IMPORTANCE OF DEVOPS PRACTICES IN SOFTWARE ENGINEERING PROCESS

Ehtesham Zafar
FAST National University
Karachi, Pakistan
K201655@nu.edu.pk

Amna Mubarak
FAST National University
Karachi, Pakistan
K201695@nu.edu.pk

Hassan Ali
FAST National University
Karachi, Pakistan
k201052@nu.edu.pk

Abstract*--With the invention of DevOps, the software development saw a positive change in the the development process. It was not only made to improve collaboration, communication, and integration between software development and operations teams but also automate the development and deployment process reducing the time, efforts and problems related to them thus providing speed, accuracy and efficiency in software releases. The current problem which are faced by many industries now a days is integration and the adoption of DevOps in their software development process due to inexperience employees and lack of material in this regard. This research papers aims to provide knowledge to integrate DevOps with Agile framework. We analyzed various literature and research articles to gather information regarding the implementation of DevOps in different stages of the software development and also the most efficient tools that would be required to achieve and implement DevOps practices in the software development life cycle. This papers presents detail about the working of DevOps in the 4 development stages of Agile model including a brief overview of the tools that would be required to best implement DevOps. This paper could be used as a basis of further modification to their operations by new and existing software houses and development teams who wishes to use DevOps practices.

***Keywords*-- DevOps, SDLC, Agile Software Development, Automation**

I. INTRODUCTION

The idea of Development and Operations (DevOps) in relatively new due to which it faces challenge of acceptance but it arrived to solve a crucial problem faced in the Software Engineering Process[1]. The need for DevOps arises from the challenges faced in traditional software development processes, such as slow delivery, poor quality, and limited collaboration between development and operations teams. DevOps addresses these challenges by automating and integrating software delivery processes, enabling faster and more frequent releases of high-quality software, and promoting collaboration between development and operations teams. This practice increase understanding the business goals along with limitations and constraints of software [3] [4].

Overall, DevOps helps organizations deliver software more efficiently and reliably, while reducing the risk of errors and improving customer satisfaction. It focuses on the convergence of standards between the Development teams and the Operations teams and it seeks to improve cooperation between both teams [2].

II. LITERATURE REVIEW

Research Paper "A proposal to systematize introducing DevOps into the software development process" proposes a framework for introducing DevOps practices in the software development process. The article argues that while DevOps has become increasingly popular, many organizations struggle with how to implement it effectively. The proposed framework consists of four stages: Awareness, Preparation, Implementation, and Continuous Improvement. In the

Awareness stage, the organization becomes familiar with the concept of DevOps and assesses its potential benefits. In the Preparation stage, the organization defines its DevOps strategy, identifies the necessary tools and processes, and trains its personnel. In the Implementation stage, the organization deploys the DevOps tools and processes and monitors their effectiveness. Finally, in the Continuous Improvement stage, the organization evaluates the effectiveness of its DevOps practices and makes adjustments as necessary. The article also discusses the challenges of implementing DevOps, including cultural resistance, tool integration, and knowledge gaps. The proposed framework aims to address these challenges by providing a systematic approach to introducing DevOps and promoting collaboration between development and operations teams.

The article "Analyzing the Behaviour of Applying Agile Methodologies & DevOps Culture in e-Commerce Web Application" by Nikhil Govil et al., published in IEEE Xplore, presents a case study on the application of Agile methodologies and DevOps culture in an e-commerce web application. The study aimed to evaluate the effectiveness of the Agile and DevOps practices in improving the quality of the application and reducing the time-to-market. The study used a mixed-methods approach, combining quantitative and qualitative data from various sources, including surveys, interviews, and performance metrics. The results showed that the application of Agile methodologies and DevOps culture had a significant impact on the quality and performance of the e-commerce web application. The application of Agile methodologies led to improved collaboration, increased flexibility, and faster delivery of software features. The application of DevOps culture led to better communication and coordination between development and operations teams, automation of repetitive tasks, and continuous monitoring and improvement of the application. The study also identified some challenges in implementing Agile methodologies and DevOps culture in the e-commerce web application, including cultural resistance, lack of tool integration, and knowledge gaps. The authors recommend addressing these challenges by providing training and support to the teams and promoting a culture of continuous learning and improvement. Overall, the article provides a valuable contribution to the field of Agile and DevOps practices by presenting a case study on their application in an e-commerce web application. The study highlights the benefits of applying Agile methodologies and DevOps culture in improving the quality and performance of software applications, as well as the challenges of their implementation. The article offers practical insights and recommendations for organizations looking to adopt Agile methodologies and DevOps culture in their software development process.

