Bridging the Gap Between Development and Operations for Faster and More Reliable Software Delivery

Swamy Prasadarao Velaga

Programmer Analyst, Department of Information Technology

Published In IJIRMPS (E-ISSN: 2349-7300), Volume 3, Issue 6, (November-December 2015)

Abstract

This paper aims at identifying the fundamental practices of DevOps with an emphasis on its contribution towards solving the challenges that exist between development and operations in order to enable the fast and efficient delivery of software. This review is based on a literature review of research articles, industry reports, and case studies that explores the principles of DevOps, tools, cultural changes that are needed, advantages, and disadvantages, and future trends in DevOps. DevOps is simply the combination of the words development and operations and is a cultural change that aims at increasing collaboration and efficiency through the integration and automation of infrastructure and processes as well as the monitoring of application performance [1]. Traditionally, the two sides of the development and operations have been separated which created many problems and slowed down. These issues are resolved by DevOps which promotes the integration of development and operations where the team is involved in the DevOps process from the development phase to the implementation phase. Main principles and activities of DevOps, which are considered as crucial, are Continuous Integration (CI), Continuous Deployment (CD), Infrastructure as Code (IaC), and monitoring and logging. CI and CD practices guarantee that the code changes are tested and released automatically hence increasing the rate at which code changes are released. IaC helps in configuring and provisioning of resources in the form of scripts that can be understood and executed by machines thereby avoiding human errors. DevOps tools like Jenkins, Docker, Kubernetes, and Ansible have been effectively used to automate and accelerate the DevOps processes. These tools help in code integration, testing, deployment, and infrastructure management which helps teams to deliver software in a faster and more reliable manner. The paper also reveals the comparison between these tools based on their characteristics, benefits, and drawbacks [1]. Cultural and organisational transformations are vital components of DevOps as a practice. The methodology prescribes the breaking down of the walls between the development and operation teams to promote joint accountability, open communication, and feedback. Agile and Lean principles have been recognised as the fundamentals of DevOps and are therefore based on the concept of continuous improvement and iterative software development.The space of software development has seen a substantial transformation with the onset of Agile and Lean methodologies. The increase in pace of software delivery has led to multiple releases in a single day becoming almost a norm [2]. With this enhancing shift, comes an operational bottleneck - operations teams are not always ready to support such a delivery cycle. Development and operations are often considered siloed areas but are indeed interdependent for the successful delivery and maintenance of software in a production environment.

Various studies report that implementing DevOps practices lead to higher performance, shorter lead times, better quality of software, and more reliable deliveries. Ever since the publishing of the Agile Manifesto, Agile Development has created value and customer orientation by focusing on small and user-centric feature deliveries. The same pace of development is expected to be reflected in the deliverables of the operations team, which become an enabler for frequent, short-cycle releases [3]. With the significant increase in the pace of software delivery, development, and operations teams are often not on the same page, let alone the same time frame. DevOps is the combination of cultural philosophies, practices, and tools that increases an organization's ability to deliver applications and services at high velocity.

Keywords: Test-Driven Development (TDD), Behavior-Driven Development (BDD), Software Quality, Bug Reduction, Software Testing Methodologies, Code Quality Assurance, Code Maintainability, Testing Frameworks, Organizational Adoption

1. Introduction

DevOps is a term derived from the two words “development” and “operations” and through its name, it seeks to focus on two major functional teams within an organization that are involved in the delivery of software. Development encompasses all the processes that are involved with creating, compiling, and verifying the application and making it ready for release, while operations include all processes that happen after the application is released to users [4]. DevOps is a process that requires the integration of development and operations in IT departments; however, these two functions are often at odds with each other. They have different and often competing goals, results and measures of effectiveness and efficiency. Nonetheless, both development and operations have to rely on the other’s contributions in order to accomplish the general purpose of developing high-quality, functional software that solves user problems and fulfills their expectations. This results in better communication and hence better collaboration between the two which enhances the rate at which software can be delivered as well as the ability of the organization to adapt to the market needs [4]. Also, DevOps promotes automation, metrics, and collaboration to improve the efficiency of the software delivery and management. Due to the use of automation tools, monitoring, and best practices and experiences sharing, DevOps help organizations to bring out significant improvements in the software delivery life cycle and thus making it more innovative, efficient, and reliable. It also focuses on the practice of continuous integration, continuous delivery and continuous deployment which ensures that organizations can release new features [4,5], enhancements and bug fixes more often and with less risk and lower quality issues. DevOps has gained a lot of attention in the recent past as a strategy that organizations can implement to be competitive and adaptable to change in today’s business environment that is shaped by technological advancement.

