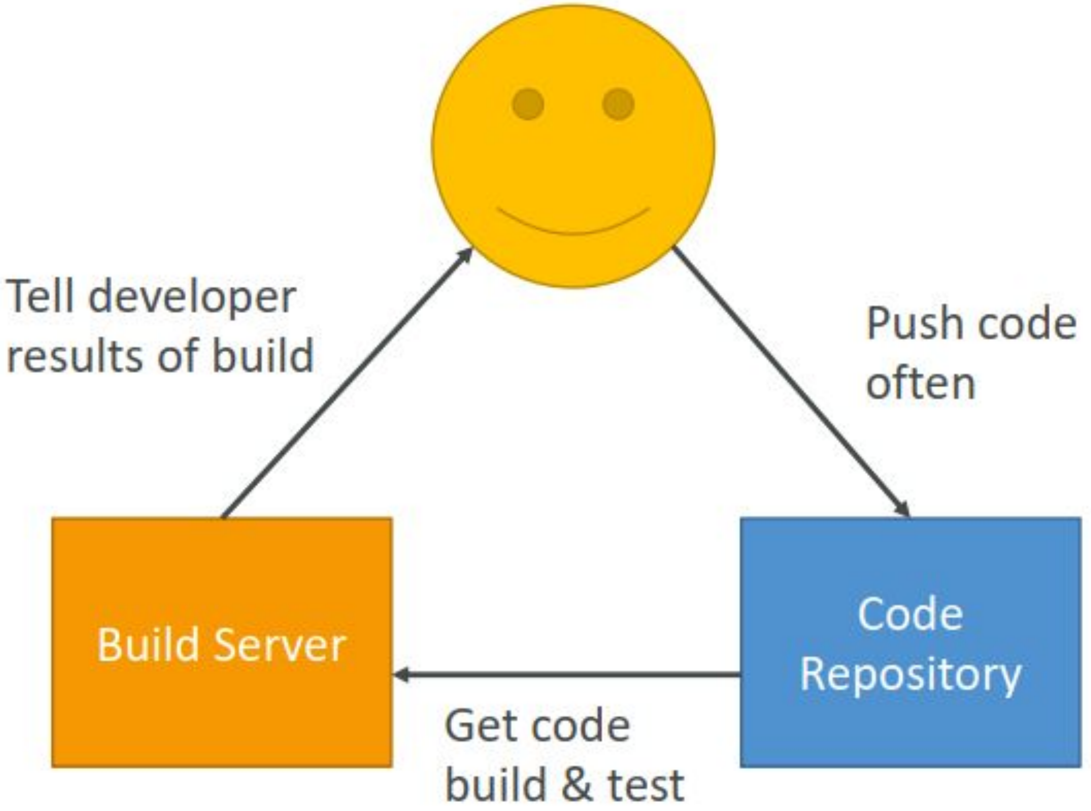
# Continuous Integration?

- Developers push the code to a **code repository** often (**GitHub / CodeCommit / Bitbucket / Gitlab etc…**)
- A **testing / build server** checks/pulls the the code from *Code Repo* as soon as it's pushed (**CodeBuild / Jenkins CI , Travis CI , Circle CI** etc )
- The developer gets feedback about the tests and checks that have passed / failed.
- Deliver faster as the code is tested
- Deploy often
- Find bugs early, fix bugs

