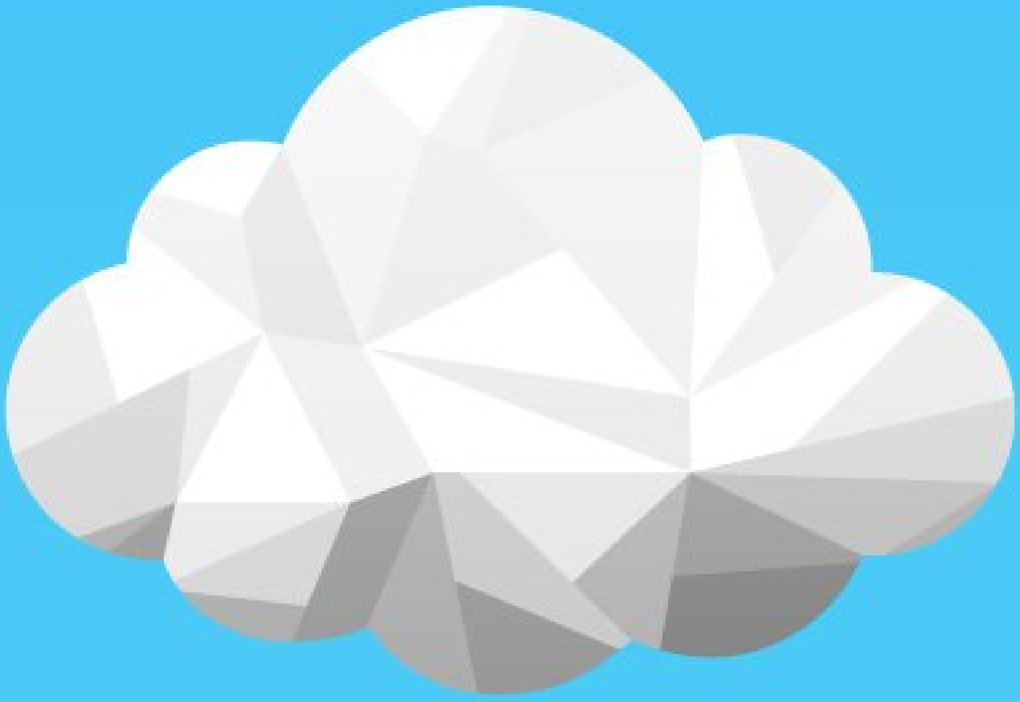
DevOps in the Cloud with AWS



617-340-3850 contact@kandasoft.com

Table of contest

1. INTRODUCTION

Software delivery has come a long way over the last decade as the business model has changed from selling as a product to increasingly providing software as a service (SaaS). This shift has also transformed software production, changing silo-ed coding shops to integrated organizations where the development, testing, deployment and service management of software work as a unit. The glue that binds these activities together is DevOps and the way it does this is by automating the build-release-operate cycle and turning it into a repeatable and consistent process.