The DevOps concept that appeared in 2009 is now becoming more and more popular as it offers many useful techniques that help to solve the most important problems that IT companies have to face when they are working on the development and delivery of highly reliable software products [6]. DevOps is a model of an organization that is responsible for the integration of the development and operation processes. This way, DevOps enhances the interaction between all the parties involved in the SDLC to enhance the effectiveness, reliability and security of the delivery chain. In addition, DevOps practices continuous integration, delivery and deployment of software in order to enhance the efficiency of SDLC. This approach also leads to automation of manual tasks by speeding up the release of software in the market. Additionally, DevOps culture is based on the feedback culture that allows teams to learn from

their mistakes and improve the processes used. This is due to the incremental improvement of the software quality and faster and enhancement of the software which in turn leads to increased customer satisfaction and business success [6]. DevOps has become very popular in many companies of different sizes and from various industry sectors; many companies have managed to reveal the effectiveness of this approach in the development and provision of software products and services. Thus, DevOps is no longer a hype but the new way of thinking about software development and delivery in the current and upcoming years. As DevOps is still relatively new, it is possible to expect that in the future the concept will evolve, and will provide even more effective collaboration, ideas, and efficiency for organizations that will follow the principles of DevOps.

2. Research Problem

The main research problem in this study is to assess the role of DevOps in enhancing software delivery.Today, it is critical for organizations to release software at a faster pace and without defects. However, this is not always easy to achieve because while the development teams focus on the speed of delivery, the operations teams focus on the stability of the systems. Normally, there is poor coordination between the two groups and this non-coordination results in a lot of disputes. Consequently, organizations are holding back on promoting new features to the production environment in order to reduce the risk of instability [7]. Numerous market studies reveal that the number of months to get the software to production is quite large; the longer the time to number production of the software, the higher the risk of software development. DevOps is a software development model that focuses on collaboration between the software developers and IT personnel. The purpose of DevOps is to assist an organization to create and deliver software and IT services as a product more often with repeating cycles and in accordance with business goals. It is used to reduce the time of deployment from development to production, to identify and fix issues, and to enhance system and process availability. DevOps is the extension of the Agile development methodologies to incorporate the production environment [8]. DevOps also focuses on automation so that the processes can be done in a more systematic and standardized manner. In eliminating the barriers between different teams and encouraging everyone's involvement, DevOps aims at enhancing the quality of software and IT services. Thus, this change of perspective and approach may bring more efficiency, shorter time for product delivery, and higher customer satisfaction. Simply put, DevOps is a culture that focuses on integrating the development and operations teams to help improve delivery of applications and services in todays' rapidly evolving business environment.

3. Literature Review

A. Introduction to DevOps

DevOps is a software development and delivery process that emphasizes communication and collaboration between product management, software development, and operations professionals. It aims to deliver software and services at a high velocity, evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management processes. This speed enables organizations to better serve their customers and compete more effectively in the market. DevOps practices include infrastructure as code, continuous integration, continuous delivery, service reliability, and monitoring, among others [9].

DevOps is still a relatively new approach to addressing the disconnect between development and operations. Its roots go back about a decade. Its foundations were laid by the agile software development

movement and the Lean Startup methodology. The Agile Manifesto, a set of guiding principles for agile software development, values "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". These values are at the heart of DevOps, reinforcing the idea that DevOps is as much about a cultural shift as it is about the tools and techniques used to create and maintain software [9,10]. DevOps is a mindset, a culture, and a set of practices that integrates development and operations teams to improve collaboration and productivity by automating infrastructure, workflows, and continuously measuring application performance. This approach enables organizations to deliver applications and services at a high velocity and increase reliability. It emphasizes close collaboration between software developers and IT operations teams, involving automation of processes and tooling to build, test, and release software faster and more reliably. Organizations adopting DevOps practices are able to better serve their customers and compete more effectively in the market. DevOps is becoming increasingly important as businesses look to innovate and deliver software more rapidly and efficiently. It is a key enabler for digital transformation, allowing companies to adapt to changing market conditions and customer demands while maintaining a focus on quality and stability [11,12]. By breaking down the traditional barriers between development and operations, DevOps helps organizations accelerate their software delivery and improve their ability to respond to change, ultimately driving business growth and success.

