IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

DevOps and Its Practices

Ravi Teja Yarlagadda
DevOps SME & Department of Information Technologyy,
USA

Abstract

DevOps entails a set of integrated activities or practices employed in automation and interlink software development processes with IT developers with the aim of building, testing, and releasing deliverables quickly and reliably. Constantly, DevOps has resulted in the use of infinite loops by the developers or practitioners in showing the relationship between development lifecycle phases. Even though the various activities or steps in a DevOps make a loop and flow sequentially, the iteration indicates that the flow must be constantly collaborative and repetitive to improve the entire lifecycle. If well explored by DevOps, various software automation trends could be ready to handle the industry's latest software and technology.

Keywords: DevOps, Software development practices

Introduction

DevOps entails a set of integrated activities or practices employed in automation and interlink software development processes with IT developers with the aim of building, testing, and releasing deliverables quickly and reliably. DevOps is an integrated term that refers to development and operations and culturally represents an interconnection between developers and operators, whose functionalities were initially based in silos (Whittle, 2014).

Constantly, DevOps has resulted in the use of infinite loops by the developers or practitioners in showing the relationship between development lifecycle phases. Even though the various activities or steps in a DevOps make a loop and flow sequentially, the iteration indicates that the flow must be constantly collaborative and repetitive to improve the entire lifecycle (Gruver, 2016). The main criteria behind DevOps formation were communication. Other developers and designers thought of the method as a collaborative way for the developers and engineers to get tasks and other operations executed as fast as possible through iterative automation.

Various factors contributed to the integration of the development and the operation teams. The groups' long-existing hurdles and issues birthed the idea as their responsibilities are highly distributed, making their operations more complex (Armstrong, 2016). As the complexity chain increased quickly, the lesser the ability to solve the issues became from an individual position. Some of the critical interventions that led to the merge are providing virtual machines in a vast network area, complex configuration of extended network devices and servers, and deploying various applications (Armstrong, 2016). Further, log collection and aggregation, monitoring services, monitoring performance of a network, and monitoring application performance had become a complex issue. Besides, the developers, operators, and engineers could not understand events that lead to these complexities, raising alerts and seeking remedy in the event they occur.

Power combination

When the distinct group of operators and developers collaborated, they designed a decent approach to solving the existing operation complexities in the real world (Gruver, 2016). These complex operations have been crunched into different sections for easy understanding. They include monitoring, management of performance, deployment automation, infrastructure automation, log management, and configuration management.

Infrastructure automation

Unlike the complex times for developing infrastructural components from nothing, nowadays, one does not have to create servers from scratches, buy energy, or establish data center connections manually. Further, the manual connection of a machine into a network is no longer essential. Since the developers and operations got integrated, most of these procedural and time-consuming processes have become ineffective and inessential altogether. Previously, these operations were manual, mundane, and only took place once an operation had developed some issues (Gruver, 2016). Through DevOps, there is an assumption that all functions can be carried out, controlled, and managed in a central place using a simple code. The introduction of cloud services provides real-time services through a web application. That does away with the need for data centers or silos for usual operations to run.

Once DevOps got introduced, most operations got automated like IT infrastructure, which solved the hurdle of being present in the data centers and silos physically to obtain hardware services or enact network modifications. The cloud services are substantially beneficial because they reduced the linear demand costs, and one can make automated provisions without actual payments for the hardware services (Meena, 2014). Various vendors render these services. They entail Azure, Heroku, Ubuntu cloud, Amazon web services, HP cloud, RackSpace cloud, and EngineYard.

Configuration management

Traditionally, developers or network users had to manually implement, execute, and configure various packages after the hardware got installed. The power of DevOps comes in to solve the hurdle through intelligent or automated configuration management. With DevOps, the installation and configuration of hardware packages are run automatically by using simple programmed commands. The computerized configuration solutions are beneficial because there is the constant and real-time deployment of servers in every event. Suppose a user needs to make alterations or modifications in many computers. In that case, they only need to run them in a single machine, and it will replicate in the other devices automatically (Meena, 2014). There are various vendors for the DevOps tools, such as Puppet, Ansible, Pallet, Chef, Salt Stack, and Bcfg2. More specifically, Amazon's OpsWorks and Rightscale vendors also supply the automated tools.