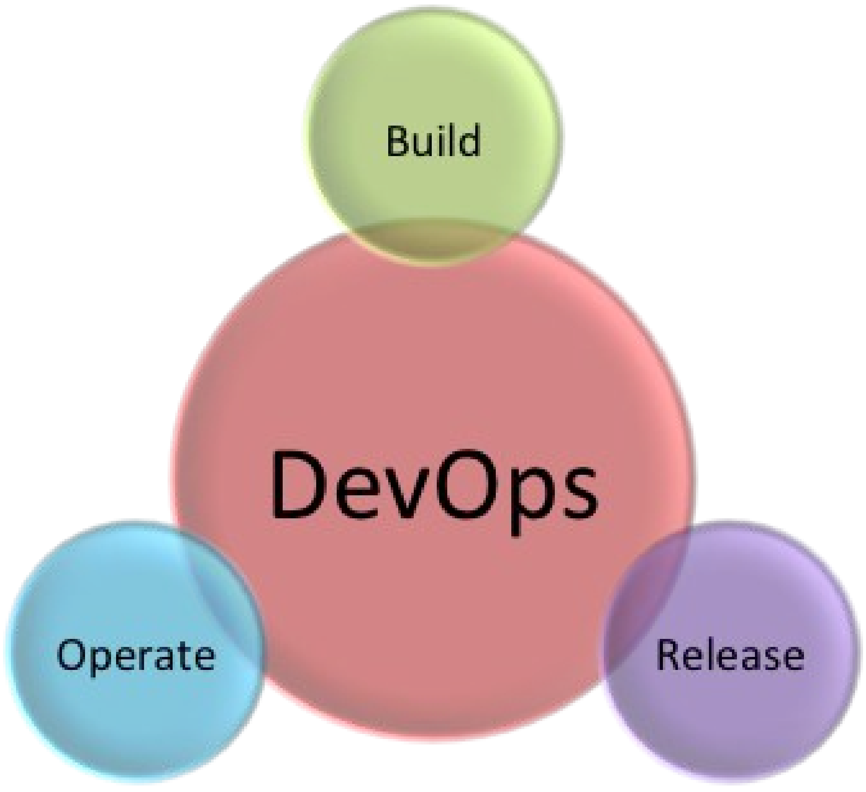


Fig 1. DevOps Cycle

The major trend pushing infrastructure automation is the simultaneous rise of the Cloud. The world has moved from bespoke hardware to a pay-by-usage economic model where infrastructure is consumed like a utility (IaaS). Cloud services like AWS provide enormous flexibility by empowering automation, rapid provisioning and scaling. DevOps with AWS makes it possible to automate infrastructure as easily as it does the deployment of the code.

One can think of how agile practices have transformed software development with rapid iterations and a continuous delivery model. In a similar way, DevOps has made software operations increasingly agile removing friction from the deployment process. At the push of a button code can be built, run through automated tests, deployed and ready to be used by customers.

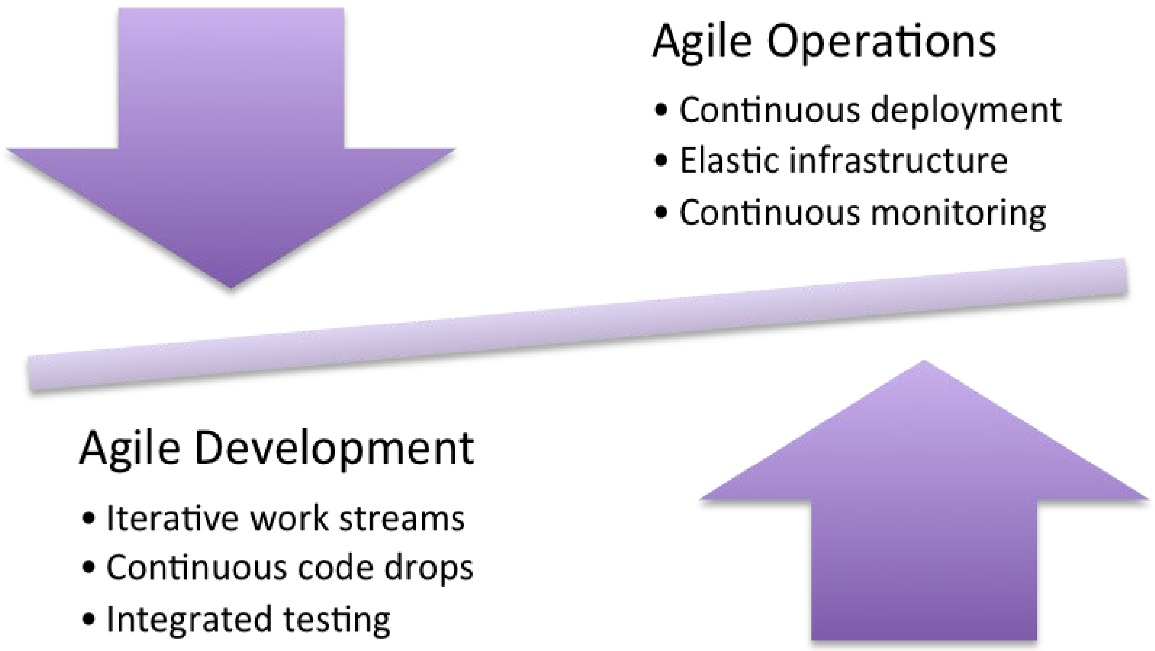


Fig 2. Agile Ops

It is important to point out, however, that automation is not an end in itself. It is a means to enable organizations to maintain fluidity in their technology management and continuously incorporate feedback to improve service quality. Automation makes it possible for technology teams to rapidly respond to

