IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

Streamlining DevOps automation for Cloud applications

Sikender Mohsienuddin Mohammad

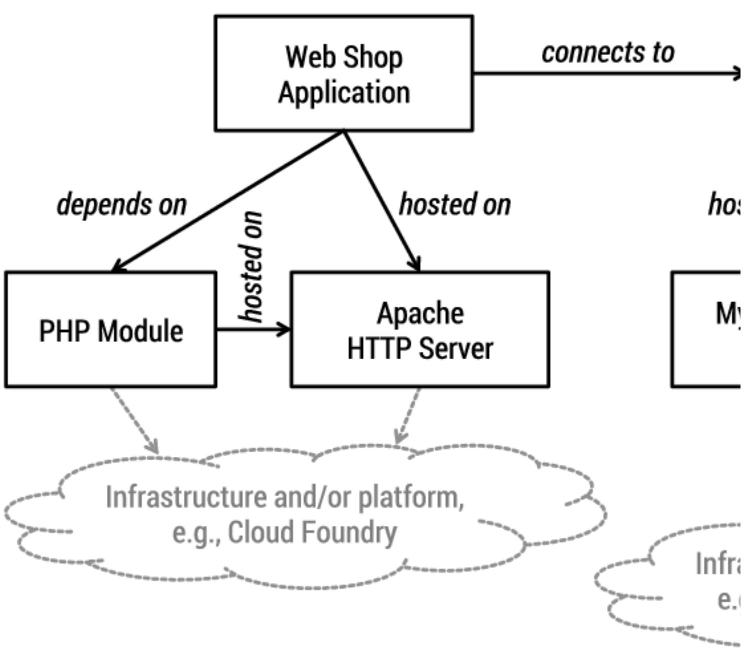
Sr. Associate, KPMG LLP, Department of Information Technology, Wilmington University
419 V street, Apt D, Sacramento, CA USA

Abstract
There are closer links between DevOps and cloud computing. DevOps aims to optimize progress for user specifications to be swiftly incorporated in application output, while the cloud enables automated supply and scaling to manage application changes. IT experts in cloud computing sometimes make errors that they could easily prevent. The question is that best practices are still undefined. These fields are very fresh, but the problem can be more about people than technology, and the challenge is often more difficult for people to overcome. DevOps transforms the face of IT processes and continuously hear new tools and features unveiled by different companies to boost the DevOps pipeline. Electric Cloud is a well-known leader in automation and ongoing delivery of DevOps and recently released a new ElectricFlow version which gives programmers a new data-driven way to onboard apps. The consumerization of IT improves job satisfaction and managing efficiency by encouraging workers to bring their own" goods into the workplace. This streamlines the way activities are conducted and offers flexibility in updates and launches in the toolchain. "Today, ElectricFlow is allowing CD's 'consumerization' to introduce organizations and workers to secure and well-controlled DevOps toolchain, introducing procedures, trends, methods, resources, and other techniques. DevOps and cloud computing do not exclude each other, but from a lay person's viewpoint, the relationship between them is often unclear and confusing. Although the technologies and infrastructure of cloud computing are concerned, DevOps is more concerned with processes and process enhancement. Yet in any organization's journey towards successful digital transformation, both DevOps and cloud computing are important.

Keywords: DevOps, Cloud applications, Automation, Streamlining DevOps, configuration management.

