WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Achieving Continuous Delivery with DevOps Automation and Agile Methodology

***Sarthak Srivastava*[1]**

[1]*Visa Inc, United States*

A B S T R A C T

Recently, there has been a growing interest in the collaboration between technology and IT activities as part of the DevOps phenomenon. This interest is shared among software engineering practitioners and scientists. DevOps is closely associated with agile and continuous software development delivery approaches, making it increasingly important in the field. The study of DevOps explores its roots, acceptance, integration, and priorities within the context of agile, lean, and ongoing delivery strategies in software development. The development of agile software based on lean principles has greatly influenced the emergence of the DevOps phenomenon. Successful implementation of DevOps requires an agile approach to software development. DevOps involves end-to-end automation in the development and delivery of software. While there is no one-size-fits-all approach to DevOps, most developers can benefit from improving the connectivity between previously isolated creation and activity silos. Although agile software development methodologies are becoming more common, many organizations struggle to achieve a frequent release rate, primarily due to departmental silos. In order to overcome these challenges, organizations are turning to DevOps to break down these silos. As digitization continues, companies are increasingly adopting DevOps practices. The transition from agile to DevOps can be divided into three phases: agile, continuous integration, and continuous delivery. Through a comprehensive case study conducted in an enterprise with a long history of DevOps implementation, we have identified a fundamental disruption in the soft skills and communication patterns that software teams are expected to possess.

Keywords: DevOps, Automation, software engineering, configuration management, Agile methodology, Continuous Integration, Continuous Delivery.

1. Introduction

The distinction between DevOps and Agile methodologies can be quite perplexing, partly due to the loose definitions provided by many marketers, which dilutes their precise meanings. However, it is not solely the ambiguity in marketing that leads to misunderstandings and confusion. The fact that DevOps and Agile focus on principles rather than specific practical activities also contributes to this misunderstanding [1].

In a software project, the execution of tasks to achieve valuable outcomes is paramount. Project management plays a crucial role in the planning, implementation, and monitoring of these tasks. As the business environment in software and product design evolves rapidly, successful projects are distinguished by effective project planning, evaluation, improvement management, and quality control. Agile methods have been widely adopted worldwide to address these challenges [1]. Software developers increasingly embrace agile development, enhancing project productivity and meeting the competitive demands of their customers. Agile methodology has emerged as an alternative to the traditional model, providing project teams with multiple options during the development process.

The methodologies for software development are continually evolving, with the application development process being handled in shorter iterations (sprints) and involving team members being modified through agile and DevOps methodologies [2]. This study aims to conduct a qualitative analysis of the responses to DevOps and agile methodologies. The primary focus will be on conducting a literature review to explore the findings of other scholars regarding DevOps automation and Agile methodology and their impact on the current information technology industry. With this research, a better understanding of the efficiency of information systems can be achieved, thanks to the contributions of DevOps.

2. Literature Review

Laanti, Salo, and Abrahamsson conducted an analysis highlighting the prevalent use of Agile and DevOps in many organizations, but also the frequent overlap or ambiguity between the two concepts [2]. While they are implemented beyond software development into organizational units, it is important to note that Agile and DevOps are not identical nor are they adversaries. Instead, leveraging Agile and DevOps in conjunction is often the most appropriate approach for teams, departments, or entities seeking to drive change. Embracing these methodologies requires flexibility in adapting to continuous change and recognizing that no single solution can address all organizational needs.

2.1 An Agile methodology

According to Laanti, Salo, and Abrahamsson, software developers are well aware of the limitations posed by heavyweight processes like the Waterfall method. Consequently, they seek ways to streamline software development and make it more agile [2]. One approach they have adopted is to incorporate end-user feedback more frequently to ensure they are on the right path. In the 1990s, numerous lightweight methodologies, including well-known methods like Scrum and Kanban, emerged for software development. The Agile Software Development Manifesto, established in 2001, formalized many of these concepts and positioned them within the framework of agile software advancement.

Pedrycz outlines the key values upheld by agile methodologies [3]:

a) People: Emphasizing the importance of interactions among teammates, clients, and stakeholders over tools and processes.

b) Immediacy: Prioritizing functional applications over exhaustive documentation.

c) Flexibility: Embracing and responding to change rather than rigidly adhering to predetermined plans.