- Bugs reported in the system
- Customer feature requests
- On-demand scaling
- Innovations in the application architecture

As businesses grow so do the demand for newer features and capabilities in the software. At the same time, the customers expect no letdown in service quality – in fact, they expect it to improve. Simultaneously satisfying both of these expectations requires an approach where all aspects of the production cycle are aligned to create rapid iterations and manage change together.

The incorporation of continuous feedback to automated build-release-operate cycles creates a virtuous circle of continuous value that is the greatest promise of DevOps.

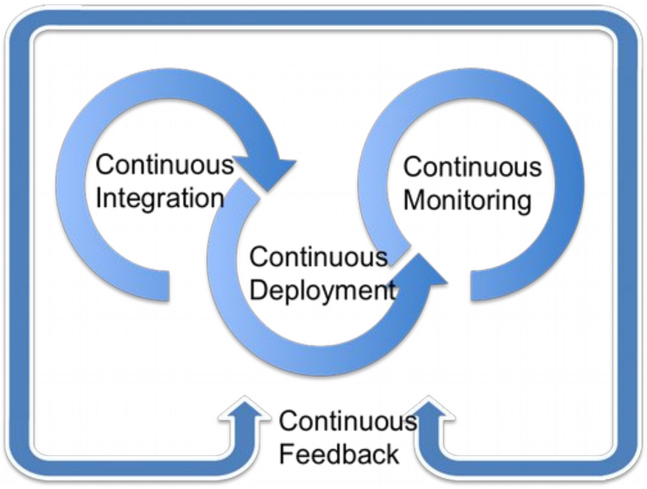


Fig 3. Continuous Value Loop

For software development organizations in particular this brings numerous benefits including

- Lower cost by removing manual work
- Reduced risk through consistent and repeatable processes
- Increased speed of delivery
- Increased productivity of team members
- Reduced outages and service disruptions

Whether you are just starting out on AWS or have an established environment, a robust DevOps regimen can significantly improve your business. Below, we provide a brief overview of the tools and best practices used to perform DevOps tasks particularly with AWS.

2. TOOLS AND PRACTICES

DevOps is a collection of processes and practices to manage the life cycle of software provisioning, delivery, security, compliance and service quality management. It strives to combine the workflows of application development and technology operations by linking together their toolchain.

The AWS approach to DevOps is to streamline and simplify the application distribution chain on the AWS cloud through an integrated toolset. Collectively, the tools provide the ability to

- Provision infrastructure resources
- Configure the application stack
- Create consistent application environments
- Code deployment and distribution
- Monitor resources and alert on events
- Administer security and compliance

Lets take a look at each of these capabilities.

2.1 Provisioning

Consider the set-up of a basic AWS environment. This would include resource allocations like

- EC2 instances
- EBS volumes
- VPC setup
- Elastic IPs

The AWS solution to automate the creation of such resources is CloudFormation. By using AWS CloudFormation you can instrument your entire AWS infrastructure using JSON configuration files that effectively serve as building blocks. So, for the above example, you would create a script in CloudFormation that contains the definition for these resources and their required configuration.

Fundamentally, the idea is to treat “infrastructure as code”. In other words, we use code to create and automate infrastructure elements in the same way application code automates your business. More importantly, infrastructure operations use the same sophistication and discipline used in application development including source control for scripts, versioning and testing.

CloudFormation templates use the concept of “stacks”, which are essentially a grouping of resources as a single unit. For example, a subnet with a collection of instances, IPs and storage for your staging environment could be a stack. CloudFormation is the orchestration tool to deploy and update stacks as well as provision, manage and update the collection of resources within the stack.

