future internet

MDPI

Article

Exploring the Benefits of Combining DevOps and Agile

Fernando Almeida [1,*], Jorge Simões [1] and Sérgio Lopes [2]

[1] Polytechnic Higher Institute of Gaya (ISPGAYA) and INESC TEC, 4400-103 Porto, Portugal; jsimoes@ispgaya.pt
[2] Polytechnic Higher Institute of Gaya (ISPGAYA) and Distance Education and eLearning Laboratory-Open University (LE@D-UAb), 1269-001 Lisbon, Portugal; ssargo@ispgaya.pt
* Correspondence: almd@fe.up.pt

Abstract: The combined adoption of Agile and DevOps enables organizations to cope with the increasing complexity of managing customer requirements and requests. It fosters the emergence of a more collaborative and Agile framework to replace the waterfall models applied to software development flow and the separation of development teams from operations. This study aims to explore the benefits of the combined adoption of both models. A qualitative methodology is adopted by including twelve case studies from international software engineering companies. Thematic analysis is employed in identifying the benefits of the combined adoption of both paradigms. The findings reveal the existence of twelve benefits, highlighting the automation of processes, improved communication between teams, and reduction in time to market through process integration and shorter software delivery cycles. Although they address different goals and challenges, the Agile and DevOps paradigms when properly combined and aligned can offer relevant benefits to organizations. The novelty of this study lies in the systematization of the benefits of the combined adoption of Agile and DevOps considering multiple perspectives of the software engineering business environment.

Keywords: software development process; operations; software engineering; information system development; team structure

Citation: Almeida, F.; Simões, J.; Lopes, S. Exploring the Benefits of Combining DevOps and Agile. *Future Internet* **2022**, *14*, 63. https://doi.org/10.3390/fi14020063

Academic Editor: Davide Tosi

Received: 12 January 2022
Accepted: 17 February 2022
Published: 19 February 2022

1. Introduction

The software development process can be viewed as a set of tasks required to produce high-quality software. The literature shows that the quality of the software produced reflects the way the process was carried out [1–3]. Although several software development processes exist, generic activities common to all of them stand out, as Sommerville [4] highlights, such as software specification (e.g., requirements definition, software constraints), software development (e.g., software design and implementation), software validation (e.g., software must be validated to ensure that the implemented functionality conforms to what was specified), and software evolution (e.g., software evolves as per customer need). The software development process provides an interaction between users and software engineers, between users and technology, and between system engineers and technology. In this sense, software development is an interactive learning process, and the result is an embodiment of knowledge gathered, transformed, and organized as the process is conducted.

A software development methodology includes a set of activities that assist in the production of software. These activities result in a product that demonstrates how the development process was conducted. Agile methodologies arise from the need to overcome the difficulties and disadvantages of applying traditional methodologies in project management and implementation. The Agile methodology assumes short periods of time between each delivery to ensure early and continuous delivery of software susceptible to evaluation [5]. In [6] it is also recognized that software implementation according to this paradigm is interactive and incremental, enabling early confirmation of whether or not

the delivered artifact meets the needs and making the respective corrections with low risk and cost.

The main social and human factors involved in the adoption of Agile methodologies are the impact on organizational culture, namely by the collaborative culture imposed on developers and the implications of being embedded in an Agile team [7,8]. Constant feedback to all team participants on the activities being carried out, and the commitment to the team's goals, are highlighted in Junker et al. [9] as key elements for a well-functioning Agile team. Feedback and collective awareness are essential as opposed to individualism and lack of communication. This view is also confirmed by Sweetman and Conboy [10] when they highlight that feedback loops are the essence of the empirical and complex processes found in software engineering that require continuous adaptation based on learning obtained daily. Furthermore, complex projects are very unpredictable and therefore need a process that incorporates unpredictability [11].

Inspired by the success of Agile methods, the DevOps (Development and Operations) movement emerged that aims to take this line of reasoning to a higher level. This movement comes to break the traditional culture where there was almost no interaction between teams and, as highlighted by Luz et al. [12], the goal is to create a culture of collaboration between development and operations teams that allows increasing the flow of completed work. In summary, it is intended to increase the frequency of deployments while increasing the stability and robustness of the production environment. Beyond a cultural change, the DevOps movement also focuses on the practices of automating the various activities required to deliver quality code into production, such as creating environments for testing, automating testing, configuring infrastructure, data migration, auditing, and security, among others [13,14].

In the literature, we can essentially find studies on DevOps that explore ways to align development teams with operations [15], the benefits that this methodology can bring to organizations [16], and the challenges that are posed [17]. However, there is a research gap in the characterization of the simultaneous adoption of Agile and DevOps in organizations. In this dimension, the number of available scientific studies is limited and they mostly present individual views of their implementation, which does not allow for a sufficiently comprehensive characterization of the benefits of their combined adoption. We acknowledge the study conducted by Hemon et al. [18], which characterizes the different phases of Agile to DevOps transition (e.g., Agile, ongoing integration, and constant delivery), while Melgar et al. [19] explore the benefits of the combined SCRUM-DevOps approach in terms of increasing speed during the deployment process and increasing the quality of software processes. In both studies, there is just one empirical case study, which makes it difficult to generalize the findings. In this sense, this study seeks to bridge this research gap and presents an analysis of the benefits that can be found by the combined approach of DevOps and Agile considering a comprehensive set of twelve case studies that are representative of practices of simultaneous adoption of both methodologies. This approach supported by multiple case studies avoids the individual limits of each company's vision and reduces the risk of bias, and allows comparing, grouping, and systematizing the main benefits of the combined adoption of both methodologies.

