DevOps-Driven Approach to Development in Cloud

Anita Nair, Sr. Cloud Architect

Abstract - DevOps, a prevalent terminology in organizations today, has evolved as a paradigm that once was SDLC. From the mammoth concept of the “WaterFall Model” to the more recent Agile Framework and the emerging DevOps methodology, the Software Development Life Cycle saw a huge business win in “Time to market”, over several other advantages like risk aversion, flexibility, requirement readjustments, collaboration etc. to name a few. This whitepaper elaborates on the various salient features of DevOps that shapes the way software releases are made today. DevOps is most suited to Cloud and this paper explains the way the two technologies synergize. Service Oriented Architecture is an architectural style that is essential in simplifying the business and this paper explains how DevOps facilitates creation of granular services that ultimately helps in attaining flexibility and agility of development and operations in Cloud.

Index Terms – DevOps, Containerization, Continuous Integration, Continuous Deployment, Service Oriented Architecture, Agile, Microservices, AWS

———————————— ◆ ————————————

1. INTRODUCTION

DevOps, a now-popular terminology in organizations, has evolved as a paradigm that once was SDLC. From the mammoth concept of the “WaterFall Model” to the more recent Agile Framework, the Software Development Life Cycle saw a huge business win in “Time to market”, over several other advantages like risk aversion, flexibility, requirement readjustments, collaboration etc. to name a few.

DevOps is one leap above Agile, where we are no longer talking about just the software but also the infrastructure. DevOps facilitates the marriage of IT and Development to such an extent that organizations now wouldn’t need two disconnected teams working in silos but fosters collaboration and unification of these two services within the business. Not only does it foster collaboration between teams but also brings about a unified approach to development that smoothens interaction between the developers, testers, project managers and end users.

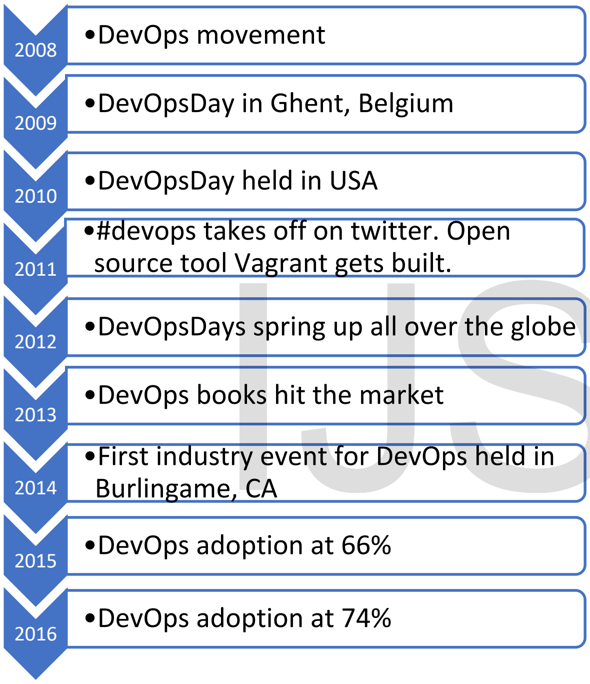
Adopting DevOps is however not simple. Every technology faces resistance in a paradigm shift and DevOps is no different. In recent years, Cloud Computing has evolved as a platform where many companies see advantage of migrating their workloads. Since Cloud offers many advantages in moving from manual processes to automation, it is an ideal environment for DevOps implementation. We would therefore be emphasizing on DevOps in the context of Cloud.

This whitepaper is divided into seven major sections. Section 1 gives a brief history of how DevOps came into being. Section 2 elaborates the salient features of the DevOps paradigm. Section 3 illustrates the tools used to realize DevOps. Section 4 explains the two complimentary technologies that aid DevOps – Cloud and Microservices. The section discusses why Cloud is most suitable for DevOps. In Section 5, we talk about the various best practices that can be leveraged while implementing the DevOps strategy in Cloud. Section 6 talks about the several challenges of DevOps in Cloud. In Section 7 we present two different use-cases that demonstrate how we utilized the various features of DevOps in Cloud for solutioning its clientele.