I. INTRODUCTION

However, the rise of cloud computing allowed cloud-optimization alerts and monitoring to help development, IT, and security teams respond to performance challenges or provide new products without an on-site presence [1]. The supervision of network operational performance has in the past required an ongoing on-site network operating center. Coloubdes is a technique that combines engineering and IT operations teams to streamline processes and procedures associated with software creation, system upgrades, and performance issues. CloudOps will assist in identifying an appropriate cloud strategy to promote the success of DevOps efforts. Improving coordination between technology and IT teams enables greater cooperation, aimed at reducing the time taken to deliver applications or upgrades, while also eliminating interruptions such as delays or downtime [1]. Combining development and testing with management and planning into a cohesive DevOps team offers a wider aspect of the system or application's strengths or weaknesses. Improved efficiency and continual effectiveness are two of the key goals of a DevOps approach and can promote these goals by digital transformation leveraging cloud services. CloudOps is short for cloud operations and the term is used to identify and define correct operating procedures to automate cloud IT services. DevOps and traditional cloud-based IT operations resulted in this. The use of the DevOps software has recently expanded rapidly. These resources are designed to create, integrate, and deliver a rapid IT service by enhancing communication among developers and operators. Everyone uses DevOps software to streamline their business processes, be it a start-up or a big organization. In the cloud, most people who use DevOps fight both a cultural and technological challenge. Heart and mind along with technology need to change. Training contributes to comprehension and acceptability. Cloud and DevOps preparation are the major players inside the organization, so you will even need to provide mentoring. You could tell everyone that they have to do it, or you can show them the way. This paper will look at how DevOps automation is streamlining cloud applications to bring about efficiency in the information and technology sector.



II. LITERATURE REVIEW

A. What matters in DevOps

DevOps is a core element of customer loyalty and the faster production of value in addition to its efforts in breaking down barriers to collaboration and coordination between technology and IT operations teams. DevOps is also intended to promote market creativity and the continuous improvement of processes. DevOps' methodology facilitates the faster, cheaper, and easier distribution of company value to the final customers of an enterprise. This interest may be more frequent launches, enhancements, or upgrades to items [1]. These can include the speed at which a new product or feature is released — all with the appropriate quality and safety standards. Or, it may concentrate on the quick finding and resolving a problem or error and then releasing it. DevOps also supports the underlying infrastructure with smooth performance, availability, and software stability, which are first designed and tested and then put into production.

B. Type of DevOps

Several common DevOps approaches can be used by companies to speed up and enhance production and releases of products. We take the form of approaches and procedures for software creation. Scrum, Kanban, and Agile are among the most popular [2]. Scrum explains how a team leader should work to speed up production and QA tasks. Scrum activities include main workflows (sprints, time boxes, regular scrum [meeting]), and defined positions (Scrum Master, owner of a product). Kanban came from the Toyota factory floor efficiency improvements [2]. Kanban sets out that the progress of WIPs on a Kanban board should be tracked. Earlier agile methods of production of applications also have a significant effect on the practice and tools of DevOps. Many DevOps approaches implemented agile programming components, including Scrum and Kanban. Some agile methods are related to a better response to evolving demands and requirements, documentation of user experience, regular support, and continuous customer reviews. Agile also prescribes shorter lifecycles of software creation rather than long conventional methods of waterfall development [3].