Previously, most of the operational environments had strict guidelines and controls on the designated person responsible for access to the production field, make modifications, and the time to instill these changes. These changes had to occur through physical interaction with hardware, especially in the data centers (Meena, 2014). Today, these demands and extraneous processes cut short the ability for development and operations to merge. However, some of the techniques and standing functions under DevOps are still unclear on their applicability and maintenance. Before one conceives on the best way to deploy models for the infrastructural establishment, they should start from the operation perspective, which is fundamental. Further, the Ops teams of developers must understand the development perspective by deploying an application in a pre-production area (Google, 2014).

Monitoring

Initially, the developers and operations team had to physically regulate or control the entire systems within a network to ensure no flaws or potential risks. These operations were less effective and less productive. With DevOps, the designs are automated to ensure that the operators, organizations, or business organizations never miss the systems' vital alerts. These systems are deeply integrated with monitoring, chatting, and ticketing tools. These tools raise signals with filtered noise (Google, 2014). Unlike in the past models where an alert would be presented in a single machine or unit, the DevOps systems send signals to multiple channels and offer the necessary step to resolve the issues, whether potential or underway.

Besides raising alerts, the DevOps enabled systems have made the on-call management simple and straightforward. The designs can establish and alter schedules and outline escalation guidelines within a single user interface. The operation team will have an idea of whose call is underway and the accountability in case of an incident. In every incident, the vital alerts always get acknowledged as they arise. Further, the monitoring systems present the ability to make advanced reports and analytics (Veritis, 2016). The automated systems have insights into success and opportunistic areas for further enhancements. The systems track all alerts and foreign incidents related events. Also, the plans have robust reporting and analytical patterns for the revelation of the alert's origin, the team's performance level in terms of risk acknowledgment and resolution. Further, it oversees the method for distribution of workload to various channels in a network.

Log management

Independent of the speed of adoption to recent technological changes within business enterprises, primary capabilities such as infrastructure monitoring play a critical role in production applications' operation. As the businesses and other organizations gain momentum, the observability concept is essential for deployment success with minimal interruptions in every stage of operation and constant delivery pipeline (Kuchler, 2016). As a part of the entire process of the delivery pipeline, log and event management enables the developers and operation team to monitor and realize the application's behavior before subjection to production. Even though some entities may argue that log management adds on the procedure, effort, and time for development, it helps in the evolution of the organization's critical software development acts while minimizing issues that can be prevented during production. Further, it can help create a smooth, seamless user interaction stage and reduce the re-architect of the production solutions (Kuchler, 2016).

Testing set

The increased evolution of the technological landscape in various sectors has enhanced the massive need for log management within highly distributed systems. There have been tremendous changes in creating applications and services, deployment ability of applications across various logging as a service vendor, and containers ’pervasiveness. As far as the service building capacities with different languages, the need for collection, monitoring, and tracking data across an interconnected network has gone high (Bass et al., 2015). Initially, the log management process was a time-consuming and tedious activity that involved running search commands aboard all the local servers. The approach had minimal scalability in situations where the servers ’ clusters were limited to 20 servers requiring individual operators. With DevOps, there has been a high prevalence of virtualized systems with low operational costs, increasing the system numbers within a typical business enterprise. With the increased virtualization of systems under DevOps, the independent developers have gained the capacity to create designs in a parallel manner employing best-suited technologies to the production levels. These happen while sharing the surrounding for building, production, and staging. When developers use distinct technologies with different log formats, the log aggregation reduces personalization issues (Bass et al., 2015).

Observability

DevOps has come up with various principles for data monitoring from different standpoints to predict the accuracy and stability of a system. Observability is built on three integrated principles that enhance the workability of developers and operation teams. The first element is external monitoring which examines external run-ups that ensure that web applications have a modern experience for the users (Bass et al., 2015). Secondly, the metrics and distributed tracings identify links between distributed applications across a system. It also helps in error identification, and exceptions from an application and makes necessary resolution actions. Finally, the events and logs assist in providing contextual information from invested data in the circumstances. It also enables one to trace issues in a code once integrated with other principles.

Literature review

DevOps have been defined and promoted as the next change drive in the IT field as technology changes and assume different perspectives. DevOps is based on the realities of the specific organization and industrial application (Contributor, 2014). Every industry is seeking after DevOps due to its ability to create a collaborative environment. Among the broad area of application, DevOps has attracted high popularity in the manufacturing sector. Considering that interlocked processes and activities highly surround the industry, DevOps fits into space. Further, the manufacturing market has seen constant expansion with speedy delivery and innovativeness, driving the integrated development and operations.