The article "Qualifying Software Engineers Undergraduates in DevOps - Challenges of Introducing Technical and Non-technical Concepts in a Project-oriented Course" by Isaque Alves and Carla Rocha, published in IEEE Explore, addresses the challenges of introducing DevOps concepts to undergraduate software engineering students. The study aimed to evaluate the effectiveness of a project-oriented course in qualifying students in DevOps and to identify the challenges faced by both students and instructors in teaching and learning DevOps concepts. The study used a mixed-methods approach, combining surveys and interviews with students and instructors, as well as analysis of the course materials and assignments. The results showed that the project-oriented course was effective in introducing DevOps concepts to undergraduate students and provided them with practical experience in applying these concepts in software development projects. The course also improved the students' communication and collaboration skills and

their understanding of the importance of automation and continuous integration and delivery. The study also identified several challenges in teaching and learning DevOps concepts, including the need for students to learn technical and non-technical concepts, the lack of practical experience of some students, and the need for instructors to adapt to new teaching methods and technologies. The authors recommend addressing these challenges by providing more hands-on experiences and exercises, promoting collaboration and communication among students and instructors, and using tools and technologies that support DevOps practices. Overall, the article provides a valuable contribution to the field of DevOps education by presenting a case study on the challenges and opportunities of introducing DevOps concepts to undergraduate software engineering students. The study highlights the importance of practical experience and collaboration in teaching and learning DevOps, as well as the need for instructors to adapt to new teaching methods and technologies. The article offers practical insights and recommendations for educators and organizations looking to qualify software engineering students in DevOps practices.

III. AGILE COMBINED WITH DEVOPS

Agile methodology works best with DevOps because it provides a flexible and iterative approach to software development, which aligns well with the principles of DevOps. The Agile methodology emphasizes collaboration, frequent communication, and continuous delivery of working software, which are essential components of DevOps culture. DevOps and agile are two of the most popular mechanism that organizations must adopt together to stay ahead of the competition[5].

To answer the question of using Agile model over other models, lets discuss its benefits. The main benefit of using Agile over other models is that it allows teams to quickly respond to changing requirements and customer feedback, which is crucial in today's fast-paced and dynamic software development environment. The Agile methodology also promotes early and frequent testing and feedback, which helps to identify and address issues early in the development process, reducing the risk of defects and delays. This aligns well with the DevOps culture of continuous testing and deployment, where software changes are quickly and safely released to production.

According to the survey [6], 75% of respondents claimed that adopting agile with DevOps enhanced staff retention and recruiting compared to 30% of respondents who simply used agile. This is more than just a fact. The productivity of experts in the IT sector increased by around 45%, while consumer satisfaction with the services they receive increased by about 30%.

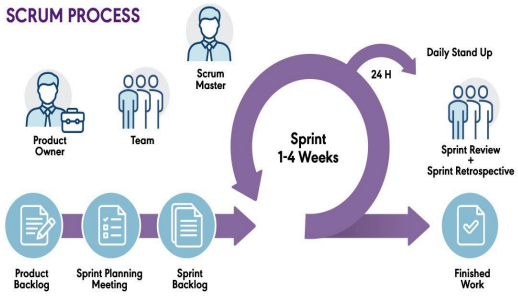

Fig. 1 Agile - Scrum Framework [8]

IV. DEVOPS IMPLEMENTATION IN SOFTWARE EVELOPMENT LIFE CYCLE

One of the main benefit of DevOps is that it provides better collaboration and understanding among the different teams working on the same product [7]. Agile works on iterative plus incremental model which means that it includes small continuous cycles of (planning, implementation, testing and deployment). Collaboration between these components is very crucial in delivering a successful software product whilst also managing the

deadlines and maintaining customer trust at very stage.

A. Planning Stage

DevOps may assist at this stage by fostering communication between the development and operations teams as well as other stakeholders like business analysts and product owners. Teams may make sure that software development is in line with corporate objectives and client demands by utilizing agile approaches and technologies like user stories and backlog management. As a result, DevOps provides a transparency to every stakeholder about what the work is being done, how the work is being done when the work will be done.

B. Implementation/Coding Stage

Software development's coding phase can be aided by DevOps in a number of ways, including by offering practices and tools that promote teamwork, automation, and continuous integration and delivery.

