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

DevOps with Modern Software Development

Salman Asif S, Anala M R

Abstract*: DevOps is the form of software engineering practices for software used to have better collaboration among teams and help deliver the business product in a timely manner and meet the business requirements. The growing demands from the customers and the competition among similar products give rise to better principles in DevOps than ever before. There exists current trend in industry to adopt cloud based solutions as compared to other solutions. This gives rise to certain evolution in the way DevOps is done in industry. This main purpose of this paper is to address the changes in the way DevOps is carried in industry and the scope of it due to evolving industry needs and newer emerging tools. The paper is structured in to provide an introduction to modern DevOps and the necessity of adoption of DevOps, DevOps applied in SDLC phases, tool concerning DevOps, Evolution of DevOps in modern software development and conclusion.

***Keywords*: DevOps, Agile, Continuous Integration , Continuous Delivery, Agile**

I. INTRODUCTION

Technology has been a vital feature of the industry in past two decades. Scientists in software engineering are implementing new forms of languages, architectures, technological systems[3]. Overall, the emphasis is mainly on optimizing product engineering processes by increasing reusability, comprehensibility of specifications, time and cost-effectiveness of product delivery, and many other quality-enhancing functions.

Consistent high quality is the most critical necessity, either in a discrete production process or in a continuous production method, more focus needs to be given to producing a product that is satisfactory to consumers. According to International Standard Organization ISO 9000 quality is characterized as the entirety of the characteristics of the product as a whole in order to satisfy the specified and implied needs in accordance with its ability[6].

DevOps combines the two realms of development and operation through automated growth, delivery and network management. This is an organizational transition in which cross-functional departments collaborate on ongoing technical functionality implementations instead of fragmented organizations conducting tasks independently. [1]. These practices involve four main concerns:

- Quick change into production.
- Employ automated testing to find errors.
- Reducing or removing errors that occur during the deployment process.
- Looking into and fixing faults in the system.[4].

This approach helps to deliver software products much faster when compared to traditional IT way of delivering where development and operations teams are separated and each have distinct responsibilities. DevOps integrates well with Agile methodology of development in continuous small sized iterations. Agile is concerned with small rapid quick releases with feedback from the customers and DevOps is considered to have a better collaboration between software development teams and operation teams. Together the combination of agile and DevOps principles allows to have an effective strategy to have a set up to continuous delivery in small upgrades with faster feedback and scope for change in the product regarding scalability, security, software testing, incident management [1].

In a highly competitive world, it is obvious that there should be versatile and simpler approach to consumer demands and service distribution for any product to succeed on the market[2]. Various trends have been observed during the past few years in software industry one of the largest trend is the movement of services to the cloud and increasing adoption of microservice architecture to facilitate scalability and optimize the use of resources.

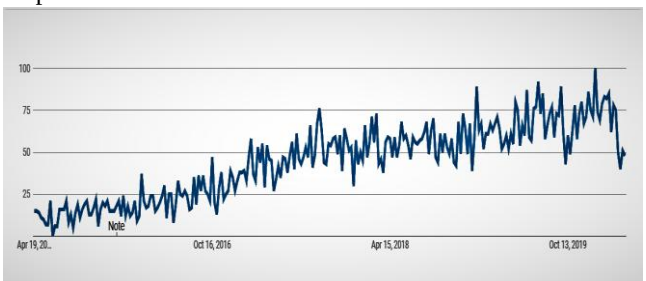

Fig1: Google Trend Data for rise in DevOps searches('Note' represents data collection system received enhancement)

II. DEVOPS IN SDLC PHASES

The software development lifecycle consists of the following phases

1.Continuous Planning
2.Continuous Integration
3.ContinuousMaintenance
4.Continuous Deployment
5.Continuous Testing
6.Continuous Monitoring

Revised Manuscript Received on May 05, 2020.
* Correspondence Author
Salman Asif S*, Computer Science dept, RV College of Engineering, Bengaluru, India. Email: salman1798@gmail.com
Sachin R Doddaguni, Computer Science dept, RV College of Engineering, Bengaluru, India. Email: sachinrd199@gmail.com
Anala M R, Computer Science dept, RV College of Engineering, Bengaluru, India. Email: analamr@rvce.edu.in

