Microservice CI/CD Pipeline

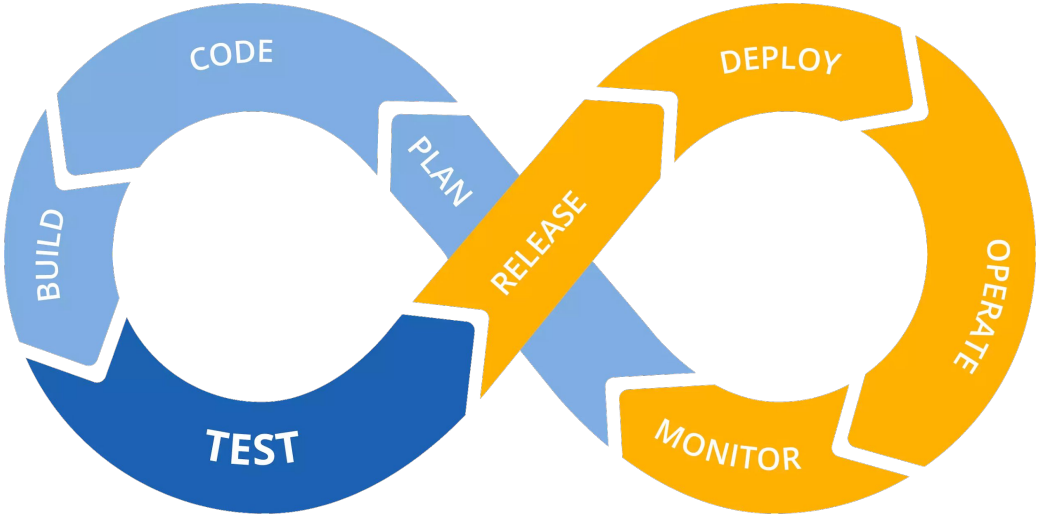
CLARUSWAY
WAY TO REINVENT YOURSELF

Table of Contents

- What is DevOps?
- What is Continuous Integration?
- What is Continuous Delivery/Deployment?
- DevOps Setup
- Microservice Petclinic

1 What is DevOps?

What is DevOps?