Manufacturing calls for increased collaboration and streamlined communication, which are the cornerstone and basis for DevOps (Veritis, 2016). DevOps makes massive growth in different areas; the manufacturing sector also benefits in three other areas: innovation, automation, and collaboration. Integration of the three factors caused streamlined communication and increased production in the industry. Further, the teams work better towards the same objective, hence fast product delivery and minimal marketing time (Humble and Farley, 2011).

Collaboration

According to research, lack of collaboration has resulted in many firms' failure, especially in the automotive industry. Collaboration is the cultural intuition of an organization, and every entity looks forward to the achievement of such levels with ease. However, the process is not easy as some may project, but that's where DevOps chips in (Veritis, 2016). DevOps comes in to integrate the development and operational groups of an organization. Most of the automotive manufacturing industries rely on silos for awareness of the tasks they should perform. However, DevOps helps bridge the gap between the Ops and Dev team and make them aware of their responsibilities (Bird, 2016). That results in a quality enhancement in every step of development, as they are tied up with testing and preparing the products for production. SIEMEN serves as the best example of manufacturing companies, which have harnessed strong collaboration.

Automation

Automation is the process of enhancing software operations to reduce the human workforce. Most machines are automated to ensure fast production and delivery of products to meet users 'demands (Bird, 2016). However, the automation process is not entirely covered. Automation has helped manufacturing industries to covers some ground on introduced solutions to time-consuming activities. However, to achieve more robustness in the area, the manufacturers ought to embrace DevOps. For industries to see success under DevOps, they must follow the four steps for installation, including steering inefficiency, testing, deploying, and operating. Enhanced automation is a company that has harnessed these automation gears. The industry has applied inefficiency steering and testing to transform their operation way from a procedural point to a product-based setting. That has then changed the production through optimized development (Humble and Farley, 2011). They have also employed automated deployment and operation as their driving force to accelerate the

development process. Finally, that has promoted transparency, easy deployment, and quick release of deliverables to the market.

Innovation

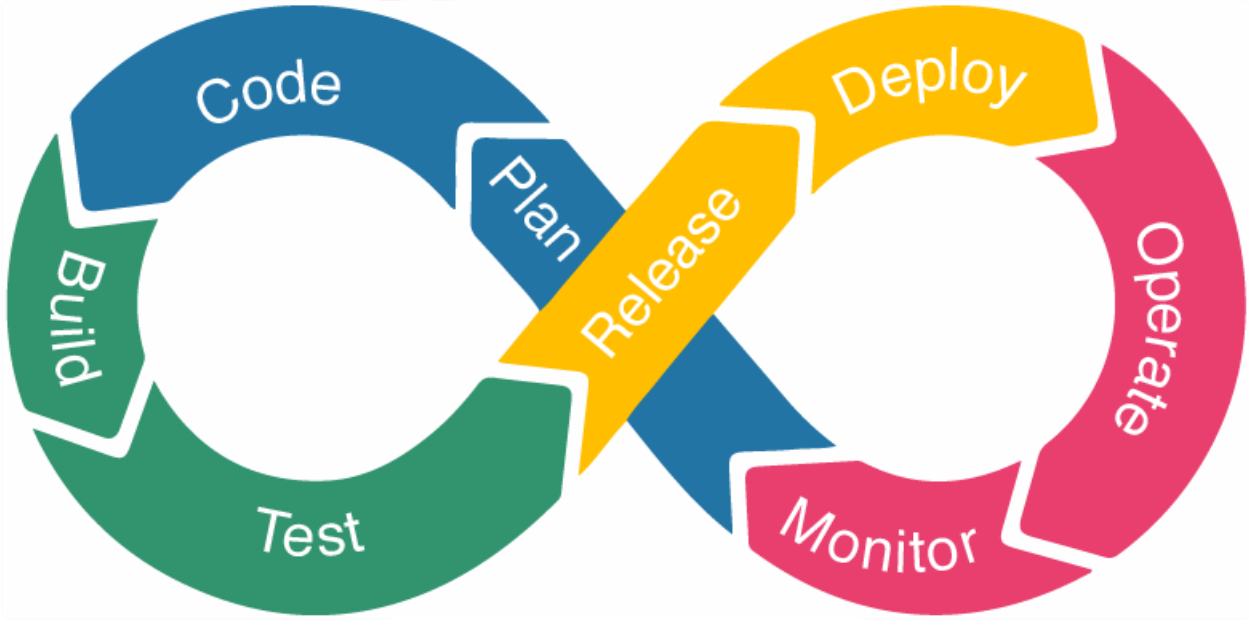
To reduce the pressure that comes from the development cycles, industries have ensured quality and production levels are optimal. That has opened up the opportunity for innovation. Innovativeness has offered the automotive sector an advantage in competition in the market. DevOps have helped the industries achieve the highest innovation level (Humble and Farley, 2011). DevOps enables organizations to reduce the amount of time taken in production or activity execution allowing the business to make informative decisions through process transfer to the teams.

