Journal of Engineering Design and Analysis
Volume 3, Issue 2 - 2020, Pg. No. 104-105
Peer Reviewed Journal

Article

DevOps

Mayank Khator

TCS, Bangalore, Karnataka, India.

I N F O

E-mail Id:
mayankkhator@gmail.com
Orcid Id:
https://orcid.org/0000-0003-3469-6168
How to cite this article:
Khator M. DevOps. *J Engr Desg Anal* 2020; 3(2): 104-105.

Date of Submission: 2020-11-11
Date of Acceptance: 2020-12-23

A B S T R A C T

All were focused on a central issue that was ending up more pervasive over a developing number and assortment of IT associations: How would we conquer any hindrance amongimprovement and activities for the advancement of the business? Let me put in a simpler way. Have you seen Google or Amazon or Facebook or any of these kinds of giants go down for their work/ enhancements? Downtime (Time in which Application is inaccessible) of such giants can cause them in millions. Hence to sort with this a culture of continuous integration and Development was alluded as Development plus Operation (DevOps). let's emphasize on the expression "DevOps" which regularly alludes to the rising proficient development that backers a community-oriented working connection among improvement and IT activities, bringing about the quick stream of arranged work (i.e., high convey rates) while at the same time expanding the dependability, steadiness, flexibility, and security of the creative environment.

Keywords: Build, Test, Production, DevOps

Introduction

There's always been a feud between the operations team and developers; both accuse the other of being inept. Until you live in the shoes of both worlds for a while, you'll keep thinking this way. hence DevOps is the sophisticated unification of Operations and Development. A typical DevOps workflow looks like this:

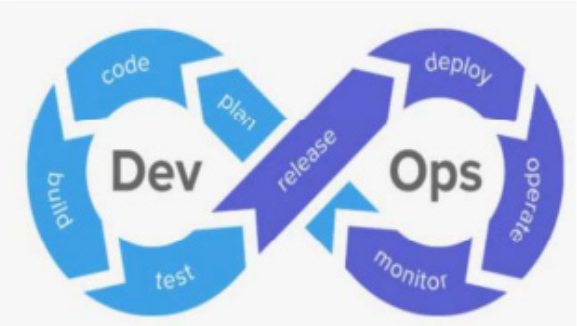

Figure 1

Plan for the next release Code new features Build the code for testing Test the new and old code Release the code to the production staging area Deploy the code to production Monitor for issues Get user Feedback&Repeat with the feedback gained. DevOps was initially viewed as a procedure discovered just in huge organizations (e.g., Netflix Google) and new businesses. Inside the DevOps people group, these sorts of associations are frequently alluded cloud-locals," "conceived in the cloud," or "unicorns.". Today, the usage of DevOps can be found in three fundamentally extraordinary gatherings of adopters, all tested by the quickly expanding pace of use discharge. A. Day by day to month to month discharges B. Continuous Delivery and Integration C. Continuous Operation & Assessment High-level view of Continuous Delivery & Integration:

Major advantages which companies have witnessed are way across the books, but let us focus on some important points on why the industry has been so quick to adopt DevOps principles:

Shorter Development Cycles, Faster Innovation

With a combined development and operations team, applications are ready for use much more quickly. This is

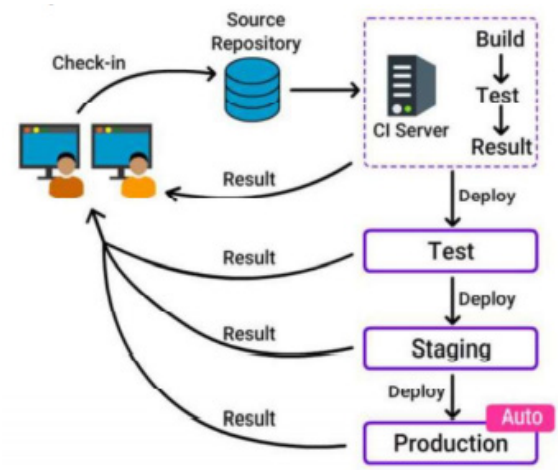

Figure 2

important, since companies succeed based on their ability to innovate faster than their competitors do. In fact, Kevin Murphy from Red Hat estimates that shorter development cycles translate to bringing an application to market 60 percent faster than with traditional approaches. Isn't that interesting?

Reduced Deployment Failures, Rollbacks, and Time to Recover

Time to recover is an important issue, because some failure has to be expected. But recovery is much faster when the development and operations teams have been working together, exchanging ideas and accounting for both teams' challenges during development. Your customer would be damm happy on it.

Improved Communication and Collaboration

It's no longer a matter of "turning over" the application of operations and waiting to see what happnes. Operations doesn't need to wait for a different team to troubleshoot and fix a problem. The process becomes increasingly seamless as all individuals work toward a common goal. Gosh no troubleshooting would be this easy!

Increased Efficiencies

Increased efficiency helps to speed the development process and make it less prone to error. There are ways to automate DevOps tasks. Continuous integration servers automate the process of testing code, reducing the amount of manual work required. This means that software engineers can focus on completing tasks that can't be automated. Yeah, Its Cooler than you thought!

Reduced Costs and IT Headcount

All of the DevOps benefits translate to reduced overall costs and IT headcount requirements. According to Kevin Murphy from Red Hat, DevOps development teams require 35 percent less IT staff and 30 percent lower IT costs. Now, this is Insane!

Conclusion

The industry has spoken, and it is implementing DevOps at a rapid rate. Organizations are eager to take advantage of faster application delivery, enhanced innovation, more stable operating environments and performance-focused employee teams. There have been multiple tools invented and helping in successful implementation of DevOps. Obviously covering such a humongous topic over some 800 words is never a justification but hope this gave a picturization of what is DevOps and How it works and its major perks. Just to add on everyday its getting bigger and better. DevOps is Almost Future.

References

1. Henderson-Sellers B. Process metamodelling and process construction: examples using the OPEN Process Framework (OPF). *Ann Softw Eng* 2002; 14(1&4): 341-362.
2. Henderson-Sellers B, Serour M, McBride T et al. Process construction and customization. *Journal of Universal Computer Science* 2004; 10(3). online journal accessible at http://www.jucs.org
3. Kumar K, Welke RJ. Methodology engineering: a proposal for situation specific methodology construction. In Challenges and Strategies for Research in Systems Development (eds. W.W. Cotterman and J.A. Senn), J. Wiley, Chichester 1992; 257-269.
4. Method fragment definition. FIPA Document, http://www.fipa.org/activities/methodology.html, (Nov 2003).
5. OMG. Software Process Engineering Metamodel Specification, Version 1.0, Object Management Group, formal. 2002.
6. Ralyté J. Towards situational methods for information systems development: engineering reusable method chunks, Procs. 13th Int. Conf. on Information Systems Development.
7. Advances in Theory, Practice and Education Vilnius Gediminas Technical University, Vilnius, Lithuania 2004; 271-282.
8. Ralyté J, Rolland C. An assembly process model for method engineering, Advanced Information Systems Engineering), LNCS2068, *Springer* 2001: 267-283.
9. Ralyté J. Reusing scenario based approaches in requirement engineering methods: CREWS method base. Proc. of the 10th Int. Workshop on Database and Expert Systems Applications.

ISSN: 2582-5607