CHAPTER 1

INTRODUCTION

1.1 Overview:

Devops is a new trend which has emerged from collision between two old trends namely ‘agile system administrations’ or ‘agile operations’ and other is the understanding the value of collaboration between development and operation staff in each and every stages of DevOps life cycle. ‘Dev’ means all the developers involved in production stage. ‘Ops’ means the term used for system engineers, system administrators, security professionals and various other disciplines or job title. So, totally ‘DevOps’ does not differentiate between any of the system administrator sub-disciplines. DevOps is a practice in which operations and developers participate together in each stage of DevOps life cycle like from development stage to production stage.

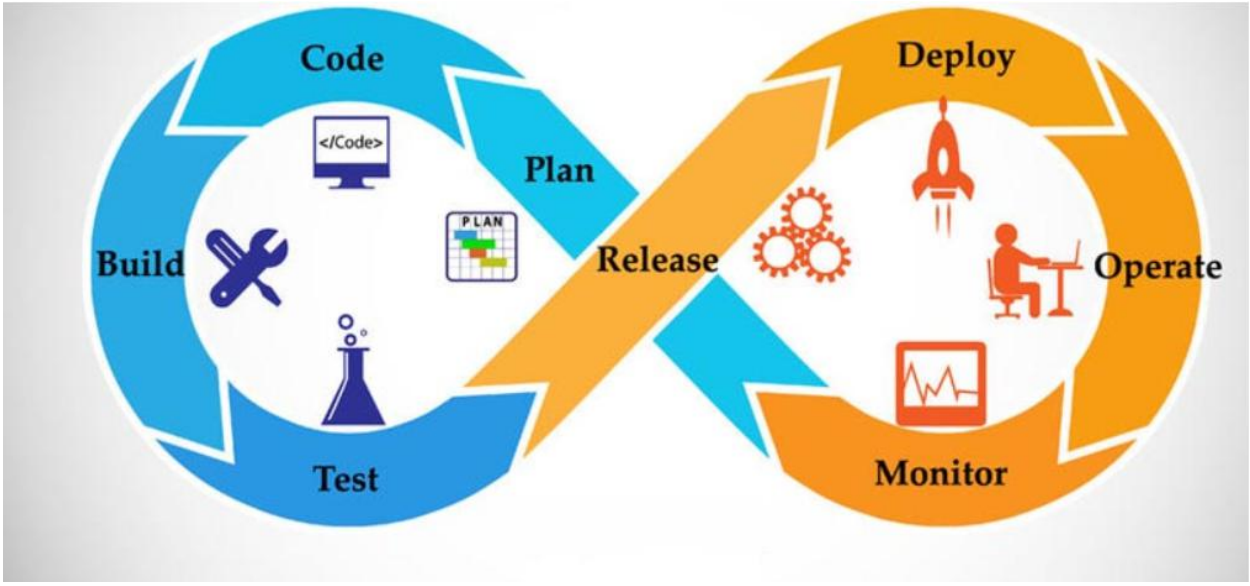

Fig.1.0 DevOps Life Cycle

The speed of application development and application change is increasing and with it, the demand of “Rolls Royce like” quality. Companies look to build new capabilities with high expectations being placed on the IT department. Although the IT industry has taken huge leaps in technology innovation, the quality of application development projects has been lagging. Many IT projects run inefficiently, missing implementation deadlines and causing outages during or

after implementation and therefore, costing significantly more than anticipated. DevOps is the new development that addresses such inefficiencies. It connects development, quality assurance, and technical operations personnel in a way that the entire 'build-release-run-repeat'process operates as a factory, having clear roles and responsibilities and well-defined inputs and outputs DevOps is positioned right in the Peak of Inflated Expectations.In simple terms, DevOps refers to an umbrella concept that encompasses people, processes, and technologies required to connect development to execution."Connect" in this context means ensuring that the development changes are tested and deployed in a way that is efficient and does not interrupt ongoing operations.

1.2 Objectives:

The aim of DevOps is to revolutionize the transition, de-risk IT deployments, eliminate the excuse "but-it-works-on-my-system", and break the silos between developers, testers, release managers, and system operators. The products and tools developed in this area focus on maximizing predictability, visibility, and flexibility, while maintaining stability and integrity. DevOps, in itself, is not a new concept. A development-to-operations lifecycle has existed for quite some time. The latest developments include the ambition to industrialize, automate, and connect the entire process covering infrastructure, application, as well as business changes. The prime focus is on outage reduction and quality improvement.Developers always want to deliver the changes in the product as soon as possible whereas the operation team want reliability and stability in the product. This situation was explained clearly in "wall of confusion" by Lee Thomson is shown in figures 4 & 5. This wall of confusion not only gives the mentalities of two teams but also the tools they practice. DevOps bridge the gap between the development and operations for better and faster results.