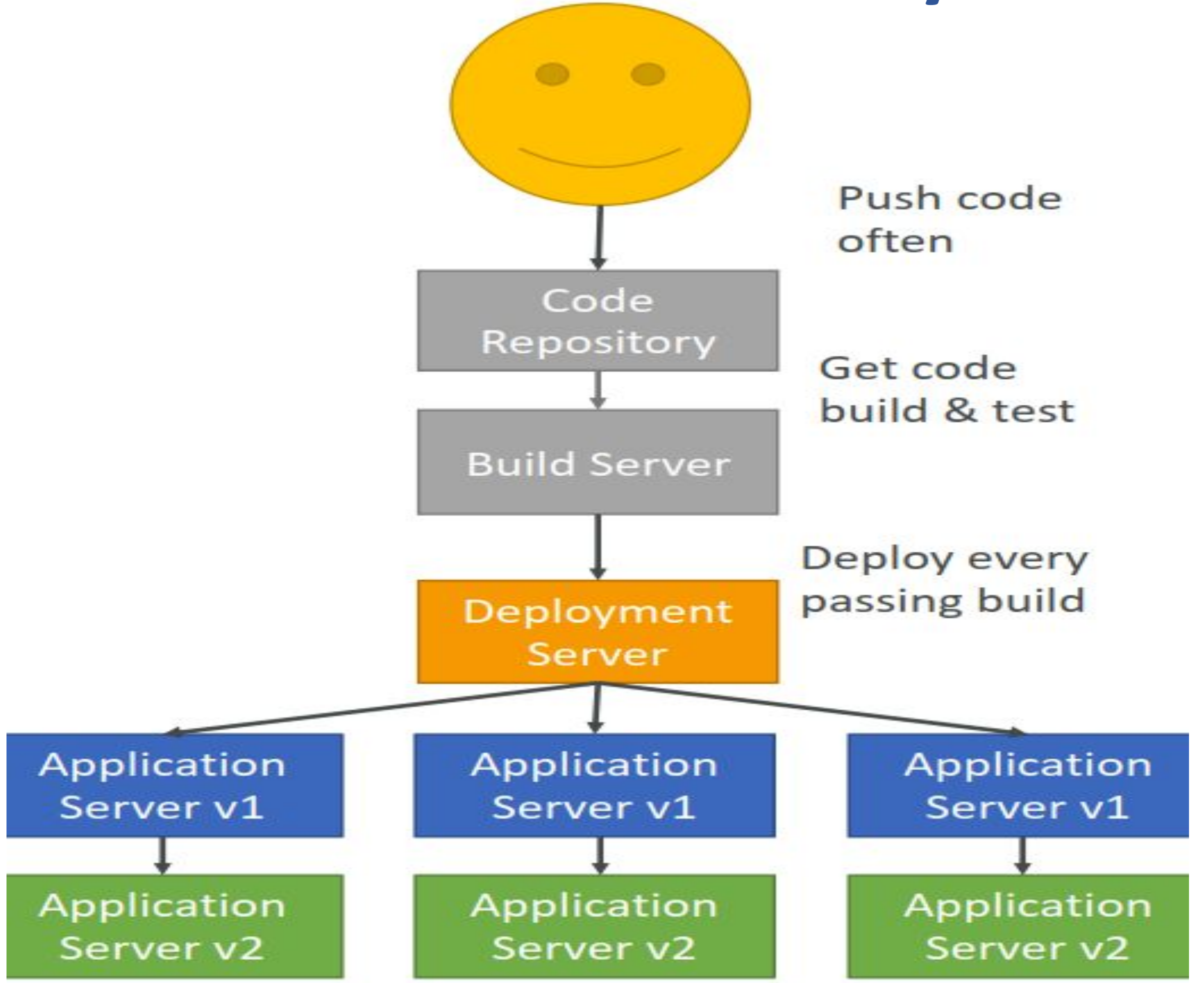
# Continuous Integration



## Continuous Delivery?

- Ensure that the software can be released reliably whenever needed
- Quick Deployments
- Shift from “one release every 3 months” to ”5 releases a day”
- That usually means automated deployment
  - **CodeDeploy**
  - **Jenkins CD**