One of the main differences in the approach of the traditional developers is based on coding. Initially, the industrial developers employed various application features and merged them at the very end of coding. Once the bugs and flaws occur, the developers have to carry out the entire work to seek corrections. That has resulted in unnecessary delays in detecting bugs manually through integration. DevOps has brought a substantial transformation in the field through CI/CD phases whereby developers deliver updated codes at a relatively faster rate and with high frequency. With such actions, the developers can sample and cross-check code in small units and perform constant compilation (Bob, 2016). The testing, security, and UI components are usually automated for regular code checks, and it's time to shift to the repository. Naturally, that results in early detection of bugs and faster resolution, thus enhancing the developers' productivity. Once the deployments have been taken care of through pipeline release, the deliverables' delivery becomes continuous.

DevOps cycles

The DevOps cycles form natural cycles which are looped and employed by the practitioners to mark their lifecycles. Despite the sequential flow of the DevOps lifecycles, the loop indicates the urge for regular collaboration and repetition for phase improvement. A DevOps lifecycle entails six repetitive phases representing the capabilities, processes, and techniques crucial for development. The loop has left and right sides, which are different operational capabilities, processes, and tools. Each face encounters separate but constant collaboration of teams and communication to ensure velocity, alignment, and quality of the process are maintained. The lifecycles are planning, building, integration and deployment, monitoring, operating, and responding through feedback passage.

Before development, each team should be well equipped with open tools and rules at every development stage. The devices are customizable to suits the developer's needs and aspirations (Bob, 2016). The process results in the fast development of superior quality software, which is reliable. The fact that the stages are iterative, they make bug recognition easy and quicker. The processes iterate as shown below;



Planning

The process involves a problem statement and a scope definition to identify the required resources that would be in use throughout the period. Secondly, the developers overview the latest systems and state their objectives. Thirdly, they establish the study's practicability and feasibility to determine the period it would take to accomplish. Fourthly, the team considers the potential threats, risks, constraints, security, and integrations for the system. Finally, they compile a feasibility report for the whole project (Tutorialspoint, 2012).

Building

Once the initial stage of project planning gets accomplished and approved, the developer makes the system designs and changes the SRS document into a logical component with detailed specifications for implementation through coding. That is followed by the creation of contingencies, team training, and maintenance through an operation plan. Once the design is complete, the documents get passed to the implementation level. Implementation entails coding using a specified source code and programming language. The system modules get combined in straining surrounding for error and defect detection (Tutorialspoint, 2012). Then a test report gets prepared through resource allocation once the system has been integrated.

Monitoring

The monitoring process seeks to identify the various errors and bugs in the system—DevOps chips through this phase, mainly during the maintenance and support lifecycle. Traditionally, the maintenance activities were carried through constant supervision by users or developers as they used. They implemented changes that allowed software to undergo a specific period or implemented decent requirements once a system is in operation. The process handles residual errors and resolved them manually eventually after the testing phase. Today, DevOps comes in to fix the entire hurdle by implementing simple codes that could oversee the system virtually and detect foreign components and raise alerts. The alerts get raised and resolved automatically or recommend necessary actions that the developers and operation team should undertake to fix them. The process of maintenance and support is, therefore, run automatically without the need for physical intervention. That helps the developers to refocus their energy into other productive actions (Crispin and Gregory, 2015).

DevOps Process flow

Since the introduction of the DevOps in the development cycle, the teams have moments. As the IT operations' future is set on DevOps, it's critical to understand the various processes creating flow and the best technique to implement it. The multiple principles of DevOps form a practical and cultural platform for organizations. While developer concentrates their energy on automation, collaboration, and iteration as the main principles, they also indulge in constant system improvement (Duffy, 2015). The developers and operations team conduct a regular test on the system and learn from their experience with code bugs and errors. To optimize their level of performance, they swiftly act on the system feedbacks and recommendations. That also optimizes deployment time, cost, and effort.

