate more resources, depending on the client's flexibility, bur it's challenging" "If the deadline is more important, we sometimes have to allocate additional resources to meet it, even if it stretches the team."

COMPANY C		
Themes and codes	**Reported**	**Notes and quotations**
1. Project management methodologies in use		
1.1. Scrum	x	
1.2. Kanban		
1.3. Scrum and Kanban		
2. Reasons for the choice		
2.1. Internal drivers		
2.1.1. Disciplined approach		
2.1.2. Simplicity	x	"Scrum is popular and easy to adapt because most people know it."
2.1.3. Flexibility / adaptability		
2.1.4. Predictability for the team	x	"Scrum helps us create a routine and predictability"
2.2. External influences		
2.2.1. Client requirement		
3. Core practices	x	Product department has their own dailies, separated from the development team
3.1. Scrum-related practices		

3.1.1. Sprint planning and sprints	x	"We work in two-week sprints with refinement at the end and planning at the beginning."
3.1.2. Sprint review	x	
3.1.3. Sprint retrospective	x	"Each team—product and engineering—has its own retrospective to reflect on what went well or not."
3.2. Cross-methodology Agile practices		
3.2.1. Daily syncs	x	
3.2.2. Refinement meetings	x	
3.2.3. Pair programming		
3.2.4. Milestone-based planning		
3.2.5. Time tracking		
3.2.6. Knowledge sharing sessions		
3.2.7. Weekly planning		
3.3. Agile methodologies tailoring		
3.3.1. Alternatives to retrospective per sprint		
3.3.2. No predefined periodicity for syncs		
3.3.3. Allow stories to be added during the sprint	x	"We use Scrum to give the team stability, but we're open to interrupting or adjusting if something urgent arises."; "If something critical comes up, we prioritize it immediately, even if it disrupts the sprint."
3.4. Kanban-related practices		
3.4.1. Task pulling		
4. Collaborative software and tools		
4.1. Project documentation and backlog management		
4.1.1. Zoho Sprints		
4.1.2. Confluence	x	"...Confluence for documentation to keep everything organized.";
4.1.3. Google Workspace		
4.1.4. ClickUp		
4.1.5. Jira	x	"We use Jira for project tracking and Confluence for documentation to keep everything organized.";
4.1.6 Trello	x	"Trello helps us quickly visualize what we're working on within the product team."
4.2. Communication		
4.2.1. Google Chat		
4.2.2. Google Meet		
4.2.3. Slack	x	"Slack keeps us connected, especially in a remote-first setup."
4.3. Design collaboration		
4.3.1. Figma		
4.4. Code, testing and version control		
4.4.1. Bitbucket	x	"Bitbucket for engineering"
5. Core Team Roles		
5.1. Roles		
5.1.1. Product Owner		

5.1.2. Scrum Master		
5.1.3. Developers	x	“We have the Product Team Lead, 2 Product Managers and a Product Designer or UI and UX Designer. So on the engineering side, we have the Director of Engineering (Team Lead) and then we have 2 people from iOS, 2 people from Android and 4 people who do web.”
5.1.4. QA		The Product Manager leads QA efforts, but developers are also accountable.
5.1.5. Designer	x	
5.1.6. Project Manager		
5.1.7. Architect	x	
5.1.8. Team Lead		
5.1.9. Product Manager	x	"As a Product Manager, what I do is, together with the team, the entire team, (...) I try to understand how the product, how we, what we can improve, what user problems we can solve to achieve the company's objectives."
5.2. Role overlaps	x	"We don’t have a dedicated QA team; QA is handled by the product team alongside engineering."
6. Key performance indicators		
6.1. Efficiency and productivity	x	"Burn-down charts give us a clear picture of how the sprint is progressing and help us spot if we’re falling behind."
6.1.1. Velocity		"Velocity helps us estimate capacity, but it’s not a metric we rely on to measure success. The outcome matters more."; "The engineering team owns the velocity; we only step in when we notice significant deviations or issues."
6.1.2. Lead time		
6.1.3. Timely delivery of projects	x	"Velocity helps us estimate capacity, but it’s not a metric we rely on to measure success. The outcome matters more."; "The engineering team owns the velocity; we only step in when we notice significant deviations or issues."
6.1.4. Earned value management		
6.2. Quality and continuous improvement		
6.2.1.Team feedback	x	
6.2.2. Bug tracking		
6.3. Value		
6.3.1. Client satisfaction	x	"For the product side, that's all that matters to us, if people are using it, if it solved the problem"; "Velocity helps us estimate capacity, but it’s not a metric we rely on to measure success. The outcome matters more."; "The engineering team owns the velocity; we only step in when we notice significant deviations or issues."
6.3.2. Contribution to OKRs (Objectives and Key Results)	x	"What we defined for this quarter's OKRS, was it achieved or not (...)
7. Challenges		
7.1. Organisational and structural challenges		
7.1.1. Scaling Agile		
7.1.2. Managing cross-team collaboration		
7.2. Operational and process challenges		
7.2.1. Time estimation		
7.2.2. Balancing flexibility and process rigor		

7.2.3. Balance quality and constraints		
7.2.4. Balance unexpected issues and planning	x	"Urgent tasks sometimes enter the sprint unexpectedly, but we manage based on their impact."
7.2.5. Meeting lengths	x	"Our refinement meetings are long because we aim to convince the engineering team of the strategy, it's a pain point."

COMPANY D		
Themes and codes	**Reported**	**Notes and quotations**
1. Project management methodologies in use		
1.1. Scrum		
1.2. Kanban		
1.3. Scrum and Kanban	x	"We use Kanban for all projects (...) it's our default methodology because of its simplicity and flexibility."; "When we work with outsourcing clients, we follow the full Scrum process, including sprint planning and retrospectives."
2. Reasons for the choice		
2.1. Internal drivers		
2.1.1. Disciplined approach		
2.1.2. Simplicity	x	"Sprints don't work for us because our projects are often short-term, so dividing them into two or three sprints didn't make sense."
2.1.3. Flexibility / adaptability	x	"Kanban is ideal for our team size and type of projects; it allows us to adapt as we go."
2.1.4. Predictability for the team		
2.2. External influences		
2.2.1. Client requirement		
3. Core practices		
3.1. Scrum-related practices		
3.1.1. Sprint planning and sprints		
3.1.2. Sprint review		
3.1.3. Sprint retrospective		
3.2. Cross-methodology Agile practices		
3.2.1. Daily syncs	x	"Every team member talks about their tasks, regardless of the project they're on, which fosters collaboration and problem-solving."
3.2.2. Refinement meetings		
3.2.3. Pair programming		
3.2.4. Milestone-based planning	x	"It's more of a general schedule where we try to roughly have a feature or so delivered this week."
3.2.5. Time tracking	x	"We also have time tracking. Each person at the end of their day, or when they perform tasks, as they see fit, enters how long they have been carrying out that task."
3.2.6. Knowledge sharing sessions		
3.2.7. Weekly planning		
3.3. Agile methodologies tailoring		

