International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 03 Issue: 12 | Dec -2016 www.irjet.net p-ISSN: 2395-0072

An approach of automation between development and operation by using DevOps

Mr. Gaurav kumar[1], Mr. Mukesh kumar[2]

[1]*Student of Master of Technology, Department of Computer science & Engineering, APJKTU, Lucknow, India*
[2]*Assistant professor, Department of Computer science & Engineering, APJKTU, Lucknow, India*

Abstract - *This paper provides a process for development to operation and defines the automation activity. Every developer wants easy to use tool and reduce time. At this time many companies are working on different tools but which will be suitable for a project, it depends on the requirement. This is certainly one primary causes for the perpetual tension between Development and IT Operations and sub-optimal results. The consequences of this are well-known: inadequately defined and specified environments, no repeatable procedures to deploy them, incompatibilities between deployed code and the environment, and so on. In this pattern, we will make environments early in the Development process, and enforce a policy that the code and environment be tested together. When Development is using an agile process, we can do something very simple and delicate. [1] According to Agile, we're supposed to have working, shippable code at the end of each sprint interval (typically every two weeks). Instead of having IT Operations responsible for creating the specifications of the production environment, instead, they will develop an automated environment creation process. By making tools available early, perhaps even before the software project begins, developers and QA can run and test their code in consistent and uniform environments, with controlled variance from the production environment. Ideally, the deployment mechanism we build is completely automated.[2] Tools that can be used include shell scripts, Puppet, Chef, Solaris Jumpstart, Redhat Kickstart, Debian Preseed, etc.*

Key Words: **Software, Testing, Optimization, Production, Agile transformation, Development, automation, Organization**

1. INTRODUCTION

Developments to Operations (DevOps) have a profound impact on the global IT sector in the near future. Realizing "DevOps" full potential, IT vendors have been agile enough in providing new products and services under the label "DevOps inside", at an ever increasing pace. [2]

1.1 What is DevOps?

The term "DevOps" typically refers to the emerging professional movement that supports a collaborative working relationship between Development and IT Operations, aftereffect in the fast flow of planned work (i.e., high deploy cost), while simultaneously increasing the reliability, stability, flexibilty and security of the production environment.

Why Development and IT Operations?
Because that is typically the appraisal stream that is between the business (where requirements are defined) and the customer (where value is delivered). The root of the DevOps movement is commonly placed around 2009, as the assemblage of numerous adjacent and mutually reinforcing movements:

- The Velocity Conference operation, especially the seminal (20 Deploys a Day) presentation given by John Alls paw and Paul Hammond
- The "infrastructure as code" operation (Mark Burgess and Luke Kanies), the "Agile infrastructure" movement (Andrew Shafer) and the agile system administration movement (Patrick DeBois)
- The gangly Startup movement by Eric Rise
- The continuous integration and release movement by Jez Humble
- The widespread availability of cloud computing and PaaS (platform as a service) technologies (e.g., Amazon Web Services). [3]

1.2 History of DevOps

In 2007, Patrick Debois, a software development consultant, had an objective of learning all aspects of IT. Over fifteen years, Patrick had taken on many different roles in IT in order to work in every aspect of an IT organization and gain an entire understanding of IT. He worked as a developer, network specialist officer, system administrator, tester and project manager.

Patrick had taken a consulting job for a large datacenter migration. He was in charge of the testing, which meant he was spending a lot of time with Dev and Ops.[3] Patrick had always been bothered by the differences between how Dev and Ops worked, but he became particularly thwarted with the challenges of managing work across the two groups on this datacenter migration.

Continuous integration was gaining popularity in the agile community and was moving Dev closer to deployment, but there was still nothing out there that fully crossed the divide of Dev and Ops. Patrick was confirmed there had to be a better way for these two teams to work together.

