International Journal of Soft Computing and Engineering (IJSCE)
ISSN: 2231-2307 (Online), Volume-9 Issue-6, May 2020

Understanding SDLC using CI/CD pipeline

Sachin R Doddaguni, Rohith R, Salman Asif S, Manas M N

Abstract**: : *Development of complex and quality software necessitates the use of a development model, so that the development process is efficient, reliable and faster. Software development life cycle (SDLC) is a well-defined and well-organized process used to plan, develop, deploy and maintain high quality software systems. DevOps is one recent addition to SDLC that ensures that the development and operations team collaborate to accelerate the deployment and delivery of higher quality software products. This paper throws a light on how development processes are accelerated using DevOps tactics like continuous integration and deployment (CI/CD) pipelines. however, there are several factors that prevent the organizations from using these approaches. Discovering the evolution of DevOps and its continuous practices, gives a thorough understanding of the importance of the DevOps culture. Manual deployment and testing increase the feedback time of a commit operation. The paper discusses various tools available in the DevOps community that can be used to automate various stages of continuous integration and deployment pipeline, so that the feedback time is reduced.***

Keywords: **SDLC, pipeline, DevOps, Virtualization.**

I. INTRODUCTION

We use complex software every day. life without some of this software is hard to think. Software is getting more complex day by day. competition between multiple software with the same functionality necessitates a faster development process without compromising on the security and dependability of the software. This has led to a situation where completion of projects in a timely manner is imperative to any successful product development. Consequently, various software development life cycle models were developed to meet various non-functional requirements of a project. SDLC stands for software development life cycle. It is a set of practices and ideologies that illustrates how a software project can be developed and completed effectively and efficiently. SDLC ensures that the software project is in accordance with the requirements document and meets the specified goals and objectives. This approach increases software project efficiency and across all software development processes. Several SDLC models are followed in the software development process.

Revised Manuscript Received in April, 2020.
* Correspondence Author
Sachin R Doddaguni, Student Of R.V College of Engineering, Bengaluru, Karnataka. India.
Rohith R, Student Of R.V College of Engineering, Bengaluru, Karnataka. India.
Salman Asif S, Student Of R.V College of Engineering, Bengaluru, Karnataka. India.
Manas M N, Assistant Professor Department of CSE, R.V College of Engineering, Bengaluru, Karnataka. India.

The SDLC model consists of a series of steps to ensure a successful product delivery. some of the prominent SDLC models are waterfall model, iterative model, spiral model etc. The present scenario requires a faster development process. The set of software practices and ideologies that insists higher level of collaboration between the two teams of development and operations, is known as DevOps [1]. DevOps is a culture where transparency is encouraged, i.e. various teams facilitate an open culture . It is often quoted that DevOps encourages open culture practices as in any open source project [9]. The development and operations team work together to catalyse the development process. DevOps is a variation of agile and lean models. To summarize, DevOps encompasses a set of software development techniques aimed at reducing the actionable process of software design changes. The main principles of DevOps is described by the CAMS acronym which stands for culture, automation, measurement and sharing. Adherence to these concepts is accomplished through a variety of DevOps techniques, including continuous development, regular updates, QA automation, testing of ideas as early as possible, and in-team collaboration. The advantages of DevOps over agile methodologies is presented further in the paper.