DevOps has fundamentally changed the way an IT organization works and how it gets things done. Since its inception in 2009, DevOps (coined as the "new Cloud" by market) has been adopted at a rapid pace, evolving from a niche concept to an integral part of enterprise IT strategy. This fast pace in adoption was mainly due to the immediate value realization that DevOps helps business to build better-quality products and services quickly and with greater reliability.According to IDC's "DevOps and the Cost of Downtime: Fortune 1000 Best Practice Metrics Quantified"

• The average hourly cost of an infrastructure failure is $100,000

• The average hourly cost of a critical application failure is $500,000 – $1M

• The average number of deployments per month is expected to double in 2 years

• Unplanned application downtime costs the Fortune 1000 from $1.25 billion to $2.5 billion every year.

Goals for This Work:

The goal of DevOps is not to eliminate errors/downtime, rather automate manual tasks and create repeatable processes with greater transparency, traceability, thus zeroing down on the level of manual intervention and helping organizations attain a stable and reliable IT infrastructure, ensuring uninterrupted business continuity with a level of acceptable risks. By this process, Devops help organizations attain a stable and reliable IT infrastructure. Therefore, one of the drivers of increasing interest in DevOps practices and tools is the significant reduction of unplanned downtime and the cost associated with it.

1. Strong collaboration between development and operation teams.

2. Synchronized deployment across multiple platforms.

3. Pressure to release applications to meet customer requirements or to enter into the new market.

4. Improving end user capability levels.

5. Vast usage of smart devices

6. Necessity to develop and deploy into cloud-based

applications.

7. Increasingly complex IT infrastructure.

8. Need to reduce the cost for IT industry.

1.3 Significance: RECOGNIZING BUSINESS VALUE OF DEVOPS

DevOps applies agile and lean principles in the complete software deployment process to enhance the speed of delivery of product or service from the initial release to the production release and to the feedback given by the client based on the release. DevOps return our investment in these three areas,

1. Enhanced Customer Experience

Delivering an enhanced product for the customer leads to build loyalty and increase in market share. To deliver an enhanced product we need to continuously obtain and respond to the customer's feedback faster and perform required changes suggested by the customer.

2. Increased capacity to Innovate

Lean thinking approaches are used in modern organizations to increase their innovation capacity. Their goals are to utilize the resources efficiently for other activities by reducing waste and rework. An example of lean thinking in organization is A-B testing in which large organizations asks a small group of users to test and rate two or more sets of software having different capabilities then the better capability set is picked up for the users and unsuccessful version is rolled back. This A-B testing is reliable only if efficient and automated mechanisms are adopted such as DevOps.

3. Faster time to value:

This involves in development of new culture and practices and automating the project leads to fast and reliable delivery process throughout the production phase. This DevOps can be worked as a business capability with the tools for release planning, predictability and success. The DevOps main goal is to deliver the value faster and in efficient way and the value definition changes with organization or with the project.

CHAPTER 2

EVOLUTION

2.1 History:

Patrick Debois, who devised the name “DevOps” in 2009 and he is also called as “The Father of DevOps” devised the name “DevOps” in 2009. The word DevOps says itself that it formed by coalescing the two words “Development” and “Operations”. DevOps is the collaboration of development and deployment of software. DevOps is the portmanteau of development and operations. It is a software development method that escalates to the amalgamation between software development team and operations team. This is the time to change the old technology to new technology like DevOps. - “Time to stop wasting money, time to start delivering great software and building systems that scale and last” – Patrick Debois.

Just like anything in the modern world, and IT in particular, web and mobile app development is basically a truckload of complex procedures. However, the most important ones will always be the following: writing features, testing the code, finding flaws (debugging), and deployment of those features to production. Each of these stages involves a certain team of specialists: developers write code, Quality Assurance (QA) engineers are responsible for testing while system administrators deploy stuff. As you can see, each of these teams has its own area of responsibilities.The result of such “division of labor”, if you like, is that each team member focuses and relies on different criteria to evaluate their efficiency. E.g. for developers, it’s mostly about how fast they can implement or fix a certain feature. QA engineers are absorbed by the processes of finding every flaw to make the application work as designed. System administrators, on the other hand, do their best to keep the whole thing up and running without any crashes and make sure deployments go smoothly.