2. A SHORT HISTORY OF DEVOPS

“Necessity is the mother of invention” – DevOps came about as a by-product of the ego-struggle between different teams and roles. Patric Debois, a software developer who had assumed different roles in large enterprises, helped plant the seeds of DevOps movement in 2008 at an Agile conference at Toronto. In 2009, two Flickr employees made a seminal talk called “10+ Deploys per Day: Dev and Ops Cooperation at Flickr”. They made a case that the only sensible way to develop, build and deploy applications is to make Development and Operations transparent. Debois got inspired by viewing the presentation and organized his own conference in Ghent, Belgium, called the DevOps day. In 2010, the first DevOps Day gets organized in US, with the help of John Willis, an enterprise system management expert and Kris Buytaert, a Linux and Open source consultant. In 2011, Cameron Haight from Gartner predicted that by 2015, 20 percent of global 2000 businesses will embrace

DevOps. The DevOps community started to build open source tools like Vagrant that worked with existing configuration management tools like Puppet and Chef. In 2012, DevOps days began springing up in many cities around the world, from Bangalore to Boston. In 2013, many DevOps related books started appearing, like, "The Phoenix Project", "Implementing Lean Software Development", "The Lean Startup" etc. In 2014, DevOps was embraced by brands such as Nordstrom, Target and LEGO. The "DevOps Enterprise: The Agile, Continuous Delivery and DevOps Transformation Summit", the first industry event focused on helping enterprise software organizations, was held in Burlingame, CA.



2008 •DevOps movement

2009 •DevOpsDay in Ghent, Belgium

2010 •DevOpsDay held in USA

2011 •#devops takes off on twitter. Open source tool Vagrant gets built.

2012 •DevOpsDays spring up all over the globe

2013 •DevOps books hit the market

2014 •First industry event for DevOps held in Burlingame, CA

2015 •DevOps adoption at 66%

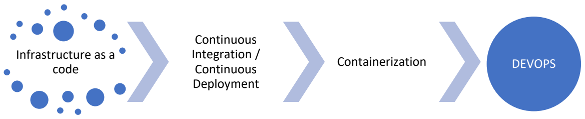
2016 •DevOps adoption at 74%

In a survey published in January 2016 by the SaaS cloud-computing company Right scale, DevOps adoption increased from 66 percent in 2015 to 74 percent in 2016. And among larger enterprise organizations, DevOps adoption is even higher — 81 percent.

3. SALIENT FEATURES OF DEVOPS

There are three key features that define DevOps. To be able to treat infrastructure the way developers treat code is a leap in thinking that has allowed deployments of infrastructure to be automated. This feature enables collaboration between the operations and the development teams so that both teams are equally well-equipped to handle releases.

Continuous Integration/Continuous Deployment is the terminology given to that feature of DevOps where the primary focus of the developer would be confined to writing code and checking in. The rest of the process of building, testing, integrating and deploying to production would be a continuously automated process.



Infrastructure as a code

Continuous Integration / Continuous Deployment

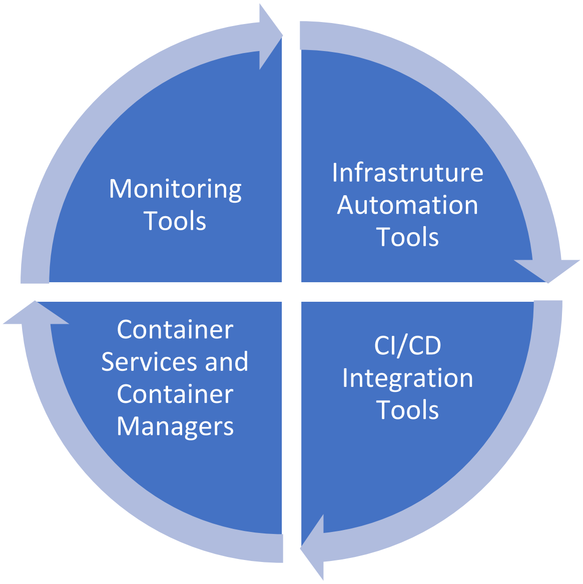
Containerization

DEVOPS

While virtualization technology provides an environment for an application to run in a virtual machine with an OS and run-time of choice, it is not entirely light-weight. An alternative to virtualization is containerization wherein the application one deploys is contained in a software container that has all elements of a desired OS and its run-time. This makes the process of deployment extremely efficient and light-weight.