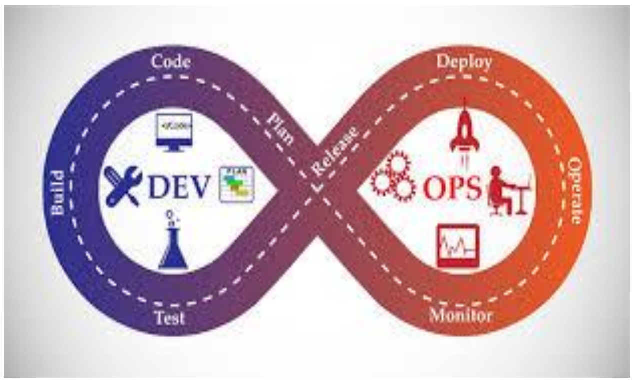

Fig1: DevOps life cycle

Continuous practises is one of the outcomes of incorporating DevOps culture in the organization. The two continuous practices used are continuous integration and continuous deployment. Continuous delivery is also a continuous practise but is left to the organization as a choice. The paper further discusses various continuous practices and its evolution. The remaining part of our paper is presented as follows: Section II presents DevOps and Various Continuous practices. Section III describes a typical CI/CD pipeline. Section IV discusses enabling techniques and technologies in the DevOps world. Section V and VI concludes the paper.

II. DEVOPS AND CONTINIOUS PRACTICES

DevOps is a set of practices that insists a reduces developer-operations gap in any organization [5]. DevOps is a relative newcomer compared to other SDLC versions. In order to successfully initiate DevOps, developers would need to experience a cultural change. It is where Technology combines a modern SDLC platform with Operations. DevOps as a software practice is proved to improve the software quality and accelerate the development process [2]. Cross-functionality, mutual roles, and trust are encouraged within a DevOps framework. DevOps expands the continuous development principles of the agile movement to continuous integration and release. In order to support continuous updates, DevOps supports the automation of changes, configuration and release processes. DevOps applies the agile concepts to the entire development system of applications. The main goals of DevOps have identified in [6] as following:

- Deliver substantial market value through continuous and high-quality delivery of services [6].
- Encouraging simplicity and agility W.R.T to development practices and human involvement [6].
- Development and operations team collaborating together with trust and shared ownership [6].
- Prepared for dynamic changes [6].

DevOps ensures all these goals are reached, by performing continuous practices at various phases of the software delivery life cycle. The various continuous practices incorporated in the DevOps SDLC model are:

A. CONTINUOUS PLANNING

Planning is the first phase in any SDLC model. project is planned before moving into 5the next phase. Agile methodology ensures that the software product is flexible to changing requirements and conditions. DevOps Encourages this by having a prioritized product backlog and a feedback system where customer feedback is recorded and incorporated into the plan after the small product delivery.

B. CONTINUOUS INTEGRATION

Continuous integration refers to adopting to changes in the early stages of development. Any change done to the code is is shared immediately to the team and validated for correctness. This means that changes to the code is not kept local for a long period of time and is committed immediately to the central repository. If the commit is successful, the build is generated to a central repository. The build process is automated to be triggered as soon as a change is detected, and the automated process runs various tests against the changed code to check for correctness. This whole process repeats itself after each change committed.

C. CONTINUOUS TESTING

Continuous testing and testing in the early stages reduces the effort required to diagnose and correct the errors. A test suite is automated to be executed against each build generated. Every test case possible is automated in the test-suite. Lot of tools are available to automate the testing process. Consequently, the cycle time of the pipeline is reduced and further accelerates the software delivery process. Continuous testing evaluates the risk of the immediate release.

D. CONTINUOUS DEPLOYMENT

Continuous deployment is a critical stage in a software delivery pipeline. The traditional method of manually checking whether the build works in another environment or not is a time-consuming process and error prone. Deploying code to production environment can be automated. The provisioning of hardware can also be automated as depicted by the term “infrastructure as a code” [5]. continuous deployment and continuous delivery are two different stages of the pipeline although they both are abbreviated as CD. continuous delivery is a manual stage after a successful deployment is done.