Unlike the Waterfall approach, which focused on delivering a "finished product," agile methodology acknowledges that software development should be iterative and incremental. This means that with each new software release, customers can expect the implementation of new features or improvements to existing functions. Agile methodologies facilitate this by breaking down the design of software into manageable units known as "user stories." According to Pedrycz, this underscores Agile's customer-centric approach, enabling developers to gather feedback more rapidly and align the product with business needs [3].Agile methodology also promotes adaptive planning, evolving development, early and continuous delivery, and ongoing improvement. These aspects empower developers to respond promptly and flexibly to customer requirements, software complexities, and other external factors.

2.2 The Modern DevOps

Agile methodology emerged as a response to the limitations of waterfall methodologies, whereas DevOps was not specifically developed as a solution to Agile. Although these two concepts are distinct, businesses have started to recognize the benefits of using them together, leading to improved performance and efficiency. Within the modern business landscape, two critical areas have gained prominence: IT operations (ITOps) and Development Operations (DevOps). ITOps focuses on ensuring safety, compliance, and reliability, while DevOps is responsible for designing and delivering new products to end-users. While ITOps provides reliability and security across the network to meet customer needs, DevOps complements it with flexibility, thorough analysis, and effective communication during the implementation of new software.

Over the past decade, organizations have made efforts to transition from traditional IT teams to dedicated DevOps groups or introduce agile methodologies to their software development projects, as stated by Pedrycz [3]. Through these organizational changes, multiple connections between Agile and DevOps have surfaced. Agile teams emphasize automated design, test automation, and Continuous Integration (CI). DevOps teams often rely on these tools, along with metrics and monitoring systems, configuration management, virtualization, and cloud computing, as highlighted by Forsgren and Humble [4]. Agile appeared as a transformative paradigm for software developers who were dissatisfied with the limitations of the waterfall approach. However, Agile is not without its challenges. Forsgren and Humble identify common issues with Agile planning, including missed deadlines, incompatibility between completed elements due to separate scrum or groups, and the splitting of old features when adding new ones due to a lack of coordination with DevOps and ITOps [4]. These challenges can all be traced back to breakdowns in communication.Forsgren and Humble suggest that DevOps addresses this gap by emphasizing communication within its own realm and across other departments, as developers and operators need to coordinate closely [4]. DevOps works in collaboration with ITOps to maintain secure and stable testing environments, and it also makes sense to involve other departments, such as marketing, in deploying new applications to enhance customer service. Supporters of using both Agile and DevOps in appropriate business contexts consider DevOps as an extension of the Agile methodology. While Agile focuses on cross-functional teams typically comprising designers, testers, and developers [5], DevOps takes it a step further by introducing an operator who facilitates the transition between software development and implementation. With its inherent focus on communication across teams, DevOps helps automate processes and promotes transparency for all teams involved.

3. DevOps Vs Agile

The Once the background and contextual dimensions are clear, it becomes crucial to examine the comparisons between DevOps and agile methodologies [6]. Agile methodology is a software development approach that enhances the feedback loop between users and software developers, while DevOps encompasses a set of cultures, practices, and tools that break down silos and promote collaboration among teams. Although DevOps and agile methodologies share certain similarities, they are not identical, and some individuals argue that DevOps is superior to agile methodology. To gain a deeper understanding and eliminate any ambiguity, it is essential to delve into the specifics. By analyzing the differences at a higher level, we can gain insights into the distinct areas in which these two methodologies operate.

3.1 Differences

The distinctions between agile and DevOps methodologies can be summarized as follows:

a) Agile methodology is primarily a project management strategy, whereas DevOps focuses on optimizing the software development pipeline.

b) Agile emphasizes flexibility in requirements and feature development, while DevOps places greater emphasis on continuous integration and software deployment [8].

c) Agile methodologies often align with specific frameworks like Scrum, DaD, LeSS, and SAFe, whereas DevOps does not necessarily adhere to any particular framework. d) Agile methodology prioritizes operations, while DevOps focuses on operational efficiency and automation.

The divergence between the two methodologies becomes evident after the software development phase. In both DevOps and Agile, software is developed, tested, and deployed. However, pure agile tends to conclude after these three stages, whereas DevOps encompasses ongoing operations [9]. In DevOps, the monitoring and enhancement of software are continuous. DevOps takes on the responsibility of engineering, where technology is treated as a business and software is treated as a business asset. DevOps is closely associated with cost reduction, while agile methodology is more aligned with lean principles and waste reduction. Terms such as agile project management and minimum viable product (MVP) hold significance in agile methodology. Agile methodology emphasizes empiricism (adaptation, transparency, and inspection) rather than predictive steps [10]. Furthermore, a well-managed and automated production pipeline serves little purpose if it doesn't deliver value to clients [11]. The confusion surrounding DevOps and Agile arises from the overlapping fundamental principles. Terms like "continuous delivery" can be associated with both methodologies, highlighting the importance of collaboration, speed, and feedback loops. It is crucial to comprehend their distinctions, how they can complement each other, and gain clarity on their unique contributions [12].