The rest of this manuscript is organized as follows: Initially, a theoretical contextualization of the DevOps model is performed and the similarities between DevOps and Agile are explored. Next, the methodology and associated methods adopted in the study are described. This is followed by the presentation of the results and discussion of their relevance to the perception of the benefits of the combined adoption of DevOps and Agile. Finally, the conclusions are enumerated. It is also in this last phase of the manuscript that the limitations of the study are addressed and some suggestions for future work are provided.

2. Literature Review

2.1. DevOps Concepts and Model

In 2009, Paul Hammond and John Allspaw presented the talk "10+ Deploys Per Day: Dev and Ops Cooperation at Flickr" [20]. They explained how the developers' teams (Dev) and operations teams (Ops) could contribute to more agile and scalable software development. Tight integration between Dev and Ops to safely achieve several software deployments (more than 10) in a single day was a disruptive idea regarding software development and its evolution. Later, Patrick Debois coined the term DevOps (Development and Operations) and created the DevOps Day event [21]. Although the DevOps movement has been discussed for more than a decade [13–15,22] it still lacks a unique formal definition. For Wiedemann et al. [23], the lack of a unique definition could be intentional to allow each team to choose the definition that better suits its needs. Nevertheless, several authors proposed definitions such as the one by Leite et al. [13]—"DevOps is a collaborative and multidisciplinary effort within an organization to automate continuous delivery of new software versions while guaranteeing their correctness and reliability"—and others view it as a combination of values, principles, methods, practices, and tools [24]. Some other common definitions can be found in [23].

One of the key points in the execution of a project is the approach used to manage it. The traditional approach based on the waterfall model looks at the project in a linear way with several events, while in the iterative approach the development of software is undertaken through successive progress [25]. Therefore, it is common that the system is presented still incomplete or with some deficient parts. The objective is that the refinement of the product happens in stages until the desired result is achieved. The software development process does not end with the release of the code, but only when it closes the feedback loop between those who write the code and those who use it. DevOps aims to remove the barriers between traditionally independent teams: development and operations. Under the DevOps approach, these teams should work together across the entire software life cycle, from development and testing through deployment to operations. More than only a technical subject, DevOps deals with the organizational and human issues that arise in the software life cycle. It promotes a culture of collaboration, integration, and communication between teams to reduce the disconnect between them while assuring the delivery of software in an agile, safe, and stable way. According to Rajapakse et al. [14], the DevOps movement is currently a widely adopted software development approach having as a major benefit the ability to deploy releases more frequently and at a higher rate.

Some related concepts used in conjunction with DevOps are Continuous Integration, Continuous Delivery, and Continuous Deployment. As noted in [14], these concepts are considered key practices within DevOps, but are not always clearly used, as stressed by Stahl et al. [24]. Continuous Integration is a development practice where developers frequently integrate the code they produce, that has successfully passed testing, to the project under development [24]. Those integrations occur typically once a day. Continuous Delivery is a development practice where the software is kept in a reliable deployable state at any time [14]. Potentially, after every change, the software can be released. This leads to several release candidates that are evaluated. The deployment to production is made manually by a team member, with the appropriate authority, who decides when and which candidate should be released [14]. Apart from Continuous Delivery, in Continuous Deployment, release candidates resulting from software changes are automatically deployed to production without the intervention of any team member [14,24].

Regarding how organizations could adopt DevOps and measure its success, the CALMS framework is considered a foundational model for DevOps. CALMS is an acronym for Culture-Automation-Lean-Measurement-Sharing. CALMS was created by Jez Humble, co-author of The DevOps Handbook. The acronym highlights the five core elements of CALMS [23]:

- Culture: a cultural change focusing on collaboration and integration between developers' team and operations' team;

- Automation: a high level of automation to achieve continuous delivery running each code change through automated tests;
- Lean: the application of lean principles to increase efficiency and reduce complexity;
- Measurement: keeping key performance indicators for measuring performance and identifying where improvements can be achieved;
- Sharing: knowledge and best practice should be shared in the organization and across organizational boundaries.

Security issues concerning DevOps led to the spread of another term: DevSecOps. It adds "Security" to "Development" and "Operations", which were already part of the DevOps term. According to Rajapakse et al. [14], security is often treated as a non-functional requirement, handled at a later stage of the software development life cycle. Under DevSecOps, security should be built into every part of the DevOps life cycle. The purpose of the DevSecOps philosophy is to align the speed of code delivery with building secure code, merged into one streamlined process.

The application of DevOps still must deal with some problems and concerns that can limit its use (e.g., resistance to change, organizational vision, legacy systems) [26]. Misuse of the term, lack of clear guidelines and, as already mentioned, the lack of a clear definition creates some ambiguity about how to use DevOps principles. Those principles presuppose that, before DevOps, development teams and operations teams were working independently with almost no knowledge about each other's work. This lack of knowledge across teams is not, in general, as deep as DevOps assume. The whole software development process is improved with better collaboration between teams, as DevOps advocates, but it does not mean that DevOps teams did not previously cooperate. Another concern around DevOps is that its adoption rate is still low.

There is a close relation between DevOps and Agile methods in software development. According to Leite et al. [13], DevOps is an evolution of the Agile movement since software development under Agile favors small release iterations with customer reviews.

2.2. Similarities and Differences between DevOps and Agile

In the context of Software Engineering, as discussed in this paper, DevOps can be understood as a behavioral evolution of Agile development [27], which was gradually conceived through practical experiences of implementation in software development. However, it is important to point out that the Agile method has its focus directed specifically to software development [28], while DevOps aims to involve the software development area in the implementation and operation of the software developed or still under development, which shows us that DevOps processes are being implemented within the Agile processes.