3.3.1. Alternatives to retrospective per sprint		
3.3.2. No predefined periodicity for syncs		
3.3.3. Allow stories to be added during the sprint		
3.4. Kanban-related practices		
3.4.1. Task pulling	x	
3.4.2. WIP limits	x	
4. Collaborative software and tools		
4.1. Project documentation and backlog management		
4.1.1. Zoho Sprints		
4.1.2. Confluence		
4.1.3. Google Workspace	x	"we have folders in Drive, that is, each project also has its own foWder"
4.1.4. Clickup	x	"We moved to ClickUp for task management, but we're considering alternatives due to issues with exporting time tracking data."
4.1.5. Jira		
4.1.6 Trello		
4.1.7. Notion	x	"More for content"
4.2. Communication		
4.2.1. Google Chat		
4.2.2. Google Meet		
4.2.3. Slack	x	"Slack channels are integrated with ClickUp, though notifications can get overwhelming."
4.2.4. Discord	x	"We also use Discord for dailies. And we usually spend all day on Discord, in one room, we are always connected to each other."
4.3. Design collaboration		
4.3.1. Figma	x	"Figma helps us create layouts and gather client feedback efficiently."
4.4. Code, testing and version control		
4.4.1. Bitbucket		
5. Core Team Roles		
5.1. Roles		
5.1.1. Product Owner		
5.1.2. Scrum Master		
5.1.3. Developers	x	
5.1.4. QA		
5.1.5. Designer	x	
5.1.6. Project Manager	x	
5.1.7. Architect		
5.1.8. Team Lead	x	
5.1.9. Product Manager		
5.1.10. Business Manager	x	"Business managers are the ones who have direct contact with the client".

5.2. Role overlaps	x	QA: "Final testing is shared between the Project Manager, Designer, and Business Managers to ensure thoroughness."
6. Key performance indicators		
6.1. Efficiency and productivity		
6.1.1. Velocity		
6.1.2. Lead time		
6.1.3. Timely delivery of projects	x	"Team members log time spent on tasks to improve estimation accuracy for future projects."; "We assess whether the project was delivered within the agreed-upon timeline, though delays are common."
6.1.4. Earned value management		
6.2. Quality and continuous improvement		
6.2.1.Team feedback	x	"We try to find out if (...) the team was satisfied with that project or how the project was going, whether the team liked it or not, whether we should continue in that direction or not (...) What problems did we have in this project, which we were able to resolve in advance in future ones?"
6.2.2. Bug tracking		
6.3. Value		
6.3.1. Client satisfaction	x	"We also always want the client to be satisfied with our project, yes, there is also a metric, and what can we improve in the next projects in this sense, that is, try to achieve what the client wants more quickly,"
6.3.2. Contribution to OKRs (Objectives and Key Results)		
7. Challenges		
7.1. Organisational and structural challenges		
7.1.1. Scaling Agile		
7.1.2. Managing cross-team collaboration		
7.2. Operational and process challenges		
7.2.1. Time estimation	x	"(...) We were able to try to find out if it was delivered within that timeframe, which usually almost never happens."
7.2.2. Balancing flexibility and process rigor		
7.2.3. Balance quality and constraints		
7.2.4. Balance unexpected issues and planning		
7.2.5. Meeting lengths		
7.2.6. Workload balance	x	"Developers often work across projects, so we try to balance workloads to avoid burnout."
7.3. Client interaction challenges		
7.3.1. Delays in client deliverables	x	"Clients don’t always deliver required materials on time, which delays development and integration."
7.3.2. Changing requirements and priorities	x	"Sometimes we get to the end and the client says “I didn't want it like that, I wanted it another way”, when it was presented one way at the beginning and accepted.."

COMPANY E		
Themes and codes	**Reported**	**Notes and quotations**
1. Project management methodologies in use		
1.1. Scrum		
1.2. Kanban		
1.3. Scrum and Kanban		
1.4. Scrum and Scrumban	x	"When it's a product or software project that's going to take time, it has to be Scrum. We define the sprints very well and stick to them without making changes during the sprint."; "In smaller or more iterative projects, we typically use Kanban... It's easier to manage and allows flexibility."; "Even when we do Kanban, it's a mix of Kanban and Scrum"
2. Reasons for the choice		
2.1. Internal drivers		
2.1.1. Disciplined approach	x	"When things need to be managed very well and because there are very concrete deliverables that cannot fail, that is typically Scrum. "
2.1.2. Simplicity	x	
2.1.3. Flexibility / adaptability		
2.1.4. Predictability for the team		
2.2. External influences		
2.2.1. Client requirement	x	"And many times it is also because of the client, the client says "we want Scrum and we even want to participate in the dailies", so it is often imposed by the client, other times it is because we are there."
3. Core practices		
3.1. Scrum-related practices		
3.1.1. Sprint planning and sprints	x	"We try to have at least 5 or 6 sprints planned...so that we don't change anything for the entire time."
3.1.2. Sprint review	x	
3.1.3. Sprint retrospective	x	
3.2. Cross-methodology Agile practices		
3.2.1. Daily syncs	x	"We try to keep daily meetings to 10–15 minutes, but sometimes they go up to 40."
3.2.2. Refinement meetings		
3.2.3. Pair programming		
3.2.4. Milestone-based planning	x	
3.2.5. Time tracking	x	"Each employee records the time they work on the project with date and time and description."
3.2.6. Knowledge sharing sessions		
3.2.7. Weekly planning		
3.3. Agile methodologies tailoring		"We don't follow Scrum to the letter. There are some projects where we take more, others where we take less."
3.3.1. Alternatives to retrospective per sprint		
3.3.2. No predefined periodicity for syncs		
3.3.3. Allow stories to be added during the sprint		

3.3.4. Estimation in time, not in story points	x	"Story points, we don't use them. (...) We abandoned it, because you realized and a sprint planning meeting, which should have lasted half an hour or an hour, lasted all afternoon, with endless discussions."
3.4. Kanban-related practices		
3.4.1. Task pulling	x	
3.4.2. WIP limits		
3.5. Primarily Scrumban practices		
3.5.1. Kanban with Sprints	x	"Even when we do Kanban, it's a mix of Kanban and Scrum... we try to make Sprints."
4. Collaborative software and tools		
4.1. Project tracking, documentation and backlog management		
4.1.1. Zoho Sprints		
4.1.2. Confluence		
4.1.3. Google Workspace	x	“We allocate resources weekly using Google Sheets—it’s flexible and practical.”; "Google Meet for video calls"
4.1.4. Clickup		
4.1.5. Jira	x	"We still have a few projects where we use Jira."
4.1.6 Trello		
4.1.7. Notion		
4.1.8. Odoo	x	"This ODOO... each employee records the time they work on the project with date and time and description"
4.1.9. GitHub	x	"GitHub, which is the platform we use to manage code, and we use, for example, the Scrum and Kanban tools that GitHub has, those boards"; “GitHub allows us to tag issues, set priorities (P1 to P5), and keep everything tracked.”; "We can share it with the customer at any time... We take the customer's GitHub account, add it..."
4.2. Communication		
4.2.1. Google Chat		
4.2.2. Google Meet	x	"Google Meet for meetings"
4.2.3. Slack	x	"We use Slack to communicate"
4.2.4. Discord	x	"Discord with a few clients"
4.3. Design collaboration		
4.3.1. Figma	x	
4.4. Code, testing and version control		
4.4.1. Bitbucket		
4.4.2. Github	x	"GitHub, which is the platform we use to manage code..."
5. Core Team Roles		
5.1. Roles		
5.1.1. Product Owner	x	
5.1.2. Scrum Master		
5.1.3. Developers	x	
5.1.4. QA	x	
5.1.5. Designer	x	
5.1.6. Project Manager		
5.1.7. Architect		