4. TOOLS TO REALIZE DEVOPS

The DevOps industry has succeeded through the adoption of several DevOps tools that have hit the market in the last few years. The following are several areas where open source and third-party tools have been maturing:


Monitoring Tools
Infrastructure Automation Tools
Container Services and Container Managers
CI/CD Integration Tools

The Infrastructure Automation tools like Chef, Puppet, Ansible etc. have wide adoption amongst the DevOps community. What determines the choice of the selected tool varies between users. But on an average users want a tool that is cost-efficient, easy to learn and has a simple interface.

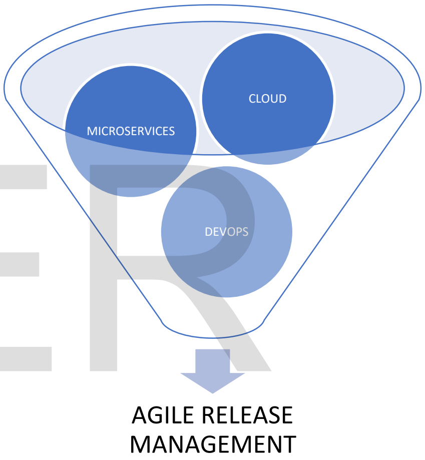
CI/CD Integration tools make the lives of developers, testers and project managers easy. Some tools in the industry that are popular are: Jenkins, TeamCity, Travis CI, Go CD, Bamboo, CircleCI etc. to name a few. These come as free or paid versions. There are self-hosted and hosted versions. The best choice depends on how much control over the build and deploy process one wants to have.

Container services and container managers cater to the needs of the end-users for portability reasons. Docker is the most popular container service today and some of the container clustering and orchestration tools that have hit the market today are Docker Swarm, Kubernetes, Mesos, Fleet etc. The parameters that determine their choice are: networking, high availability, ease of deployment and good service discovery.

Monitoring Tools in DevOps are of utmost importance to ensure service uptime and optimal performance. They come under two categories: Infrastructure Monitoring and Application Monitoring. Examples of Infrastructure monitoring tools are Nagios, Zabbix, New Relic Infrastructure etc. Examples of application monitoring tools are New Relic, AppDynamics etc.

5. COMPLIMENTARY TECHNOLOGIES: CLOUD AND MICROSERVICES

Two modern technologies that complement DevOps are Cloud Computing and Microservices. Both these technologies have been gaining traction over the recent years and they aid the concept of DevOps to a large extent. In this section we will see how they synergize to strengthen the Agile Framework for release management.



Cloud Computing is a technology that has been pervading the IT landscape for the past decade where enterprises see benefit in moving infrastructure from their own on-premise data centers to the datacenter of a third-party vendor, from cost, reliability, performance and security point of view. Among several advantages, agility is a huge reason for the popularity of Cloud. DevOps also focusses on agility and Cloud makes it possible for infrastructure to be immediately available when demand necessitates.

Apart from Agility, the features of DevOps are easily realized in Cloud. Infrastructure in Cloud is packaged as "Machine Image" and instantiating a virtual machine is nothing but a run of a script. Hence Cloud is a natural place for maintaining

Infrastructure as a Code. Automating Continuous Integration and Deployment is easily realizable in Cloud.

The feature of elasticity that Cloud renders is an advantage when one talks of implementing DevOps. Bringing up Test Environments, running tests and staging Production environments is a seamless process irrespective of the load on the machines. Cloud also lends itself easily to Containerization where it is easy to spin off and cluster Docker containers.

Micro services on the other hand, is an architectural approach to system design that evolved from Service Oriented Architectural framework. In this approach, a system is broken down into granular components that are self-contained so that each component can be individually developed, tested, built and deployed, leading to the architectural advantage of loose-coupling. The concept of Docker, where a component of the application can be packaged in individual containers has played a pivotal role in realizing the micro services approach to architecture.

Cloud lends itself well to loose-coupling and when Cloud, Micro service architectural framework and Containerization synergize, one ends up building powerful enterprise architectures.