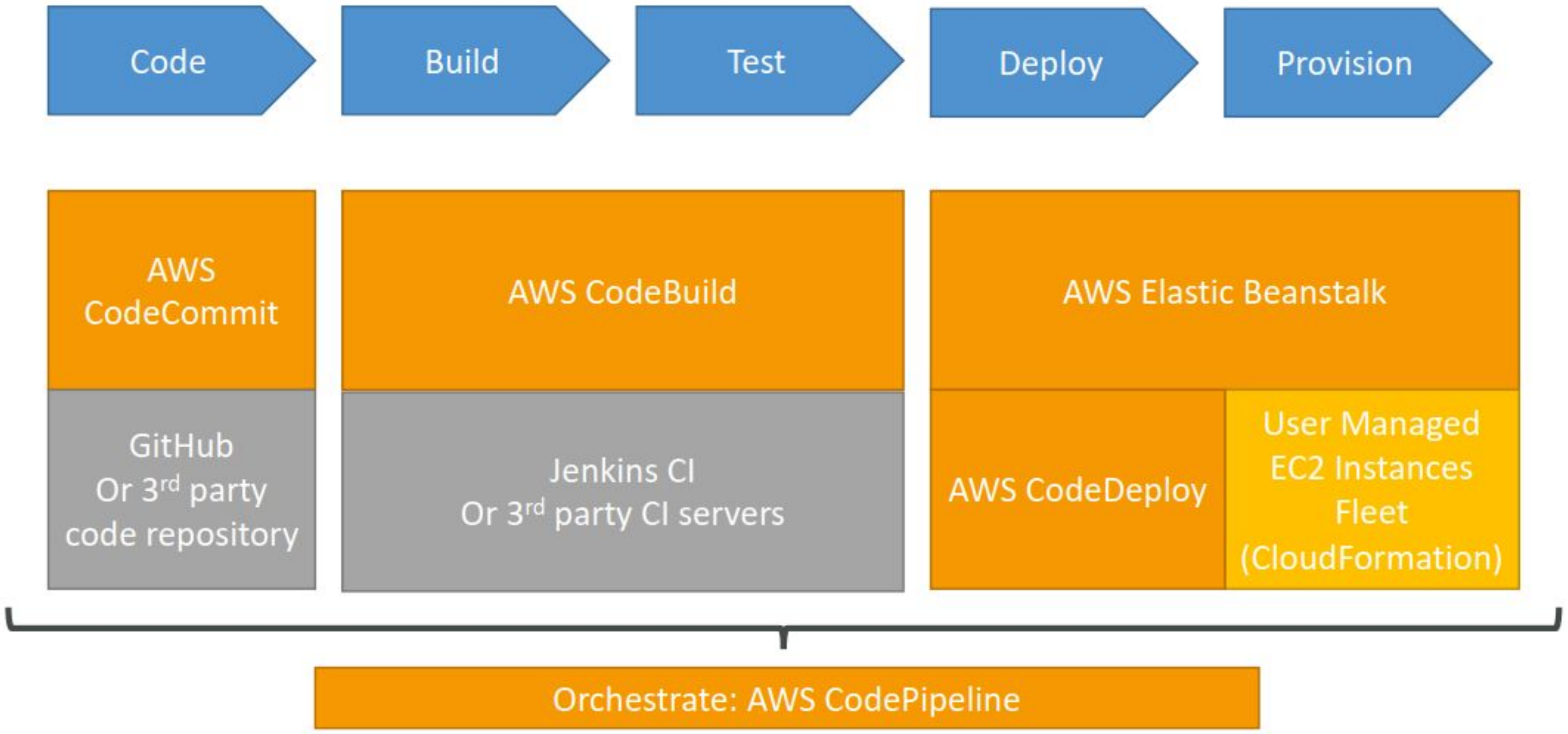
# Continuous Delivery?



## Continuous Delivery & Continuous Deployment?

- **Continuous Delivery:**
  Ability to deploy often using automation
  May involve a manual step to “approve” a deployment
  The deployment itself is still automated and repeated!
- **Continuous Deployment:**
  Full automation, every code change is deployed all the way to production
  No manual intervention of approvals

# Technology Stack for CICD


Code
Build
Test
Deploy
Provision
AWS CodeCommit
GitHub Or 3rd party code repository
AWS CodeBuild
Jenkins CI Or 3rd party CI servers
AWS Elastic Beanstalk
AWS CodeDeploy
User Managed EC2 Instances Fleet (CloudFormation)
Orchestrate: AWS CodePipeline

## Components of AWS CodeDeploy

- **Application**: It represent the application name that needs to get deployed. This name also ensures the correct combination of the deployment group, configuration, and revision.
- **Deployment configuration**: It consists of a set of deployment rules and some condition of deployment success and failure used by CodeDeploy at the time of deployment.
- **Deployment group**: It represents the individual tagged instances or an instance in auto scaling groups where the deployment needs to be done.
- **Deployment type:** The deployment strategy used to make the latest application available on instances or auto scaling after the deployment. AWS CodeDeploy provides two different deployment types:

  In-place deployment

  Blue-green deployment

## Components of AWS CodeDeploy

- **Revision**: It is basically an archive file, which contains source code, deployment scripts, and the **appspec.yml** file. It is stored in the S3 bucket or **GitHub/CodeCommit repository.**
- **Service role**: In case of AWS CodeDeploy, a service role generally has the following permissions:

- Reading the tags of instances and auto scaling groups to identify the instances in which an application can get deployed.