In 2008, Andrew Shafer posted an idea for an agile infrastructure "birds of a feather" session at the Agile 2008 Conference. Patrick Debois saw the post and went to the session. Unfortunately, he was the only one who showed up. The idea was so poorly received that Andrew didn't even show up to his own discussion.

Fortunately, Patrick was so excited to see that someone else was interested in solving the challenges of Dev and Ops working together that he tracked down Andrew and they decided to start a Google group named Agile System Administration.

In 2009, John Allspaw, senior vice president of technical operations at Flickr, and Paul Hammond, director of engineering at Flickr, gave a presentation at the O'Reilly Velocity Conference in San Jose, "10+ Deploys per Day: Dev and Ops Cooperation at Flickr." The presentation laid the groundwork for how Dev and Ops can effectively work together to improve software deployment.

Patrick Debois watched the presentation in Belgium via a live stream and was inspired to start his own conference, DevOps Days, in Ghent, Belgium. The conference brought together an energetic group of forward-thinking minds trying to improve software deployment. What may be even more important is that this group of people kept the conversation going on Twitter with the hashtag #DevOps Days. In an effort to save Twitter character space, people soon dropped days and the hashtag became #DevOps.

In 2010, the following year, DevOps Days were held in Australia and the U.S. Over time, there were more and more DevOps Days that were hosted in different countries and cities around the world. The face-to-face meetings ignited more and more people to get energized about DevOps until it had become a full-on grassroots movement.

In 2011, Up until 2011, the DevOps movement has been fueled by individuals and open source tools with little attention from analysts or vendors. But in 2011, the movement began to go main stream, catching the attention of analysts like Cameron Haight from Gartner and Jay Lyman of 451 Research. Big vendors started to market DevOps.

In 2012, By 2012 DevOps was quickly turning into a buzzword and DevOps Days continued to grow.

In 2013, the public thirst for DevOps information inspired several authors to write books on the topic. Notable examples are *The Phoenix Project* by Gene Kim, Kevin Behr and George Spafford and Implementing Lean Software Development by Mary and Tom Poppendiek.

In 2014, large companies such as Target, Nordstrom and LEGO became some of the first companies to bring DevOps into the enterprise. [4]

2. PROPOSED SYSTEM

The key technological breakthrough in DevOps is the creation of an automated continuous delivery pipeline. This new way of delivering software has a profound impact on the processes in development, quality assurance and operations. "Automating inefficient processes leads to automating inefficiency." (Jyrki Kasvi) Sandstorm used the quote to point out that setting up automation technology alone is not sufficient. While automation promises benefits in lead times, application stability etc., implementing this technology and processes is not entirely challenge-free. [5]

Table -1: Drives the Need for DevOps

The need for higher collaboration between development and operations components	47%
A bigger need for simultaneous deployment across different area	41%
Pressures from the company to release apps more quickly to meet customer demand or enter new organization	41%
Need to improve the end customer relationship	39%
The increasing use of mobile devices (Smartphone/tablets)	35%
The increasing need to develop or deploy cloud based applications	31%
An increasingly complex IT infrastructure that is part physical, part virtualized and part cloud	28%
Need to reduce IT cost	16%

Table -2: DevOps Significant Benefits

Metric	Percent Improvement
Increased relationship between departments	23%
Enhanced quality of our deployed applications	22%

Increased numbers of customers using our software/services	22%
New software/services that would otherwise not be possible/scrutinize	21%
lesser employees working on developing and deploying our software/services	21%
Diminished time-to-time market for our software/service	20%
An increase in revenue	19%
Our 1software/service made available across more platforms	19%
a reduction in spend on development and operations	18%
Increased frequency of deployments of our software/services	17%

2.1 Area of DevOps

This part will detail the key areas of DevOps. DevOps is no longer considered as a trendy subject of those working with the latest development environment. Present information goes beyond technology into processes and people, and there are many different approaches to bringing them together (Riley, 2014). Humble and Molesky (2011) purpose that DevOps consists of four dimensions: Culture, Automation, Measurement and Sharing. These four dimensions each target the specific problems discussed in the previous section, the main goal being the alignment of incentives of all stakeholders, particularly Development, Quality Assurance (QA) [6] and Operations personnel. Furthermore, several sources add Lean to the definition as well. Generally this has been dubbed as the "CALMS" model: Culture, Automation, Lean, Measurement and sharing respectively. This model defines the most essential points-of-view of DevOps. These areas also describe DevOps as a "flow" (Riley 2014).