An area in Cloud called the "Function as a Service" or "Serverless" architecture has gained huge popularity amongst the community who are interested in taking micro-services to the next level. FaaS provides enormous cost savings and the offering is completely managed. It is also an architecture which is above the concept of DevOps and is popularly known for being "No-Ops" that is better known as "SuperOps". The fact that server provisioning is not needed to deploy the serverless framework makes it irrelevant to formalize continuous integration and deployment. Serverless is in the rudimentary stages of maturity and still needs wide adoption but as far as DevOps goes, it is a perfect example of integrating the Development and Operations functions in the enterprise.

6. DEVOPS IN CLOUD - BEST PRACTICES

While implementing DevOps in Cloud, the following are few best-practices one must take care to implement in order for the solution to be effective:

1. **Select a DevOps solution that is Vendor Neutral: While many Cloud Vendors offer DevOps solutions, it may be advisable not to use them unless there is compelling reason, so that one may avoid vendor lock-in. Several open source DevOps tools can be evaluated and leveraged for most needs.**
2. **Take care of security: Depending on the data that is put in Cloud, one should chose a DevOps strategy. Using Virtual Machines is more secure than Containerization for the obvious reason that the host operating system is secured by the hypervisor layer in case of virtualization. If your data and application in Cloud needs compliance and high levels of security, a better option to choose from Containerization and Virtualization is the later.**
3. **Include automated performance test suite: Your DevOps strategy should be designed such that the unit tests, regression tests, performance tests and integration tests are packaged to run in an automated fashion when the deployment stages from Development to Test to Production. With the elastic nature of Cloud at one's disposal, performance tests can be easily accomplished by loading the system to maximum amount.**
4. **Use containers: Containers are light-weight, packaged solutions that allow easy porting of code in different platforms. It is always advisable to use container technology completely while designing the DevOps strategy. The industry has several open source solutions to manage clustering of container services, which could be employed.**
5. **Not every application is Cloud-able or Docker-able: Don't force every application to be migrated to Cloud and don't force Dockerization of all applications. Especially applications that are monolithic and hard to break down into micro**

services. Example: COBOL or Mainframe applications.

7. DEVOPS IN CLOUD – CHALLENGES

In the previous sections we have seen the opportunities that prevail in DevOps in Cloud. In this section, we will see that not everything is rosy in the DevOps path and we will see why the industry faces certain challenges while choosing a DevOps strategy. The following are some of the biggest challenges we see in implementing DevOps in Cloud:

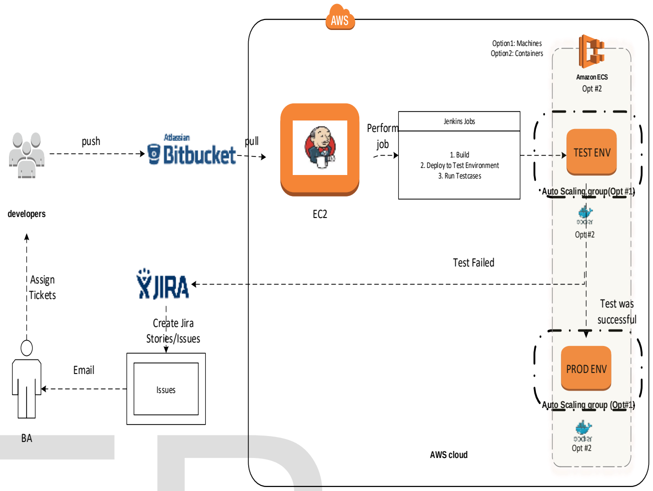
1. **The first and foremost challenge that DevOps faces is to build a collaborative culture within the enterprises to bring together the Development and Operations that is the crux of the DevOps movement. This is difficult to attain because a shift in paradigm is not easily assimilated and there is always a steep learning curve.**
2. **Another challenge is that many enterprises still have legacy systems that are hard to migrate to Cloud and DevOps-ify. It is very difficult to rewrite such applications so that they become cloud native.**
3. **Many companies focus of CI/CD while ignoring test automation. A DevOps solution is not complete unless there is Continuous Testing in place and many resources do not support the entire gamut of test suits ranging from unit, performance, security, regression and integration testing.**
4. **There are many DevOps tools available in the market. Many are open source too but a unified tool is difficult to find and one must employ a set of tools for a DevOps solution. Choosing the right tools for the right workloads and applications is a challenge that needs an expert eye.**

8. CASE STUDIES

7.1 Case Study 1

One of our Telecom clients who had a huge presence in AWS Cloud wanted a DevOps solution that would allow them to leverage the server-less technology seamlessly so that their development cycles are fully automated, agile and portable. They did not want to be vendor-locked however, they wanted their deployment fleet to be in AWS. We gave them a cost-effective solution by leveraging the open source solution (Jenkins) that could integrate the products (as plugins) from Atlassian they already use for Source Control (Bit bucket) and Project Management (Jira).