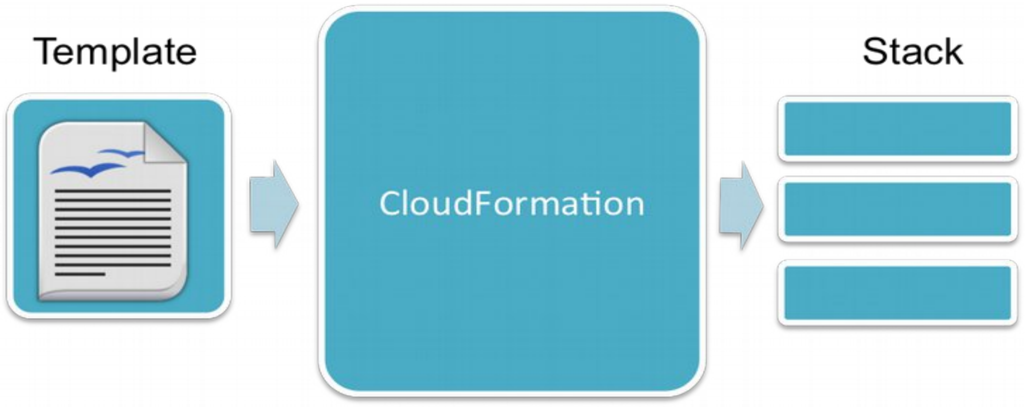


Fig 4. CloudFormation process

Creating resources programmatically provides enormous flexibility and elasticity for managing compute resources allowing you to spin up new environments quickly with relative ease. As most businesses have multiple types of environments e.g. development, staging, production, you can create multiple templates for each type. Versioning ensures you keep track of the state of the environment at the time it was created.

2.2 Configuration

Static templates for building pieces of infrastructure are one thing, managing the dynamic configuration of your application stack within them is quite another. Application code depends on various other software components like web servers, application servers, file servers and databases among others. Configuration management tools help you control and manage such resources for your application and their lifecycles. For example, you would use these for things like

- Provisioning of application components
- Configuration of application components
- Application deployment
- Software updates
- Access control

There are many open source configuration management tools available to DevOps teams, most notably Chef, Puppet and Ansible. In principal, they all provide the ability to create "recipes" for managing the set-up of application environments. Within the AWS ecosystem, Amazon provides the OpsWorks tool that integrates with Chef and provides a comprehensive interface to automate and orchestrate the configuration chain.

OpsWorks is organized into four different management components

- Stack
  - top level container for your resource set
- Layer
  - units within the stack. e.g. web server (Nginx), db (MySQL)
- Instances
  - server instances to which the layers are mapped
- App
  - the application build

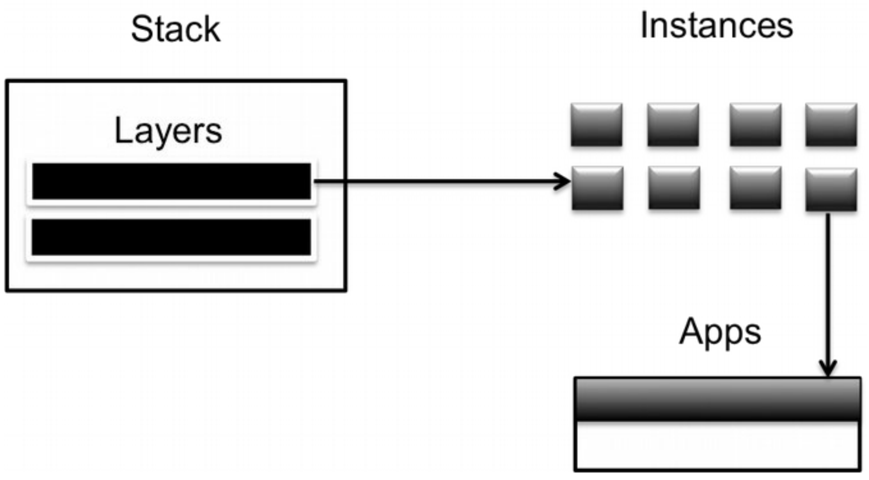


Fig 5. OpsWorks structure

As illustrated above, each stack consists of one or more layers, which are mapped to instances where the application code is deployed. The goal is to provide maximum ability to adapt to changes, scale the application as demanded by a growing business and extend the technology stack to incorporate innovations faster and better.