III. III.CI/CD PIPELINE

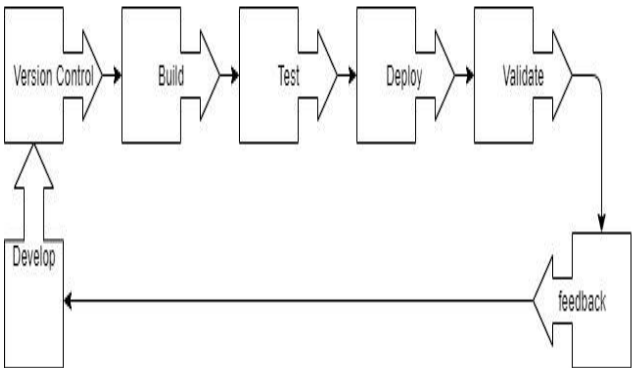

Fig2: stages of a CI/CD pipeline

Fig2 depicts the stages of a CI/CD pipeline. Each stage is performed as a continuous process. Various tools are used for automation of processes at various stages of the pipeline. The figure also implies that the DevOps is derived from agile techniques. After production level deployment is done, customer feedback is acquired, and changes are made in accordance to any changed requirements.

IV. TOOLS AND APPROACHES

Test Driven development and continuous integration has accelerated the software delivery process. Various tech breakthroughs have enabled an easy shift to the DevOps culture in any organization. various enabling techniques and technologies are described in this section.

A. MICROSERVICES ARCHITECTURE

Many companies that have adopted DevOps ideologies have a strong desire towards speed od development and scalability. Companies are extremely benefited when DevOps is combined with microservices architecture. Microservices architecture emphasizes a paradigm shift from monolithic architecture to decomposed solutions. In microservices architecture, independent development and modularized application ensures division of labour. Independent development enables multiple languages and tools to be used in the software development cycle. Development cycle based on microservices architecture and continuous practices reduces the feedback time by a substantial amount with more frequent software releases. Decomposed solutions ease the maintainability and testability features of a software product. Microservices architecture promotes decentralized operations in an organization.

Multiple independent services ensure the scalability of the application.

B. VIRTUALIZATION AND CONTAINERS

Virtualization is a process that abstracts various resources like CPU, memory etc and creates a pool of such resources. Virtualization ensures an efficient application deployment method by acquiring required amount of resources from the pool of resources. A virtual machine is an abstraction of IT hardware. Virtualization simulates a hardware environment without the need for separate hardware components. Containers on the other hand abstract OS level components in addition to hardware virtualization. OS level kernels are virtualized to mimic runtime environments. Containers are usually preferred over virtualized machines [8]. Containers have performance advantages over VMs. Containers have relatively lower start-up times when compared to VMs because of the OS level abstraction. It never means that containers have completely replaced virtualization. Moreover, techniques like containers inside a VM are used as shown in Fig3.Both containers and VMs are used so that resources are used as efficiently as possible [8].