In addition to the research by Cloud providers, major product and tool vendors, DevOps market movement has been catalyzed by increased Cloud adoption, emergence of concepts like containerization, Platform-as-a-Service, micro-service architecture, service virtualization, and a strong contribution from the open-source community with several cost-optimized DevOps enabling tools. The early adopters or the “DevOps Unicorns” like Netflix, Amazon, Google, Etsy,

and Snapchat have continuously innovated and showcased successful DevOps model variants like NoOps, ChatOps, and SmartOps. In a recent study, Rackspace interviewed 700 IT decision makers and found 55% of the organizations had already implemented or adopted DevOps and are looking for enhancements. Further 31% of them plan to use DevOps in the next 2 years. This adoption is among the largest in the side of technology for the initial implementation of tools. Today, the DevOps trend not only goes beyond technology implementation and management but also focuses on a positive organizational change brought across its processes, cultural shift, and security and compliance aspects of the DevOps platform. Over the next 5 years, the DevOps market looks very positive, with many sources forecasting double-digit growth and a higher adoption rate as larger enterprises begin to understand the benefits DevOps can bring in terms of cost reduction and agility.

2.2 Evolution:

If DevOps had a birth certificate, the father's name would be penned in as Patrick Debois. Patrick was interested in learning IT from every perspective, and in 2007, he began working on a large data center migration where he was in charge of testing. During this project, he realized that the frustrations experienced in projects such as these are from the constant switching back and forth between the development side of the problem and the silo of operations on the other side of the fence. He recognized that a lot of time and effort was wasted navigating the project between these two worlds, but the divide between them seemed too wide to bridge.

Later, in 2008 during an Agile conference held in Toronto, Canada, a man by the name of Andrew Shafer tried to put together a meetup session entitled "Agile Infrastructure." When Patrick showed up for the session, he was the only one there. Andrew had received so much negative feedback from his posting that not even he showed up to his own session. However, Patrick was so excited to learn of a like-minded person that he hunted him down at the conference and had that talk in the hallway. They formed a discussion group for other people to post their ideas for how to solve this divide between development and operations later that year.

Initially, the interest was pretty tame and not a whole lot came of it. In June of 2009, John Allspaw and Paul Hammond gave a talk entitled "10+ Deploys a Day: Dev and Ops Cooperation

at Flickr." Our friend Patrick happened to watch the streaming video of that presentation at his home in Belgium, and it instantly resonated with him. He realized this was exactly the solution for which he had been looking. Emboldened by this presentation, he put out a call to have a gathering of developers and system administrators to get together and discuss the best ways to start bridging the gap between the two disparate fields.

He named the event DevOpsDays, occurring in the last days of October in 2009. This event garnered a fair amount of attention from experts in both fields and sparked lively debates over Twitter where the hashtag was soon shortened to simply DevOps. It wasn't long before some of the smaller tech enterprises were attempting to put together DevOps practices as well as tools built to aid these newly forming teams. DevOps had managed to achieve a grassroots following that was starting to put their ideas to use.

Finally, in March of 2011, Cameron Haight of Gartner presented his predictions for the trajectory of DevOps over the next few years. His positive outlook on its impact on the industry lead to more attention for the DevOps movement, and it wasn't long before enterprises of all sizes were beginning to adopt these new practices. DevOps had officially caught on as the next big thing since Agile for the IT industry.

CHAPTER 3

KEY CONCEPTS AND LITERATURE SURVEY

3.1 Salient Properties and Principles of DevOps :

The principles of DevOps are Iterative, Incremental, Collaboration, Quantification, automation, integration and Holistic . Lets discuss below few of the key features of DevOps.

1. Collaboration

Collaboration between the two departments stands at the very base of the DevOps operations. While it does require cross-functional training and discipline as well as may result in the changing of predefined roles, such collaboration ultimately leads to higher efficiency and effectiveness.

2. Quantification

It is necessary to have a scale on whose basis a practice and its effects can be judged. By quantifying the number of deployments, failures, time taken for each deployment, up time etc before employing DevOps, you can successfully judge the impact and improvement of DevOps on your work culture and environment.

3. Automation

While the developing department generally has a high level of automation, we don't observe the same with the operations side. By integrating the same level of automation in both departments, we achieve a level of consistency and efficiency across the board that would not otherwise be possible.

4. Holistic

Holistic system means thinking about the entire system and the ecosystem around it.