5.1.8. Team Lead	x	
5.1.9. Product Manager		
5.1.10. Business Manager		
5.1.11. DevOps	x	
5.2. Role overlaps	x	"The company is small and, therefore, in some contexts, I have to act as Product Owner. We do a little bit of everything. I just don't do development, because that's no longer possible, it's no longer compatible. And I also do a bit of Team Lead."
6. Key performance indicators		
6.1. Efficiency and productivity		
6.1.1. Velocity		
6.1.2. Lead time	x	Time spent on tasks is tracked using tools like Odoo to monitor budget alignment.
6.1.3. Timely delivery of projects	x	"We look at everyone's timesheet, the project's timesheet, and understand whether the deviation is significant or not."
6.1.4. Earned value management		
6.2. Quality and continuous improvement		
6.2.1.Team feedback	x	Informal retrospectives focus on overall success or failure.
6.2.2. Bug tracking		
6.3. Value		
6.3.1. Client satisfaction		
6.3.2. Contribution to OKRs (Objectives and Key Results)		
7. Challenges		
7.1. Organizational and structural challenges		
7.1.1. Scaling Agile		
7.1.2. Managing cross-team collaboration		
7.1.3. Lack of structured KPIs	x	“We should have more specific KPIs. QA happens, but it’s not quantified in reports—it’s more ad hoc.”
7.2. Operational and process challenges		
7.2.1. Time estimation	x	"If there’s a significant deviation in hours, we act quickly, either renegotiating with the client or adjusting our internal efforts."
7.2.2. Balancing flexibility and process rigor	x	“We don’t follow Scrum or Kanban rigidly; it’s a blend that works for us, but explaining it to clients can be tricky.” “Our retrospectives are informal, and while it works now, a more structured approach might be better for larger projects.”
7.2.3. Balance quality and constraints		
7.2.4. Balance unexpected issues and planning		
7.2.5. Meeting lengths		
7.2.6. Workload balance		
7.3. Client interaction challenges	x	Many difficulties reported in dealing with clients, such as pressure to abandon or deprioritize features during development, only to reverse their decisions later, disrupting planned workflows, or demands for frequent feedback sessions that often lead to inefficiencies.
7.3.1. Delays in client deliverables		

7.3.2. Changing requirements and priorities	x	"In the end we will understand, and almost always at the end, that what during the Sprint he [the client] wanted to abandon, almost always is what he sees and he says 'ok, that's it!'"
7.3.3. Overwhelming communication processes	x	"A client who wants, for example, daily, practically daily, to do retrospectives, to see what is happening, wants to see results. It is very difficult to manage such a client."

COMPANY F		
Themes and codes	**Reported**	**Notes and quotations**
1. Project management methodologies in use		
1.1. Scrum		
1.2. Kanban		
1.3. Scrum and Kanban		
1.4. Scrum and Scrumban		
1.5. Unidentified hybrid approaches	x	"We say that our approach is hybrid... always adapted and inspired by various methodologies depending on the situation." "Our approach is always adapted to the context, inspired by various methodologies but never rigidly tied to one."
2. Reasons for the choice		
2.1. Internal drivers		
2.1.1. Disciplined approach		
2.1.2. Simplicity		
2.1.3. Flexibility / adaptability	x	"Hybrid methodologies give us the flexibility to maintain structure where it's needed while adapting to unexpected changes without losing momentum."
2.1.4. Predictability for the team		
2.2. External influences		
2.2.1. Client requirement		
3. Core practices		
3.1. Scrum-related practices		
3.1.1. Sprint planning and sprints		
3.1.2. Sprint review		
3.1.3. Sprint retrospective		
3.2. Cross-methodology Agile practices		
3.2.1. Daily syncs	x	"Daily syncs create a safe environment for team members to share difficulties and seek support."
3.2.2. Refinement meetings	x	"Backlog refinement is essential for keeping the team motivated. An unorganized backlog of 50 tasks can feel overwhelming, so we clean up what no longer makes sense and reprioritize what does." "We use this time to re-evaluate tasks—what's still relevant, what's outdated, and what should take priority." "Refactoring tasks and system health improvements often fall into the backlog, and refinement ensures these don't get permanently pushed aside."
3.2.3. Pair programming		
3.2.4. Milestone-based planning	x	"Weekly planning involves the entire product team, setting goals and ensuring alignment."
3.2.5. Time tracking		

3.2.6. Knowledge sharing sessions	x	
3.2.7. Weekly planning	x	"We start the week with a weekly planning, which is a meeting in which we discuss all ongoing and future projects to clarify any doubts, if anyone needs help, we define priorities, we define objectives, that is, it is a time to discuss the week's plan, the tasks that are allocated to you and we have this moment that is effectively with the entire product team, that is, with the data analyst, with the PMs, with the designer and with the engineers and with me."
3.2.8. Post-mortem	x	"Post-mortems are reflections on incidents that had significant user impact, such as when we couldn't process payments. It's a chance to learn and document solutions for the future." "We don't just resolve the issue; we share the knowledge with the entire team so that if it happens again, anyone can address it—even if the original person isn't available." "The key to a good post-mortem is ensuring that solutions are documented and shared, not just fixing the immediate problem." "Post-mortems are about asking: What went wrong? How can we prevent it? And how can we ensure the knowledge is spread across the team?"
3.2.9. Decision logs	x	"We always have a document related to the project, with all the decisions that were made and what the final implementation of the solution was."
3.3. Agile methodologies tailoring		
3.3.1. Alternatives to retrospective per sprint	x	"We always do a retrospective after a big project, just to talk about the project, and then we have team retrospectives, usually every 3 months."
3.3.2. No predefined periodicity for syncs		
3.3.3. Allow stories to be added during the sprint		
3.3.4. Estimation in time, not in story points	x	"Each point is a day"
3.4. Kanban-related practices		
3.4.1. Task pulling		
3.4.2. WIP limits		
4. Collaborative software and tools		
4.1. Project tracking, documentation and backlog management		
4.1.1. Zoho Sprints		
4.1.2. Confluence		
4.1.3. Google Workspace	x	"Docs, Sheets, Drive for documentation"
4.1.4. ClickUp		
4.1.5. Jira		
4.1.6 Trello		
4.1.7. Notion	x	"We work a lot with Notion, which is a documentation tool mainly"
4.1.8. Odoo		
4.1.9. GitHub		
4.1.10. Linear	x	"Linear organizes tasks by state (e.g., to do, in progress, QA) with clear ownership and expected durations."
4.2. Communication		
4.2.1. Google Chat		
4.2.2. Google Meet	x	"Google Meet for meetings"