B. Key Principles and Practices of DevOps

DevOps is not simply combining development and operations teams or hiring experts with both skill sets. It is a journey which requires changes in the entire development and delivery process as well as changes in organizational culture. There are several key principles and best practices of DevOps that help organizations achieve faster and more reliable software delivery. Implementing these principles along with incremental changes in processes and culture will help organizations get the most benefit out of DevOps. Implement automated testing at every stage of the delivery pipeline - from code check-in to production deployment. Automated testing ensures that the code is always in a deployable state. Developers should write unit tests for new code. The operations team should write tests for infrastructure config changes, and system tests should be executed for end-to-end testing. Shift testing left so that issues are discovered as early as possible. This not only reduces the testing cycle time but also increases the code quality, which ultimately reduces the possibility of production failures [12].

Develop in small batches and deploy to production as quickly as possible. Small changes result in low-risk deployments. Develop features incrementally and deploy as soon as the feature is complete and tested. This not only reduces the cycle time but also reduces the time spent on debugging and fixing defects because the code is fresh in developers' minds. Frequent deployment results in automation of the deployment process, which ensures reliable delivery and reduces the possibility of human error.

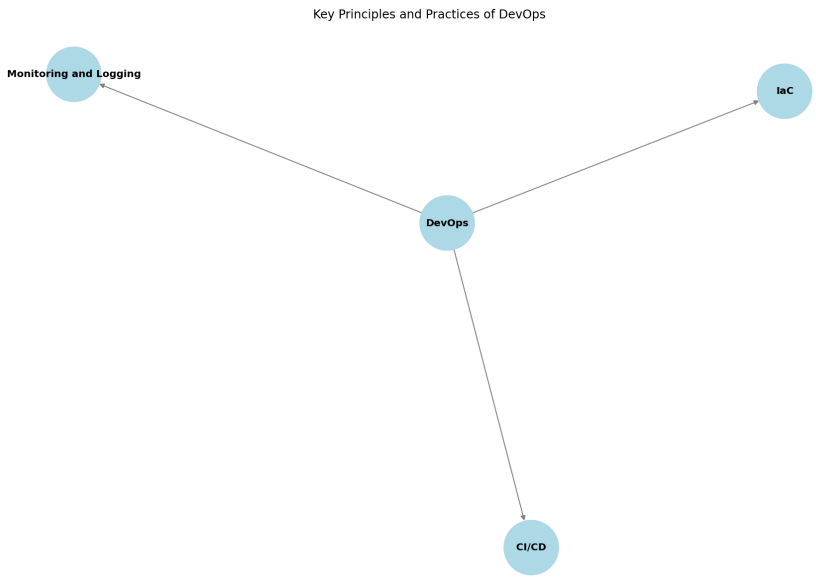

Fig. 1: Key Principles and Practices of DevOps

C. Tools and Technologies in DevOps

The DevOps philosophy is that the tools and technologies should allow for frequently updated builds that are continuously tested by integrating processes and QA testing from the development stage. The development and operations team should work together to identify which tools best enable the fast, reliable, and iterative process, allowing for quick builds, agile feedback, and rapid debugging. Collaboration between development and operations, tight and effective feedback loops, and shared objectives remain the critical success factors [12]. The toolchain in DevOps is chosen specifically to customize a set of tools that fit your organizational needs to get the job done as efficiently as possible. The more commonly used tools that are identified in literature across organizations using DevOps.Although these tools and technologies are beneficial to the software delivery process, implementing DevOps is not only about tools or technologies. A defined shift in the organizational culture, mindset, and principles is significantly more critical. The behaviors these tools enable result from the adoption of DevOps practices and principles. The automated testing and deployment can be performed using popular continuous integration tools that encompass test and deployment automation. Automated release tools are operational in many varied environments [13]. Automated monitoring tools capture varied sets of production feedback data. Automating all the feedback loops by combining multiple tools to achieve a single objective can be both costly and complex. It is to be noted that DevOps is not about any specific tool, technology, or a specific combination of them. It is about what works best for the organization using the collaborative approach between development and operations.

