The necessity of a lead person to monitor development stages of the DevOps pipeline

Betool Alnamlah, Sara Alshathry, Norah Alkassim, Nor Shahida Mohd Jamail
College of Computer and Information Sciences, Prince Sultan University, Riyadh, Saudi Arabia

Article Info

Article history:

Received Apr 14, 2020
Revised Jun 27, 2020
Accepted Aug 18, 2020

Keywords:

Agile
Deployments
DevOps
Integration
Iterations
Pipeline

ABSTRACT

The development to operations model, known as DevOps, refers to a framework that integrates the development and operation phases of organizational applications. DevOps bridges the gap between these two processes that traditionally prolonged the implementation of these applications. This framework aims at enhancing service delivery and releasing subsequent updates based on real-time customer feedback. Consumers in this modern era prefer tailor-made services, and a slight downtime from one supplier can quickly shift their preference to the company’s competitors. However, DevOps involves sophisticated coding activities, and there is a need for someone to oversee these processes to ensure developers are not overwhelmed. The objective of this survey is to clarify the significance of a lead person to monitor the DevOps pipeline and report any failure detected in each development stage. The study utilized meta-analysis of verified tertiary resources from other researchers that took a deep dive into the DevOps model and how it works towards organizational goals in various firms. All institutions involved in multiple business platforms should take advantage of new technologies to scale up their service delivery and net profit. If adopted and appropriately monitored, DevOps can revolutionize the capabilities of most business applications by minimizing downtimes and providing uninterrupted services.

Corresponding Author:

Betool Alnamlah
Department of Computer and Information Sciences
Prince Sultan University
3349 Rafha St, Salah Ad Din, Riyadh 12435 7413, Saudi Arabia
Email: batool_alnamlah@hotmail.com

1. INTRODUCTION

The transition between application development and implementation phases has received a tremendous boost thanks to the adoption of the DevOps techniques. Organizations strive to establish modern capabilities, with high expectations placed on the latter’s IT department. Although there is a notable technological advancement towards innovation, prolonged lagging has also been witnessed in project development as well as the quality of applications. A considerable number of projects are hindered by missed implementation deadlines, inefficiency, and outages throughout the implementation period [1].

The DevOps framework comes handy as it strives to combine the traditional waterfall development lifecycle into a single-phase process. Developed firms prefer using engineers to implement projects by automating all the processes from system design, and merely engaging the security team when testing the system. The model not only streamlines the entire development lifecycle but also minimizes unnecessary expenses significantly [2].

Agile principles play a significant role in DevOps through continuous integration and continuous delivery to eliminate potential downtimes. A remarkable milestone of this model is the breaking of monotonous silos between the system developers, testers, release managers, and the eventual operators [3]. It also reduces the handover processes as both teams are involved in the entire process.

Having explored the beneficial value and overview of the DevOps model, it is equally essential to make it more efficient. Efficiency is only achievable if the development pipeline is streamlined and technical errors are minimized. The next sections of this research will discuss at length why it is necessary to have a lead person to monitor the entire DevOps lifecycle

Developers are always involved in the coding duties and rarely create time to do comprehensive scrutiny of the final software version. They only rely on feedback from end-users to check for bugs and release new updates. The comments may come with criticism, and by the time the teams act on the information, consumers may have already shifted to competitors [4]. With increased specialization, developers tend to shift their focus and dwell on developing more sophisticated systems rather than assessing the latter's loopholes. Undetected errors may emerge unexpectedly, prompting the team to halt their current tasks and look back into the failure. Such mishaps pose as drawbacks to the company's overall performance and revenue projection. Due to the increasing demand for better products and services, developers work for longer hours to meet these demands and remain ahead of competitors [5].