4.2.3. Slack	x	"Slack for internal communication with channels by department and project"
4.2.4. Discord		
4.3. Design collaboration		
4.3.1. Figma	x	"Figma for design and prototyping"
4.4. Code, testing and version control		
4.4.1. Bitbucket		
4.4.2. Github	x	"Github for code development and maintenance."
4.4.3. Insomnia	x	"Insomnia to test APIs"
5. Core Team Roles		
5.1. Roles		
5.1.1. Product Owner		
5.1.2. Scrum Master		
5.1.3. Developers	x	
5.1.4. QA	x	
5.1.5. Designer	x	
5.1.6. Project Manager		
5.1.7. Architect		
5.1.8. Team Lead		
5.1.9. Product Manager	x	
5.1.10. Business Manager		
5.1.11. DevOps	x	"DevOps provides support at the infrastructure and CI level"
5.1.12. Release Manager	x	"The Delivery Manager is the facilitator in carrying out tasks/projects and Release Manager coordinates and manages the product launch process"
5.1.13. Delivery Manager	x	
5.1.14. Data Analyst	x	"The Data Analyst is the one who analyses data to provide insights that inform business and development decisions."
5..15. Tech Lead	x	"We have a Tech Lead who leads technical requirements and the implementation of projects/tasks."
5.2. Role overlaps	x	"Despite my Delivery Manager title, I still oversee QA and release management responsibilities."
6. Key performance indicators		
6.1. Efficiency and productivity		
6.1.1. Velocity		
6.1.2. Lead time	x	
6.1.3. Timely delivery of projects		
6.1.4. Earned value management		
6.1.5. Delivery by assignee	x	"Total points delivered by assignee. Each point equals one day; monthly reviews highlight deviations from expectations."
6.1.6. Delivery by task type	x	"This allows us to check, on a monthly basis, what the team is most focused on. If it's projects, if it's improvements, bugs... the distribution of work."
6.1.7. Delivery by priority	x	Regular planning is affected by urgent unexpected tasks. "Urgent tasks are reviewed to determine if they were proactive or reactive and whether they could've been avoided."

6.2. Quality and continuous improvement		
6.2.1.Team feedback	x	Retrospectives and post-mortem meetings
6.2.2. Bug tracking	x	
6.3. Value		
6.3.1. Client satisfaction		
6.3.2. Contribution to OKRs (Objectives and Key Results)		
7. Challenges		
7.1. Organisational and structural challenges		
7.1.1. Scaling Agile	x	"Our methodologies evolve with the maturity of the team; what worked for 10 people wouldn't work now for 30." "It's not about applying a textbook approach but finding what works for us at our size and with our constraints."
7.1.2. Managing cross-team collaboration		
7.1.3. Lack of structured KPIs		
7.1.4. Distributed teams	x	"Feedback must be organized efficiently to keep remote engineers productive."
7.2. Operational and process challenges		
7.2.1. Time estimation		
7.2.2. Balancing flexibility and process rigor		
7.2.3. Balance quality and constraints		
7.2.4. Balance unexpected issues and planning		
7.2.5. Meeting lengths		
7.2.6. Workload balance	x	"We monitor individual contributions to detect signs of burnout or inefficiency."
7.3. Client interaction challenges		
7.3.1. Delays in client deliverables		
7.3.2. Changing requirements and priorities		
7.3.3. Overwhelming communication processes		

COMPANY G		
Themes and codes	**Reported**	**Notes and quotations**
1. Project management methodologies in use		
1.1. Scrum		
1.2. Kanban		
1.3. Scrum and Kanban		
1.4. Scrum and Scrumban		

1.5. Unidentified hybrid approaches	x	"We don't follow a strict methodology but rather take ideas from multiple ones and adapt them to what works for us."
2. Reasons for the choice		
2.1. Internal drivers		
2.1.1. Disciplined approach		
2.1.2. Simplicity		
2.1.3. Flexibility / adaptability	x	"We don't have projects that last longer than 3-4 months, so we need flexibility to switch between them."
2.1.4. Predictability for the team		
2.2. External influences		
2.2.1. Client requirement		
3. Core practices		
3.1. Scrum-related practices		
3.1.1. Sprints and sprint planning		
3.1.2. Sprint review	x	
3.1.3. Sprint retrospective		
3.2. Cross-methodology Agile practices		
3.2.1. Daily syncs	x	
3.2.2. Refinement meetings		
3.2.3. Pair programming		
3.2.4. Milestone-based planning	x	"Deadlines are project-based; tasks are allocated weekly with an estimated time to completion."; "For clients with specific deadlines, we plan milestones and review progress at each stage."
3.2.5. Time tracking	x	
3.2.6. Knowledge sharing sessions		
3.2.7. Weekly planning	x	"We plan weekly and adapt day-to-day if the client fails to deliver content we switch to another project to avoid downtime."; Fridays are reserved for planning the next week, reviewing pending tasks, and adapting to client delays.
3.2.8. Post-mortem		
3.2.9. Decision logs		
3.3. Agile methodologies tailoring		
3.3.1. Alternatives to retrospective per sprint	x	"At the end of the year, we review all projects to see what succeeded and what could improve."
3.3.2. No predefined periodicity for syncs		
3.3.3. Allow stories to be added during the sprint		
3.3.4. Estimation in time, not in story points	x	"We have the estimated time for the task"
3.4. Kanban-related practices		
3.4.1. Task pulling		
3.4.2. WIP limits		
4. Collaborative software and tools		
4.1. Project tracking, documentation and backlog management		
4.1.1. Zoho Sprints		

4.1.2. Confluence		
4.1.3. Google Workspace	x	Google Calendar: Ensures team members have access to their schedules and task deadlines.
4.1.4. Clickup		
4.1.5. Jira		
4.1.6 Trello	x	"Trello gives us a Kanban-style view of the overall project.."
4.1.7. Notion		
4.1.8. Odoo		
4.1.9. GitHub		
4.1.10. Linear		
4.1.11. PHC	x	"PHC tracks task hours, while Trello shows the project's overall state across departments." "Trello gives us a Kanban-style view of the overall project, while PHC is used to track task hours and progress." "We create a macro-level Kanban board for projects but manage tasks individually in PHC with specific deadlines."
4.2. Communication		
4.2.1. Google Chat		
4.2.2. Google Meet		
4.2.3. Slack		
4.2.4. Discord		
4.3. Design collaboration		
4.3.1. Figma		
4.4. Code, testing and version control		
4.4.1. Bitbucket		
4.4.2. Github		
4.4.3. Insomnia		
5. Core Team Roles		
5.1. Roles		
5.1.1. Product Owner		
5.1.2. Scrum Master		
5.1.3. Developers	x	
5.1.4. QA		
5.1.5. Designer	x	
5.1.6. Project Manager	x	
5.1.7. Architect		
5.1.8. Team Lead		
5.1.9. Product Manager		
5.1.10. Business Manager		
5.1.11. DevOps		
5.1.12. Release Manager		
5.1.13. Delivery Manager		
5.1.14. Data Analyst		
5..15. Tech Lead	x	"I'm the Tech Lead"
5.2. Role overlaps	x	For example, the CEO is the tech lead, acts as project manager and can even code.