These principles have extended their applicability to lean-agile models to activities that focus on automation and tools provision for fast deployment. The automation process has enabled operators and other IT specialists to draw their effort into seamless processes by adopting the agile model, which focuses on integration, iteration, delivery, and deployment of the software. These are some of the activities that compound the collaboration process throughout the pipeline's entire development process.

DevOps benefits

DevOps has brought massive solutions to the industrial platform. Most industries and organizations, especially in the United States, have harnessed DevOps' benefits through the fast adoption of methodology and tools in their development processes. For instance, Enhanced Automation, based in America, has shut the traditional pains relating to the dragging IT operations (Opsgenie, 2015). The organization has entirely focused on automation, collaboration, and flexibility through agile benefits. These organizations have attracted the following benefits; first, they spend less time marketing their products or services as they are up-to-date and solving real-time issues. Therefore, they deliver their components at a faster rate to the market. Secondly, the

quality of return on investment has increased immensely overtime. Thirdly, there has been high customer satisfaction as they acquire services and products on a real-time basis following faster delivery (Opsgenie, 2015). Fourthly, there has been high operation efficiency through reduced operation time, with minimum cost and effort. That is actualized through automation. Fifthly, the improved collaboration has created a field for developers and IT operators to resolve the existing issues using simple codes. These codes finally help in the fast detection of errors and problems and fix them.

These development teams seamlessly foresee changes through processes and culture of reduced risks of miscommunication and process misalignment. With clear and constant communication, efficiency and product quality get enhanced. Moreover, regular integration, deployment, and testing promote fast process development and error detection.

Implementation of DevOps

During the initial stages of DevOps installation in the organization, the process always seems tedious and daunting. That is because the operation does entail process shifting and cultural shifting as a whole. When an organization decides to go for DevOps adoption, it must consider the gradual implementation of stages (Bass et al., 2015). The phases should be installed one at a time. Depending on the current positioning of an organization, the management may consider employing the agile methodology.

The sequential implementation process follows the following steps; first, the organization must establish the agile development model. Secondly, consider cloud computing by shifting their infrastructure, software, and platform to the clouds. That means abandoning the traditional dependency on the clouds for service delivery (Bass et al., 2015). It also makes the data centers and silos inessential, as the organization's processes are available on a real-time basis. Thirdly, the organization must adapt its operations to CI/CD pipeline workflow. Fourthly, they should automate their software deployment process to reduce human interactions and enhance the fast delivery of products in the market. Fifthly, they should automate the software testing for the discovery of errors and defects in codes. Finally, they must ensure continuous deployment upon implementation.

Organizations must understand that automated DevOps brings along tooling and infrastructural shifts. If the organization does not have proper tools and infrastructure for process support, it risks creating gaps in the DevOps process flow. Therefore, for the intelligent creation of a legit DevOps process, an organization must integrate and automate all stages using an agile methodology (Bass et al., 2015). Further, a firm may consider the application of visuals, which determine the DevOps processes, personnel, working them out, and process timelines. These assist in the implementation process by keeping all participants in the same phase.

DevOps Tools

As far as the IT field's development and operational processes are mindset-related, various tools must be incorporated. Therefore, the organization must consider multiple devices during DevOps implementation. Git is one of the simple tools based on a distributed source code that allows the developer and operators to track the development progress. One can navigate easily back and forth to various regulations and leverage other components such as workflow options, staging areas, and branching environments. Lucidchart is a visual feature that assists developers and IT engineers to create rapid but easy to comprehend processes and data. Further, during initial developments using DevOps, the team must be trained and handed with documentations to track the development progress and keep all members updated (Lucidchart, 2015).

Typically, containerization has shifted to the grouping into various units and clusters for automation and distribution (Kar, 2015). Kubernetes makes the most common DevOps tool and open-source system that helps developers shift their projects to the following levels. These are just a few tools employed in the process of DevOps development process, among others. Nevertheless, the IT engineers still face the hurdle of tool management. Independent of the device in the application, Lucidchart straightens any arising issues relating to the development process.

Future

If well explored by DevOps, various software automation trends could be ready to handle the industry's latest software and technology. The new technology and automated software will make machines in the industries to be autonomous. The main issue with autonomous machinery is the DevOps' ability to convince the corporate, operators, and drivers that these self-driven machines are accurate and safe to ride on or operate (Abouzaid, 2016). Nevertheless, the entire DevOps process will modernize the future through intensive automation, collaboration, and integration to speed up fast delivery.