Sharing feedback, best practices, and knowledge Adherence to these principles is achieved through a number of DevOps practices that include continuous delivery, frequent deployments, QA automation, validating ideas as early as possible, and in-team collaboration.

In short, the main principles of DevOps are automation, continuous delivery, and fast reaction to feedback. You can find a more detailed explanation of DevOps pillars in the **CAMS** acronym:

Culture represented by human communication, technical processes, and tools

Automation of processes

Measurement of KPIs.

3.2 Literature Survey :

This paper deals with a systematic literature review which focus on the general term DevOps. The literature review includes all relevant literature which doesn't rely only on one research methodology, one set of journals, or one geographic region. (Webster and Watson, 2002, p. 5) To understand the whole process and technique in detail, the following section explains how the literature was filtered and selected. The search was applied to three databases and two of them have a specific focus on information technology. The third one is a recognized economical database. 2.1 Review method & Search string The search string in all databases is DevOps. The choice of a very general search term helps to get a very wide overview. Within the databases the search was not filtered, and no date range was conducted. The search is processed in IEEE Xplore, ACM Digital Library and EBSCO Information Services. All three databases are recognized research databases in the field of information technology or economics.

DevOps is an evolution of the agile movement [8]. Agile Software Development advocates small release iterations with customer reviews. It assumes the team can release software frequently in some production-like environment. However, pioneer Agile literature puts little emphasis on deployment-specific practices. No Extreme Programming (XP) practice, for example, is about deployment [57]. The same sparse attention to deployment is evident in traditional software engineering processes [92] and books [105, 115]. Consequently, the transition to production tends to be a stressful process in organizations, containing manual, error-prone activities, and, even, last-minute corrections [82]. DevOps proposes a complementary set of agile practices to enable the iterative delivery of software in short cycles effectively.

From an organizational perspective, the DevOps movement promotes closer collaboration between developers and operators. The existence of distinct silos for operations and development is still prevalent: operations staff are responsible for managing software modifications in production and for service levels [48]; development teams, on the other hand, are accountable for continuously developing new features to meet business requirements. Each one of these departments has its independent processes, tools, and knowledge bases. The interface between them in the pre-DevOps era was usually a ticket system: development teams demanded the deployment of new software versions, and the operations staff manually managed those tickets. In such an arrangement, development teams continuously seek to push new versions into production, while operations staff attempt to block these changes to maintain software stability and other non-functional concerns.

Theoretically, this structure provides higher stability for software in production. However, in practice, it also results in long delays between code updates and deployment, as well as ineffective problem-solving processes, in which organizational silos blame each other for production problems. Conflicts between developers and operators, significant deployment times, and the need for frequent and reliable releases led to inefficient execution of agile processes. In this context, developers and operators began collaborating within enterprises to address this gap. This movement was coined “DevOps” in 2008 [68]. In the Continuous Delivery book [82], Humble advocates for an automated deployment pipeline, in which any software version committed to the repository must be a production-candidate version.

After passing through stages, such as compilation and automated tests, the software is sent to production by the press of a button. This process is called Continuous Delivery. A variant is the continuous deployment [80], which automatically sends to production every version that passes through the pipeline. Many authors closely relate DevOps to continuous delivery and deployment [42, 46, 47, 49]. Yasar, for example, calls the deployment pipeline a “DevOps platform” [49]. Besides automating the delivery process, DevOps initiatives have also focused on using automated runtime monitoring for improving software runtime properties, such as performance, scalability, availability, and resilience [2, 3, 8, 40]. From this perspective, “Site Reliability Engineering” [58] emerged as a new term related to DevOps work at runtime.