6. Key performance indicators		
6.1. Efficiency and productivity		
6.1.1. Velocity		
6.1.2. Lead time		
6.1.3. Timely delivery of projects	x	"We review estimated versus actual time at the semester or project's end to identify deviations."
6.1.4. Earned value management		
6.1.5. Delivery by assignee		
6.1.6. Delivery by task type		
6.1.7. Delivery by priority		
6.2. Quality and continuous improvement		
6.2.1.Team feedback	x	"At the end of the year, we review all projects to see what succeeded and what could improve."
6.2.2. Bug tracking		
6.3. Value		
6.3.1. Client satisfaction		
6.3.2. Contribution to OKRs (Objectives and Key Results)		
7. Challenges		
7.1. Organisational and structural challenges		
7.1.1. Scaling Agile		
7.1.2. Managing cross-team collaboration		
7.1.3. Lack of structured KPIs		
7.1.4. Distributed teams		
7.2. Operational and process challenges		
7.2.1. Time estimation	x	"Support tasks can be unpredictable. Some are quick and simple, while others take significant time to resolve, disrupting the planned schedule."
7.2.2. Balancing flexibility and process rigor		
7.2.3. Balance quality and constraints		
7.2.4. Balance unexpected issues and planning	x	"Support tasks can vary from two-minute fixes to complex problems that disrupt the schedule."
7.2.5. Meeting lengths		
7.2.6. Workload balance		
7.3. Client interaction challenges		
7.3.1. Delays in client deliverables	x	"If the client doesn't deliver by Friday, their work waits another week."
7.3.2. Changing requirements and priorities		
7.3.3. Overwhelming communication processes		

COMPANY H		
Themes and codes	**Reported**	**Notes and quotations**
1. Project management methodologies in use		
1.1. Scrum	x	"We use Scrum but adapted to our realities, strengths, and weaknesses—it's not the book version."
1.2. Kanban		
1.3. Scrum and Kanban		
1.4. Scrum and Scrumban		
1.5. Unidentified hybrid approaches		
2. Reasons for the choice		
2.1. Internal drivers		
2.1.1. Disciplined approach		
2.1.2. Simplicity		
2.1.3. Flexibility / adaptability	x	"Scrum fits us because it allows us to deliver value consistently and adapt to changes."
2.1.4. Predictability for the team		
2.2. External influences		
2.2.1. Client requirement		
3. Core practices		
3.1. Scrum-related practices		
3.1.1. Sprints and sprint planning	x	
3.1.2. Sprint review	x	"We combine planning, review, and retrospective into one session to save time, especially for team members in hybrid or remote work."
3.1.3. Sprint retrospective	x	
3.2. Cross-methodology Agile practices		
3.2.1. Daily syncs		
3.2.2. Refinement meetings		
3.2.3. Pair programming		
3.2.4. Milestone-based planning		
3.2.5. Time tracking		
3.2.6. Knowledge sharing sessions		
3.2.7. Weekly planning		
3.2.8. Post-mortem		
3.2.9. Decision logs		
3.3. Agile methodologies tailoring		
3.3.1. Alternatives to retrospective per sprint		
3.3.2. No predefined periodicity for syncs		
3.3.3. Allow stories to be added during the sprint	x	"We aim to avoid changes during a sprint, but commercial needs or legal obligations can force adjustments."
3.3.4. Estimation in time, not in story points		

3.3.5. Sprint review, retrospective and planning in one single session	x	"We combine planning, review, and retrospective into one session to save time, especially for team members in hybrid or remote work."
3.4. Kanban-related practices		
3.4.1. Task pulling		
3.4.2. WIP limits		
4. Collaborative software and tools		
4.1. Project tracking, documentation and backlog management		
4.1.1. Zoho Sprints		
4.1.2. Confluence		
4.1.3. Google Workspace		
4.1.4. ClickUp	x	"We use ClickUp and we also have some part that is internally developed software."
4.1.5. Jira		
4.1.6 Trello		
4.1.7. Notion		
4.1.8. Odoo		
4.1.9. GitHub		
4.1.10. Linear		
4.1.11. PHC		
4.1.12. Power BI	x	"Since ClickUp has APIs, we will fetch the data and work in Power BI to make our own charts."
4.1.13. Software developed in-house	x	"We use ClickUp and we also have some part that is internally developed software."
4.2. Communication		
4.2.1. Google Chat		
4.2.2. Google Meet		
4.2.3. Slack		
4.2.4. Discord		
4.2.5. Microsoft Teams	x	
4.3. Design collaboration		
4.3.1. Figma		
4.4. Code, testing and version control		
4.4.1. Bitbucket		
4.4.2. Github		
4.4.3. Insomnia		
5. Core Team Roles		
5.1. Roles		
5.1.1. Product Owner	x	
5.1.2. Scrum Master		
5.1.3. Developers	x	
5.1.4. QA	x	
5.1.5. Designer		
5.1.6. Project Manager		

5.1.7. Architect		
5.1.8. Team Lead		
5.1.9. Product Manager		
5.1.10. Business Manager		
5.1.11. DevOps		
5.1.12. Release Manager		
5.1.13. Delivery Manager		
5.1.14. Data Analyst		
5..15. Tech Lead		
5.1.16. Business Expert	x	"Every team has at least one business expert, sometimes more, depending on the area. A person who has more knowledge of the business area"
5.2. Role overlaps		
6. Key performance indicators		
6.1. Efficiency and productivity		
6.1.1. Velocity		
6.1.2. Lead time		
6.1.3. Timely delivery of projects	x	"We use Power BI to analyse task completion rates, estimated versus actual times, and project delivery dates."; "We evaluate whether tasks were completed on time and how much effort went into them."
6.1.4. Earned value management		
6.1.5. Delivery by assignee		
6.1.6. Delivery by task type		
6.1.7. Delivery by priority		
6.2. Quality and continuous improvement		
6.2.1.Team feedback	x	"Retrospectives help us identify what worked, what didn't, and how to do better."
6.2.2. Bug tracking	x	"We assume that the lower the number of bugs and errors, the higher the satisfaction. However, a single bug can be serious enough to cause the customer to have a low level of satisfaction (...) And so this [metric] is not as real as it could be [...] it's necessary to do customer surveys to gather this information. (...) for quality, bug tracking is important".
6.3. Value		
6.3.1. Client satisfaction	x	"A customer success survey is being developed to gauge satisfaction beyond reported errors."
6.3.2. Contribution to OKRs (Objectives and Key Results)		
7. Challenges		
7.1. Organisational and structural challenges		
7.1.1. Scaling Agile	x	"As the team grows, maintaining efficient workflows and avoiding process overload is a priority."
7.1.2. Managing cross-team collaboration		
7.1.3. Lack of structured KPIs		
7.1.4. Distributed teams		