- Performing operations on instances, auto scaling groups, and ELB.

- Accessing SNS for publishing information & CloudWatch for alarm monitoring

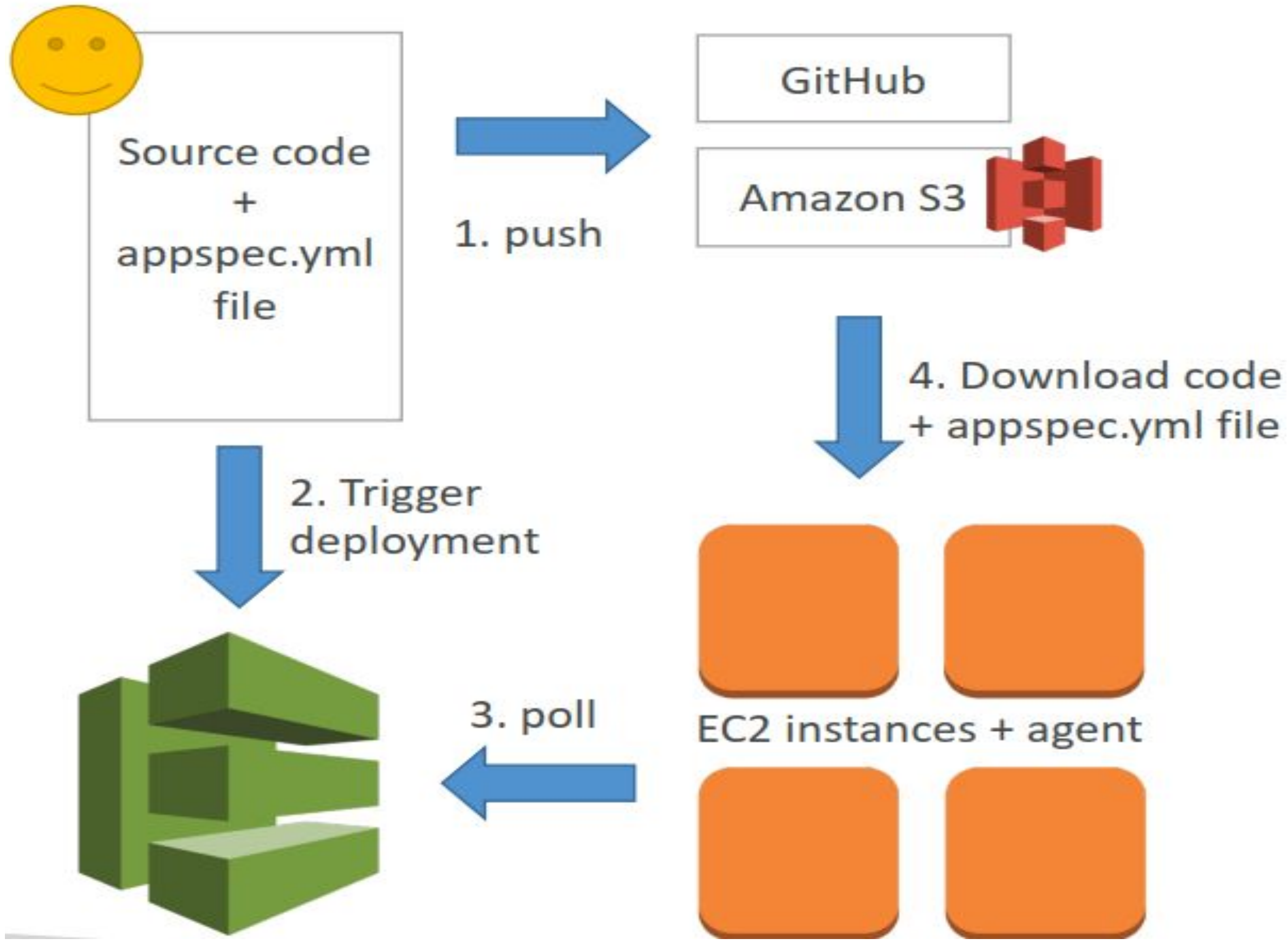
# How CodeDeploy Works

- Each EC2 Machine (or On Premise machine) must be running the **CodeDeploy Agent.**
- The **CodeDeploy Agent** is continuously polling AWS CodeDeploy for work to do.
- CodeDeploy sends **appspec.yml** file.
- Application is pulled from **GitHub or S3**.
- EC2 will run the deployment instructions.
- CodeDeploy Agent will report of success / failure of deployment on the instance.