Conclusion

Before one understands the DevOps definition and purpose, they must know their industry and their position. Some systems require a high degree of automation and collaboration, while others don't require complex processes. However, these processes call for alignment between applications and requirements and quick developments, and innovative innovation testing. Once an industry has a constant delivery to the market, DevOps can help release small batches for effectiveness and efficiency in the market.

References

[1] AbouZaid, A. (2016). Continuous delivery and maturity model: DevOps. Retrieved September 18, 2019, from http://tech.aabouzaid.com/2016/01/continuous-delivery-and-maturity-model.html

[2] Armstrong, S. (2016). DevOps for networking: Boost your organization's growth by incorporating networking in the DevOps culture. Birmingham, UK: Packt Publishing.

[3] Atlassian. (2015) DevOps: Breaking the Development Operation Barriers. Retrieved February 26, 2021, from https://www.atlassian.com/devops

[4] Bass, L., Weber, I., & Zhu, L. (2015). DevOps: A software architect’s perspective. Old Tappan, NJ: Pearson Education.

[5] Bird, J. (2016). DevOps sec: Delivering secure software through continuous delivery. Sebastopol, CA: O'Reilly Media.

[6] Bob, R. (2014) The DevOps Life Cycle. Retrieved February 26, 2021, from https://devops.com/the-devops-life-cycle/

[7] Contributor. (2014) DevOps with purpose: It's about your application—retrieved February 26, 2021, from https://devops.com/devops-with-a-purpose-applications/#:~:text=A%20common%20goal%20cited%20for,operations%20teams%3B%20and%20automation%20tools.

[8] Crispin, L., & Gregory, J. (2015). More agile testing: Learning journeys for the whole team. Boston, MA: Addison-Wesley.

[9] Duffy, M. (2015). DevOps automation cookbook: Over 120 recipes covering essential automation techniques. Birmingham, UK: Packt Publishing.

[10] Google. (2014) 2014 State of DevOps Report. Retrieved February 26, 2021, from https://services.google.com/fh/files/misc/state-of-devops-2014.pdf

[11] Gruver, G. (2016). Start and scaling DevOps in the enterprise. Pennsauken, NJ: BookBaby.

[12] Humble, J., & Farley, D. (2011). Continuous delivery. Boston, MA: Pearson Education

[13] Kar, S. (2015) Best 2014 DevOps tools and trends that define DevOps' future in 2015. Retrieved February 26, 2021, from https://siliconangle.com/2015/01/13/best-2014-devops-tools-and-trends-that-define-the-future-of-devops-in-2015/

[14] Kuchler, K. (2016) Log Management: What DevOps Teams Need to Know. Retrieved February 26, 2021, from https://devops.com/log-management-what-devops-teams-need-to-know/

[15] Lucidchart. (2015) Understanding the DevOps process flow. Retrieved February 26, 2021, from https://www.lucidchart.com/blog/devops-process-flow#:~:text=The%20DevOps%20process%20flow%20is,feedback%2C%20delivery%2C%20and%20deployment.

[16] Meena, S. (2014) Configuration Management. Retrieved February 26, 2021, from https://www.slideshare.net/sahilsk/configuration-management-stackexpress-20140610

[17] Microsoft. (2013) 32 principles and practices of successful continuous Integration, Continuous Delivery, and DevOps. Retrieved February 26, 2021, from http://davidfrico.com/devops-principles.pdf

[18] Opsgenie. (2015) Modern incident management for operating always-on services. Retrieved February 26, 2021, from https://www.atlassian.com/software/opsgenie/what-is-opsgenie?

[19] Tutorialspoint. (2012) System Development Life Cycle. Retrieved February 26, 2021, from https://www.tutorialspoint.com/system_analysis_and_design/system_analysis_and_design_development_life_cycle.htm

[20] Veritis. (2016) DevOps Implementation in Manufacturing Sector: Meet the Culture-Driven Approach. Retrieved February 26, 2021, from https://www.veritis.com/blog/devops-implementation-in-manufacturing-sector-meet-the-culture-driven-approach /

[21] Whittle, D. (2014) An Introduction to DevOps. Retrieved February 26, 2021, from https://devops.com/introductiontodevops/