CHAPTER 4

WORKING

4.1 ARCHITECTURE :

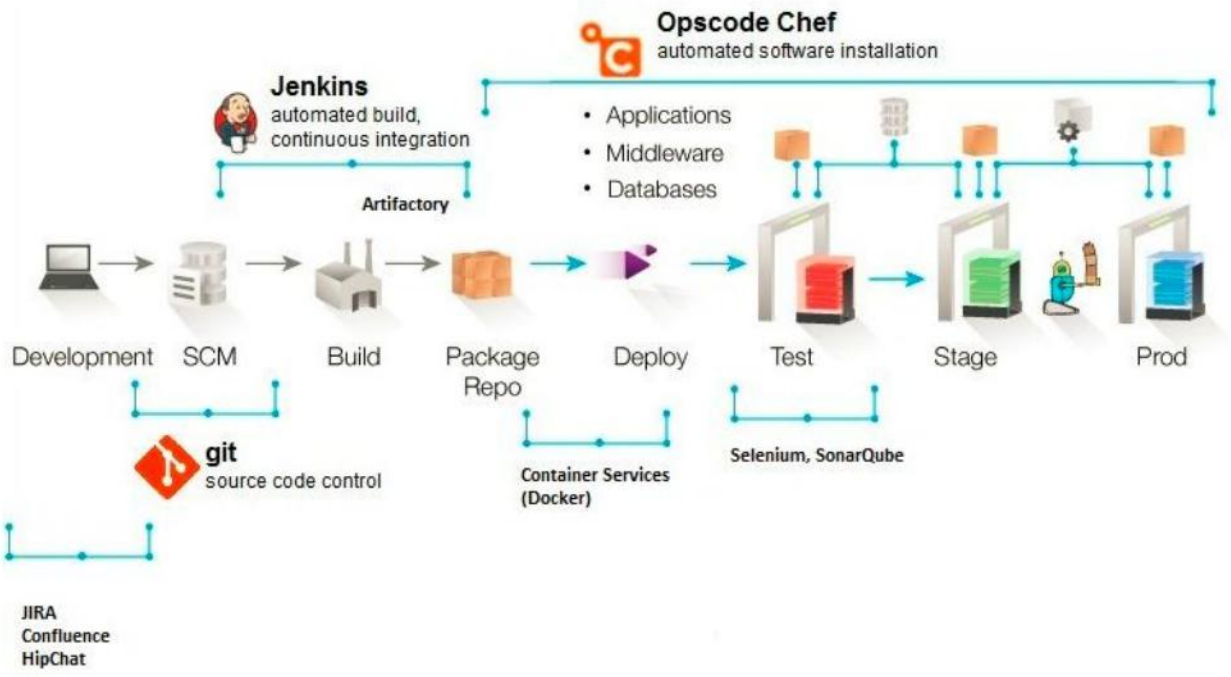

Fig 4.1: Architecture of DevOps System

Architecture of Proposed DevOps System :

The proposed system architecture contains the three phases known as DEV, STG and PRD. These three phases are explained here with some of the tools required to work in these phases.

A. DEV:

Dev is the kind of server in which the complete development of the project is done and uses various tools in this phase. After completion of the development process the project is sent for testing and this is tested in the test server.

B. TEST

Test is also a kind of server in which the testing takes place for the developed project. This testing will have some test cases written for testing of project. Testing can be manual or automated.

C. STG

STG also known as staging in which all the test cases passed and the project configuration and acceptance testing by the customer is done. If the customer agrees then it goes to deployment otherwise it will be changed according to the instructions given by the client.

D. PRD

PRD server is used for the production purpose. After accepting by the customer, the complete project after passing through all the tests will be deployed and released into the market.

TOOLS USED,

1. APACHE MAVEN

It is a Software Management tool works with Project Object Model (POM). It is used as a reporting, building, documenting from an informational central source point and can be used for building Java based projects.

2. GIT SOFTWARE

Git is a version control system for tracking the changes in the local machine or the distributed system among the multiple people in the project. Its primary use is to Source code management (SCM) in software development but it is able to track the any number of changes done in the project. In distributed system its aim is to provide the speed, data integrity, availability and support for the distributed and nonlinear work flows.

3. JENKINS

Jenkins is a continuous integration (CI) server or a tool written in java language. It is open source software for download.

4. ANSIBLE

Ansible is the automation software that provides configuration management, software provisioning and application deployment. A platform was created by the Michael Dehaan, who is the author of the provisioning server application cobbler and a co-author of the funcframework for remote administration. It is the part of Fedora Linux which is owned by Red Hat Inc., and is available for the Red hat Enterprise Linux, Cent OS, Oracle Linux via extra packages for Enterprise Linux (EPEL), Scientific Linux and for remaining OS's also. It is used for automating the tasks like software installation etc.,

In practice, tools, methods, and technologies are seldom deployed on green-field sites and having implemented DevOps for more than a decade now, we believe that the key to successful implementation is to:

• Define a clear target

• Establish a clear transformation plan

• Actively manage the plan execution

DevOps implementation should not be merely perceived as deploying a new tool like CodeStream or Docker. It should be viewed from a bigger perspective and should be planned and executed in an efficient manner. Poorly planned DevOps implementation may result in significantly higher costs. DevOps implementation starts with creating a business case, mapping a way for code migration between environments (considering people, processes, and technology), and placing focus on the target. Understanding the 'As-is' scenario, mapping the 'To-be' scenario, and estimating the benefits of moving to the 'To-be' scenario are critical for success. DevOps implementation should be backed by a strong business case. Every environment does not benefit from full or partial DevOps deployment. For instance, environments with little change requirements may not benefit from DevOps implementation at all. In our experience, many DevOps projects have failed due to the absence of a strong business rationale or a poorly planned start.