Currently, in the professional community of Information and Communication Technologies (ICT) there is a growing consensus, in practice, that DevOps can be understood as "DevOps = Agile + Lean + IT service management (ITSM)" [29]. In its method and processes, DevOps adopts characteristics of frameworks related to the technical area of Agile software development together with ICT management processes. Complementarily, other relevant methodologies (e.g., Extreme Programming, Dynamic Systems Development Method, Kanban, SCRUM) have approaches intrinsically related to the Agile philosophy [30–33]. SCRUM is very well-known and intensely used in the Agile method, and is generally enhanced by the Kanban tool, for managing the workflow of software development, but which also fits very well into the DevOps development process itself [34]. Table 1 presents the problems or gaps in the Agile method that are solved by adopting the DevOps method.

Table 1. Problems with Agile development and DevOps solution (adapted from [35]).

Problem with Agile Development	DevOps Solution
Delivery of new features to the customer is often delayed.	DevOps tools are used to test and release new features as they are completed.
Completed software components are not compatible with each other.	Open interfaces and test automation make it possible to divide development into independent yet compatible parts.
Quality of the product is not ensured properly prior to release.	DevOps tools and practices help automating quality assurance and reduce the need for repetitive manual work.
New features break old functions.	The quality of existing functions is ensured quickly and automatically after each change.
Budget goals and deadlines are missed.	The tools and procedures of DevOps increase the transparency and predictability of the development work.
Developer teams and IT operations crews are not cooperating.	Developer teams and IT operations crews agree upon responsibilities together. Their goals are unified.

In the context of the Agile method workflow, under the SCRUM framework implementation, DevOps presents four metrics [36] that are directly related to software delivery through enhanced software engineering: i—waiting time, ii—deployment frequency, iii—mean time to restore (MTTR), and iv—change failure percentage. These metrics can be implemented in the execution cycle of the Agile by SCRUM approach, enhancing the qualitative process of the organizational performance of software companies, in meeting the planned objectives, because it increases in a relevant way the level of monitoring and maturity of the activities performed in each work cycle.

Organizations that implement Agile methodologies focus on productivity and process optimization by reducing execution time [37]. However, software development environments are shrouded in constant change, with continuous interactions of teams that focus on deliverable products. DevOps processes can improve the interactivity of the development teams, improving the integration process of the stakeholders of a project in an Agile environment, facilitating the continuity of the processes in a more balanced and stable way.

According to [38], the DevOps culture can be implemented to carry out an incident management process for deliverable products, allowing Agile development teams to be continuously monitored, because it integrates the software deployment process by monitoring operations, as provided by the DevOps framework. Therefore, DevOps enhances the stability of the Agile cycle, as we mentioned before; that is, while Agile focuses on productivity in product deliverables in a more technical approach around ICT [39], DevOps directs its processes to checking the level of effectiveness of what is produced in the work cycles of the development teams. Therefore, in this central aspect of integration, there will be a tendency toward an improved qualitative increment in the scope of Software Engineering.

Within what we have discussed so far, it was verified that the processes originated from Agile methodologies are the structural base of DevOps [18,39], and that the union of these methodologies increases the level of intelligence of the information system that is generated in the work cycles, because it implements several functions, such as Developers, System Architects, Product Owners, Release Engineers, and Testers. Therefore, the level of collaboration is elevated with professionals that are beyond the initially foreseen roles, for example, in the SCRUM approach that is currently used in Agile, promoting the creation of cross-functional teams in the context of the DevOps approach.

The approach of development teams that implement Agile methodologies, to operations teams as proposed by DevOps, tends to accelerate the software release process, with studies [40] indicating that in addition to acceleration, there is an increase in software quality in terms of reliability and maintainability. As a result, the deliverable product meets the conditions foreseen in the essential objectives of the project, from software development

(Agile), as in deployment and testing in operations (DevOps). However, despite DevOps decreasing the gap between Developers and Operators, in terms of standardization, the DevOps methodology lacks a simple approach or a roadmap to be followed for its implementation in an organization [41], leaving it up to companies to define their standards and metrics. This can present as a complicating factor, very dependent on the maturity level of organizations and work teams, which will have to define their specific integration processes in a DevOps approach.

3. Methodology

This study adopts a qualitative methodology to explore the benefits of the combined adoption of DevOps and Agile. This type of methodology is used in the context of social sciences and engineering and, according to Merriam and Tisdell [42], is concerned with a level of reality that cannot be quantified, exploring meanings, aspirations, beliefs, values, and attitudes that correspond to a deeper space of relationships, processes, and phenomena, and cannot be reduced for operationalization of variables. Dyba et al. [43] recognize that the software industry presents lines of research that are not only limited to exploring technical software engineering issues, but also need to look at the challenges of the intersection between technical and non-technical aspects. In this sense, adopting a purely quantitative approach is insufficient. Moreover, phenomena addressed in the field of project development in a DevOps and Agile paradigm are complex and unique. Therefore, the qualitative approach adopted in this study allows for exploring and understanding the relationships and activities performed by organizations in their software development activity.

In the scope of this study, twelve case studies were incorporated that report the simultaneous inclusion of DevOps and Agile methodologies in their software development teams. These cases correspond to reports and press releases from commercial vendors of reference in this area. The data come from secondary sources and the authors have not made any changes to the press releases that represent the view of each manufacturer. No summarization of the press releases has been made. It cannot be guaranteed that there is no risk of bias since the identified benefits come from press releases from commercial companies in the area and that have commercial goals in the market. However, to minimize this risk, external and internal validity mechanisms were used. To ensure external validity, multiple case studies were used of companies in different geographic areas, and to ensure internal validity, the same identifier was used to associate similar benefits between the case studies.