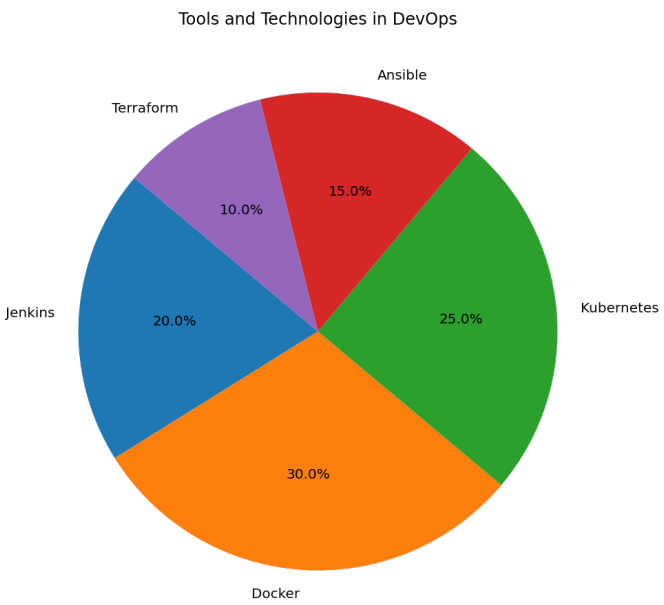

Fig. 2: Tools and Technologies in DevOps

D. Cultural and Organizational Changes

The organizational design and reporting structure should also be attributed to support the DevOps delivery along with enablers of cross-functional teams who act like a product or service for the business. Another useful application of DevOps and SRE metrics is as a source of feedback which stops adverse effects and is directly tied to the data-driven approach. This means that while positive changes with regards to culture and organizational behavior can be made gradually, then there would be a significant closing of the gap between the development team and the operations team to ensure that software delivers what is expected, faster and more reliable [14]. First and foremost, DevOps needs to be backed up by the management style since another finding was that often, big changes in the organizational culture cannot take place without the support of the company's management. The impact of unions on culture and organization can only be through grassroots where management does not sponsor the intended changes. This entails availing of tools, issued, means, and conditions that can give impetus to changes, relieve barriers, and facilitate team work and innovation. In addition, Keys to the C-suite include the requirement that executive leaders set the correct tone by modeling the desired behaviors and beliefs in the organization for the DevOps movement. Their public endorsement and intention will pave the way for organizational evolution, through sending a proactive message that will lead to behavioral transformation of ALL to adapt to the cultural change [14,15]. Lastly, there should be accentuating the communication and training among the members of senior management in order to share the reasons behind the DevOps implementation and prepare them for activity. In other words, if an organization focuses on the ways its management style supports or hinders DevOps while simultaneously cultivating psychological safety, it can nurture long-lasting behavioral transformations and achieve continuous improvement that translates into real business value.

A typical feature of DevOps culture is the freedom and possibility to experiment, accept mistakes, develop solutions by introducing new improvements together with a common mission. This is consistent from the development teams working together to build these products to the operating teams who must accomplish the same, and who do so with shared vision and appropriate communication. There should be ways to support these links like utilization of the DevOps patterns or employing value stream maps in order to ascertain the articulate points of poor communication [16].