Table -3: components of DevOps

Culture	Hearts & Minds Embrace Change Hearts
Automation	CI/CD "Infrastructure as Code"
Lean	Focus on producing for the end-user Small batch sizes
Metrics	Measure everything Show the Improvement
Sharing	Open information sharing Collaboration

3. EXPERIMENT

3.1 DevOps tools

Earlier we briefly discussed some of the tools used in DevOps; here are some of the key tools and practices you need to know.

3.1.1 Source Code Repository

A source code repository is a place where developers check in and adjust code. The source code repository manages complete code that is checked in, so developers don't write over each other's work. Source control has probably been around for forty years, but it's a major component of continuous integration. Popular source code repository tools are Git/GitLab, Subversion, Cloudforce, Bitbucket and TFS.

3.1.2 Build Server

The build server is an automation tool that compiles the code in the source code repository into executable code base. Popular tools are Jenkins, SonarQube and Artifactory.

3.1.3 Configuration Management

Configuration management defines the configuration of a server or an environment. Popular configuration management tools are Ansible, SaltStack, Puppet and Chef.

3.1.4 Virtual Infrastructure

Amazon Web Services and Microsoft Azure are examples of virtual infrastructures. Virtual infrastructures are provided by cloud vendors that sell infrastructure or platform as a service (PaaS). These infrastructures have APIs to allow you to programmatically create new machines with configuration management tools such as Puppet and Chef.

There are also private clouds. For example, VMware has vCloud. Private virtual infrastructures allow you to run a cloud on top of the hardware in your data center.

Virtual infrastructures combined with automation tools to empower organizations practicing DevOps with the ability to configure a server without any fingers on the keyboard. If you want to test your brand-new code, you can automatically send it to your cloud infrastructure, build the environment and then run all of the tests without human intervention. [7]

3.1.5 Pipeline Orchestration

A pipeline is like a manufacturing assembly line that happens from the time a developer says, "I think I'm done," all the way to the time that the code gets deployed in the production or a late-stage-pre-production environment.

3.1.6 Unifying Enterprise Software Development and Delivery

The VersionOne Enterprise Agile Platform unifies agile application lifecycle management and DevOps, providing a full picture of your entire software delivery pipeline in a single platform. VersionOne® Continuum™ for DevOps is an enterprise continuous delivery solution for automating, orchestrating, and visualizing the flow of change throughout the software delivery cycle.

3.1.7 Test Automation

Test automation has been around for a long time. DevOps testing focuses on automated testing within your build pipeline to ensure that by the time that you have a deployable build, you are confident it is ready to be deployed. [7] You can't get to the point of continuous delivery where you're fairly confident without any human intervention that your code is deployable without an extensive automated testing strategy. Popular tools are Selenium and Water.