Table 2 presents brief descriptions of the profiles of the case studies. In general, it can be concluded that the area of activity of the organizations is similar through the offer of IT products and services based on cloud architecture. We also highlight the offer of consulting services in the field of web and mobile development, and the offer of complementary services in the field of artificial intelligence, big data, and cybersecurity.

Thematic analysis is an interpretive data analysis method widely used in the social sciences and engineering and was adopted in the context of this study to identify common benefits in the combined adoption of DevOps and Agile. Flexibility is, according to Braun and Clarke [44], one of the benefits of thematic analysis. Through its theoretical freedom, the thematic analysis provides a flexible and useful research tool that can potentially provide a rich and detailed set of information about the data. Figure 1 shows the process of conducting data. The step begins with the coding process in which one coded the advantages present in each case study report. After that, an iterative process follows in which common themes of the advantages among the case studies are identified. Finally, the last step consists of accounting for the benefits identified in the case studies. This approach allows for a ranking of the top benefits.

Table 2. Profiles of the case studies.

ID	Country	Description
CS1	India	A company that operates in the IT outsourcing services market in building technological solutions in areas such as web apps, mobile apps, cloud strategy, analytics and business intelligence, testing, quality assurance services, and Agile project management. Their report looks to different ways to enable DevOps in Agile environments.
CS2	USA	Information technology company and advanced training for IT professionals in the fields of programming and technological development. Their report explores the relevance of Agile principles for deployment activities.
CS3	Canada	A global company that offers consulting services to help companies adjust their development teams by integrating new practices and technologies. Their report explores how Agile practices should be updated considering the needs of operations teams in organizations.
CS4	USA	Cloud services provider offering technology infrastructures based on public cloud, private cloud, hybrid cloud, and multi-cloud. Their report explores the difference and similarities between both paradigms.
CS5	Australia	Australian software company that develops products for software developers, project managers, and other software development teams. Their press release presents how automation processes can be implemented using a combination of both paradigms.
CS6	USA	Global business and technology consulting firm dedicated to helping organizations leverage emerging technologies and the latest business management thinking to achieve competitive advantage and mission success. Provides consulting and training services, primarily targeted at executives. Their article explores the differences between the two paradigms and suggests points of convergence between them.
CS7	USA	Company specialized in the dissemination of technological information in the field of information and communication technologies. Their press release looks to important aspects observed while combining DevOps and Agile.
CS8	USA	Multinational company in information technologies that develops automation solutions and advanced knowledge in areas such as automation, enterprise DevOps, data-driven business, and adaptive cybersecurity. Their article explores the role that Agile practices can play in DevOps.
CS9	Switzerland	Company that operates mainly in the European market in providing captivating scalable cloud-based solutions. Their article looks at the isolated benefits of each paradigm and tries to predict the benefits of their combined adoption.
CS10	India	Company that develops technological solutions for the education field and relies on the application of the Agile scalability paradigm, especially the SAFe model. Their article explores the change-driven management approach and looks at how DevOps and SCRUM address this challenge.
CS11	Germany	IT company that operates in the global market implementing cloud solutions, DevOps, software testing, quality assurance, artificial intelligence, and big data. Their press release looks at the problems in software engineering that the joint adoption of both paradigms can solve.
CS12	UK	A consulting company that aims to optimize work processes in organizations using cloud solutions, slack, and Trello services. Their press release looks at the role of the cloud and Agile methodologies in developing the DevOps paradigm.

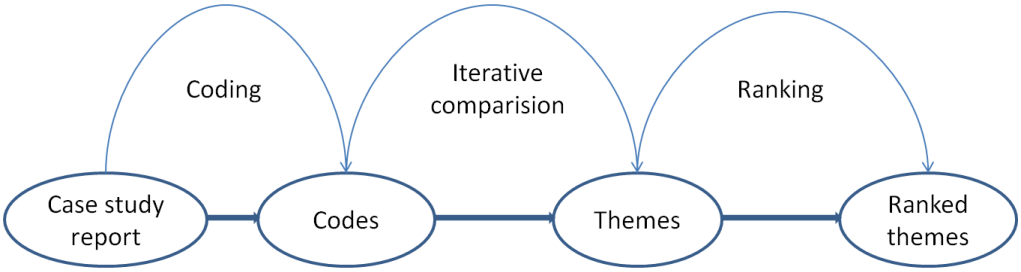

Figure 1. Thematic analysis process (authors own illustration).

4. Results

The twelve case study reports associated with each company presented in Table 2 were thoroughly read and each identified benefit was assigned a unique identifier. Each theme is identified by the acronym "BF" and a number is associated to differentiate each benefit. Common themes have the same acronym. A brief description of how each benefit is understood in the case studies is also included. Table 3 presents the identification process of the themes associated with each case study. Table 3 shows the themes for all the case studies mentioned in Table 2. We highlight the existence of themes that are common to several case studies. Although the themes are common, the vision of each case study in relation to them has some relevant oscillations, which indicate a complementary vision of the institutions present in the case studies. For example, in CS1 time to market emerges due to increased collaboration between teams, whereas in CS8 process integration is highlighted. Something similar emerges in relation to cost. Cost reduction is understood in CS6 as achieved from the existence of multi-skilled human resources, whereas in CS8 cost reduction is motivated by the effects of increased team performance.

Table 3. Identified benefits themes in the case studies.