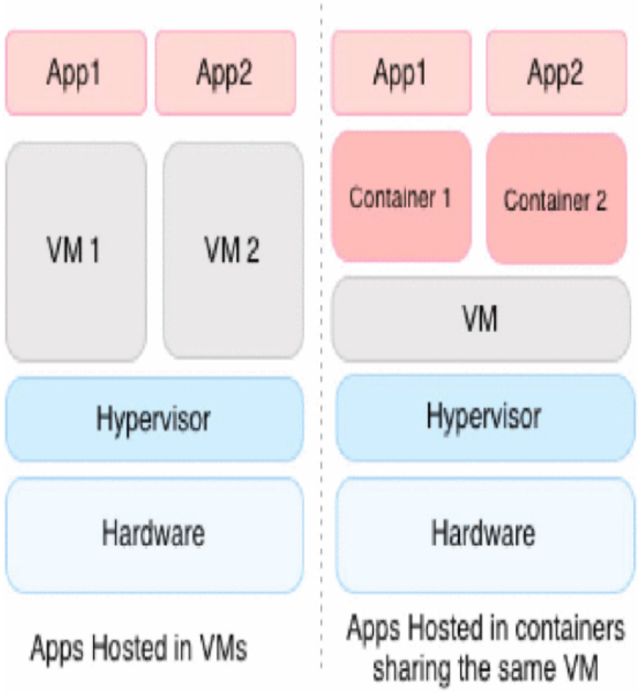

Fig3.Containers and VMs

C. INFRASTRUCTURE AS CODE

Setting up infrastructure is a phase in the software delivery pipeline and involves configuring hardware components needed for product deployment. It entails configuring virtual machines, configuring networking among the VMs and installing software drivers required for the operation of the software product. DevOps emphasizes automation at every stage of the pipeline. Infrastructure Deployment and provisioning can be automated by using a set of scripts. Here the infrastructure is also represented using a set of scripts. This practice is called "infrastructure as code".

D.BUILD AND CI TOOLS

Tools reduce the need for manual processing and human intervention in software development cycle. Build and CI tools are the automation tools in the Build phase of a DevOps lifecycle. Build operations include compiling code with dependencies and running tests against the compiled code. Some of the widely used tools in the build phase of DevOps are Maven, Gradle, Apache Ant etc. Apache Ant is usually used while building opensource applications. It uses non-procedural languages like XML for defining build processes. Maven improves over apache ant and uses XML to define the project nature. Gradle unlike maven and apache ant, uses programming language-based tools to define the build process.

There are many CI tools to choose from. Selecting appropriate tools is always the decision of the organization. Jenkins is a widely used java-based CI tool. Jenkins is a opensource tool with a lot of plugin options. Bamboo is a CI tool from Atlassian which provides a lot of advantages when used with other Atlassian products.

E. TOOLS USED IN DEPLOYMENT PHASE

Automation in the deployment phase involves treating infrastructure as code. In this phase the production environment is simulated so that production and development infrastructure configurations are similar. Like any code, the infrastructure can be version controlled and tested. Some of the most widely used configuration management tools are puppet, chef and ansible. Both puppet and chef use client server architecture and are based on ruby bases domain specific language. Ansible unlike puppet and chef doesn't use a client server architecture. Configurations are pushed to client machines using a secure shell protocol. Ansible uses simple YAML based configuration files.

E. TOOLS FOR INFRASTRUCTURE OPERATIONS

In DevOps Development and Operations team collaborate to make frequent software releases. Various tools are used to ensure reliability and stability of infrastructure operations. Tools are available for logging and monitoring of infrastructure processes. Logging is used to debug any errors in complex applications. Modern applications emphasize on including log messages at various stages of application processing. Log messages from bare metal servers, virtual machines and complex applications are uploaded to cloud and are analysed. Graylog2 is one such open source analyser. Monitoring tools are used to monitor various variables W.R.T infrastructure like CPU load, RAM usage, number of jobs running, time taken for each job, network stats etc. tools like Nagios are used to monitor the safety and warn administrators of the system constantly when issues are identified so that the administrators can take corrective measures. Nagios is an open source monitoring tool. Nagios provides a web interface with graphical representations of various variables.

V. RESULTS

Continuous practices have changed the way software processes are carried out and have changed the process of software delivery. DevOps has enabled the use of continuous practices with the use of agile techniques and ideas. Fig3 shows the continuous integration trend from the year 2004.Continuous integration has evolved in the software development market and has been the most widely used SDLC model recently.

Continuous practices enhance the consumer satisfaction in the software development process. DevOps incorporating continuous practices is an extension of agile SDLC model. Fig4 indicates the spike in usage of Jenkins tool from the year 2004. Jenkins is an opensource CI/CD tool and is well known among DevOps engineers. Jenkins has recently been replaced by lot of other tools .