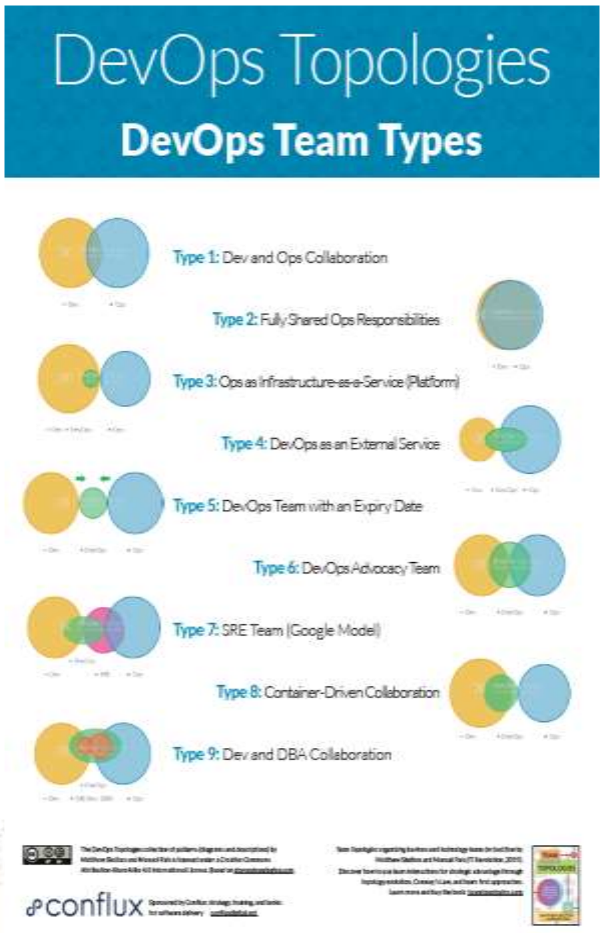

DevOps Topologies
DevOps Team Types
Type 1: Dev and Ops Collaboration
Type 2: Fully Shared Ops Responsibilities
Type 3: Ops as Infrastructure-as-a-Service (Platform)
Type 4: DevOps as an External Service
Type 5: DevOps Team with an Expiry Date
Type 6: DevOps Advocacy Team
Type 7: SRE Team (Google Model)
Type 8: Container-Driven Collaboration
Type 9: Dev and DBA Collaboration
conflux

Fig i: DevOps team types [3]

c. Toolchain in DevOps

DevOps followers also use other DevOps-friendly devices in their "toolchain" of DevOps. The purpose of these tools is to further enhance the software delivery process (or the "pipeline") at different levels, shorten and simplify it. Some such tools often support key automation, collaboration, and alignment concepts of DevOps between engineering and business teams. The following provides a range of instruments used in various DevOps lifecycle processes. The planning process contributes to identifying market interests and requirements. Jira or Git are sample methods for monitoring and handling established problems.

III. DevOps VS CLOUDOPS

DevOps is a methodology that incorporates IT development teams for streamline software development and system updates as well as solve performance challenges, processes, and procedures. Working to improve coordination between technology and IT teams enables greater cooperation, aimed at reducing the time required to deliver applications or upgrades, while also eliminating interruptions such as delays or downtime [4]. Combining development and testing with operations and support into a cohesive DevOps team offers a wider view of the system or application's strengths or weaknesses.

Improved efficiency and continual effectiveness are two of the key goals of a DevOps approach and can promote these goals by digital transformation leveraging cloud services [5]. CloudOps is short for cloud operations and the term is used to identify and define correct operating procedures to automate cloud IT services. It is the result of DevOps and conventional Cloud-based IT processes.

Effective cloud migration patterns help in a better understanding of network resources' current capabilities instead of trying to adapt existing operational methods to the cloud platform [6]. CloudOps, therefore, needs a change in thought around the enterprise. Aside from regular procedures, however, the temporary pain is beneficial for its advantages, which include:

a) Scalability-Cloud technology ensures that without adding additional space or equipment you can increase or decrease capacity at all times. The planning and management of resources and assets are virtualized [7].

b) Automation: Cloud services automate many SDLC processes, contributing to self-healing systems and reducing device or consumer disturbances.

c) Accessibility: the absence of on-site servers enables organizations to run servers from virtually anywhere.

d) Cloud computing enables programs that share similar resources to coexist without contact [8].

e) Backup management: Because data is not centrally located or physically stored onsite, cloud infrastructure automatically incorporates disaster recovery processes.

f) Cost measurements monitor the use of cloud services, making budget allocation simpler.

g) Continuous operations: Cloud-based services can be used constantly and if the right processes exist, applications can be upgraded and used without interruption of service.