ID	Benefit
CS1	(BF1) Time to market: greater collaboration between teams reduces software delivery cycles (BF2) Automation: the combined development and production process becomes more automated to meet market needs
CS2	(BF2) Automation: continuous delivery and integration combined with fast releases lead to the automation of activities (BF3) Communication: promote constant communication between development and operational team (BF4) Mindset and culture: establishment of collaboration among teams
CS3	(BF1) Time to market: through continuous delivery from the development phases (BF5) Planning: the product backlog now includes services are products that need to be deployed, scalable, maintained, monitored, and supported as a service
CS4	(BF4) Mindset and culture: increase the quality of collaboration (BF6) Visibility: more visibility for release and upgrade processes (BF7) Risk mitigation: better identification of risks in the context of each sprint (BF8) Software quality: decrease the existence of software errors and helps to launch faster patches
CS5	(BF2) Automation: contribution for the implementation of Agile fluency model which focus on value, transparency, and alignment (BF3) Communication: amplify feedback loops between development and operational team (BF4) Mindset and culture: looks to the performance of all system instead of local departments. Furthermore, promotes learning from failure. (BF5) Planning: increase the planning dimension of unplanned events typically found in the context of operational teams

Table 3. *Cont.*

ID	Benefit
CS6	(BF1) Time to market: deployment chains cut the time needed to get a product to market (BF9). Cost: combining people and activities makes people more multi-skilled with future reduction in human resource costs
CS7	(BF2) Automation: increasing code size and complexity encourages process automation (BF3) Communication: communication between both teams is constant with feedback loops (BF10): Software quality: functional and load tests are both considered (BF11). Efficiency: project management considers performance metrics that result from combined methods in both areas
CS8	(BF1) Time to market: integrated processes make order fulfillment faster (BF6) Visibility: increased visibility over data and processes (BF9) Cost: increased productivity and team performance
CS9	(BF2) Automation: increase speed and agility to attend continuous requirements changes (BF3) Communication: smooth communication between the team and the customers by continual iteration (BF12) Flexibility: agility in the face of continuous requests for revision becomes important to make the organization competitive
CS10	(BF2) Automation: implementation of a paradigm based in continuous integration, continuous delivery, and continuous deployment (BF3) Communication: by fostering communication in the teams, constant collaboration is promoted
CS11	(BF2): Automation: shorten the development cycle by promoting the automation of repetitive tasks (BF10) Software quality: focus on end-product quality
CS12	(BF2) Automation: better performance when compared against on-premise DevOps automation. Furthermore, it contributes to eliminates human errors (BF12) Flexibility: it empowers each stage of the application delivery lifecycle

Table 4 summarizes the comparative analysis of the identified benefits. All previously identified benefits are mapped. The "ranking" attribute allows us to understand the relative importance of the benefits and to perceive which ones are transversal to several case studies and which ones emerge only in a very specific context of each organization. The benefits related to automation (BF2), communication (BF3), and time to market (BF1) stand out. These are the three main benefits that can be found in the combined adoption of Agile and DevOps. Conversely, there are other benefits that are identified in a smaller number of case studies, namely, those related to efficiency (BF11), risk mitigation (BF7), and software quality (BF8). These benefits are less relevant and arise in the specific context of each organization, which indicates that they are more difficult to replicate in other software companies.

Table 4. Comparative analysis of benefits and ranking.

Benefit	CS1	CS2	CS3	CS4	CS5	CS6	CS7	CS8	CS9	CS10	CS11	CS12	Ranking
BF1	X		X			X		X					#3
BF2	X	X			X		X		X	X	X	X	#1
BF3		X			X		X		X	X			#2
BF4		X		X	X								#4
BF5			X		X								#5
BF6				X				X					#5
BF7				X									#10
BF8				X									#10
BF9						X		X					#5
BF10							X				X		#5
BF11							X						#10
BF12									X			X	#5

5. Discussion

Although Agile and DevOps are widespread and different concepts, they can be combined and offer relevant benefits to organizations. As reported in [45], companies have problems in the process of implementing and releasing new software versions because most of the time this is a process performed manually. In addition, this approach leads to a high quantity and frequency of errors [46]. To reduce the incidence of problems and increase flexibility and automation, non-operational resources can be used and in environments that are not in production. The combined adoption of Agile and DevOps allows the developer to gain greater control over the environment, infrastructure, and applications.

The seamless integration between Agile and DevOps generates a more collaborative and Agile framework. This approach leads to a simplification and automation of model processes to make them more rational and efficient. A classic example of this benefit is given by Fabro [47] when highlighting the reduction in delivery cycles, endowing small development packages with a previously unrecognized value. Hemon-Hildgen et al. [48] also highlight the role of orchestration, which consists of automating tasks to optimize the process and reduce repetitive steps that add little to the development cycle. Finally, automated testing along the Agile and DevOps chain allows the reuse of tests between environments and makes them more sustainable [49].

Team communication is recognized in DeFranco and Laplante [50] and Schmutz et al. [51] as the main cause for product delivery failures. By starting to work together, teams can more easily track the evolution of processes from their inception, which fosters the emergence of process improvements. Cois et al. [52] recognize that the great differentiator of DevOps lies in its ability to optimize communication between the teams involved and the customer. This allows, for example, the team to involve the operations team, which enables the implementation of the ITOps model [53]. This enables it to provide a sufficiently secure development environment. However, interconnecting it with an organization's Agile teams offers more potential. For example, the marketing and sales departments can be involved in the activities covering the delivery of the releases, which allows companies to add even more value to the product by using the full potential of their available resources.

The findings further revealed a very diverse number of benefits, such as increased visibility over processes, better identification and mitigation of risks, or increased software quality. The integration of the two paradigms fosters consolidation, which allows project managers to have greater visibility of both the work of the teams and the interdependencies between them [54]. Furthermore, iterative planning between teams makes it easier to adapt in case of changes, and continuous customer feedback generates value from the beginning of the project, lowering the risks associated with development and operation [55]. In the joint Agile and DevOps paradigm, both teams share responsibility for producing functional and quality code and need to work together to achieve these common goals.