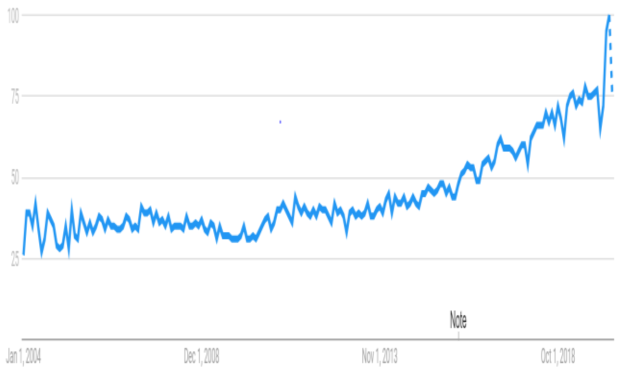

Fig3.Continuous integration

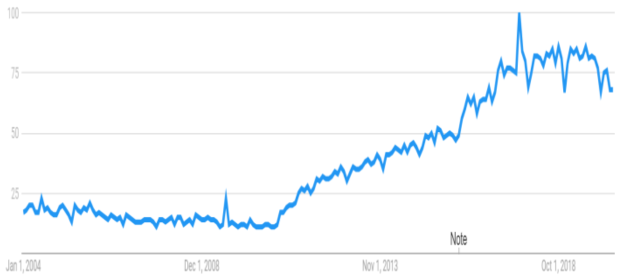

Fig4.Jenkins tool

VI. CONCLUSION

The paper illustrates how a CI/CD pipeline implements various DevOps principles. DevOps as a SDLC model is improvised version of Agile techniques. The paper describes how tools are used to automate various phases of the pipeline and how automation reduces the feedback time of a software release. It is not only DevOps ideas but also many enabling technologies and techniques that have enabled a cultural shift.

REFERENCES

1. L. J. Bass, I. M. Weber, and L. Zhu, DevOps - A Software Architect's Perspective., ser. SEI series in software engineering. Addison-Wesley, 2015.
2. P. Perera, R. Silva and I. Perera, "Improve software quality through practicing DevOps," 2017 Seventeenth International Conference on Advances in ICT for Emerging Regions (ICTer), Colombo, 2017, pp. 1-6.
3. E. Dörnenburg, "The Path to DevOps," in IEEE Software, vol. 35, no. 5, pp. 71-75, September/October 2018.
4. M. Virmani, "Understanding DevOps & bridging the gap from continuous integration to continuous delivery," Fifth International Conference on the Innovative Computing Technology (INTECH 2015), Pontevedra, 2015, pp. 78-82.
5. Understanding DevOps – Infrastructure as
6. a Code, https://sdarchitect.wordpress.com/2012/12
7. /13/infrastructure-as-code/.
8. D. L. Farroha and B. S. Farroha , "A Framework for Managing Mission Needs, Compliance, and Trust in the DevOps Environment," In Military Communications Conference pp. 288-293, 2014.
9. T. Perumal, M. N. Sulaiman and C. Y. Leong, "Internet of Things (IoT) enabled water quality monitoring system", 2015 IEEE 4th Global Conference on Consumer Electronics (GCCE), Osaka, 2015, pp. 86-87.
10. R. Dua, A. R. Raja and D. Kakadia, "Virtualization vs Containerization to Support PaaS," 2014 IEEE International Conference on Cloud Engineering, Boston, MA, 2014, pp. 610-614.
11. C. Ebert, G. Gallardo, J. Hernantes and N. Serrano, "DevOps," in IEEE Software, vol. 33, no. 3, pp. 94-100, May-June 2016.

AUTHORS PROFILE

Sachin R Doddaguni is a student .at R.V College of Engineering. he is a B.E student, currently studying in 8th sem, computer science domain. His area of interest includes applications of cloud computing and machine learning.

Salman Asif S is a student .at R.V College of Engineering. he is a B.E student, currently studying in 8th sem ,computer science domain. His area of interest includes machine learning.

Manas MN is working as Assistant professor in the department of CSE, rvce. His area of interest is Computer Architecture, Machine learning Data mining and analytics.

Rohith R is a student .at R.V College of Engineering. he is a B.E student, currently studying in 8th sem, computers science domain. His area of interest includes cloud computing and networking.