2. RESEARCH METHOD

The application delivery process in the DevOps pipeline involves specific essential stages before being released to stakeholders and customers. The pipeline initiates at the development stage where the team writes the application code, which is then integrated into a source control respiratory [6]. The integrated code is used in the next step to build the software. The developed system is then tested, and if any failure is detected, it is sent back to developers for fixing. The final step is the deployment of the final version to end-users. All stages in the DevOps pipeline are interconnected by a feedback path to ensure continuous application delivery [7]. There have six DevOps Practices and five benefits of DevOps. Below are the six DevOps Practices [8].

a) Logging and monitoring: Frequent monitoring of organizational applications is done to evaluate the satisfaction rate of end-users. It also helps developers know the kind of updates to release.
b) Continuous integration: It refers to how developers run automated tests and build by merging their codes. The practice aims at scaling software quality by locating and fixing bugs faster.
c) Communication and collaboration: Teams utilize chatting platforms of the application to share the organizational workflow and keep track of their projects.
d) Micro-services: Companies use an HTTP-based application programming interface to integrate lightweight services into one program. Every functional unit in the application serves a specific purpose.
e) Continuous delivery: The process involves uninterrupted building, testing, and releasing code changes. It is usually complemented by constant integration.
f) Infrastructure as code: It is a practice used in managing infrastructure through software and code development techniques such as continuous delivery and integration. Administrators and developers interact through the cloud's Application Programming Interface (API) framework using coded tools.

Below are the 5 Benefits of DevOp:

a) Speed: Continuous delivery and micro-services enable teams to release updates faster, get real-time customer feedback, and cope with radical changes in the market.
b) Reliability: The automation of all the DevOps sectors creates a favorable work environment for collaborators to keep the system updated.
c) The breaking down of silos: The collaboration enhances efficiency by saving the time wasted during handovers. The coding and deployment functionalities are usually treated as a unit.
d) Rapid delivery: Working products are released into the market faster, which enhances the return on investments.
e) Risk management strategy: The team quickly points out any bugs that may hinder the application's success and fixes the loophole before it reaches consumers.

2.1. The absence of a lead person impacts DevOps' success

The DevOps framework attributes its success to the commitment of developers and engineers who take up the responsibility of steering the organization's objectives [9]. If the developers are not well supported and coordinated centrally, the following setbacks are likely to be experienced:

a) Some members may feel overburdened and give themselves unplanned breaks when the demand is too high.

b) There will be improper coordination between developers and the management, and any slight inquiries can interrupt the DevOps processes.
c) Some individuals lack the integrity of working without supervision, and their laxity can drag other members behind.
d) If the developers are being outsourced, some of them may implement applications that violate the company’s policies, leading to a waste of time and resources.

Table 1 show the comparison between existing DevOps roles and DevOps Mentor. There have many roles gaps in DevOps transformational leadership and release manager. Both are not monitor and control the pipeline, not involve in ensuring each pipeline stage reflect business need and don’t have responsible to conduct a meeting between team members [10]. Only DevOps transformational leadership supports the transformation of DevOps. In remove impediments and monitor delivery time from ticket created to delivery are covered from released manager. In this study, the DevOps mentor will cover all of the roles between DevOps transformational leadership and release Manager [11].

Table 1. Comparison between existing DevOps roles and DevOps mentor

	DevOps Transformational leadership	Release Manager	DevOps Mentor
Roles			
Support transformation to DevOps.	✓	x	✓
Monitor and control pipeline.	x	x	✓
Ensure each pipeline stage reflect business need.	x	x	✓
Remove impediments.	x	✓	✓
Monitor delivery time from ticket created to delivery.	x	✓	✓
Conduct meeting with team.	x	x	✓

2.2. Proposed solution

There is a considerable gap between end-users and the DevOps team. Organizations should consider appointing an expert, probably from their developers, to overlook the entire DevOps process. The said expert must possess skills and vast experience in DevOps pipeline stages. They will be tasked with giving a green light for each development stage, as well as suggesting a revision of specific coding before progressing to the next step. Alternatively, quality assurance departments can be established to validate every step of the DevOps pipeline before proceeding to the next phase. For uninterrupted delivery, the DevOps framework needs an expert to oversee the entire process, monitor the deployment, and give developers feedback on any potential setbacks in the final application version.