Agile methodology primarily focuses on software development and deployment, while DevOps incorporates IT into the equation. Both methodologies play vital roles in software development [13]. Agile methodology has been in existence for over 20 years, whereas DevOps is a more recent discovery.

3.2 Similarities

Both DevOps and Agile methodologies revolve around the goal of developing software quickly without negatively impacting customers or operations. They share a common belief in the importance of rapid software development. Speed and consistency are key principles emphasized by both approaches [14]. Both methodologies are highly adaptable and can be integrated into various business models and industries. DevOps provides the underlying organizational culture and infrastructure from a technological perspective, while Agile enables teams to prioritize flexibility, speed, and high-quality delivery of business value to customers [15]. The complementary nature of these methodologies becomes apparent when considering the importance of well-informed project stakeholders and successful sprints, which rely on a solid foundation in both development and production contexts [16]. Conversely, Agile methodologies have also been implemented effectively in large-scale projects.

Agile methodology was developed to assist businesses in meeting current needs and fostering a positive and creative work environment. However, there is limited documentation in the literature on how Agile strategies interact with the Organizational Process Improvement Program [18]. Further advancements in technology and innovation can be shaped by corporate business models, concurrent design, multisectoral management, and proactive development [19]. When experts were asked about the major research issues in their disciplines during the 2010 XP meeting in Trondheim, Norway, the integration of Agile methodologies into large-scale projects was among the top concerns mentioned.

4. Challenges: Agile Approach

Although Agile methodologies are well-suited for small teams with direct involvement, incorporating these practices in large multi-site, multi-customer, and multi-project collaborations faces several challenges [20]. A comprehensive list of transitional challenges and nearly forty clear barriers to Agile adoption in large groups has been identified. These obstacles often stem from the complexity of such environments and the inherent friction between Agile and traditional organizations. Examples include conflicts between design processes, limitations in modifying subsystems, complex life cycles, and difficulties in aligning Agile practices with legacy frameworks. To overcome these challenges, Agile design approaches must be customized to fit existing procedures within the enterprise [21].

5. Agile and DevOps Culture

While Agile methodologies may not always directly lead to DevOps, they do bring about profound cultural changes within organizations. Adopting an Agile approach encourages a shift in development thinking [22]. Instead of viewing growth as a cumbersome process, Agile thinking promotes small, incremental changes that accumulate into significant improvements over time [23]. Businesses of all sizes have explored how they can enhance their work using Agile practices. While some companies have successfully embraced Agile, the introduction of DevOps can further drive cultural changes, such as improved communication and a stable balance between transformation and flexibility [24]. The decision to adopt both Agile and DevOps is an active choice that many analysts believe can contribute to effective decision-making and foster a better organizational culture.

6. Conclusion

Agile and DevOps may have correlations that lead people to assume they are the same, but in reality, they are distinct concepts. This misconception does a disservice to both Agile and DevOps. Both methodologies aim to help teams work faster and more efficiently, delivering high-quality work and satisfying customers. DevOps serves as a collaboration-building framework between the Technology and Operations teams, enabling rapid and automated

deployment of code to production. It accelerates product and service delivery across the organization. While Agile and DevOps share the goal of delivering value to end-users, they approach it from different angles. Agile focuses on improving efficiency for developers and release schedules, while DevOps brings together the operations team to enable continuous integration and delivery [25]. DevOps has also fostered innovative thinking beyond Agile development, incorporating multi-functional (DevOps) and end-to-end processes that impact software companies and customers. Many organizations have successfully embraced Agile practices to expedite application development, moving away from traditional waterfall strategies. Agile has gradually expanded into other areas within the technology organization, including IT and operations. Teams have become more streamlined, processes simplified, feedback loops improved, and IT departments have gained speed, all of which have a profound impact on the entire organization. To further enhance performance, DevOps and Continuous Delivery (CD) have been introduced to support and enhance agility, responsiveness, and fast time-to-market throughout the software delivery lifecycle. However, successful implementation of these methodologies requires a deep understanding by software teams. Agile and DevOps are not adversaries; rather, they are allies in the agile revolution. They can coexist and work together inclusively or exclusively in the same space.