CHAPTER 5

FUTURE SCOPE

5.1 Future scope :

The future holds more focus on security and DevSecOps, using AI/ML to improve processes, and more automation. In modern world, this is the era of cloud and its related technologies. In recent years there's been a lot of discussion about cloud, and cloud storage, like AWS, Microsoft Azure, and Google Cloud as well as private and hybrid cloud.DevOps is becoming the standard way of working for businesses. It offers businesses to deliver values in a new way to their customers quickly and efficiently. Businesses are now recognizing the power of DevOps it can bring for operational efficiency and performance.Last year, we have seen many industries adopted the solutions and practices that make up DevOps. According to a Capgemini report, 60% of companies have adopted DevOps or plan to do so in 2018. Due to this, we have seen more and more demand for DevOps expertise across the globe. As DevOps grows, following technical areas are expected to see in the upcoming years.

- DevSecOps with security tools and practices
- DevOps into IoT

There is a huge demand for DevOps engineers and expert professionals as pioneer organizations have made a move to adopt the DevOps culture.Due to the increased span of the DevOps culture and philosophy in the software industry, you can find ample opportunities to align your career goals. With DevOps certification mapped training

You can qualify for the following job roles:

- **DevOps Architect**
- **Software Tester**
- **Security Engineer**
- **Integration Specialist**
- **Release Manager**
- **Automation Engineer**

Certainly, the designations and the career path varies with the level and other academic qualifications. To gain the entry-level DevOps roles, you must be a graduate in computer science, having basic knowledge of coding, QA, testing etc. For senior roles, you must have advanced level certifications and a degree in software designing and systems architecture. 80% of Global Fortunes organizations are expected to adopt DevOps by 2019.DevOps Certifications have been designed to prepare you for two levels:

DevOps Foundation: DevOps Foundation level course lays the foundation for the DevOps basics, concepts and focuses on communication, collaboration and automation of the software development and operations processes. It helps in bridging the gap between the development and operations team and enhances productivity by improving the workflow.

DevOps Practitioner: DevOps practitioner courses are designed to best fit the requirements of the IT roles in modern enterprises. It encapsulates emerging trends of the It industry and addresses the growing needs of the IT software and operations professionals across the world.

By 2019, over 70% of routine development-life cycle tasks will be automated, with an agile DevOps pipeline driving and incubating life cycle and application development intelligence. With the overwhelming growth in the adoption of DevOps technology culture, it has been predicted that by the year 2021, more than 50% of the fortune companies would have the huge demand for DevOps certified professionals for the leading profiles like Delivery Heads and PMOs.

Key Trends in DevOps

Some of the key trends in DevOps are:

- Many organizations recognized DevOps as the new ALM methodology**.**
- Many DevOps/automation tools are being developed.
- DevOps Engineers are paid highest pays in the market.
- Ops professionals are getting their DevOps knowledge in production for testing and automation operations.
- Automation and testing is gaining prominence in the market.

Developers and Operations engineers are two different organizational teams, and if these teams are found to be on the wrong track that it signifies that you need DevOps. We have consolidated here seven signs that can help you in getting the answer whether you need DevOps or not:

- The development team is not able to detect software defects at the early age of its development
- Agile methods are used to speed up the software development process, but as soon as the application goes to production department all methods become ineffective
- Testing and development team members are not able to access resources timely and so the development process delays
- You are not able to identify the exact problems of development, testing, and production department
- Simple human errors are often creating hurdles during the development and deployment process.
- Once the app is in production, developers think that their job is over.
- At the time of the problem, both development and operation teams start blaming each other.

5.2 Advantages and disadvantages:

ADVANTAGES : The benefits of DevOps

Companies that incorporate DevOps practices get more done, plain and simple. With a single team composed of cross-functional members all working in collaboration, DevOps organizations can deliver with maximum speed, functionality, and innovation.