Figure 1 shown the roles of supervisor in the DevOps pipeline. The newly appointed leader will be tasked with the following responsibilities:

a) Ensure the project is running, and every team member has the necessary tools to accomplish their duties efficiently.
b) Utilize their effective communication skills and problem-solving abilities to solve any conflicts amongst members that might hinder the latter’s performance.
c) Monitor the entire development process from the writing of the application codes up to the release of the final version.
d) Conduct subsequent meetings with the DevOps team to get updates on their progress.
e) Prevent all forms of distractions, either internal or external and remove impediments so that the team can focus on the work at hand.
f) Draft balanced working schedules to ensure no team player is overwhelmed by duties.
g) They will act as the link between management and the developers and present their grievances to managers.

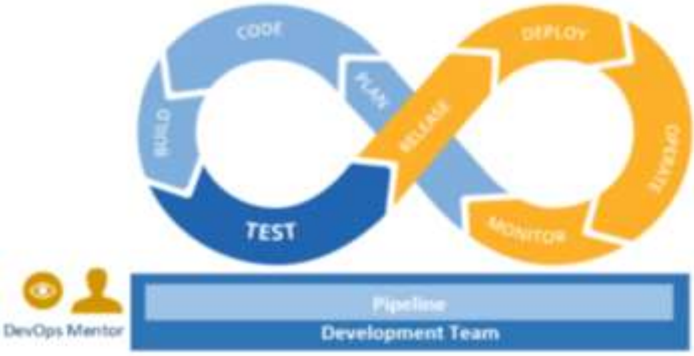

Figure 1. Supervision in the DevOps pipeline

3. RESULTS AND DISCUSSION

3.1. Software productivity in DevOps

The investigation of multi-cycle data in DevOps has attracted significant attention in the recent past. The instant release in DevOps relies on productivity, though this has not been adequately analyzed based on unnoticed cost factors in open source software and the features of merging efforts. The survey by [12] viewed the DevOps framework as an innovative method of producing estimated software, by redefining the cost and efforts it entails. The DevOps model assumes the committed outcomes to be the production cost of the project, as opposed to traditional prolonged processes that were usually accomplished by an individual [13]. Merged efforts involve the running of several scripts, codes, and handling any emerging issues as a team.

The study utilized four open-source projects with a total of 95828 issues, and 95481 commits to verify the findings. Research results indicated that the productivity rate changes with the system development lifecycle. The researchers also noted that combined coding duties by the DevOps team represent the iteration frequency of the system production, and the numbers increase significantly before essential updates are released [14]. Software productivity in DevOps, therefore, allows developers to monitor the entire development lifecycle and predict significant changes in the production processes.

It was noted that the ratio of project timeline to processes is usually low during development periods and rises drastically during the holidays and release dates [15, 16]. It is still a bit challenging to notice the developing periods since some projects have short intervals between the development and release dates. However, the peak is easily noticeable during the release.

3.2. A qualitative study of DevOps usage in practice

Companies are adopting lean and agile development approaches to speed up the rate of software development operations as well as improving the quality of their final product [17]. The organizations describe these agile processes in terms of DevOps, an umbrella concept comprised of procedures, people, and necessary technology for creating cross-functional collaborations to speed up the development of business applications [18]. The researchers had the objective of describing the various methods used by organizations in implementing the DevOps framework and the results of this approach. The researchers conducted a systematic literature review from tertiary sources to find out what other scholars had said concerning the DevOps approach.

To support their survey, they explored six companies involved in different industries at various levels and interviewed them separately. One of the frequent observations was that all the companies gave positive responses that they were happy to adopt the DevOps framework [19]. There were minor hitches noted in the transition, such as the need for restructuring to internalize the new framework.

Although most organizations have a positive response to DevOps, the latter can hardly account for any quantitative data to illustrate how this model has impacted the users so far [20]. There is also a notable absence of quality assurance personnel to perform a qualitative analysis of the DevOps model explicitly [21]. The survey concludes that there is no specific technique of measuring the effectiveness of this framework and recommended future researchers to work on such.