- Collaboration: The DevOps culture places a strong emphasis on working together with operations, other stakeholders, and developers to produce better code more quickly. Developers and operations teams can detect and address problems early in the development process, lowering the risk of errors and delays, by working closely together.

- Automation: DevOps practices and tools enable the automation of time-consuming and repetitive operations like code testing, deployment, and monitoring. Developers' time can be freed up by this automation to work on more difficult and imaginative projects, such creating new features and enhancing user experience which would also eventually result in more faster deployment building a healthy relationship with the client.

- Continuous Integration and Delivery (CI/CD): DevOps encourages continuous integration and delivery, which ensures that code modifications are swiftly tested and pushed to production. This method lowers the risk of delays and faults by identifying problems early in the development process.

- Version Control: DevOps places a strong emphasis on the use of version control tools like Git, which let developers more efficiently collaborate and manage changes to code over time. Developers can work more productively with other team members and more readily go back to prior versions of the code by using version control.

- Infrastructure as Code (IaC): IaC refers to the management of infrastructure configurations in the same manner as managing code, which is a concept that DevOps practices promote. This method can help to lower the risk of mistakes and delays while allowing developers to manage infrastructure upgrades more simply.

The tools for automation, CI/CD and Version Controlling will be discussed in section 5 in detail.

C. Testing Phase

DevOps can assist in this step by automating the testing process, including unit testing, integration testing, and acceptance testing. This ensures that quality criteria are met while reducing the time and effort needed to test software. There are various tools that can help in automating the testing process. This automation can be defined in the development pipeline so that it runs the tests in every cycle of the development phase.

D. Deployment

DevOps can be applied at this point by leveraging tools like configuration management and containerization to automate the deployment process. As a result, there is less downtime and interruption during the quick and reliable deployment of software.

Tools for automating the Deployment process will be discussed in section 5 in detail.

E. Maintenence

DevOps may assist at this stage by putting continuous monitoring and feedback loops into place, utilizing instruments like log analysis and performance monitoring. This ensures that software functions as intended and that any problems or flaws are swiftly found and fixed.

V. DEVOPS TOOLS

The planning, tracking, automation, and management tools we use define the "ground truth" of where and how work happens [9]. It is due to these tools that DevOps is successfully implemented in the Agile software development life cycle. Some essential tools are discussed below with respect to their work they perform in bringing automation and bridging the gap between development and operations.

A. Puppet

DevOps relies on configuration management solutions to automate the setup and configuration of network devices, servers, databases, and other IT infrastructure components. The ability to manage infrastructure as code provided by these technologies allows DevOps teams to deploy and maintain infrastructure more consistently, more efficiently, and with fewer mistakes.

Fig. 2 Puppet Configuration Management [10]

In this proposed work, Puppet, an open-source configuration management tool is used because it is one of the most common and recommended tool among professions due to its advance features and enhanced capabilities which makes it better suited for managing large and complex infrastructures. In a research experiment [11] a model was proposed to provide quality judgement of Puppet code and it was found that experts deemed the model appropriate and usable in practise.

B. Jenkins

The core of DevOps culture is Continuous Integration and Continuous Delivery or Deployment (CI/CD). Actually, CI/CD is an open-source server that gives experts the ability to automate various parts of the delivery process. Jenkins, an open-source continuous integration server, has the ability to automate a software project's whole workflow [12]. Your CI/CD pipeline may be configured and tailored using Jenkins to meet your unique needs. You can iterate and deliver new code as rapidly as feasible with Jenkins. You may use it to evaluate the effectiveness of each pipeline stage as well. It may be used as a straightforward CI server only for development or as a full CI/CD solution that also handles your deployment procedure.

C. Git Action

DevOps teams may automate numerous software development workflows with the aid of GitHub Actions, a CI/CD (Continuous Integration/Continuous Deployment) platform provided by GitHub. It offers a mechanism to create and execute unique processes that are sparked by a variety of occasions, such as code changes, pull requests, or manually applied triggers. DevOps teams can now automate their CI/CD pipelines, reducing the time and effort required for manual tasks such as building and testing code. This can lead to faster feedback cycles, better collaboration, and ultimately faster delivery of software. Additionally, GitHub Actions provides a seamless integration with other GitHub features, such as pull requests and code reviews, making it easier for teams to collaborate and ship high-quality software [13].