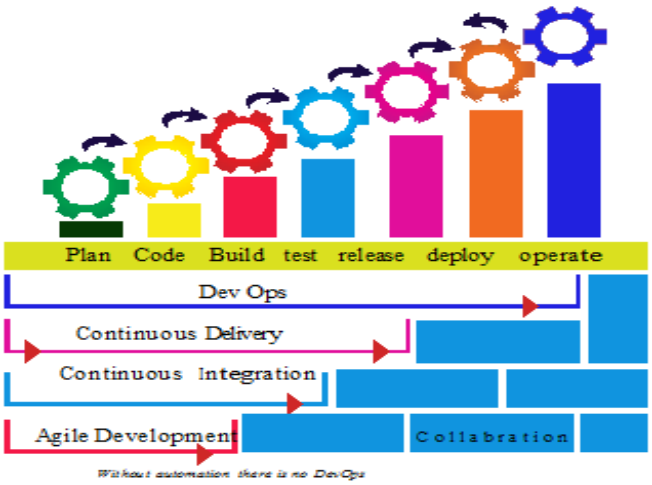

Fig-1: Without automation there is no DevOps

- Automate Provisioning - Infrastructure as Code[8]
- Automate Builds – Continuous Integration [9]
- Automate Deployments – Defined Deployment Pipeline and Continuous Deployments with appropriate configurations for the environments. [10]
- Automate Testing – Continuous Testing, Automated tests after each deployment
- Automate Monitoring – Proper monitors in place sending alerts
- Automate Metrics – Performance Metrics, Logs

4. CONCLUSION

The uniformly changing business needs and the requirement for faster time to market with the software of present day have made a paradigm shift towards a 3rd generation Software Development philosophy called DevOps. The lack of association between IT Operations and Software Development, as well as the imbalance in configuration between development, testing, and production environment, has made deploying software releases slow and painful for many organizations. [11] A different incentive between teams makes it difficult to work towards a common goal of bringing added value to customers.

A DevOps approach to software development brings down the walls between the teams and aligns incentives through a collaborative culture, automation, lean principles, measurement practices, and sharing. The benefits of DevOps have been shown to be substantial with a significantly faster time to market and increased software security. The organizational change is substantial which makes the challenges in adopting DevOps an interesting topic to research. This paper studied the challenges of DevOps by interviewing nine experts who had been involved with DevOps initiatives in their companies.

The findings were divided into four main challenge categories based on their topic. Due to the novelty of the

approach, the concept of DevOps for many is unexplained or biased which hurts the overall implementation of practices. The lack of support in both management and organizational levels is a hindrance since especially changing culture needs strong support and organizational buy-in in order to achieve. The toolset needed for DevOps is particularly diverse and finding the fit, correct usage and attitudes towards that technology is challenging. Finally, when moving to DevOps that requires a certain level of lean principles and agility, aligning existing organizational processes such as the change management process to accommodate the new way of working was found challenging.

REFERENCES

[1] Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2002). Agile software development methods. Review and analysis. Espoo: VTT.
[2] Anderson, D. (2010). Kanban Successful Evolutionary Change for your Technology Business. Sequim, Washington: Blue Hole Press.
[3] Boehm, B. W. (1988). A spiral model of software development and enhancement. Computer, 21(5), 61-72.
[4] Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., & Brandic, I. (2009). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Generation computer systems, 25(6), 599-616.
[5] Dubois, A., & Gadde, L. (2002). Systematic combining: An abductive approach to case research. Journal of Business Research, 55 (7), 553-560.
[6] Earnshaw, A (2013) what is a Devops Engineer. Online. Available at https://puppet.com/blog/what-a-devops-engineer [09.04.2016]
[7] Eficode, Devops Quick Guide. Online. Available at http://devops-guide.instapage.com/ [20.04.2016]
[8] Basil, V. R., & Turner, A. J. (1975). Iterative enhancement: A practical technique for software development. Software Engineering, IEEE Transactions, (4), 390-396.
[9] Bass, L., Weber, I., & Zhu, L. (2015). DevOps: A Software Architect's Perspective. Boston: Addison-Wesley Professional.
[10] Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M. & Kern, J. (2001). Manifesto for agile software development.
[11] Benington, H. D. (1983). Production of large computer programs. IEEE Annals of the History of Computing, (4), 350-361.

BIOGRAPHIES

Mr. Gaurav kumar,*Department of Computer science & Engineering, ITS engineering college, Greater Noida, UP, India*

Mr. Mukesh kumar ,*Department of Computer science & Engineering, ITS engineering college, Greater Noida, UP, India*