3.3. Systematic literature review on benefits-dependency network for DevOps

Although the DevOps model is yet to be investigated from the academic point of view, it has received overwhelming attention from a significant number of stakeholders. This study was conducted by [22] to figure out beneficial attributes of DevOps and challenges experienced when adopting the framework.

The research explored the central components of the DevOps model by evaluating how agile principles relate to development practices. Peer-reviewed literature was used to assess these benefits.

The findings were synthesized using the benefits dependency network to establish dependencies between DevOps practices and their beneficial values. The survey results indicated that the peer-reviewed literature did not comprehensively define the DevOps characteristics [23, 24]. The characteristics include challenges, benefits, principles, and practices of the framework. The research also noted that the availability of empirical studies on DevOps was very minimal, implying that no in-depth analysis has been conducted on this model.

The DevOps model a product of innovation that has revolutionized the software development industry. Though there is no clear cost of production during the iterations, the final release always delivers precise customer expectations. Organizations that adopt DevOps to deliver their applications often have a competitive advantage since they meet market expectations promptly. Due to the continuous deployment and integration attributes of this model, a lead person in the DevOps pipeline would ensure that each stage is mistake-free to enhance service delivery [25]. The release dates for updated software versions are usually characterized by voluminous coding activities since developers rush to beat their deadline. Quality assurance in every stage of the development eliminates repetitive tasks in the DevOps pipeline to shift more focus towards meeting the market demands, hence, faster return on investments for entrepreneurs. The DevOps pipeline is a collection of agile processes comprising of people, technology, and procedures that collaborate to enhance the delivery of reliable services by business applications. The success of DevOps techniques largely relies on the effectiveness of each development stage to ensure that different functionalities are free from bugs [26].

4. CONCLUSION

This research has established that the DevOps pipeline needs a lead person to monitor the entire software developments lifecycle. This new job description is essential for any for profit organization that aims to get the best value for their investment. The full potential of the DevOps oractices can only be realized if the technical activities are closely monitored by an expert. The overseer will also link the engineers with the management to speed up the approval of recommendations from the developers. Internal mechanisms are more effective for enhancing quality assurance rather than relying on the external consumer recommendations.

ACKNOWLEDGEMENTS

This work was partially supported by the Artificial Intelligence Data Analytics Lab (AIDA), Prince Sultan University, Riyadh, Saudi Arabia.

REFERENCES

[1] J. Angara, S. Prasad and G. Sridevi, "Towards Benchmarking User Stories Estimation with COSMIC Function Points-A Case Example of Participant Observatio", *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8, no. 5, p. 3076, 2018.

[2] N. Wilde et al., "Security for Devops Deployment Processes: Defenses, Risks, Research Directions", *International Journal of Software Engineering & Applications*, vol. 7, no. 6, pp. 01-16, 2016.

[3] J. Angara, S. Prasad and S. Gutta, "Feasibility Predictability Model for Software Test Automation Projects in DevOps Setting", *International Journal of Forensic Software Engineering*, vol. 1, no. 1, p. 1, 2019.

[4] R. Vaasanthi, V. Prasanna and S. Philip, "Analysis of DevOps Tools to Predict an Optimized Pipeline by Adding Weightage for Parameters", *International Journal of Computer Applications*, vol. 181, no. 33, pp. 33-35, 2018.

[5] R. Jabbari, N. bin Ali, K. Petersen and B. Tanveer, "Towards a benefits dependency network for DevOps based on a systematic literature review", *Journal of Software: Evolution and Process*, vol. 30, no. 11, p. e1957, 2018.

[6] J. Pernstål, R. Feldt and T. Gorschek, "The lean gap: A review of lean approaches to large-scale software systems development", *Journal of Systems and Software*, vol. 86, no. 11, pp. 2797-2821, 2013.

[7] B. Snyder and B. Curtis, "Using Analytics to Guide Improvement during an Agile–DevOps Transformation", *IEEE Software*, vol. 35, no. 1, pp. 78-83, 2018.