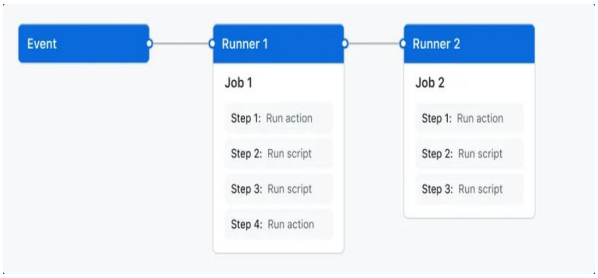

Fig. 3 Components - Github Actions [13]

D. Docker

Docker is one of the most used and recommended container platform since its launch in 2013 [12]. It is frequently cited as one of the most significant DevOps tools. Because it enables remote development and aids in automating the deployment process, Docker has largely been responsible for containerization's rise to popularity in the computer sector. Applications are compartmentalized into discrete containers, making them more secure and adaptable to different situations.

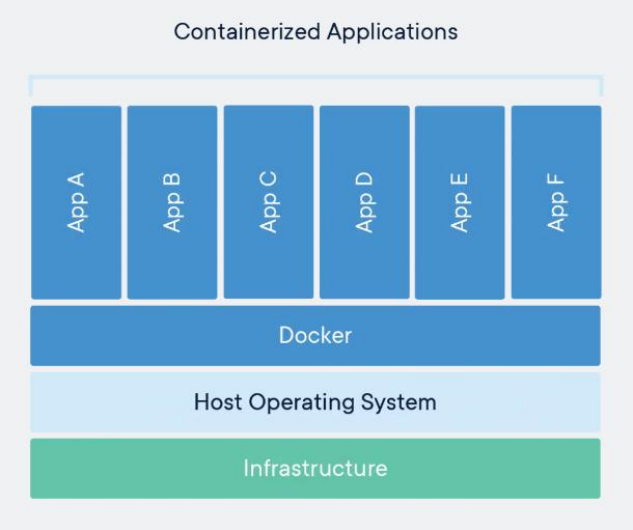

Fig. 4 Containerization - Docker [14]

Docker integrated with a CI/CD tool such as Jenkins can further improve the DevOps workflow. Docker is extensively used in the cloud since so many major cloud service providers, including Amazon Web Services and the Google Cloud Platform, offer it.

E. Kubernetes

Kubernetes is a container orchestration platform that works well with Docker or any of its alternatives to help you group your containers into logical units [12]. The task of handling hundreds or thousands of containers can be automated thanks to it. You do not need to bind your containerized apps to a particular machine while using Kubernetes. Instead, you can deploy to a group of computers called a cluster, with Kubernetes handling the scheduling and distribution of containers among the entire group. Kubernetes provides a powerful and flexible way to group containers into logical units, which enables better resource management, high availability, service discovery and load balancing, and rolling updates and rollbacks.

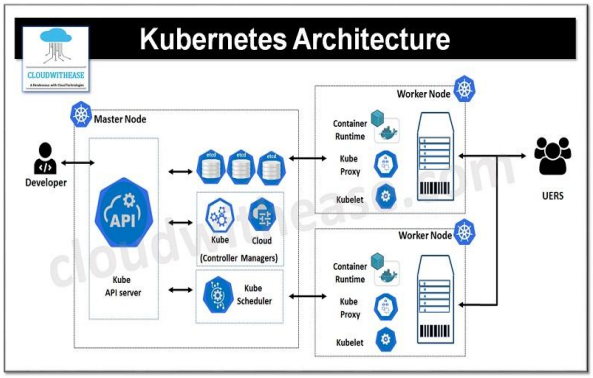

Fig. 5 Architecture - Kubernetes [15]

F. Terraform

Infrastructure as code (IaC) is important in DevOps because it enables automation, consistency, and scalability of infrastructure deployments and management. With the use of languages like YAML or JSON, it enables DevOps teams to handle infrastructure configurations as code. Teams may automate the deployment of infrastructure updates using this method, which also lowers the possibility of errors and ensures consistency between environments. Terraform is specifically build to achieve this purpose providing open-source tool for building, changing, and versioning infrastructure safely and efficiently and allowing for the automation of infrastructure deployment and management.