IV. AUTOMATION IN DEVOPS

Automation allows the developers, operators, testers, and other stakeholders in DevOps to automate the tasks performed in the creation and deployment of software. Automating activities such as integration, testing, build and software delivery, reduces delays and risks because such activities are time-consuming and error-prone if done manually [8]. Regarding the delivery of a new release, in particular, it can be a complicated work for many applications, as explained by Humble and Farley. For example, it could mean setting up and configuring web servers in the case of web applications [8]. It can also involve fixing any unpredictable errors so that the new release can run properly. Such activities are hard to do manually [9]. Delivering and deploying a new release manually is error-prone and can lead to delays and additional expenses. These problems affect primarily the operations team, further increasing the risk of tension between them and the development team. For all the aforementioned reasons, DevOps relies heavily on automation. One of the core concepts in DevOps automation is the deployment pipeline Humble and Farley. Based on Gill, Loumish, and Riat in and Huttermann in, we understood that the term delivery pipeline is often used instead. A deployment pipeline, according to Humble and Farley in is the process that changes in the source code have to go through to become visible to the end-users. In other words, it is the process of getting the software under development from version control to the production environment. DevOps optimizes the deployment pipeline, by automating every step of the pipeline, to avoid delays during the development process. In doing so, DevOps achieves continuous practices, like CI, CD, CDE, and continuous monitoring [10].

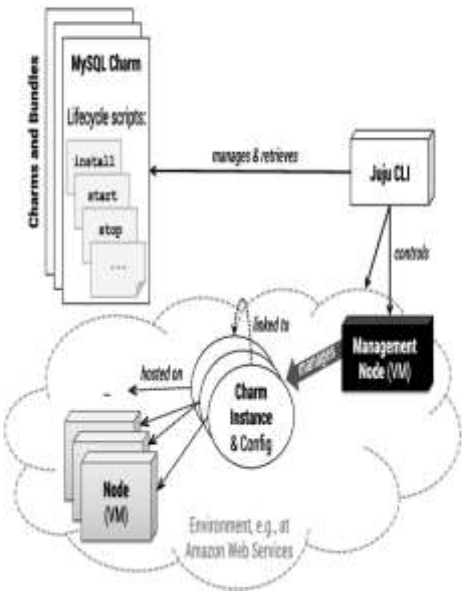


Fig ii: Streamlining DevOps automation for Cloud applications [13]

V. TOOLS IN DEVOPS

A wide variety of tools can be used to automate various steps of the development process such as source code management, build, test, and deploy. DevOps readily makes use of many of them, not just for reducing manual labor and potential errors that come with it, but also as a means to get continuous feedback during the development process so that problems can be quickly detected and fixed, as well to support better communication and collaboration between the teams. From the literature review Huttermann, we understood that tools for the automation of development activities already existed before the advent of DevOps, but many such tools renamed themselves as DevOps tools after DevOps came. Selecting the appropriate tools is important for any DevOps organization. Different tools are used to support different aspects of the development process. Tool selection depends mainly on product features and customer/user needs [11]. We identified various DevOps tools categories from the literature which are listed below:

1. Source code management (SCM) Tools: Source code management is a set of practices to track changes in the source code of the software. SCM is used for versioning and enabling teams to work together from different locations. Version control manages the changes in the source code and controls the collaboration among developers during coding. Source code management can be and should be used to track changes in any artifact that can evolve during the software development process, such as scripts for configuring systems or networks [12]. Through SCM tools developers can share their code and can work from multiple locations. The most used and popular tool in version control is GitHub because it supports distributed systems and is a freemium and an open-source tool. Other popular source code tools include Subversion, Mercurial, and Bitbucket.

2. ***Build Tools:*** Build is the process of preparing an executable program or set of programs out of a set of source code files for a particular software product. This can involve the compilation of the source code into machine instructions for specific computer architecture, but it can involve other steps such as handling dependencies [14]. For example, in case the source code uses an external library. There are many tools available for building software applications, such as Ant, Maven, Gradle, MS Build, NANT. Gradle is the most widely used tool as it combines features from other tools, such as Ant, Maven, Gant, and MSBuild.

3. **Continuous Integration (CI) Tools:** Developers integrate and merge code in an automated way. During this process, the code is submitted to the common source code of the software under

development for building and testing. In case something went wrong, CI tools typically give related feedback immediately to alert the developers. Travis CI, Jenkins, TeamCity, and Codeship fall into this category. Jenkins is one of the most popular tools among them.