Finally, it is recognized that in recent years there has been a growing adoption of the term "DevOps culture", as a counter position to DevOps implementations based only on tools. In DevOps culture, it is advocated that software development and infrastructure teams work together towards the same goal [56,57]. As Clavier and Kaminski [58] argue, DevOps not only optimizes development processes but changes the way employees think about their products and interact with customers. The combined Agile and DevOps approach allows the leveraging of these benefits by enhancing empathy among team members and unites sectors that previously worked independently and without personal connection. Furthermore, as Venugopal [59] acknowledges, when there is trust between teams, then it also increases the freedom that professionals have to experiment and innovate, without the problems of incompatibility and miscommunication as there would be with separate teams.

6. Conclusions

This study demonstrates that the Agile and DevOps paradigms are not incompatible but can bring benefits to organizations when properly aligned. While Agile brought about a fast delivery model aligned with customer expectations, DevOps optimized this system. In this sense, an alternative that usually gives great results is the adoption of both methodologies. They not only complement each other but also help companies to face changes in a team.

Changing the strategy and methodology of a team can be a delicate process full of obstacles. Therefore, organizations must address this challenge in a cross-cutting way within the organization to avoid isolated silos that do not contribute to collaborative work. Agile creates a space for more agile work with partial deliveries, while DevOps creates an environment conducive to managing these processes, with effective communication.

This study offers both theoretical and practical relevant contributions. In the theoretical dimension, this study has enabled the identification of a set of benefits of the combined adoption of both paradigms through the adoption of multiple case studies of software companies in the international market. The study identifies a total of 12 benefits and allows us to explore the relative relevance of each of them. From a practical perspective, the benefits identified are relevant to companies that, having adopted Agile and DevOps alone, have not yet taken steps towards the combined adoption of both models. The findings of the study made it evident that the two models are not incompatible, but when combined they can amplify their impacts on organizations.

Limitations and Future Research Directions

This study presents some limitations. Firstly, the case studies included in this study come from secondary sources, which does not allow us to deepen the knowledge on the themes with the use of interviews that may evidence the application of both paradigms. Furthermore, the case studies come from companies with commercial purposes, which may not give a totally unbiased view of the benefits to the organizations or represent very specific groups of the population. Nevertheless, this study adopted external and internal validation mechanisms to reduce this risk of bias. As future work it is recommended that the business view be complemented with a scientific view of the benefits of combining DevOps and Agile and, to this end, a systematic review of the literature in the field can be conducted. Moreover, the qualitative approach used does not allow us to systematically identify all the advantages and make a rigorous quantification of these benefits. As future work, a quantitative study based on a large dataset is suggested to identify the advantages of the combined adoption of both paradigms considering also different sectors of activity of the organizations. It would also be relevant to consider the degree of maturity in the implementation of Agile and DevOps in these organizations and, thus, explore its relevance in the benefits found, since it is expected that some of the benefits may be more easily achieved by organizations with lower levels of maturity in these processes.