References

[1] Dingsøyr, T., Nerur, S., Balijepally, V., & Moe, N. B. (2012). A decade of agile methodologies: Towards explaining agile software development. Journal of Systems and Software, 85(6), 1213-1221.

[2] Laanti, M., Salo, O., & Abrahamsson, P. (2011). Agile methods rapidly replacing traditional methods at Nokia: A survey of opinions on agile transformation. Information and Software Technology, 53(3), 276-290.

[3] Pedrycz, W. (2006). Quantitative logic-based framework for agile methodologies. Journal of Systems Architecture, 52(11), 700-707.

[4] Forsgren, N., & Humble, J. (2015). DevOps: Profiles in ITSM Performance and Contributing Factors. SSRN Electronic Journal.

[5] Dzamashvili Fogelström, N., Gorschek, T., Svahnberg, M., & Olsson, P. (2010). The impact of agile principles on market-driven software product development. Journal of Software Maintenance and Evolution: Research and Practice, 22(1), 53-80.

[6] Drury, M., Conboy, K., & Power, K. (2012). Obstacles to decision making in Agile software development teams. Journal of Systems and Software, 85(6), 1239-1254.

[7] Forsgren, N., & Humble, J. (2015). DevOps: Profiles in ITSM Performance and Contributing Factors. SSRN Electronic Journal.

[8] Amorim, A., Mira da Silva, M., Pereira, R., & Gonçalves, M. (2020). Using agile methodologies for adopting COBIT. Information Systems, 101496.

[9] Kamel, M., Bediwi, I., & Al-Rashoud, M. (2010). Planned Methodologies vs. Agile Methodologies under the Pressure of Dynamic Market. Journal of King Abdulaziz University-Engineering Sciences, 21(1), 19-35.

[10] Spinellis, D. (2016). Being a DevOps Developer. IEEE Software, 33(3), 4-5.

[11] Saiedian, H., & Dale, R. (2000). Requirements engineering: Making the connection between the software developer and customer. Information and Software Technology, 42(6), 419-428.

[12] Saeeda, H., Arif, F., Mehmood Minhas, N., & Humayun, M. (2015). Agile Scalability for Large Scale Projects: Lessons Learned. Journal of Software, 10(7), 893-903.

[13] Wettinger, J., Breitenbücher, U., Falkenthal, M., & Leymann, F. (2016). Collaborative gathering and continuous delivery of DevOps solutions through repositories. Computer Science - Research and Development, 32(3-4), 281-290.

[14] Sidky, A., Arthur, J., & Bohner, S. (2007). A disciplined approach to adopting agile practices: The agile adoption framework. Innovations in Systems and Software Engineering, 3(3), 203-216.

[15] Petersen, K., & Wohlin, C. (2009). A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case. Journal of Systems and Software, 82(9), 1479-1490.

[16] Tan, C., & Teo, H. (2007). Training future software developers to acquire agile development skills. Communications of the ACM, 50(12), 97.

[17] Ståhl, D., & Bosch, J. (2014). Modeling continuous integration practice differences in industry software development. Journal of Systems and Software, 87, 48-59.

[18] Turley, R. T., & Bieman, J. M. (1994). Identifying essential competencies of software engineers. In ACM Conference on computer science (pp. 271-278).

[19] Ambler, S., & Lines, M. (2012). Disciplined agile delivery. IBM Press, Upper Saddle River, N.J.

[20] Hill, P. (2011). Practical software project estimation. McGraw-Hill, New York.

[21] Hüttermann, M. (2012). DevOps for developers. Berkeley, CA: Apress.

[22] Gregory, J., & Crispin, L. (2015). More agile testing. Addison-Wesley, Upper Saddle River, N.J.

[23] Novak, I. (2012). Beginning Windows 8 application development. Wiley, Indianapolis, IN.

[24] Rico, D., Sayani, H., Sone, S., & Safari, an O'Reilly Media Company (2009). The Business Value of Agile Software Methods. J. Ross Publishing.

[25] Conboy, K., Coyle, S., Wang, X., & Pikkarainen, M. (2011). People over Process: Key Challenges in Agile Development. IEEE Software, 28(4), 48-57.