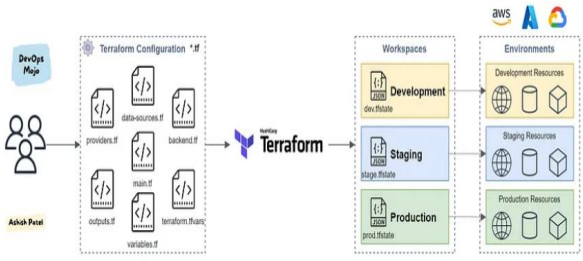

Fig. 6 Workspace - Terraform [16]

VI. RESULT

DevOps made a positive change in the software development process by eliminating the challenges and problems faced by the development, operational and strategic team before its launch. DevOps provides multiple benefits but the key benefits due to which it is highly used in practise are faster delivery, improved collaboration among different teams, increased efficiency and productivity due to its automation in the development life cycle which helps the programmers to focus on providing more features and not worrying about the deployment part, better quality and reliability due to its automation in the testing process and enhanced security it provides in the software. The paper also discusses the benefits in detail of using Agile methodology with DevOps operations to gain maximum benefit of the model. DevOps implementation in each of the 4 stages of iterative/incremental model is discussed to better understand the pros of using DevOps in development. Finally the tools are discusses in details which are used together to achieve benefits of DevOps at its fullest. All of these tools presented in the paper have their own purpose and when implemented together would create an environment to help the software development process integrate with DevOps practices.

REFERENCES

[1] A. Wiedemann, N. Forsgren, M. Wiesche, H. Gewald, and H. Krcmar, “The DevOps Phenomenon,” Queue, vol. 17, no. 2, pp. 40:93–40:112, Apr. 2019.

[2] D. Taibi, V. Lenarduzzi, and C. Pahl, “Continuous architecting with microservices and DevOps: A systematic mapping study,” in Communications in Computer and Information Science, 2019, vol. 1073, pp. 126–151.

[3] Aprna Tripathi, Varsha Kumari, Nikhil Govil, "Impacts of PLC Reducer on Software Design Cohesiveness", International Journal of Innovative Technology and Exploring Engineering,

Volume-8 Issue-8, pp. 2871- 2875, June 2019.

[4] V. Kumari, A. Tripathi, N. Govil and S. Pundhir, "A Proposed Model: Linearly Extensible Triplet Network (LETN)," 2019 4th International Conference on Information Systems and Computer Networks (ISCON), Mathura, India, 2019, pp. 1-6.

[5] N. Govil, M. Saurakhia, P. Agnihotri, S. Shukla and S. Agarwal, "Analyzing the Behaviour of Applying Agile Methodologies & DevOps Culture in e-Commerce Web Application," 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), Tirunelveli, India, 2020, pp. 899-902, doi: 10.1109/ICOEI48184.2020.9142895.

[6] https://www.blueprintsys.com/blog/top-4-challenges-agile-and-devops/ last accessed on 16 April 2023.

[7] N. Govil, M. Saurakhia, P. Agnihotri, S. Shukla and S. Agarwal, "Analyzing the Behaviour of Applying Agile Methodologies & DevOps Culture in e-Commerce Web Application," 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), Tirunelveli, India, 2020, pp. 899-902, doi: 10.1109/ICOEI48184.2020.9142895.

[8] https://www.pm-partners.com.au/the-agile-journey-a-scrum-overview/ last accessed on 18 April 2023

[9] M. Kersten, "A Cambrian Explosion of DevOps Tools," in *IEEE Software*, vol. 35, no. 2, pp. 14-17, March/April 2018, doi: 10.1109/MS.2018.1661330.

[10] https://www.javatpoint.com/puppet-configuration-management last accessed on 20 April 2023.

[11] E. van der Bent, J. Hage, J. Visser and G. Gousios, "How good is your puppet? An empirically defined and validated quality model for puppet," *2018 IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, Campobasso, Italy, 2018, pp. 164-174, doi: 10.1109/SANER.2018.8330206.

[12] https://raygun.com/blog/best-devops-tools/ last accessed on 20 April 2023.

[13]https://docs.github.com/en/actions/learn-github-actions/understanding-github-actions last accessed on 20 April 2023.

[14]https://www.docker.com/resources/what-container/ last accessed on 20 April 2023.

[15] https://cloudwithease.com/what-is-kubernetes/ last accessed on 20 April 2023.

[16] https://medium.com/devops-mojo/terraform-workspaces-overview-what-is-terraform-workspace-introduction-getting-started-519848392724 last accessed on 20 April 2023.