DevOps is a new term that primarily focuses on improved collaboration, communication, and integration between software developers and IT operations.

Prior to agile, the dominant waterfall development methodology was a sequence starting with a requirements phase where 100% of the system under development was defined up front. This approach is inflexible and monolithic.

Agile model focuses on building iterations of working software that would evolve over time with the use of numerous tools such as IDEs, unit test frameworks and code optimizers. Agile software development focuses on the collaboration between the business and its developers, whereas, DevOps focuses on the collaboration between developers, IT operations(system administrators, database administrators, network engineers, infrastructure architects, and support personnel) and security teams.

DevOps practices vary with the task: With application development, DevOps focuses on code building, code coverage, unit testing, packaging, and deployment. With infrastructure, DevOps focuses on provisioning, configuration, orchestration, and deployment. But in each area the underlying principles of version management, deployment, roll back, roll forward, and testing remain the same.

**Infrastructure as Code** refers to developing infrastructure provisioning in a declarative way and storing the configurations in a version management system.

AWS CloudFormation provides templates to define and model AWS resources in a repeatable and reliable way. The collection of resources under management is called a “stack”, which is managed through the AWS Management Console, AWS Command Line Interface, or AWS CloudFormation APIs. When a stack is created or updated in the console, events are displayed showing the status of the configuration. If an error occurs, the stack is rolled back to its previous state. Amazon Simple Notification Service (Amazon SNS) helps manage these events.

**Continuous Deployment** refers to the automatic deployment of production-ready application code. If a deployment fails, it can be automatically rolled back to a previous version.

AWS CodeDeploy provides the ability to deploy applications across an Amazon E2C fleet with minimum downtime, centralizing control and integrating with existing software release or continuous delivery process.

AWS CodePipeline is a continuous delivery and release automation service that aids smooth deployments. AWS CodePipeline allows the developer’s build, test and release process to be automated.

AWS CodeCommit is a secure, highly scalable, managed source control service that hosts private Git repositories. CodeCommit eliminates the need to operate one’s own source control system or worry about scaling. It supports the standard functionality of Git, allowing it to work seamlessly with existing Git-based tools. CodeCommit’s online code tools can also be used to browse, edit, and collaborate on projects.

In AWS Elastic Beanstalk, code changes deployments are stored as “application versions,” and infrastructure changes are deployed “saved configurations.” An example of a saved configuration is an AWS Elastic Beanstalk configuration that uses Elastic Load Balancing and Auto Scaling rather than a single instance.

AWS Elastic Beanstalk and AWS OpsWorks gives the option of defining which instances in which layers should be updated when deployments are made.

Blue-Green Deployment is a DevOps deployment practice that uses domain name services (DNS) to make application deployments. The strategy involves starting with an existing (blue) environment while testing a new (green) one. When the new environment has passed all the necessary tests and is ready to go live, traffic is redirected from the old environment to the new one via DNS. The old production environment can be kept for backup purposes, decommissioned, or used for back-end services like database deployment and failover.

**Automation** focuses on the setup, configuration, deployment, and support of infrastructure and the applications that run on it in a standardized and repeatable manner.

AWS elastic Beanstalk automation can be used to create environments or deploy applications. As the environment is being created, AWS Elastic Beanstalk automatically logs events on the management console providing feedback on the progress and status of the launch. Once complete, the application can be accessed by using the defined URL.

AWS OpWorks provides even more levels of automation with additional features like integration with configuration management software (Chef) and application lifecycle management. Application lifecycle management is used to define when resources are set up, configured, deployed, un-deployed, or terminated. Both AWS Elastic Beanstalk and AWS OpsWorks support application versioning, continuous deployment, and infrastructure configuration management.

**Monitoring-** Communication and collaboration is fundamental in a DevOps strategy, i.e., feedback is critical. In AWS, feedback is provided by two core services: Amazon CloudWatch and AWS CloudTrail.

Amazon cloud Watch allow developers to monitor in real time all AWS resources and the application they have created, configure alarms to send notification when errors occur, configure notifications in numerous formats( email, Amazon SNS, and Amazon Simple Queue Service) and trigger auto scaling up and down of Amazon EC2 instances. This allows system administrators and support teams to focus on other value-added business needs.

AWS CloudTrail is a logging system. It stored logs of all AWS API calls (whether they come directly from a user or on behalf of a user by an AWS service) in an Amazon S3 bucket. Log files are encrypted using Amazon S3 server-side encryption (SSE).

**Security**

Infrastructure and company assets need to be protected. The AWS Identity and Access Management service (IAM) allows the developer centrally manage users and security credentials such as passwords, access keys, and permissions policies that control which AWS services and resources users can access. IAM can be used to create roles that are used widely within a DevOps strategy. An IAM role defines a set of permissions to access the resources that a user or service needs.

**Conclusion:**

Firstly, define infrastructure as code using AWS CloudFormation. Secondly, define how the applications are going to use continuous deployment with the help of services like AWS CodeDeploy, AWS CodePipeline, and AWS CodeCommit. Thirdly, at the application level, use containers like AWS Elastic Beanstalk and AWS OpsWorks to simplify the configuration of common architectures. These services make it easy to include other important services like Auto Scaling and Elastic Load Balancing. Finally, use the DevOps strategy of monitoring (AWS CloudWatch) and solid security practices (AWS IAM).