Author Contributions: Conceptualization, F.A., J.S. and S.L.; methodology, F.A.; validation, F.A., J.S. and S.L.; formal analysis, F.A.; investigation, F.A., J.S. and S.L. writing—original draft preparation, F.A., J.S. and S.L.; writing—review and editing, J.S. and S.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Dyba, T.; Dingsoyr, T. Empirical studies of agile software development: A systematic review. *Inf. Soft. Tech.* **2008**, *50*, 833–859. [CrossRef]
2. Ergasheva, S.; Kruglov, A. Software Development Life Cycle early phases and quality metrics: A Systematic Literature Review. *J. Physics. Conf. Ser.* **2020**, *1694*, 012007. [CrossRef]
3. Panwar, D.; Tomar, P.; Kumar, P. Innovative methods to make the component-based software development process more effective to produce quality software. *J. Stat. Manag. Syst.* **2017**, *20*, 765–775. [CrossRef]
4. Sommerville, I. *Software Engineering*; India Education Services: Bengaluru, India, 2018.
5. Shore, J.; Warden, S. *The Art of Agile Development*; O'Reilly Media: Newton, MA, USA, 2021.
6. Petersen, K.; Wohlin, C. A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case. *J. Syst. Soft* **2009**, *82*, 1479–1490. [CrossRef]
7. Gregory, P.; Taylor, K. Defining Agile Culture: A Collaborative and Practitioner-Led Approach. In Proceedings of the IEEE/ACM 12th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE), Montreal, QC, Canada, 27 May 2019. [CrossRef]
8. Tolfo, C.; Wazlawick, R.S.; Ferreira, M.G.; Forcellini, F.A. Agile methods and organizational culture: Reflections about cultural levels. *J. Soft Maint. Evol. Res. Pract.* **2011**, *23*, 423–441. [CrossRef]
9. Junker, T.L.; Bakker, A.B.; Gorgievski, M.J.; Derks, D. Agile work practices and employee proactivity: A multilevel study. *Hum. Relat.* 2021; *in press*. [CrossRef]
10. Sweetman, R.; Conboy, K. Portfolios of Agile Projects: A Complex Adaptive Systems' Agent Perspective. *Proj. Manag. J.* **2018**, *49*, 18–38. [CrossRef]
11. Brink, T. Managing uncertainty for sustainability of complex projects. *Int. J. Manag. Proj. in Bus.* **2017**, *10*, 315–329. [CrossRef]
12. Luz, W.P.; Pinto, G.; Bonifácio, R. Building a collaborative culture: A grounded theory of well succeeded devops adoption in practice. In Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, Oulu, Finland, 11–12 October 2018.
13. Leite, L.; Rocha, C.; Kon, F.; Milojicic, D.; Meirelles, P. A Survey of DevOps Concepts and Challenges. *ACM Comp. Surv.* **2019**, *52*, 127–162. [CrossRef]
14. Rajapakse, R.N.; Zahedi, M.; Babar, M.A.; Shen, H. Challenges and solutions when adopting DevSecOps: A systematic review. *Inf. Soft Tech.* **2022**, *141*, 106700. [CrossRef]
15. Wiedemann, A.; Wiesche, M.; Gewald, H.; Krcmar, H. Understanding how DevOps aligns development and operations: A tripartite model of intra-IT alignment. *Eur. J. Inf. Syst.* **2020**, *29*, 458–473. [CrossRef]
16. Jabbari, R.; bin Ali, N.; Petersen, K.; Tanveer, B. Towards a benefits dependency network for DevOps based on a systematic literature review. *J. Soft: Evol. Proc.* **2018**, *30*, e1957. [CrossRef]
17. Joby, P. Exploring DevOps: Challenges and Benefits. *J. Inf. Tech. Dig. World* **2019**, *1*, 27–37. [CrossRef]
18. Hemon, A.; Lyonnet, B.; Rowe, F.; Fitzgerald, B. From Agile to DevOps: Smart Skills and Collaborations. *Inf. Syst. Front.* **2020**, *22*, 927–945. [CrossRef]
19. Melgar, A.S.; Osores, J.; Osores, R.; Relaiza, H.R.; Flores, J.A.; Orihuela, V.H.; Lozano, R.A. DevOps as a culture of interaction and deployment in an insurance company. *Turk. J. Comp. Mat. Educ.* **2021**, *12*, 1701–1708. [CrossRef]
20. Hammond, P.; Allspaw, J. 10+ Deploys Per Day: Dev and Ops Cooperation at Flickr [Video]. 25 June 2009. Available online: https://www.youtube.com/watch?v=LdOe18KhtT4 (accessed on 28 December 2021).
21. Frederic, P. The Incredible True Story of How DevOps Got Its Name [Web Log Message]. 6 May 2014. Available online: https://newrelic.com/blog/nerd-life/devops-name (accessed on 28 December 2021).
22. Fitzpatrick, L.; Dillon, M. The Business Case for Devops: A Five-Year Retrospective. *Cutter. IT J.* **2011**, *24*, 19–27.
23. Wiedemann, A.; Forsgren, N.; Wiesche, M.; Gewald, H.; Krcmar, H. Research for Practice: The DevOps Phenomenon. *Com. ACM* **2019**, *62*, 44–49. [CrossRef]