[8] C. Trubiani, P. Jamshidi, J. Cito, W. Shang, Z. Jiang and M. Borg, "Performance Issues? Hey DevOps, Mind the Uncertainty", *IEEE Software*, vol. 36, no. 2, pp. 110-117, 2019.

[9] B. Ramesh, L. Cao and R. Baskerville, "Agile requirements engineering practices and challenges: an empirical study", *Information Systems Journal*, vol. 20, no. 5, pp. 449-480, 2007.

[10] B. Snyder and B. Curtis, "Using Analytics to Guide Improvement during an Agile–DevOps Transformation", *IEEE Software*, vol. 35, no. 1, pp. 78-83, 2018.

[11] B. Jambunathan and D. Y. Kalpana, "Design of devops solution for managing multi cloud distributed environment", *International Journal of Engineering & Technology*, vol. 7, no. 33, p. 637, 2018.
[12] L. Zhu, L. Bass and G. Champlin-Scharff, "DevOps and Its Practices", *IEEE Software*, vol. 33, no. 3, pp. 32-34, 2016.
[13] A. Balalaie, A. Heydarnoori and P. Jamshidi, "Microservices Architecture Enables DevOps: Migration to a Cloud-Native Architecture", *IEEE Software*, vol. 33, no. 3, pp. 42-52, 2016.
[14] J. Roche, "Adopting DevOps practices in quality assurance", *Communications of the ACM*, vol. 56, no. 11, pp. 38-43, 2013.
[15] C. Virmani, A. Pillai and D. Juneja, "Clustering in Aggregated User Profiles Across Multiple Social Networks", *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 7, no. 6, p. 3692, 2017.
[16] J. Smeds, K. Nybom and I. Porres, "DevOps: A Definition and Perceived Adoption Impediments", *Lecture Notes in Business Information Processing*, vol. 5, no. 2, pp. 166-177, 2015.
[17] C. Cois, J. Yankel and A. Connell, "Modern DevOps: Optimizing software development through effective system interactions", *2014 IEEE International Professional Communication Conference (IPCC)*, vol. 7, no. 3, pp. 74-87, 2014.
[18] N. Ferry, F. Chauvel, H. Song, A. Rossini, M. Lushpenko and A. Solberg, "CloudMF", *ACM Transactions on Internet Technology*, vol. 18, no. 2, pp. 1-24, 2018.
[19] R. Colomo-Palacios, E. Fernandes, P. Soto-Acosta and X. Larrucea, "A case analysis of enabling continuous software deployment through knowledge management", *International Journal of Information Management*, vol. 40, no. 5, pp. 186-189, 2018.
[20] S. Deshpande and R. Ingle, "Preferences based Customized Trust Model for Assessment of Cloud Services", *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8, no. 1, p. 304, 2018.
[21] L. Chen, "Continuous Delivery: Huge Benefits, but Challenges Too", *IEEE Software*, vol. 32, no. 2, pp. 50-54, 2015.
[22] J. Kirchoff, C. Koch and B. Satinover Nichols, "Stakeholder perceptions of green marketing: the effect of demand and supply integration", *International Journal of Physical Distribution & Logistics Management*, vol. 41, no. 7, pp. 684-696, 2011.
[23] M. Shahin, M. Ali Babar and L. Zhu, "Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices", *IEEE Access*, vol. 5, no. 3, pp. 3909-3943, 2017.
[24] G. Claps, R. Berntsson Svensson and A. Aurum, "On the journey to continuous deployment: Technical and social challenges along the way", *Information and Software Technology*, vol. 57, no. 8, pp. 21-31, 2015.
[25] B. Fitzgerald and K. Stol, "Continuous software engineering: A roadmap and agenda", *Journal of Systems and Software*, vol. 123, no. 4, pp. 176-189, 2017.
[26] G. Marchetto, R. Sisto, M. Virgilio and J. Yusupov, "A VNF modeling approach for verification purposes", *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 4, p. 2627, 2019.