7.2. Operational and process challenges		
7.2.1. Time estimation		
7.2.2. Balancing flexibility and process rigor		
7.2.3. Balance quality and constraints		
7.2.4. Balance unexpected issues and planning	x	"We allocate 20-30% of sprint time to support, but emergencies can still disrupt plans."; "We respect the sprint backlog 70% of the time, but critical tasks sometimes force changes."
7.2.5. Meeting lengths		
7.2.6. Workload balance		
7.3. Client interaction challenges		
7.3.1. Delays in client deliverables		
7.3.2. Changing requirements and priorities		
7.3.3. Overwhelming communication processes		

COMPANY I		
Themes and codes	**Reported**	**Notes and quotations**
1. Project management hodologies in use		
1.1. Scrum		
1.2. Kanban		
1.3. Scrum and Kanban	x	"We use Scrum for development and Kanban for project nagement."
1.4. Scrum and Scrumban		
1.5. Unidentified hybrid approaches		
2. Reasons for the choice		
2.1. Internal drivers		
2.1.1. Disciplined approach		
2.1.2. Simplicity		
2.1.3. Flexibility / adaptability	x	"Since tasks can change or new priorities arise, we need a flexible roach."
2.1.4. Predictability for the team		
2.2. External influences		
2.2.1. Client requirement		
3. Core practices		
3.1. Scrum-related practices		
3.1.1. Sprints and sprint planning	x	
3.1.2. Sprint review	x	
3.1.3. Sprint retrospective		
3.2. Cross-methodology Agile actices		
3.2.1. Daily syncs	x	"Daily stand-ups help address any doubts or challenges in tasks."
3.2.2. Refinement meetings		

3.2.3. Pair programming		
3.2.4. Milestone-based planning		
3.2.5. Time tracking	x	"We don't do story points, we do time tracking."
3.2.6. Knowledge sharing sessions		
3.2.7. Weekly planning		
3.2.8. Post-mortem		
3.2.9. Decision logs		
3.3. Agile methodologies tailoring		
3.3.1. Alternatives to retrospective per sprint	x	"We do retrospectives more at the end of projects than every two ks."
3.3.2. No predefined periodicity for syncs		
3.3.3. Allow stories to be added during the sprint	x	"Our sprint is two weeks, but tasks can be re-evaluated and pped if something more important appears."
3.3.4. Estimation in time, not in story points	x	"We don't do story points, we do time tracking."
3.3.5. Sprint review, retrospective and planning in one single session		
3.3.6. Mid-sprint review	x	"In the middle of the sprint, we can analyse what can actually be e by the end or if something more important has come up and we e to remove tasks"
3.4. Kanban-related practices		
3.4.1. Task pulling		
3.4.2. WIP limits		
4. Collaborative software and tools		
4.1. Project tracking, documentation nd backlog management		
4.1.1. Zoho Sprints		
4.1.2. Confluence		
4.1.3. Google Workspace		
4.1.4. ClickUp		
4.1.5. Jira	x	"Jira covers everything we need, from project management to elopment workflows."
4.1.6 Trello		
4.1.7. Notion		
4.1.8. Odoo		
4.1.9. GitHub		
4.1.10. Linear		
4.1.11. PHC		
4.1.12. Power BI		
4.1.13. Software developed in-house		
4.2. Communication		
4.2.1. Google Chat		
4.2.2. Google Meet		
4.2.3. Slack		
4.2.4. Discord		

4.2.5. Microsoft Teams	x	"We use Microsoft, Microsoft Teams"
4.3. Design collaboration		
4.3.1. Figma		
4.4. Code, testing and version control		
4.4.1. Bitbucket		
4.4.2. Github		
4.4.3. Insomnia		
5. Core Team Roles		
5.1. Roles		
5.1.1. Product Owner		
5.1.2. Scrum Master	x	
5.1.3. Developers	x	
5.1.4. QA		
5.1.5. Designer	x	
5.1.6. Project Manager	x	
5.1.7. Architect		
5.1.8. Team Lead		
5.1.9. Product Manager	x	
5.1.10. Business Manager		
5.1.11. DevOps		
5.1.12. Release Manager		
5.1.13. Delivery Manager		
5.1.14. Data Analyst		
5..15. Tech Lead	x	"Technical coordinator who always provides, basically, support for roject implementation"
5.1.16. Business Expert		
5.2. Role overlaps	x	The Scrum Master can be handled by a Project Manager or by a duct Owner. QA is also done by project manager and designer
6. Key performance indicators		
6.1. Efficiency and productivity		
6.1.1. Velocity		
6.1.2. Lead time		
6.1.3. Timely delivery of projects	x	"We evaluate time estimated versus time spent for each module."
6.1.4. Earned value management		
6.1.5. Delivery by assignee		
6.1.6. Delivery by task type		
6.1.7. Delivery by priority		
6.2. Quality and continuous provement		
6.2.1.Team feedback	x	
6.2.2. Bug tracking		
6.3. Value		
6.3.1. Client satisfaction	x	

6.3.2. Contribution to OKRs (Objectives and Key Results)		
7. Challenges		
7.1. Organisational and structural challenges		
7.1.1. Scaling Agile		
7.1.2. Managing cross-team collaboration		
7.1.3. Lack of structured KPIs		
7.1.4. Distributed teams		
7.2. Operational and process allenges		
7.2.1. Time estimation	x	Estimating the time required for tasks can be inaccurate, especially n unexpected complexities arise during implementation.
7.2.2. Balancing flexibility and process rigor		
7.2.3. Balance quality and constraints		
7.2.4. Balance unexpected issues and planning		
7.2.5. Meeting lengths		
7.2.6. Workload balance		
7.3. Client interaction challenges		
7.3.1. Delays in client deliverables		
7.3.2. Changing requirements and priorities	x	"Midway through the project, clients sometimes request additional ures or changes that weren't initially defined."; We've had situations re clients at the end say, 'This isn't exactly what I had in mind,' even gh mock-ups were approved earlier."
7.3.3. Overwhelming communication processes		