The following was the architecture we proposed to them:

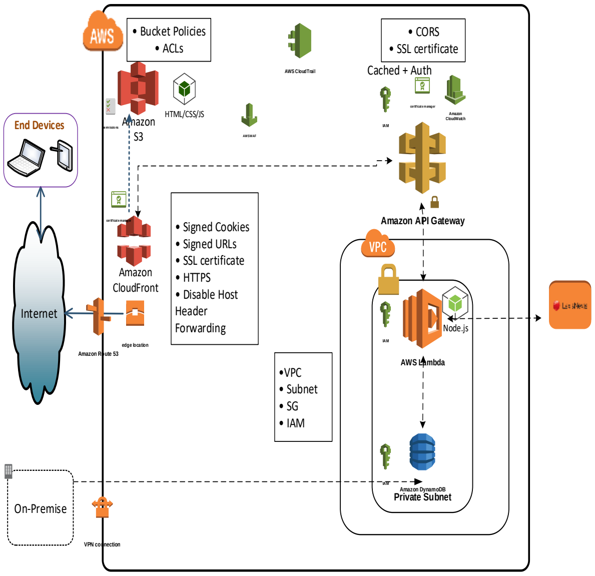


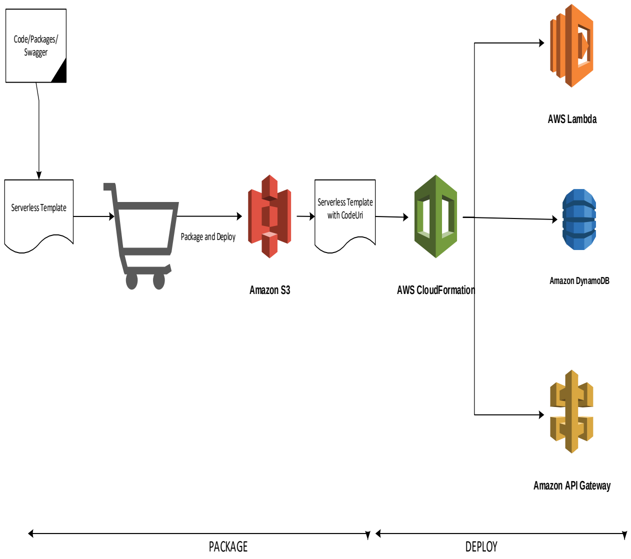
Developers push their code to Bitbucket and the Jenkins server in AWS cloud pulls the change and kicks off a bunch of jobs which include, build, and deploy to test environment and running automated test cases. Once the test cases pass, the production environment is staged. If any test case fails, jira tickets are logged and an email is sent to the Business Analyst, who assigns the tickets to the developers.

For containerizing the application, Docker containers were used and the containers were orchestrated by AWS's Elastic Container Service. Application monitoring was done with the help of AppDynamics.

7.2 Case Study 2

One of our clients on AWS cloud wanted a very scalable and cost-effective architecture for their Portal Application. We offered them a serverless microservices architecture on AWS Cloud where their web services were exposed through AWS's Cloudfront serving off API Gateway, which in turn was powered by Lambda functions on the back-end that interacted with a DynamoDB DB Tier .





A DevOps solution is in-built in serverless. A way to manage the services needed for the serverless deployment is through CloudFormation SAM (Serverless Application Model), wherein the entire package of code that needs to be deployed is packaged in a "Swagger" and CloudFormation is used to deploy the services.

9. REFERENCES

1. *M. Virmani, "Understanding DevOps & Bridging the Gap from Continuous Integration to Continuous Delivery"*, Proc. 5th Int'l Conf. Innovative Computing Technology, *pp. 78-82, 2015.*

2. *Prashant Agrawal, Neelam Rawat, "Devops A New Approach To Cloud Development & Testing", Issues and Challenges in Intelligent Computing Techniques (ICICT) 2019 International Conference on, vol. 1, pp. 1-4, 2019.*