There are technical benefits:

- Continuous software delivery
- Less complexity to manage
- Faster resolution of problems

There are cultural benefits:

- Happier, more productive teams

- Higher employee engagement
- Greater professional development opportunities And there are business benefits:
- Faster delivery of features
- More stable operating environments
- Improved communication and collaboration
- More time to innovate (rather than fix/maintain)

DevOps is hot. This sizzling buzzword is on the tip of every tongue in the IT world, from Development, Testing and QA through IT Operations. At DEVOPSdigest, we have talked a lot about what DevOps is and how you get there – but what's the point? Why go through all this trouble? What advantages can be gained from adopting a DevOps strategy? To explore the answers to these questions, DEVOPSdigest asked experts from across the industry – including consultants, analysts and the leading vendors – for their opinions on the most significant advantages of DevOps.

In recent expert commentaries, we have seen that there are at least 17 different ways to define DevOps, and at least 30 different tools that can support DevOps, so it is no surpirse that this list of DevOps advantages is just as varied and complex.

5.3 Challenges to Implementation :

DevOps and its challenges can be discussed from three perspectives: engineers, managers, and researchers. Engineers benefit from (1) qualifying themselves for a DevOps position, a technically hard task guided by over 200 papers, 230 books, and 100 tools [63]. Engineers also need to (2) learn how to re-architect their systems to embrace continuous delivery. Managers want to know (3) how to introduce DevOps into an organization and (4) how to assess the quality of already-adopted DevOps practices. Managers and engineers, also referred to as practitioners, share the necessity of choosing the best automation toolset. Finally, academic researchers (5) conduct studies to determine the state of practice in DevOps, thereby contributing to discussions among engineers and managers, and (6) educate a new generation of software engineers on DevOps principles and practices.

UNRESOLVED CHALLENGES Throughout this survey, we have shown that overviewing DevOps concepts and tools raises some issues to debate. The previous section also listed some not-so-straightforward implications for practitioners and researchers.

Some of these missteps are all too common:

1.Component versioning and tracking: Software products are complex and contain multiple components, so manually tracking them is impractical. DevOps tools can map components, making continuous deployment simpler and reducing build errors.

2.Environment variability: Software and apps are tested in different environments — dev, test and production and each of these can be configured differently. Tracking apps to environments is complex, and mapping environments to build versions is also complicated. Tools that track build versions and map them to environment parameters help alleviate this issue.

3.Continued reliance on manual processes and infrastructure deployment: Human intervention leads to human error. It also prevents repeatability, which is essential for continuous iteration. When there are hundreds of environments at play, virtualization offers the advantages of speed, cost and flexibility. Tools that automate processes and deployments play a core role in DevOps.

4.Lack of consideration for scalability: DevOps projects usually start small. Unfortunately, many teams proceed as if the project will stay small. In initial design and testing, consider how to make processes scale or even if they *can* scale. If your project can't scale, you haven't designed it correctly.

5.3 Applications :

The future holds more focus on security and DevSecOps, using AI/ML to improve processes, and more automation. In modern world, this is the era of cloud and its related technologies. In recent years there's been a lot of discussion about cloud, and cloud storage, like AWS, Microsoft Azure, and Google Cloud as well as private and hybrid cloud.DevOps is becoming the standard way of working for businesses.

Unless you've been living under the proverbial rock these days you have no doubt heard about DevOps and the groundswell building around it. Today DevOps is being defined, promoted, touted and discussed as the next game changer in IT. Rather than argue about the best

way to define DevOps in a general sense, this article will focus on something hopefully more useful – how do you determine what DevOps means for your organization?

A common goal cited for DevOps is to enable faster release and deployment cycles by taking advantage of agile development methodologies; improved collaboration between business stakeholders, application development and operations teams; and automation tools. Beyond these elements, DevOps also requires a cultural acceptance of the need to focus on the flow of work across teams vs. simply optimizing individual teams and their specific units of work.

There are infinite benefits to earning the highly reputed and globally recognized certification. You can aim to rise high in your software development and operations career. The DevOps career path has a lot in store for you. It is not a highly profitable career, in view of high salaries, but also it is the most challenging role available in the software industry. Taking the salary stats under consideration, a DevOps engineer can mint an average package ranging between $104,000 and $129,230.

5.4 LIMITATIONS:

Given the vast amount of work on DevOps, it was not feasible to conduct an exhaustive search in all available sources. We have not considered academic workshops, and we mentioned only a few books in the field. It would not be practical to try to cover every existing work on DevOps and to condense it all in this single survey. However, by focusing on journal and conference papers, and by applying the snowballing process, we aimed to cover the primary literature in the field. We decided not to expand the query string to avoid artificially favoring some of the DevOps concepts over others. However, the appearance of the keyword “DevOps” was not mandatory in the snowballing process so that we could select vital work tackling highly related subjects such as continuous delivery. We did not find, for example, a paper reporting a case of failed DevOps adoption. We did not address this issue in this study.