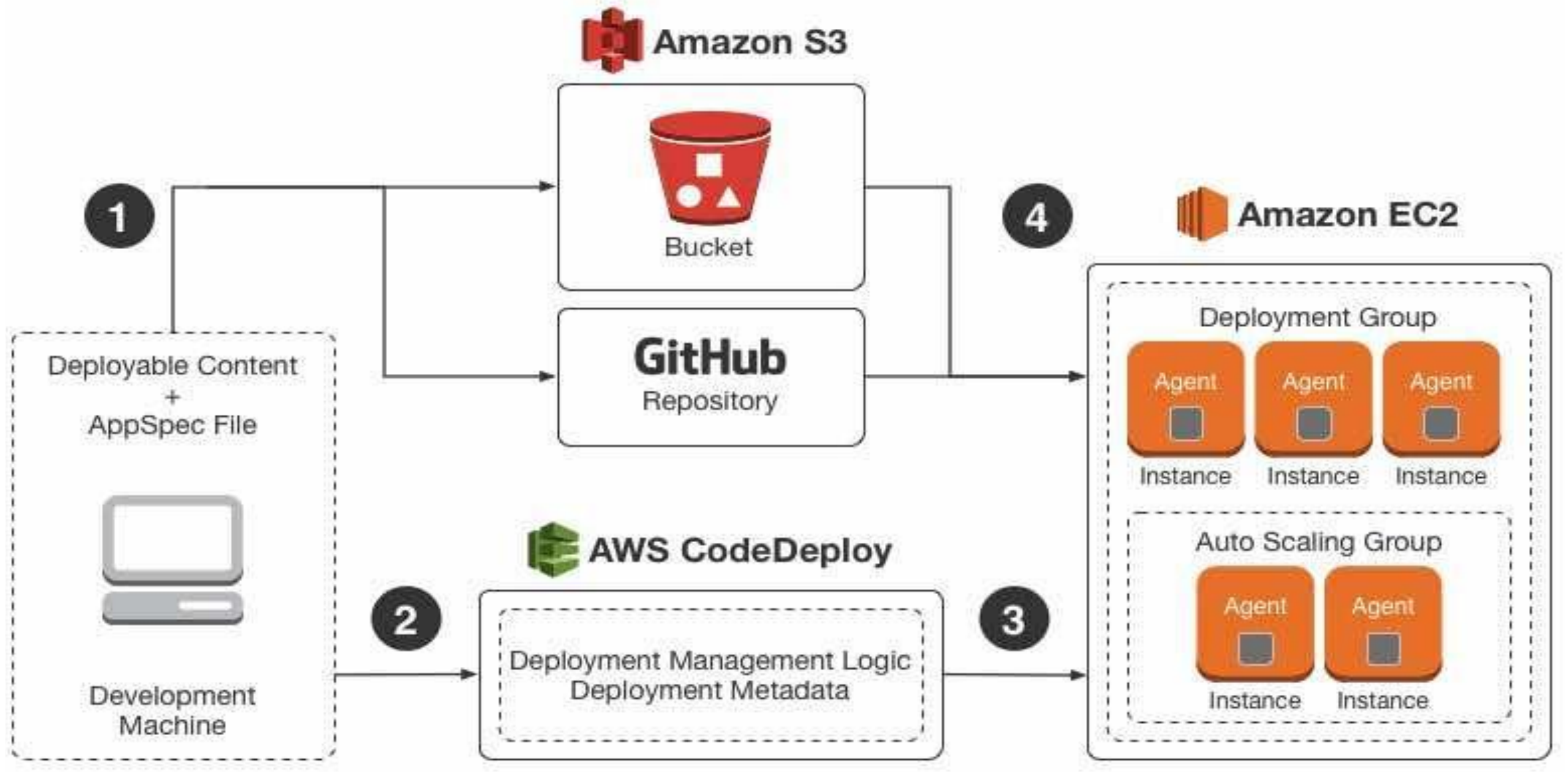
# How CodeDeploy Works

- EC2 instances are grouped by deployment group (dev / test / prod)
- Lots of flexibility to define any kind of deployments.
- CodeDeploy can be chained into CodePipeline and use artifacts from there.
- CodeDeploy can re-use existing setup tools, works with any application, auto scaling integration.
- Note: Blue / Green only works with EC2 instances (not on premise)
- CodeDeploy does not provision resources.