4. ***Configuration Management:*** It is defined as the process of establishing and maintaining the consistency of software products throughout their life cycle, by tracking changes to any artifacts used and managing the different versions of each artifact. For example, a user request for a new feature has to be tracked by the developers throughout its journey from a requirement to an implemented feature in the final software product. This may involve tracking any related changes to the source code as well as the test that were written for this feature [15]. There are many tools available for configuration, such as Chef, Puppet, Ansible.

5. Cloud Tools: Cloud tools integrate deployment and collaboration to support DevOps practices, so they are often used by DevOps practitioners. Popular cloud tools include but are not limited to Microsoft Azure and IBM services [16]. Amazon web services provide a variety of services for DevOps. For example, Amazon Beanstalk supports Continuous Deployment.

6. Automated Testing: Testing is an important process in the automated DevOps pipeline. DevOps combines with cloud software and testing which is called Testing as a Service. It improves the collaboration and quality of the software product. DevOps always continuously makes testing in combination with automation. Tools such as Cucumber, Selenium, JMeter, etc. are some of the examples of testing tools [17]. Containers: Containers are generally used for developing the platforms and deploying the applications in the infrastructure. Containers reduce the time between the developing code and 1production. For DevOps people, containers are easy for deploying and maintaining. Docker Containers and Kubernetes for orchestrating the containers are the important ones. Containers are really helpful for DevOps developers as they lower overheads [18]. In microservices, each service is divided into smaller parts called micro. Developers and Operators work together on each smaller part so that coordination among team members will be improved. Containers help microservices to deploy independently as they are operated in isolated environments [19].

8. Deployment tools: The purpose of these tools is to handle the deployment of new releases automatically. After every change in the source code, tests are performed, and the new updated version will be released without manual work [20]. Many of the big software companies Facebook, Netflix, GitHub uses continuous deployment to improve the efficiency of their work [21]. Tools like Capistrano, Jenkins, Ansible are used for deployment.

*9. **DevOps Database Tools Airaj, (2016):*** Database management tools are responsible for handling the data, the metadata, procedures, and the database schemas. In DevOps, databases can be used as code (DbaC), which means they are treated the same as the source code of the software under development and go through the same process, i.e. continuous delivery and continuous deployment [23]. Tools that are often used for database management in DevOps include DBmaestro, MongoDB.

10. Monitoring tools: These tools monitor CPU load, RAM, memory space, and try to solve the infrastructure problems which might affect the business solutions. Nagios is the most used monitoring tools. Other than Nagios, NewRelic, Cacti is also used for monitoring [24]. Monitoring tools are more beneficial to cloud applications. Examples of monitoring tools are New Relic, Graphite.

11. Collaboration Tools: DevOps is mainly established based on trust, open communications, and good collaboration. DevOps encourages teams to share responsibility, ideas, and goals. Tools like Jira, Slack are mainly used for collaboration [25].

VII. CONCLUSION

DevOps is developing theory, paradigm and allows consumers to create new software features or products more quickly and faster. DevOps's approach encourages direct and continuous coordination between the application development teams (Dev) and the IT operating team (Ops) of their partners in collaboration, participation, visibility, and accountability [26]. Every step of the DevOps process has its intimate relationship: from initial planning to coding, design, testing, and release, to implementation, service, and ongoing monitoring. This closer connection between 'DEV' and 'OPS.' This partnership encourages continuous input from customers to further strengthen, develop, test, and deploy. The quicker, more consistent disclosure of the necessary changes or additions is one consequence of these efforts. In some regions, DevOps goals can be divided into four categories: history, authorization, initiatives, and communication (CAMS) and tools from DevOps. Such tools can streamline and coordinate production and operational workflows, automating time-consuming, manual and static installation, creation, monitoring, implementation or surveillance activities previously

REFERENCES

[1] Boehm, B.,2006, "A view of 20th and 21st-century software engineering," in Proceedings of the 28th international conference on software engineering, pp. 12–29.

[2] Riungu-Kalliosaari, L., Mäkinen, S., Lwakatare L. E., Tiihonen, J. and Männistö, T.,2009, “DevOps Adoption Benefits and Challenges.