When looking into implementing DevOps into your organization, it’s just as important to consider the disadvantages of implementing DevOps as it is the benefits. The decision to implement DevOps cannot be taken lightly. When contemplating a switch to DevOps, one of the first steps is to have a thorough understanding of the challenges that need to be overcome.

The first challenge your organization will encounter is a lack of thorough understanding of the key DevOps advantages and disadvantages. It is vital to understand not just the benefits but also the disadvantages of using DevOps.There is help though–DevOps can offer powerful tools that enable your organization to maximize the way you operate.

Many definitions of DevOps have emerged, and it is important that an organization reach an agreement as to what their definition will be. This agreement is informed by several things that DevOps is often thought to be but is not.

Before the beginning of your DevOps implementation project you'll want to map out standards within your organization and optimize workflows. In addition to there not being one globally accepted definition of DevOps, there is also no standard set of DevOps processes and procedures. Everyone's roles will need to be redefined. New processes and new procedures will need to be developed, deployed, and training provided. Communicating all of these changes across your entire organization is essential in ensuring that your DevOps initiative will be worthwhile and successful.At all times remember that imperfect or half-baked DevOps can be far worse than no DevOps at all.

CONCLUSION

The future of DevOps is very promising, and many more companies are set to accept this methodology. DevOps methodologies are itself changing with new tools and technologies coming in. We hope, in this article, we have discussed ideas around DevOps future scope and how it is going to revolutionize the industry further.In spite of the literature presents an increasingly interest on DevOps, a comprehensive systematic review about DevOps culture does not exist. Even more, the definition of DevOps remains unclear in the scientific literature despite the previous efforts, such as [3, 7], made in this direction. Therefore rather than define "DevOps culture", we prefer to characterize it in order to understand its current status and address further research. This review reveals that the soft side of DevOps is not always confessed among practitioners and researchers but it is always presented in software development [17]. It seems that culture is a term that everyone thinks they understand and it has become a powerful aspect of identity.

In fact, culture is very related to human factors [25]. As a result of the characterization process, we identified 13 attributes. The most frequently attributes in the 23 primary studies were 7: (i) Communication, (ii) Collaboration, (iii) Feedback (Continuous and immediate), (iv) Responsibility (personal/mutual), (v) Improvement cycle, (vi) Sharing Knowledge, and (vii) Transparency. However, there is a relatively scarce number of primary studies related to this topic, although it is slowly growing in the scientific literature.

REFERENCES

[1] X. Bai, M. Li, D. Pei, S. Li, and D. Ye. 2018. Continuous Delivery of Personalized Assessment and Feedback in Agile Software Engineering Projects. In Proceedings of the 40th International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET '18). 58–67. Code: I818.

[2] Armin Balalaie, Abbas Heydarnoori, and Pooyan Jamshidi. 2016. Microservices Architecture Enables DevOps: Migration to a Cloud-Native Architecture. IEEE Software 33, 3 (2016), 42–52. Code: A2.

[3] A. Basiri, N. Behnam, R. de Rooij, L. Hochstein, L. Kosewski, J. Reynolds, and C. Rosenthal. 2016. Chaos Engineering. IEEE Software 33, 3 (2016), 35–41. Code: A76.

[4] Len Bass. 2018. The Software Architect and DevOps. IEEE Software 35, 1 (2018), 8–10. Code: I33.

[5] 2017. xMatters Atlassian DevOps Maturity Survey Report 2017. (2017). https://www.xmatters.com/press-release/ xmatters-atlassian-2017-devops-maturity-survey-report/, accessed on Jun 2018. [52] 2018. How Netflix Thinks of DevOps. (2018).

[6] Capgemini, Technovision 2016, https://www.capgemini.com/blog/cto-blog/2015/11/technovision-2016-build-release-run-repeat Capgemini, TechnoVision 2015, http://www.capgemini.com/blog/cto-blog/2014/11/welcome-to-technovision-2015

[7] Gartner, Cool Vendors in DevOps, 16th April 2014, G00262716, by Ronni J. Colville, Jim Duggan, Jonah Kowall, Colin Fletcher Rodríguez, P., Haghighatkhah, Lwakatare, L.E.,

[8] Journal of Systems and Software. 123, 176–189 (2017). 6. Lwakatare, L.E., Kuvaja, P., Oivo, M.: Relationship of DevOps to Agile, Lean and Continuous Deployment. In: International Conference on Product-Focused Software Process Improvement. pp. 399–415. Springer (2016).