Scientific Journal of Impact Factor (SJIF): 4.72

e-ISSN (O): 2348-4470
p-ISSN (P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 4, Issue 10, October -2017

An approach of IT performance by automation and continuous deployment by using different tools

Devops

Mohd asfar [1], pradeep kumar[2], raj kumar goel[3]

[1]computer science and engineering, NIET greater noida
[2]computer science and engineering, NIET greater noida
[3]computer science and engineering, NIET greater noida

Abstract — *this paper describes IT performance using the continuous deployment model, and presents the key factors of effective information technology (IT) team performance in the context of DevOps and IT service management. It then presents the development of a construct to measure IT team performance in the context of DevOps, based on three factors, two throughput measures (lead time to deploy and deployment frequency) and a solidity measure (mean time to recover); these correspond to EOQ variables batch size (measured as deployment frequency) and transaction cost (lead time to deploy and mean time to recover). Based on a sample of 7,522 IT professionals worldwide, we conduct a hierarchical cluster analysis and find that the throughput and stability measures move together, consistent with EOQ. The analysis reveals three IT performance profiles: high, medium, and low. Further analysis shows that innovations can be used to impact these IT performance profiles. Implications for research and practice are discussed.*

Keywords- *Software architecture, DevOps, continuous deployment.*

I. INTRODUCTION

With increasing complexity of software-intensive systems, the role of software architecture as a means of understanding and managing large-scale software intensive systems has been increasingly becoming important [1]. It has been recognized that different domains and contexts bring different challenges for architects. Hence, a given context can have considerable impact on architectural design decisions and organizational practices for architecting related activities and artifacts. Development and Operations (DevOps) in the context of Continuous Deployment (CD) is an emerging software industry movement to bridge the gap between development and operations teams [2]. CD is defined as the ability to frequently and reliably put new releases into production, with as much automation as possible [2, 11]. It is argued that an architect should make a software system’s design as much simple and fine-grained as possible and try to remove those architectural elements that can be obstacle to deployment automation [11]. DevOps/CD practices necessitate an extensive use of infrastructure automation techniques, which can reduce the complexity of deployment and operations to a very large extent. Adopting and supporting DevOps/CD involve a large number of Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee included that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights challenges because organizational processes and tools may not be ready to support highly complex and challenging nature of DevOps/CD. It is argued that one of the most pressing challenges which the organizations may encounter is how software applications should be (re-)architected to support DevOps practices such as Continuous Integration (CI), Continuous Delivery (CDE) and Continuous Deployment (CD) [2, 5]. A sound architecture helps ensure a desired level of quality attributes (e.g., deploy ability, testability, and log ability) and can enable short feedback cycle time including instant feedback from operations [3]. However, there has been a little research on what implications DevOps/CD can have on architecting [3, 5]. We assert that there is an important and urgent need of research to gain a deep understanding of how DevOps/CD adoption can influence the architecting processes and their outcomes in an organization. Therefore, this PhD research seeks architectural elements, practices, and patterns that support DevOps and continuous deployment.

1.1 What is DevOps?

The term “DevOps” typically relates 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 seminar(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]

II. THEORETICAL FRAMEWORK

Software is typically developed in long-running projects with infrequent releases because the transaction cost of pushing out software to users is very high [e.g., 9]. For example, ITIL Service Transition describes eight major processes spanning multiple functions throughout organizations in order to make changes to services [1]. These processes are thus expensive to operate and have a lead time typically measured in weeks or longer [e.g., 10].The Economic Order Quantity (EOQ) model was a precursor to Lean manufacturing [e.g., 11, 12, 13], and was derived by was derived Ford W. Harris in 1913 [14]. It models the economic trade-offs involved in deciding the optimal batch size, based on the transaction cost and the holding cost.

III. 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.

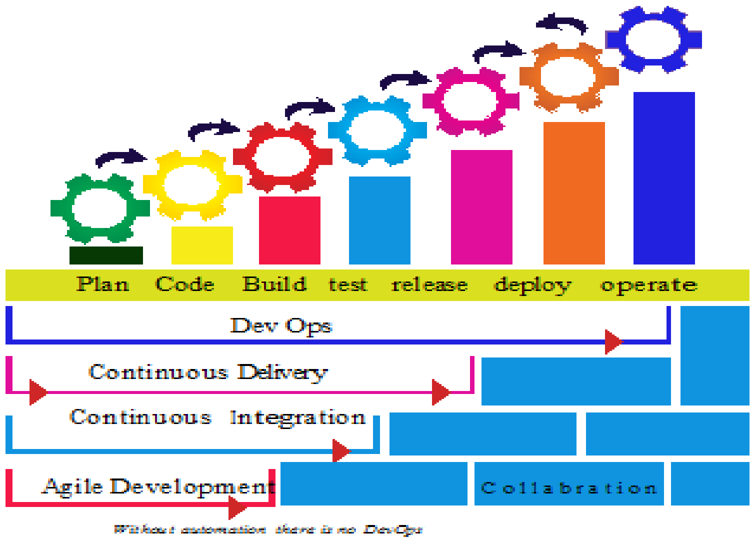

Fig-1: Without automation there is no DevOps

- 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

IV. 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] Niederman, F.; Brancheau, J. C.; and Wetherbe, J. C. (1991) Information Systems Management Issues for the 1990's. MIS Quarterly, 15(4), 475-495.

[2] NIST (2011). The NIST Definition of Cloud Computing. Recommendations of the National Institute of Standards and Technology. Gaithersburg: Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology.

[3] Null, C. (2015) 10 Companies Killing it at DevOps. Techbeacon blog. Online. Available at: http://techbeacon.com/10-companies-killing-it-DevOps [11.04.2016]

[4] Nygard, M. (2007). Release it!: design and deploy production-ready software. Pragmatic Bookshelf. Boston: Addison-Wesley.

[5] Ohno, T. (1988). Toyota production system. Cambridge, Mass.: Productivity Press.

[6] Paul, F. (2014) The Incredible True Story of How DevOps Got Its Name. New Relic Blog. Online. Available at: https://blog.newrelic.com/2014/05/16/DevOps-name/ [09.04.2016]

[7] PE International. (1995) Justifying Infrastructure Investment. IT Management Programme Report. Egham, United Kingdom: Centre for Management Research, PE International.