24. Stahl, D.; Mårtensson, T.; Bosch, J. Continuous Practices and Devops: Beyond the Buzz, What Does it All Mean? In Proceedings of the 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA), Vienna, Austria, 30–31 August 2017. [CrossRef]
25. Larman, C.; Basili, V.R. Iterative and Incremental Development: A Brief History. *Computer* **2003**, *36*, 47–56. [CrossRef]
26. Tozzi, C. 5 Problems with DevOps [Web Log Message]. 12 January 2021. Available online: https://www.itprotoday.com/devops-and-software-development/5-problems-devops (accessed on 28 December 2021).
27. Lwakatare, L.E.; Kuvaja, P.; Oivo, M. Relationship of DevOps to Agile, Lean and Continuous Deployment. In *Product-Focused Software Process Improvement*; Abrahamsson, P., Jedlitschka, A., Nguyen Duc, A., Felderer, M., Amasaki, S., Mikkonen, T., Eds.; Lecture Notes in Computer Science; Springer International Publishing: Cham, Switzerland, 2016; pp. 399–415. [CrossRef]
28. Wang, C.; Liu, C. Adopting DevOps in Agile: Challenges and Solutions. Adopting DevOps in Agile: Challenges and Solutions. 29 June 2018. Available online: http://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1228684&dswid=5071 (accessed on 2 January 2022).
29. Galup, S.; Dattero, R.; Quan, J. What Do Agile, Lean, and ITIL Mean to DevOps? *Com. ACM* **2020**, *63*, 48–53. [CrossRef]
30. Hema, V.; Thota, S.; Kumar, S.N.; Padmaja, C.; Krishna, C.B.; Mahender, K. Scrum: An Effective Software Development Agile Tool. *IOP Conf. Ser. Mater. Sci. Eng.* **2020**, *981*, 022060. [CrossRef]
31. Santos, P.S.; Beltrão, A.C.; Souza, B.P.; Travassos, G.H. On the benefits and challenges of using kanban in software engineering: A structured synthesis study. *J. Soft Eng. Res. Dev.* **2018**, *6*, 13. [CrossRef]
32. Fojtik, R. Extreme Programming in development of specific software. *Procedia Comput. Sci.* **2011**, *3*, 1464–1468. [CrossRef]
33. Sani, A.; Arbain, A.F.; Jeong, S.R.; Ghani, I. A Review on Software Development Security Engineering using Dynamic System Method (DSDM). *Int. J. Comp. Applic* **2013**, *69*, 33–44. [CrossRef]
34. Mousaei, M.; Gandomani, T.J. DevOps Approach and Lean Thinking in Agile Software Development: Opportunities, Advantages, and Challenges. *J. Soft Eng. Int. Syst.* **2020**, *5*, 1–10.
35. Hamunen, J. Challenges in Adopting a Devops Approach to Software Development and Operations. 23 July 2016. Available online: https://aaltodoc.aalto.fi/handle/123456789/20766 (accessed on 2 January 2022).
36. Marnewick, C.; Langerman, J. DevOps and Organizational Performance: The Fallacy of Chasing Maturity. *IEEE Soft* **2021**, *38*, 48–55. [CrossRef]
37. Subramanian, A.; Krishnamachariar, P.K.; Gupta, M.; Sharman, R. Auditing an Agile Development Operations Ecosystem. In *Research Anthology on Agile Software, Software Development, and Testing*; International Management Association, Ed.; IGI Global: Hershey, PA, USA, 2022; pp. 1154–1176. [CrossRef]
38. Faustino, J.; Pereira, R.; Alturas, B.; Silva, M.M.D. Agile Information Technology Service Management with DevOps: An Incident Management Case Study. *Int. J. Agile Syst. Manag.* **2020**, *13*, 339–389. [CrossRef]
39. Dörnenburg, E. The Path to DevOps. *IEEE Soft* **2018**, *35*, 71–75. [CrossRef]
40. Céspedes, D.; Angeleri, P.; Melendez, K.; Dávila, A. Software Product Quality in DevOps Contexts: A Systematic Literature Review. In *Trends and Applications in Software Engineering*; Mejia, J., Muñoz, M., Rocha, Á., Calvo-Manzano, J.A., Eds.; Advances in Intelligent Systems and Computing; Springer International Publishing: Cham, Switzerland, 2020; pp. 51–64. [CrossRef]
41. Nybom, K.; Smeds, J.; Porres, I. On the Impact of Mixing Responsibilities Between Devs and Ops. In *Agile Processes, in Software Engineering, and Extreme Programming*; Sharp, H., Hall, T., Eds.; Lecture Notes in Business Information Processing; Springer International Publishing: Cham, Switzerland, 2016; pp. 131–143. [CrossRef]
42. Merriam, S.B.; Tisdell, E.J. *Qualitative Research: A Guide to Design and Implementation*; John Wiley & Sons: Hoboken, NJ, USA, 2015.
43. Dyba, T.; Prikladnicki, R.; Rönkkö, K.; Seaman, C.; Sillito, J. Qualitative research in software engineering. *Emp. Soft Eng.* **2011**, *16*, 425–429. [CrossRef]
44. Braun, V.; Clarke, V. *Thematic Analysis: A Practical Guide*; SAGE Publications: Thousand Oaks, CA, USA, 2021.
45. Danesh, A.S.; Saybani, M.R.; Danesh, S.Y. Software release management challenges in industry: An exploratory study. *Afri. J. Bus. Manag.* **2011**, *5*, 8050–8056. [CrossRef]
46. Ogheneovo, E. Software Dysfunction: Why Do Software Fail? *J. Comp. Commun.* **2014**, *2*, 25–35. [CrossRef]
47. Fabro, V. The Unified Value of Agile and DevOps. 14 December 2020. Available online: https://www.insight.com/en_US/content-and-resources/tech-journal/winter-2020/the-unified-value-of-agile-and-devops.html (accessed on 5 January 2022).
48. Hemon-Hildgen, A.; Rowe, F.; Monnier-Senicourt, L. Orchestrating automation and sharing in DevOps teams: A revelatory case of job satisfaction factors, risk and work conditions. *Eur. J. Inf. Syst.* **2020**, *29*, 474–499. [CrossRef]
49. Ali, N.; Daneth, H.; Hong, J.E. A hybrid DevOps process supporting software reuse: A pilot project. *J. Soft Evol. Proc.* **2020**, *32*, e2248. [CrossRef]
50. DeFranco, J.F.; Laplante, P.A. Review and Analysis of Software Development Team Communication Research. *IEEE Trans. Prof. Commun.* **2017**, *60*, 165–182. [CrossRef]
51. Schmutz, J.B.; Meier, L.L.; Manser, T. How effective is teamwork really? The relationship between teamwork and performance in healthcare teams: A systematic review and meta-analysis. *BMJ Open* **2019**, *9*, e028280. [CrossRef] [PubMed]
52. Cois, C.A.; Yankel, J.; Connell, A. Modern DevOps: Optimizing software development through effective system interactions. In Proceedings of the IEEE International Professional Communication Conference (IPCC), Pittsburgh, PA, USA, 13–15 October 2014. [CrossRef]

53. Kumar, N.; Gondkar, R. Role of ITOps in DevOps. In Proceedings of the International Conference on Innovative Computing & Communication (ICICC), New Delhi, India, 19–20 February 2021.
54. Reifer, D. Is Merging Agile and DevOps Worth the Pain? 17 January 2019. Available online: https://www.cutter.com/article/merging-agile-and-devops-worth-pain-501791 (accessed on 5 January 2022).
55. Ozanich, A. DevOps Lifecycle vs Agile Methodology: Learning the Difference. 18 November 2021. Available online: https://blog.hubspot.com/website/devops-vs-agile (accessed on 5 January 2022).
56. Ebert, C.; Gallardo, G.; Hermantes, J.; Serrano, N. DevOps. *IEEE Soft* **2016**, *33*, 94–100. [CrossRef]
57. Luz, W.P.; Pinto, G.; Bonifácio, R. Adopting DevOps in the real world: A theory, a model, and a case study. *J. Syst. Soft* **2019**, *157*, 110384. [CrossRef]
58. Clavier, P.; Kaminski, A. How We Applied a DevOps Mindset to Manage Our People Data. 15 January 2021. Available online: https://tdwi.org/articles/2021/01/15/biz-all-apply-devops-mindset-to-manage-people-data.aspx (accessed on 7 January 2022).
59. Venugopal, D. DevOps: Driving Innovation with Old Habits. 1 September 2020. Available online: https://devops.com/devops-driving-innovation-with-old-habits/ (accessed on 7 January 2022).