COMPANY J		
Themes and codes	**Reported**	**Notes and quotations**
1. Project management methodologies in use		
1.1. Scrum	x	
1.2. Kanban		
1.3. Scrum and Kanban		
1.4. Scrum and Scrumban		
1.5. Unidentified hybrid approaches		
2. Reasons for the choice		
2.1. Internal drivers		
2.1.1. Disciplined approach		
2.1.2. Simplicity		
2.1.3. Flexibility / adaptability	x	"Flexibility. We believe in the concept of Scrum, but (...) Scrum has to exist to guide us, to help us organise, and not to hinder us."
2.1.4. Predictability for the team	x	"There is a structure. We try to, even in the most tense moments, with more pressure, to avoid changes in our routines"

2.2. External influences		
2.2.1. Client requirement		
3. Core practices		
3.1. Scrum-related practices		
3.1.1. Sprints and sprint planning	x	"We do the sprint [review and] retrospective and start the new one immediately. We do it in the same meeting to optimise the time."
3.1.2. Sprint review	x	
3.1.3. Sprint retrospective	x	
3.2. Cross-methodology Agile practices		
3.2.1. Daily syncs	x	"We have a daily meeting every day, with all the company members, every morning. (...) The objective is to talk a little about your reality at that moment, what you are doing, what projects you are working on, so that other colleagues can know what other colleagues are doing, what projects, and if there is any point of contact, if anyone needs help, anything. So, within each project we also have a daily. The general daily is 10 minutes, where everyone fits in (...) and then the project daily is usually 15 minutes."
3.2.2. Refinement meetings	x	"And based on what was agreed in the Sprint [planning], it will go into more detail, “ok, let's create this macro task that we talked about in the Sprint, now let's break it down into several”
3.2.3. Pair programming		
3.2.4. Milestone-based planning	x	They have replaced estimation per story by milestone-based planning.
3.2.5. Time tracking		
3.2.6. Knowledge sharing sessions		
3.2.7. Weekly planning		
3.2.8. Post-mortem		
3.2.9. Decision logs		
3.3. Agile methodologies tailoring	x	"It's always 'Scrum but'. I think all companies adapt (...), no one does it by the book". The company avoids what the participant called "Agile formalities" (e.g., story points, excessive meetings)
3.3.1. Alternatives to retrospective per sprint		
3.3.2. No predefined periodicity for syncs		
3.3.3. Allow stories to be added during the sprint	x	"Yes, tasks appear in the middle [of the sprint]."
3.3.4. Estimation in time, not in story points		
3.3.5. Sprint review, retrospective and planning in one single session	x	"At the same meeting, we start reviewing and retrospecting the sprint and start the new one straight away. We do it in the same session to optimise time."
3.3.6. Mid-sprint review		
3.3.7. No estimation per story	x	"We also took the estimation part out of the equation because in most cases it was counterproductive, it caused burnout in the team, frustration and we were distracting ourselves from the essential."
3.4. Kanban-related practices		
3.4.1. Task pulling		
3.4.2. WIP limits		

4. Collaborative software and tools		
4.1. Project tracking, documentation and backlog management		
4.1.1. Zoho Sprints		
4.1.2. Confluence		
4.1.3. Google Workspace		
4.1.4. Clickup		
4.1.5. Jira		
4.1.6 Trello		
4.1.7. Notion		
4.1.8. Odoo		
4.1.9. GitHub		
4.1.10. Linear		
4.1.11. PHC		
4.1.12. Power BI		
4.1.13. Software developed in-house		
4.1.14. GitLab	x	"We have GitLab and we define all the tasks and milestones there. So as auxiliary tools we have Teams for project documentation, and we also have Coda, we put the technical documentation there."
4.1.15. Coda	x	"Coda, we put the technical documentation there."
4.1.16. Microsoft Teams	x	"So as auxiliary tools we have Teams for project documentation"
4.2. Communication		
4.2.1. Google Chat		
4.2.2. Google Meet		
4.2.3. Slack		
4.2.4. Discord		
4.2.5. Microsoft Teams	x	
4.3. Design collaboration		
4.3.1. Figma		
4.4. Code, testing and version control		
4.4.1. Bitbucket		
4.4.2. Github		
4.4.3. Insomnia		
5. Core Team Roles		
5.1. Roles		
5.1.1. Product Owner	x	
5.1.2. Scrum Master	x	
5.1.3. Developers	x	
5.1.4. QA		
5.1.5. Designer		
5.1.6. Project Manager		
5.1.7. Architect		

5.1.8. Team Lead		
5.1.9. Product Manager		
5.1.10. Business Manager		
5.1.11. DevOps		
5.1.12. Release Manager		
5.1.13. Delivery Manager		
5.1.14. Data Analyst		
5..15. Tech Lead	x	"At the beginning of the project, the roles are defined: who is who, who is the tech lead, who is developer"
5.1.16. Business Expert		
5.2. Role overlaps	x	Tech lead acts as Product Owner and Scrum Master.
6. Key performance indicators		
6.1. Efficiency and productivity		
6.1.1. Velocity		
6.1.2. Lead time		
6.1.3. Timely delivery of projects	x	They evaluate if the sprints end on time (checking whether they achieve the sprint goal or if they are late) This ongoing evaluation helps them gauge their performance throughout the project.In the end, they also evaluate if the project itself is delivered on time, meeting its overall deadline. "The KPIs are the delivery of the project on time, and, before that, the good execution of the sprints which already gives us a forecast."
6.1.4. Earned value management		
6.1.5. Delivery by assignee		
6.1.6. Delivery by task type		
6.1.7. Delivery by priority		
6.2. Quality and continuous improvement		
6.2.1.Team feedback	x	
6.2.2. Bug tracking		
6.3. Value		
6.3.1. Client satisfaction	x	"At the end of the project, we have customer surveys"
6.3.2. Contribution to OKRs (Objectives and Key Results)		
7. Challenges		
7.1. Organisational and structural challenges		
7.1.1. Scaling Agile		
7.1.2. Managing cross-team collaboration		
7.1.3. Lack of structured KPIs	x	The interviewee reported a lack of structured performance tracking, particularly for task regressions, sprint efficiency and how the project is progressing compared to the plan.
7.1.4. Distributed teams		
7.2. Operational and process challenges		
7.2.1. Time estimation		
7.2.2. Balancing flexibility and process rigor		

7.2.3. Balance quality and constraints		
7.2.4. Balance unexpected issues and planning		
7.2.5. Meeting lengths		
7.2.6. Workload balance		
7.3. Client interaction challenges		
7.3.1. Delays in client deliverables		
7.3.2. Changing requirements and priorities	x	"Inputs that we receive from the client and that also change"; "we focus, for example, on the requirements 100% and then, after a while, the requirements have to be changed"
7.3.3. Overwhelming communication processes		