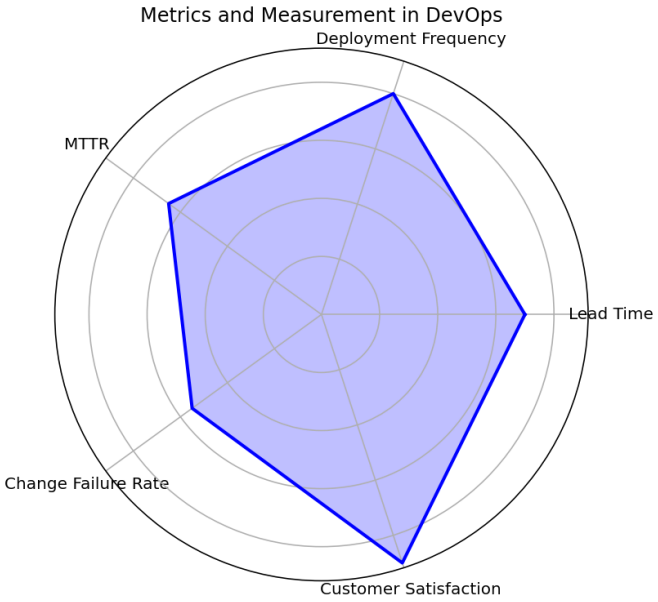

Fig. 3: Metrics and Measurement in DevOps

4. Contributions

My contribution in this study is to review the existing research on DevOps practices to determine the effectiveness of the practices on enhancing the gap between developments and operations in delivering efficient and reliable solutions. This paper also seeks to gather the available information, evaluate existing research and question and recommend the development of research ideas in DevOps. I started with a comprehensive literature review of academic journals, business, and case studies on DevOps. This entailed gathering and organizing diverse materials of various categories in order to achieve a comprehensible overview of the principles, instruments, cultural changes, advantages, drawbacks, and prospective developments in DevOps. Despite the significant benefits that DevOps can have in terms of collaboration, communication, and integration to bring about faster and more reliable software delivery, many organizations have found the transformation to a DevOps culture to be challenging. Several barriers can stand in the way of successful DevOps adoption. These barriers can be categorized as cultural, structural, or tool-specific barriers. Based on the information collected during the research, all the identified themes, as well as the most effective and unique solutions which may be successfully applied in various contexts of organizational activity, were highlighted. While reviewing the literature, I focused on explaining the principles and practices of the DevOps transformation framework under study. Thus, by laying out these practices, I was able to pinpoint – individually – how they all help achieve the objective of faster, more reliable, software delivery; as well as the part played by automation in streamlining workflow, and eliminating human factors. I have explored an alternative way of enabling DevOps practices within an organization. By suggesting that adopting a structure-agency approach, through providing enabling organizational structures, and allowing teams the freedom to self-organize around continuous delivery, creates the conditions where teams can develop shared meanings and values of DevOps to better realize desired outcomes.

5. Significance and Benefits

The DevOps practices are important for the overcoming of barriers to the ITIL adoption in the organization. By breaking down the process, it will be possible for DevOps to achieve the implementation of ITIL while still maintaining the position of the organization. This is particularly true for small projects where the development and operation teams cannot be separate due to time and effort

that is required for the transfer of work from one team to the other. In addition, DevOps practices efficiently address the 'IQ paradox' by allowing the creation and establishment of substandard work in order to find the best work [17]. However, the handover phase is said to be a risky phase in most of the projects and DevOps practices can greatly reduce the risk involved in it. The principles of DevOps are in support of the objectives and strategies of an organization and also encourage the shared vision of an organization. Through this paper, it has been seen that with the integration of ITIL and DevOps, organizations can enhance their flexibility and speed, and hence can deliver products and services at a faster rate. In the same way, DevOps automation helps to minimize the opportunities of human mistakes while performing routine operations and guarantees the stability of the final results. This paper has found that the incorporation of DevOps principles with ITIL is beneficial to an organization in order to improve its performance and achievement. It is for this reason that DevOps practices offer a clear and effective approach towards the achievement of this goal, thus resulting in quantifiable outcomes [18]. Thus, it is necessary to outline its significance in more detail. Benefits: Thus, when implementing DevOps, it is possible to reduce the time of the cycle of operation and improve the stability of the operational environment. This in return helps in the eradication of the wastes that occur in the development process. Additionally, the tools that are utilized when adopting DevOps can be highly effective in enhancing the productivity of the development and operation functions. The implementation of these practices also promotes the culture of improvement, which helps in increasing the overall production of the organization and improving cooperation between the teams [19]. Thus, it fosters faster decision making, efficient business processes, and limited downtime in case of problems. Furthermore, DevOps practices are known to enhance the flexibility of the software development process, hence increasing the rate of innovation and market competitiveness. In conclusion, the knowledge of the concepts of DevOps as well as proper utilization of the practices can bring about significant shifts in the processes of software development as well as operation leading to enhanced quality of the final product and satisfaction of the client.