While everything that can be done with OpsWorks can also be done directly though Chef, OpsWorks provides a layer of management convenience that reduces the learning curve. Your choice of tool will also depend on the skill and background of your team. For instance, a team that is trained in Python may choose to go with Ansible (a Python based tool) to give them tighter control and leverage existing skill set.

Of course, it is possible that your application may not need this level of configuration flexibility and complexity. In that case you can use a PaaS service like AWS Elastic Beanstalk that manages all your provisioning needs. For example, you may have a standard webapp architecture with a web server and database. We should note however, that Beanstalk requires your application to be stateless so consider if that might be a limitation for you. It is also possible to start with Beanstalk to get a quick deployment and then migrate to a more sophisticated OpsWorks based configuration architecture as your application grows. At Kanda, we have experience with both models and have helped many clients to transition and scale out their application architecture.

2.3 Environment Management

A constant source of frustration for development teams is when a piece of code works on a developer system but doesn’t work outside of it. The reason is a “set-up-your-own” attitude where developers get to configure and customize their environment themselves according to personal preferences. This inevitably leads to a host of configuration inconsistencies creating problems that are difficult to identify and debug and cost valuable resource hours to resolve.

The solution is to create self-contained images for all resources with proper configuration. There are a number of ways DevOps admins can choose to accomplish this. Within the AWS ecosystem a popular technique is to use Amazon Machine Images (AMI). These are instance independent images that have all requisite software installed and configured. If you use EBS volumes exclusively, then creating simple disk images can work just as well.

While AMIs are good for creating images for entire VMs, you can also take application isolation a step further by using a tool like Docker. Instead of managing individual VMs for each app, Docker uses a more complex approach of creating application “containers” to provide a finer level of isolation. One of the key benefits of this approach is component reuse – a mechanism where base packages can be repurposed to create larger ones. However, the added complexity is only warranted when you have dozens of machine images with lots of overlapping components.

2.4 Monitoring

Operational environments are live and organic entities where resources are continuously being consumed. As such, it is imperative to have a robust system in place to keep an eye on what resources are doing in real time. Monitoring systems keep track of numerous metrics for tracking things like CPU, Disk, RAM usage and network latency levels to allow your teams to

- Watch the status of each resource and its utilization
- Alert and notify teams of problems as they occur
- Audit resources to identify potential issues before they happen

The principal tool in AWS for monitoring is CloudWatch. It provides the ability to configure notifications to events around standard resource metrics as well as custom ones. Additionally, you can also create policies to trigger automated actions in response to those events.

If you don't want to have a tight dependency on AWS then you can consider tools like Zabbix or Nagios instead of CloudWatch for your monitoring needs. They are functionally equivalent though require a bit more effort to set up.

2.5 Deployment

For application deployment, DevOps enables agile software delivery by operating an automated cycle of continuous integration and continuous deployment. Continuous integration means that all working copies of the code are continuously merged to a main stream. Continuous delivery is the idea that code is tested and available in a releasable state at any given time. The objectives to are to:

- Maintain integrity of the source tree in a collaborative multi-user environment
- Provide the ability to create custom deployment workflows
- Automate deployments that can be scheduled or on-demand
- Establish consistency of the build-release lifecycle

There are numerous tools available to achieve this like Jenkins and Bamboo. AWS essentially provides a convenient layer around these tools to facilitate ease of use in the cloud environment. These include