COMPANY K		
Themes and codes	**Reported**	**Notes and quotations**
1. Project management methodologies in use		
1.1. Scrum	x	
1.2. Kanban		
1.3. Scrum and Kanban		
1.4. Scrum and Scrumban		
1.5. Unidentified hybrid approaches		
2. Reasons for the choice		
2.1. Internal drivers		
2.1.1. Disciplined approach		
2.1.2. Simplicity		
2.1.3. Flexibility / adaptability	x	
2.1.4. Predictability for the team	x	"Our methodology allows us the flexibility to make adaptations along the way, while giving us the stability of a routine."
2.2. External influences		
2.2.1. Client requirement		
3. Core practices		
3.1. Scrum-related practices		
3.1.1. Sprints and sprint planning	x	
3.1.2. Sprint review	x	
3.1.3. Sprint retrospective		
3.2. Cross-methodology Agile practices		
3.2.1. Daily syncs	x	
3.2.2. Refinement meetings		
3.2.3. Pair programming		
3.2.4. Milestone-based planning		
3.2.5. Time tracking		
3.2.6. Knowledge sharing sessions		

3.2.7. Weekly planning		
3.2.8. Post-mortem		
3.2.9. Decision logs	x	"Documenting decisions helps us evolve consistently and ensures that future decisions are made thoughtfully, based on a history. It also ensures that we are all on the same page, understanding why this specific work is being done, why this specific task, what value will this create".
3.3. Agile methodologies tailoring		
3.3.1. Alternatives to retrospective per sprint	x	"As a small team we communicate all the time, we found that we don't need a ceremony for reflection. If someone feels that something is not right, they communicate. If they find a better way to do something, they share with the team to improve. We encourage this attitude."
3.3.2. No predefined periodicity for syncs		
3.3.3. Allow stories to be added during the sprint	x	"Scrum is here to help us organize our work, not not limit our work. We need to move fast, test and adapt. If we follow all Scrum rules, we wouldn't be as agile as we are (...) It is not uncommon to add multiple tasks to the sprint since they are related to the sprint goal. We actually leave some space for them when planning"
3.3.4. Estimation in time, not in story points		
3.3.5. Sprint review, retrospective and planning in one single session	x	"Review and sprint planning are done in a single session, they are interconnected"
3.3.6. Mid-sprint review	x	"A mid-sprint review is a check-up moment in the middle of the sprint. Where are we now? It helps reorganise priorities, adjust the plan without waiting until the end of the Sprint."
3.3.7. No estimation per story		
3.4. Kanban-related practices		
3.4.1. Task pulling		
3.4.2. WIP limits		
4. Collaborative software and tools		
4.1. Project tracking, documentation and backlog management		
4.1.1. Zoho Sprints		
4.1.2. Confluence	x	"We have our technical documentation in Confluence"
4.1.3. Google Workspace	x	"Collaborative documents and spreadsheets, more complex things that require formulas or formatting that Notion can't handle are in Google Drive"
4.1.4. ClickUp		
4.1.5. Jira	x	"The technical team uses Jira, the Content & Marketing team uses Notion. And then we have links from one to another to connect all team tasks and guarantee we are all in the same page"
4.1.6 Trello		
4.1.7. Notion	x	"The technical team uses Jira, the Content & Marketing team uses Notion. And then we have links from one to another to connect all team tasks and guarantee we are all in the same page"
4.1.8. Odoo		
4.1.9. GitHub		
4.1.10. Linear		
4.1.11. PHC		
4.1.12. Power BI		

4.1.13. Software developed in-house		
4.1.14. GitLab		
4.1.15. Coda		
4.1.16. Microsoft Teams		
4.2. Communication		
4.2.1. Google Chat		
4.2.2. Google Meet	x	"Meetings, dailies when we are not all at the office, all done with Google Meet"
4.2.3. Slack	x	"Slack is our main communication channel"
4.2.4. Discord		
4.2.5. Microsoft Teams		
4.3. Design collaboration		
4.3.1. Figma	x	"We use Figma for mockups and prototyping, but also to create mind maps, diagrams to explain features and flows..."
4.4. Code, testing and version control		
4.4.1. Bitbucket	x	"Bitbucket for code management and version control"
4.4.2. Github		
4.4.3. Insomnia		
5. Core Team Roles		
5.1. Roles		
5.1.1. Product Owner	x	
5.1.2. Scrum Master	x	
5.1.3. Developers	x	
5.1.4. QA		
5.1.5. Designer	x	
5.1.6. Project Manager		
5.1.7. Architect		
5.1.8. Team Lead	x	
5.1.9. Product Manager	x	
5.1.10. Business Manager		
5.1.11. DevOps		
5.1.12. Release Manager		
5.1.13. Delivery Manager		
5.1.14. Data Analyst		
5..15. Tech Lead	x	
5.1.16. Business Expert		
5.2. Role overlaps	x	"The Product Manager (the CEO) or by the Tech Lead can act as Scrum Master. The Tech Lead is also a developer. The Product Owner is the Content Team Lead. Having many hats in a small company is the norm."
6. Key performance indicators		
6.1. Efficiency and productivity		
6.1.1. Velocity	x	"Velocity is not just for us to plan sprints. In a recent project, improving velocity over sprints was also an indicator that we were

		becoming more familiar with a technology. We realized we could deliver value faster."
6.1.2. Lead time		
6.1.3. Timely delivery of projects	x	"One of the big KPIs is time spent versus time planned."
6.1.4. Earned value management		
6.1.5. Delivery by assignee		
6.1.6. Delivery by task type		
6.1.7. Delivery by priority		
6.2. Quality and continuous improvement		
6.2.1.Team feedback	x	
6.2.2. Bug tracking		
6.3. Value		
6.3.1. Client satisfaction	x	"We base much of our development decisions on direct customer feedback or indirect feedback - through the usage and engagement data we have. Knowing whether something was successful or not has a lot to do with how it was received by users versus the effort it involved, essentially in terms of time, since all of our development is in-house."
6.3.2. Contribution to OKRs (Objectives and Key Results)		
7. Challenges		
7.1. Organisational and structural challenges		
7.1.1. Scaling Agile		
7.1.2. Managing cross-team collaboration		
7.1.3. Lack of structured KPIs	x	"We need to think properly about this subject of KPIs. We are very focused on producing and as a small team with few resources, things like these, less urgent, are postponed."
7.1.4. Distributed teams		
7.2. Operational and process challenges		
7.2.1. Time estimation	x	"Our biggest challenge is undoubtedly time estimation because our work is not repetitive. We are always developing new features with different technical challenges and it is very difficult to understand the level of effort and how long it will take us before we get our hands dirty."
7.2.2. Balancing flexibility and process rigor		
7.2.3. Balance quality and constraints		
7.2.4. Balance unexpected issues and planning		
7.2.5. Meeting lengths	x	"We want the whole team to be involved. This means entropy sometimes. For example, our status meetings, our dailies, can last more than an hour."
7.2.6. Workload balance		
7.3. Client interaction challenges		
7.3.1. Delays in client deliverables		
7.3.2. Changing requirements and priorities		

7.3.3. Overwhelming communication processes		