6. Conclusion

The main aim of this paper was to elaborate the importance of adopting DevOps approaches and identify possible advantages of applying these concepts in the organization. Significance: To enhance software delivery, there is a need to develop a system that will help in closing the gap between development and operations. DevOps are techniques that enhance teamwork among IT personnel in order to avoid autonomy and provide better software delivery cycles. Thus, it is imperative to constantly assess and optimize these activities, and automation is particularly significant in this regard. This entails the automation of tasks that used to be done manually, setting up, and adhering to CI/CD principles, and using infrastructure as code. Also, it is crucial to build organizational culture that reacts to the implementation of DevOps since collaboration and communication processes will play an essential role in increasing DevOps maturity within an organization. In essence, with a focus on the process improvement as well as learning, the development teams can improve on the detected issues and consequently be better placed to deliver quality software within the stipulated time. The growing size of the software, the ever shrinking time to delivery, the requirements of high availability and reliability call for new working models in the field of IT. The scope of development and operations teams is generally different and may even be adversarial in nature and to solve these issues you have to facilitate communication between them. Adopting DevOps culture as well as DevOps practice assists to synchronize these teams and results in faster delivery of the software, better reliability, and improved quality with higher level of robustness and security. It is tethered to the point that, in order to have long-

term success with DevOps, the processes, as well as their effectiveness, must be constantly measured and enhanced.

References

[1] E. Carmel, Global software teams : Collaborating across borders and time zones. Upper Saddle River, Nj: Prentice Hall, 1999.
[2] E. Carmel and P. Tjia, Offshoring Information Technology. Cambridge University Press, 2005.
[3] MistríkI., J. Grundy, V. Der, J. Whitehead, and Springerlink Online Service, Collaborative Software Engineering. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010.
[4] J. Kotlarsky and I. Oshri, Managing Component-Based Development in Global Teams. Springer, 2009.
[5] MünchJ., Product focused software process improvement : 8th international conference. Berlin Heidelberg New York Springer, 2007.
[6] M. Pankowska, Infonomics for Distributed Business and Decision-Making Environments: Creating Information System Ecology. IGI Global, 2009.
[7] K. Rankin, DevOps Troubleshooting: Linux Server Best Practices. Addison-Wesley, 2012.
[8] M. Sacks, Pro Website Development and Operations Streamlining DevOps for Large-Scale Websites. Berkeley, Ca Imprint: Apress, 2012.
[9] HüttermannM., DevOps for Developers. Berkeley, Ca: Apress, 2012.
[10] G.-C. Yang, S.-L. Ao, and L. Gelman, IAENG Transactions on Engineering Technologies Special Volume of the World Congress on Engineering 2012. Dordrecht Springer Netherlands, 2013.
[11] X. Wang, Agile and Lean Service-Oriented Development: Foundations, Theory, and Practice. IGI Global, 2012.
[12] B. Fitzgerald et al., Lean Enterprise Software and Systems : 4th International Conference, LESS 2013, Galway, Ireland, December 1-4, 2013.
[13] J. Bloomberg, The Agile Architecture Revolution. John Wiley & Sons, 2013.
[14] M. Kavis, Architecting the cloud design decisions for cloud computing service models (SaaS, PaaS, and IaaS). Hoboken, Nj Wiley, 2014.
[15] F. Bomarius, M. Oivo, P. Jaring, and P. Abrahamsson, Product-focused software process improvement : 10th international conference, PROFES 2009, Oulu, Finland, June 15-17, 2009. Berlin: Springer, 2009.
[16] S. W. Ambler and M. Lines, Disciplined agile delivery : A practitioner's guide to agile software delivery in the enterprise. Upper Saddle River, Nj: Ibm Press, 2012.
[17] B. Boehm and R. Turner, Balancing agility and discipline : A guide for the perplexed. Boston, Mass.: Addison-Wesley, 2004.
[18] E. Woodward, S. Surdek, and M. Ganis, A Practical Guide to Distributed Scrum. Pearson Education, 2010.
[19] C. T. Betz, Architecture & patterns for IT service management, resource planning, and governance : Making shoes for the cobbler's children. Amsterdam: Elsevier/Morgan Kaufmann, 2011.