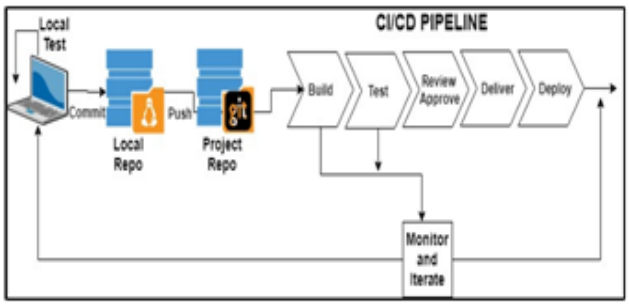

Fig2: DevOps Colloboration

A. Continuous Planning

Planning is the first phase in SDLC and is the most crucial initial step taken to ensure that product that is being developed is relevant to customer requirements and business needs. Traditionally project planning involved separate decisions taken by each team such that it would benefit the respective team in the most effective way. Modern software planning focuses on the combined final output that is most suitable for the customer as well as the business. Triaging is followed along with agile methodology where teams each representing from product, development and design discuss to form a collective decision and subsequently plan for a short duration typically lasts from 1-4 weeks. DevOps helps to do this by constant feedback channel with customers and the opportunity to assign priority to the commodity backlog all the way, bringing the market context directly into account. There is a constant mechanism for preparing a specific portion – implementing – getting input – responding to input and changing schedule as necessary and the cycle repeats[5].

B. Continuous Integration

Continuous integration of the code is the foundation of the continuous development process. Continuous code development also constructs software from libraries, meaning that development issues can be found early. There are a number of digital building devices like Ant, Travis CI and Jenkins[7]. Continuous integration is essentially about early deployment, don't hold improvements confined to your workplace for a long period, then communicate the adjustments with the team and verify how the application acts continuously[5]. Not only do they collaborate inside component departments, but they incorporate across component borders at product development level. Furthermore, method optimization at this level relates to the accomplishment of automation in such a way that, as soon as the developer provides the update, the construct systems senses it and activates the create, executes stability checks and positions the construct on the server. There must be a constant repeatable process during the manufacturing cycle. [5].

C. Continuous Maintenance

The goal of continuous maintenance is to increase production, resilience and quality through automation. Continuous maintenance is working to boost accuracy and reliability in an IT organization. Continuous maintenance will help enforce the strategy of program branching. Such a program allows the pruning of divisions that are no longer required[7].

Continuous maintenance dynamically stores data frequently. Archival doesn't only erase objects forever. Only objects can be classified as historical and no longer important to the production of the commodity. Continuous maintenance is especially relevant for organizations that utilize various forms of repositories to serve specific technologies. Continuous maintenance shall insure that all forms of repositories are preserved[7].

D. Continuous Deployment

Continuous development is a software innovation strategy, with departments working to create useful applications over a limited amount of time and make sure it is usable efficiently at all time[8]. DevOps methodology suggests that the whole provisioning of resources will be stored as code in the archive of source code – the idea is named Infrastructure as a code (IAAC)[5].

E. Continuous Testing

Software testing is a crucial component of software production systems that businesses depend on to produce goods and services with good quality. Time standards for compilation and testing product improvements place a lower limit to how easily the organization can distribute technology to customers[9].

Manual validation procedures have to be tested for automated possibilities and in certain situations there should be options to automate the same. The product distribution mechanism should be able to run the test suite on any dynamically created a pplications without any user input, allowing strides towards the eventual aim of being able to effortlessly deliver a better quality product release [5].

F. Continuous Monitoring

Software performance monitoring is to analyze the software's behavioral details at runtime, and assess if system activity is compatible with intended actions during system operation. It will maintain the software behavior's integrity, which can be used to identify which localize software faults. Off-line runtime monitoring is online system path collection method, while evaluating the off-line results. Online runtime analysis is the method of obtaining the direction of the system and reviewing the data at run time. Intrusive runtime monitoring is the method of injecting code into a software program to both track the actions of the device and evaluate the data gathered. Non-intrusive control of runtime is the mechanism that monitors a machine from outside[10].

III. MODERN DEVOPS TOOLS AND APPROACHES

Certain tools are have been developed over a period of time to assist with devops process for organizations to ship code faster to production and allow customer to have more access to newer features and Right usage of tools allows companies to gain a strategic advantage over certain businesses and help address customer requirements efficiently. Tools are broadly classified in the following format.