[3] Tessem, B., and Iden, J.,2008, “Cooperation between developers and operations in software engineering projects,” in Proceedings of the 2008 international workshop on Cooperative and human aspects of software engineering, pp. 105–108.

[4] Erich, F. M. A., Amrit, C., and Daneva, M.,2017, “A qualitative study of DevOps usage in practice,” Journal of Software: Evolution and Process, vol. 29, no. 6, p. e1885.

[5] Humble, J. and Farley, D., 2010, “Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation (Adobe Reader). Pearson Education.

[6] Sacks, M., 2012, Pro website development and operations, Apress, [New York].

[7] Wettinger, J., Vasilios, A., and Leymann, F.,2015, "Automated Capturing and Systematic Usage of DevOps Knowledge." Proceedings of the IEEE International Conference on. IEEE Computer Society.

[8] Gill, A.Q., Loumish, A., Riyat, I., and Han, S.,2010, "DevOps for information management systems," VINE Journal of Information and Knowledge Management Systems, vol. 48, no. 1, pp. 122–139.

[9] Bou G.G., and Gill, A.,2017., "DevOps: Concepts, Practices, Tools, Benefits, and Challenges," PACIS2017.

[10] Nybom, K., Smeds, J., and Porres, I.,2006, “On the impact of mixing responsibilities between devs and ops,” in International Conference on Agile Software Development, pp. 131–143.

[11] Nielsen, P. A., Winkler, T. J., and Nørbjerg, J.,2017, “Closing the IT Development-Operations Gap: The DevOps Knowledge Sharing Framework.,” in BIR Workshops.

[12] Ebert, C., Gallardo, G., Hernantes, J., and Serrano, N. ,2016, “DevOps,” IEEE Software, vol. 33, no. 3, pp. 94–100.

[13] Bell, T. E., and Thayer, T. A.,2006, "Software requirements: Are they a problem," in Proceedings of the 2nd international conference on Software engineering, pp. 61–68.

[14] Laplante, P. A., and Neill, C. J.,2004, “The demise of the waterfall model are imminent and other urban myths," ACM Queue, vol. 1, no. 10, pp. 10–15.

[15] Royce, W. W.,1970, “Managing the development of large software systems. proceedings of IEEE WESCON,” Los Angeles, pp. 328–388.

[16] Royce, W.,1998, “Software project management. Pearson Education India.

[17] Manifesto, A.,2001, “Manifesto for agile software development,”.

[18] Chow, T., and Cao, D.-B.,2008, “A survey study of critical success factors in agile software projects,” The Journal of Systems & Software, vol. 81, no. 6, pp. 961–971.

[19] de França, B. B. N., Jeronimo Junior, H., and Travassos, G.H.,2016, “Characterizing DevOps by hearing multiple voices,” in Proceedings of the 30th Brazilian Symposium on Software Engineering, pp. 53– 62.

[20] Langenhan, D., 2015, "VMware vRealize orchestrator cookbook Master the configuration, programming, and interaction of plugins with Orchestrator to efficiently automate your VMware infrastructure. Birmingham, UK: Packt Publishing.

[21] Huttermann, M., 2012, “DevOps for developers. Apress, 2012.

[22] Smeds, J., Nybom, K., and Porres, I.,2015, “DevOps: a definition and perceived adoption impediments,” in International Conference on Agile Software Development, pp. 166–177.

[23] Debois, P.,2016, “DevOps: A software revolution in the making,” Journal of Information Technology Management, vol. 24, no. 8, pp. 3–39,46.

[24] Jones, C.,2016, “A Proposal for Integrating DevOps into Software Engineering Curricula,” in Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment, pp. 33–47.

[25] Cilliers, F., and Greyvenstein, H.,2012, "The impact of silo mentality on team identity: An organizational case study." SA Journal of Industrial Psychology 38.2, pp.75-84.

[26] Erickson, J., Lyytinen, K., and Siau, K.,2005, “Agile Modeling, Agile Software Development, and Extreme Programming: The State of Research,” JDM, vol. 16, no. 4, pp. 88–100, Oct. 2005.