# How CodeDeploy Works



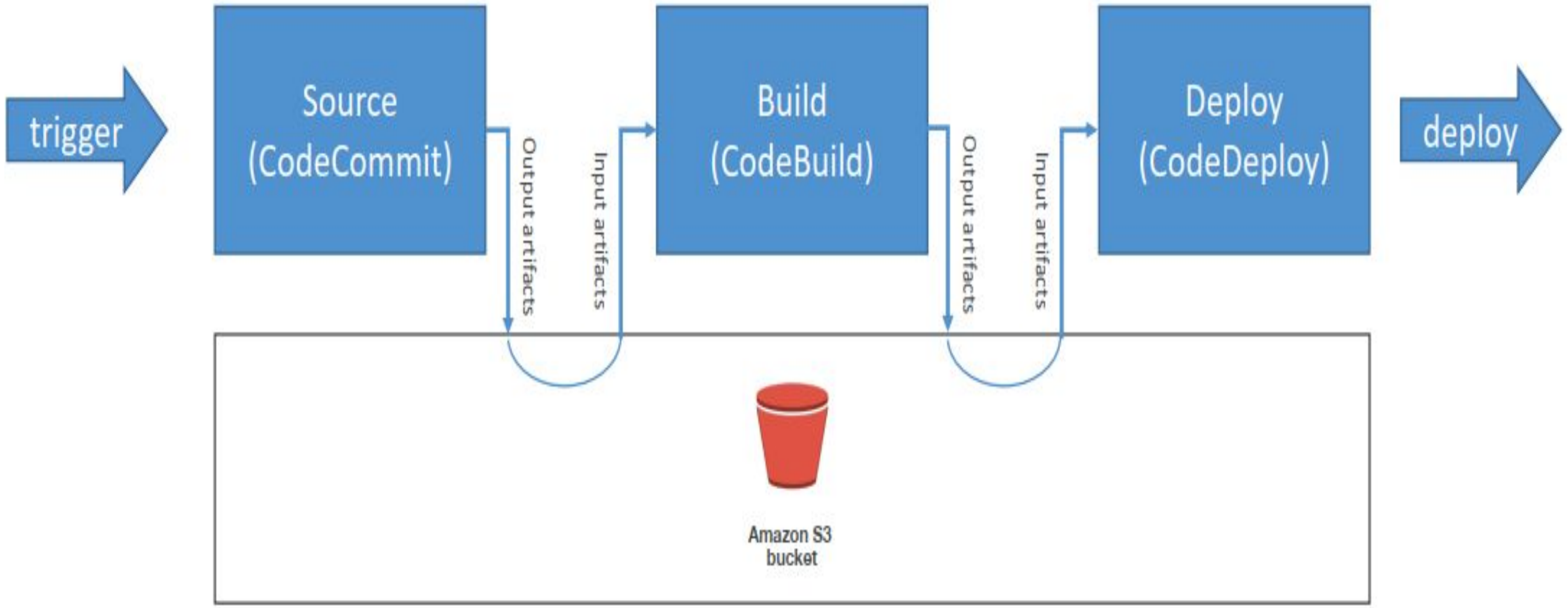
# How CodeDeploy Works



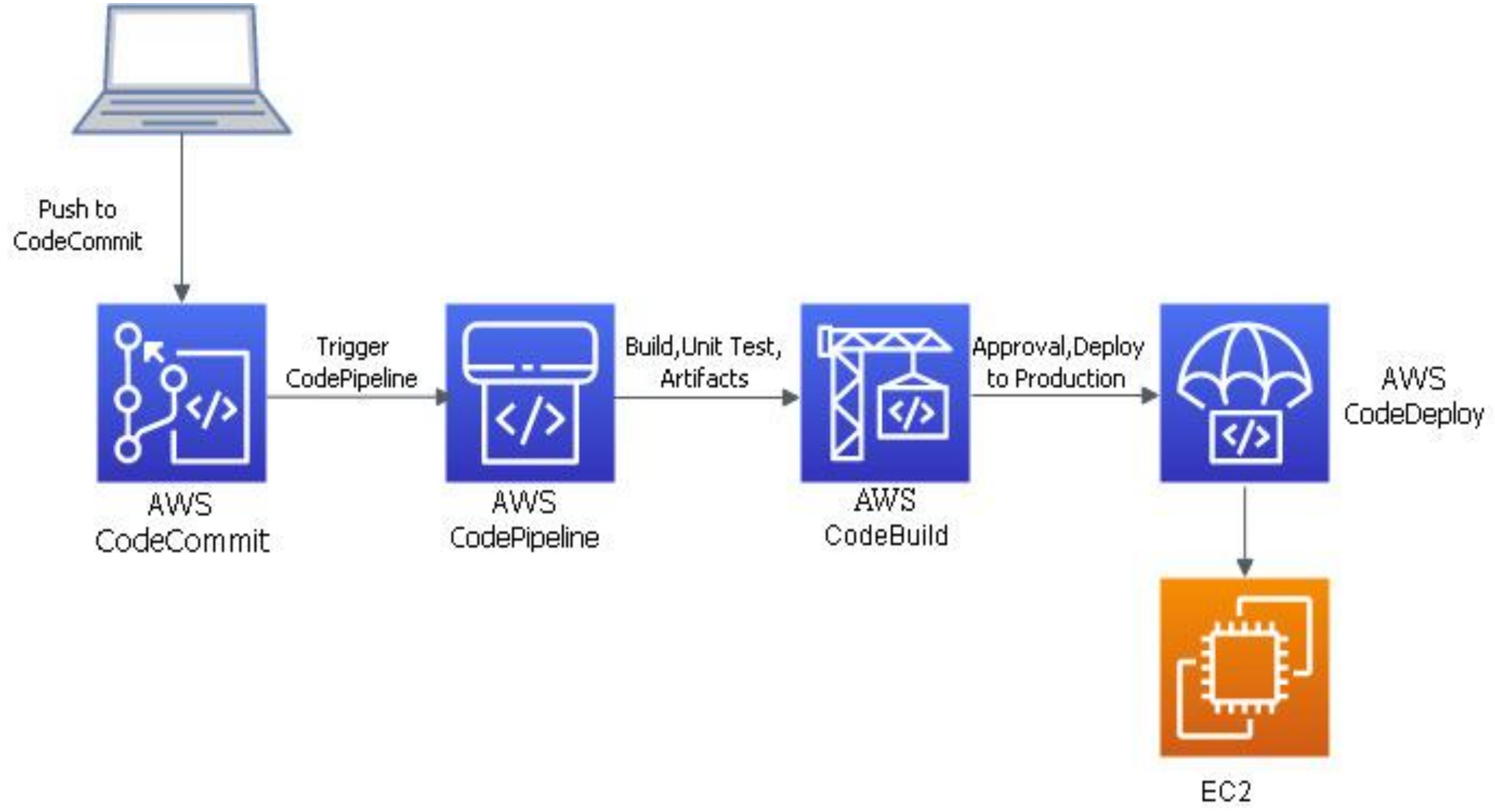
## CodeDeploy Features

- AWS CodeDeploy is the deployment service provided by AWS that automates the deployment of the application to Amazon EC2 instance, Elastic Beanstalk, and onpremise instances.
- CodeDeploy can deploy application files stored in Amazon S3 buckets, GitHub repositories, or BitBucket repositories.
- Lots of flexibility to define any kind of deployments.
- CodeDeploy can be chained into CodePipeline and use artifacts from there.
- CodeDeploy can re-use existing setup tools, works with any application, auto scaling integration.
- Support for AWS Lambda deployments, EC2
- CodeDeploy does not provision resources.

# AWS CodePipeline Artifcats



# AWS CICD Flow


Push to CodeCommit
AWS CodeCommit
Trigger CodePipeline
AWS CodePipeline
Build,Unit Test, Artifacts
AWS CodeBuild
Approval,Deploy to Production
AWS CodeDeploy
EC2