A. Automatic Commit and deploy
B. Automated Testing
C. Version Control
D. Containerization or Virtualization

A. Automatic Commit and deploy

Some of the challenges faced until automatic and continuous integration was adopted were:

1. Developer had to wait much longer for the test results, and therefore lose a ton of time that should have been invested on research and creating a new product.
2. Developers tend to go into all of the source code and repair the problem that they find.
3. No constant input from the development department or the evaluation department at all times.

Consider the Nokia case study that followed the Nighttime Construct strategy, where the source code was taken from the repository only at night. Code is pulled over each commit after the integration and then immediately create, check accompanied by correct feedback. Now the developer would need to go over the last commit's source code and not the entire software[11]. Tools like Jenkins, Atlassian Bamboo, Gitlab CI/CD are used for build plans.

B. Automated Testing

Software testing is a quality management control operation that includes implementing software under test (SUT) in an effort to detect flaws, such as glitches. Testing has traditionally been generally limited to the latter stages of the life cycle of software developmen. and often skipped altogether as resources or funds have been limited but agile methodology has changed the approach. One essential part of research is how to assess whether or not the SUT performance is right[14].

Tools like Selenium, Cucmber, Cypress, Appium help in Automated Testing to remove manual testing which is error prone and does not cover all the cases. Therefore certain manual testing will be maintained for exception or edge cases.

C. Version Control

A Version Control System (VCS) handles modifications to any text, but is usually used to track and maintain source code or scripts applications. Furthermore, a VCS is often referred to as revision control or source management. Two forms of version control system exist: centralized VCS and distributed VCS. Client-server model is a centralized version management framework. A copy from the shared server is reviewed by customer. Both these details and records are collected from the central archive and recovered there. The centralized version management framework, on the other side, is a peer-to-peer solution[12].

GitHub is a repository that serves Git for the managed documents. This provides a free or charge-fee version control server, a proprietary server that can be accessed. Git should be combined with GitHub and the archive can be viewed from anywhere via the Web, instead of maintaining a single registry server[12] .

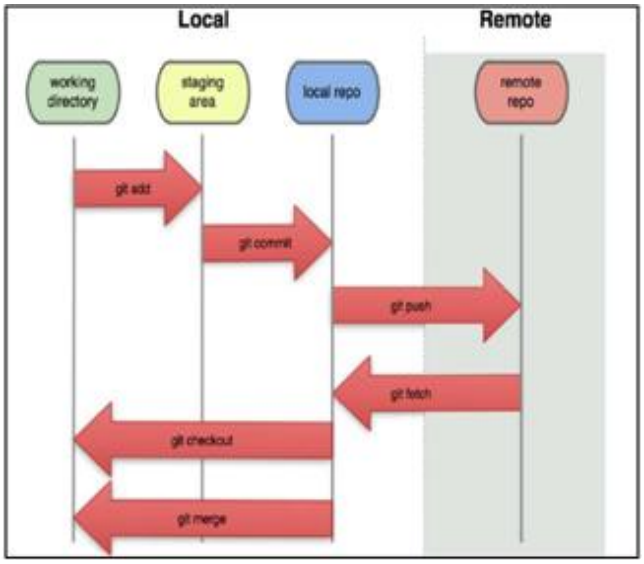

Fig. 3: GIT Working Flow

D. Containerization or Virtualization

Containerization should be used as an alternative to the traditional virtualization because it can reduce operating costs and thus increase the usage of data centers[13].

Docker is a platform that uses a Docker server to containerise. Hypervisors are found with virtual environments, and Docker lets operate many versions with software on the same computer. Every of these is performed in an independent container system. The program is bundled in the jar, which can thus be used on every system that has enabled the Docker driver. A container is a closed computing framework which contains all the data which resources that the program requires to run properly. It is possible to run several containers on the same server, and share the resources[12].