AWS CodeCommit

- source/version control, Git-based

AWS CodePipeline

- to create workflows for orchestrating build and deployment steps

AWS CodeDeploy

- to move application through to destination environment instances

This is illustrated below

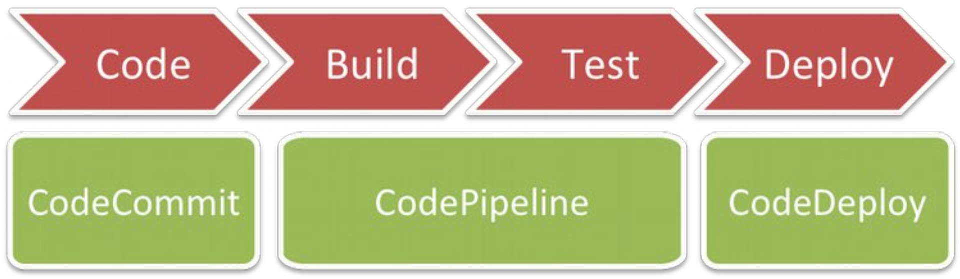


Fig 6. Build-deploy pipeline

Of course, many teams will opt to use Git or Jenkins natively. Also, teams using JIRA for Agile development might have a preference for Bamboo to take advantage of built-in integration with that tool. Ultimately the choice doesn't matter – what's important is to standardize on a toolset and stick with it.

2.6 Security and Compliance

Ensuring an exceptional level of security for every environment is a must for any ops team. This responsibility falls into a few categories

- Infrastructure security, controlling access to network and resources
- Data security including data at rest and in motion

AWS handles these by providing

- Built-in VPC network firewalls
- IAM (Identity and Access Management) service for granular access control to all AWS resources
- Encryption service for data at rest
- TLS for data in motion across all services

While networks created in AWS are protected through firewalls, creating publicly accessible end points always requires great care and a thorough knowledge of networking.

A key area of stringent security protocols is regulatory compliance. AWS provides the necessary tools for managing compliance with standards like SOC, HIPPA, HITEC among others. However, even with all the tools provided by AWS this is a complex process that requires expert guidance to implement. At Kanda, we leverage our vast experience and expertise in AWS to help our clients successfully navigate through the intricacies of the compliance process.

3. CHALLENGES

With all its promise there are several challenges to successfully implementing DevOps. One mentioned above is the lack of standardized tooling. We can address that to some degree by working within the AWS ecosystem and leveraging its built-in tools.

However, DevOps is a complex set of practices connecting people, process and tools. For enduring success an organization has to address all of these three aspects.

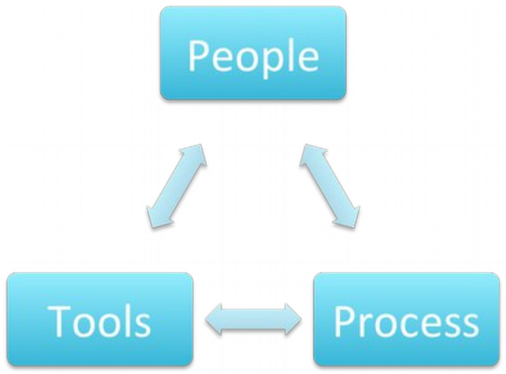


Fig 8. Aspects of DevOps org

"People" here includes team organization and culture as well as skills and talent. Earlier, we described how DevOps merged agile development with agile operations. To make that work teams need to have a very collaborative culture focused on quality. Creating agile ops also means managing an operations backlog similar to a product backlog for continuous updates to infrastructure.

A robust DevOps team requires a broad range of knowledge of tools and technologies. A typical DevOps engineer has a deep background in systems engineering and working proficiency in all layers of the software development stack. More importantly, what they also need is enormous amount discipline and experience because ultimately success comes from consistency and execution of repeatable processes. Staffing, training and managing such an operation can pose significant organizational challenges. In this respect, Kanda Software provides a clear advantage with its deep pool of talent, broad base of skillsets, knowledge of best practices and a record of success in many diverse environments.

The AWS approach to DevOps aims to provide the entire range of tooling for your needs without demanding that you use them entirely giving you the freedom to choose only the tools you need for the required task. At Kanda, we help our clients understand the best tools for their business and provide them with the guidance and service they need to realize the promise of continuous value with DevOps.

CONTACT US

Kanda Software

200 Wells Avenue, Newton MA 02459

✆ **617-340-3850**
✉ **contact@kandasoft.com**