IV. RESULTS AND DISCUSSION

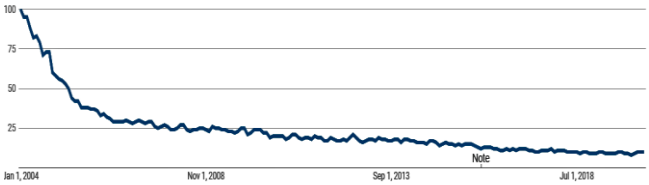

Fig4: Google Trend Data for Virtual Machine searches('Note' represents data collection system received enhancement)

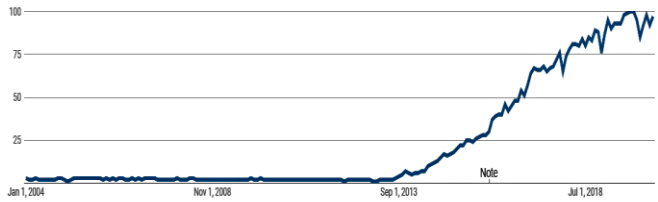

Fig5: Google Trend Data for Docker searches('Note' represents data collection system received enhancement)

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

As it can be seen from the above graph that the software industry is moving away from virtual machines and the trend for virtual machines has fallen to almost complete standstill, It is also interesting to note that the trend for adoption of containerization techniques has increased over the past decade. Result of this is because of having easy to integrate and implement applications which run on docker which uses the containerization technique.

Fig6: Google Trend Data for Git searches('Note' represents data collection system received enhancement)

From the above graph which shows the result of change in version control strategy with higher adoption rate for Git in industry and is becoming a standard across the world.

From the above results we can conclude that the way DevOps is being approached is changing which in turn is resulting in a more efficient and easy to develop standard across the industry.

V. CONCLUSION

For brief, DevOps provides a systematic solution to the full distribution network of enterprises. DevOps integrates developers and operations department to merge individuals, procedures and technologies with an agile, efficient and expense-effective automated software production.

REFERENCES

1. C. Ebert, G. Gallardo, J. Hernantes and N. Serrano, "DevOps," in IEEE Software, vol. 33, no. 3, pp. 94-100, May-June 2016.
2. M. Rajkumar, A. K. Pole, V. S. Adige and P. Mahanta, "DevOps culture and its impact on cloud delivery and software development," 2016 International Conference on Advances in Computing, Communication, & Automation (ICACCA) (Spring), Dehradun, 2016, pp. 1-6.
3. 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.
4. L. Bass, "The Software Architect and DevOps," in IEEE Software, vol. 35, no. 1, pp. 8-10, January/February 2018.
5. 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.
6. P. Jain, A. Sharma and L. Ahuja, "The Impact of Agile Software Development Process on the Quality of Software Product," 2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, 2018, pp. 812-815.
7. C. Pang and A. Hindle, "Continuous Maintenance," 2016 IEEE International Conference on Software Maintenance and Evolution (ICSME), Raleigh, NC, 2016, pp. 458-462.
8. L. Chen, "Continuous Delivery at Scale: Challenges and Opportunities," 2018 IEEE/ACM 4th International Workshop on Rapid Continuous Software Engineering (RCoSE), Gothenburg, Sweden, 2018, pp. 42-42.
9. K. Herzig, "Testing and Continuous Integration at Scale: Limits, Costs, and Expectations," 2018 IEEE/ACM 11th International Workshop on Search-Based Software Testing (SBST), Gothenburg, 2018, pp. 38-38.
10. L. Gao, M. Lu, L. Li and C. Pan, "A survey of software runtime monitoring," 2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS), Beijing, 2017, pp. 308-313.
11. A. Agarwal, S. Gupta and T. Choudhury, "Continuous and Integrated Software Development using DevOps," 2018 International Conference on Advances in Computing and Communication Engineering (ICACCE), Paris, 2018, pp. 290-293.
12. J. Shah, D. Dubaria and J. Widhalm, "A Survey of DevOps tools for Networking," 2018 9th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), New York City, NY, USA, 2018, pp. 185-188.
13. J. Watada, A. Roy, R. Kadikar, H. Pham and B. Xu, "Emerging Trends, Techniques and Open Issues of Containerization: A Review," in IEEE Access, vol. 7, pp. 152443-152472, 2019.
14. D. Towey and T. Y. Chen, "Teaching software testing skills: Metamorphic testing as vehicle for creativity and effectiveness in software testing," 2015 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE), Zhuhai, 2015, pp. 161-162.

AUTHORS PROFILE

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.

Dr. Anala M. R., is an Associate Professor at R.V College of Engineering. She has over 18 years of experience in teaching. Main area of research interest is Computer Architecture, High Performance Computing, Distributed Systems and Parallel programming. She has guided 40 UG projects and 20 PG projects. She has many publications in international journals and conferences.