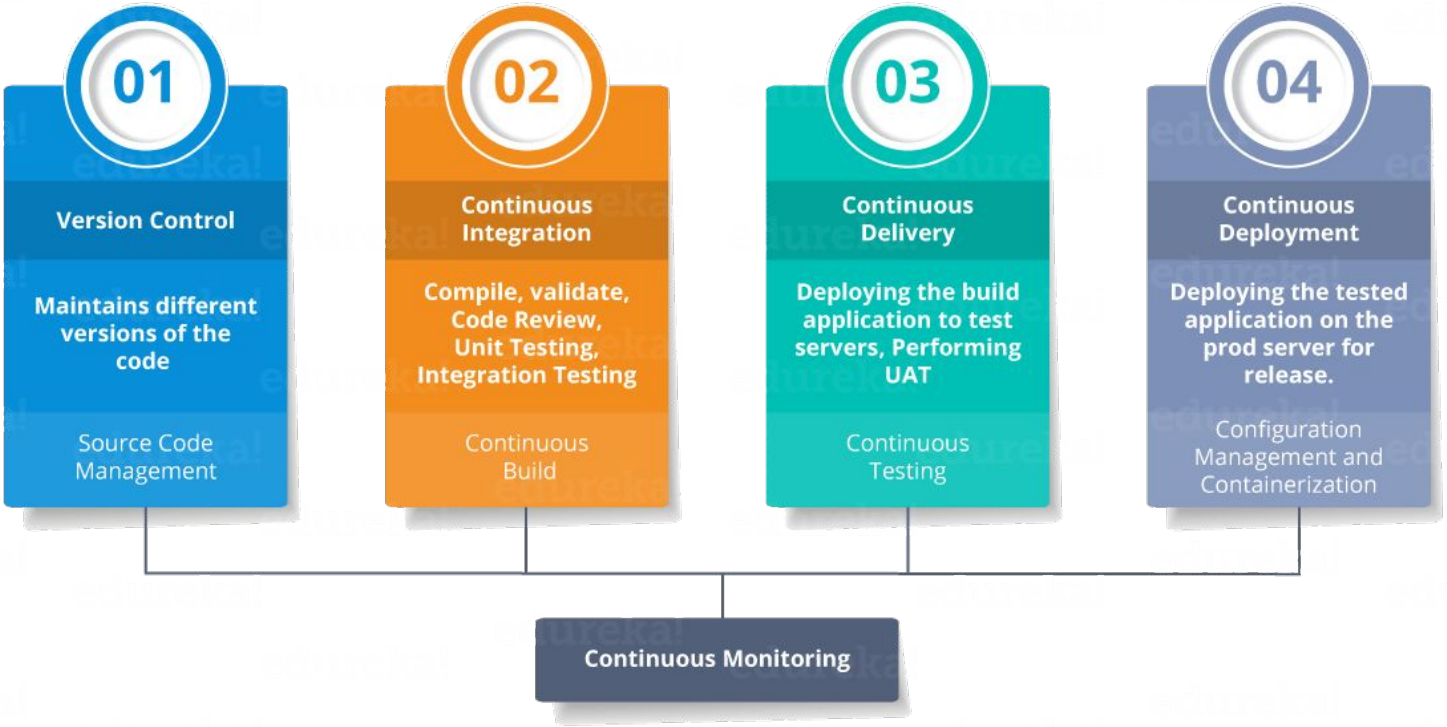


DevOps is a software development approach which involves:

- continuous development,
- continuous testing,
- continuous integration,
- continuous deployment
- continuous monitoring

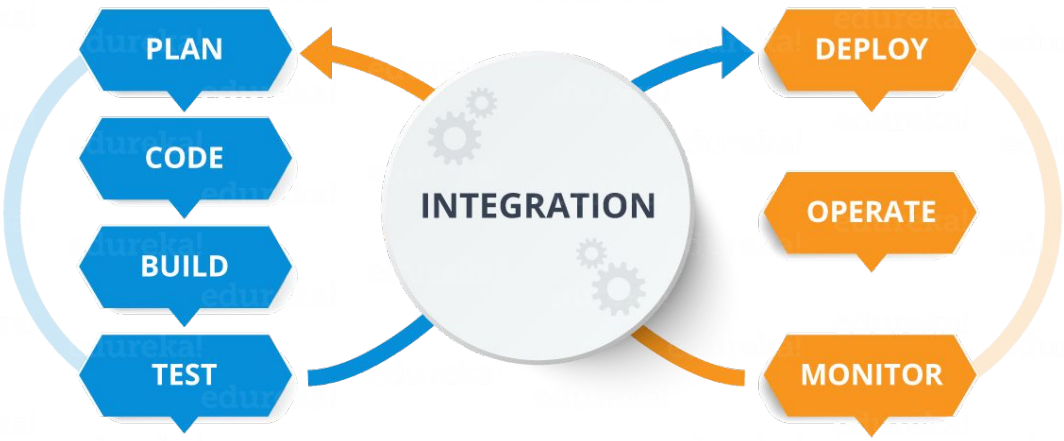
of the software throughout its development lifecycle

What is DevOps?



2 What is Continuous Integration?

What is Continuous Integration?

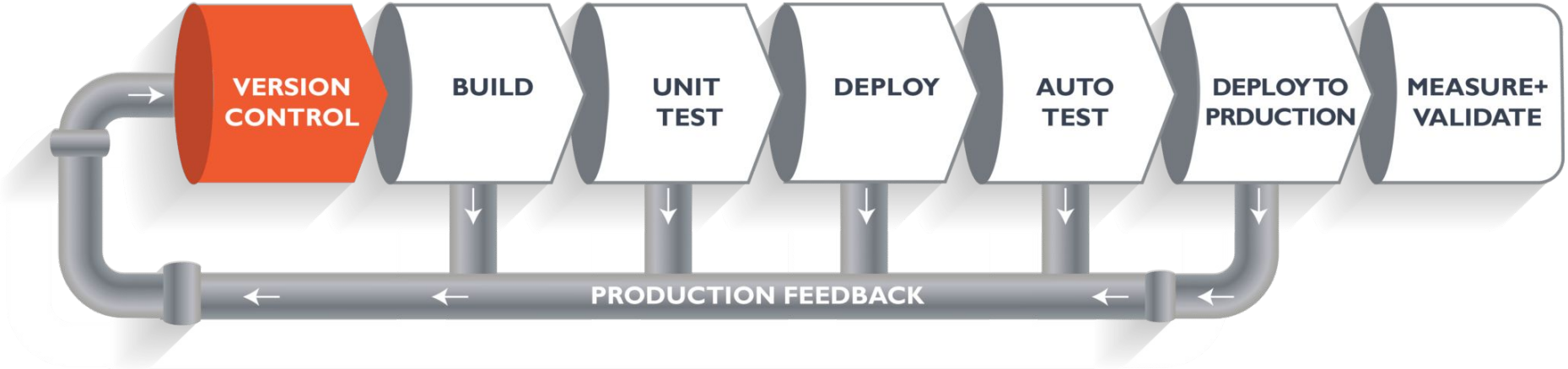


Continuous integration is a software development method where members of the team can integrate their work at least once a day. In this method, every integration is checked by an automated build to search the error.

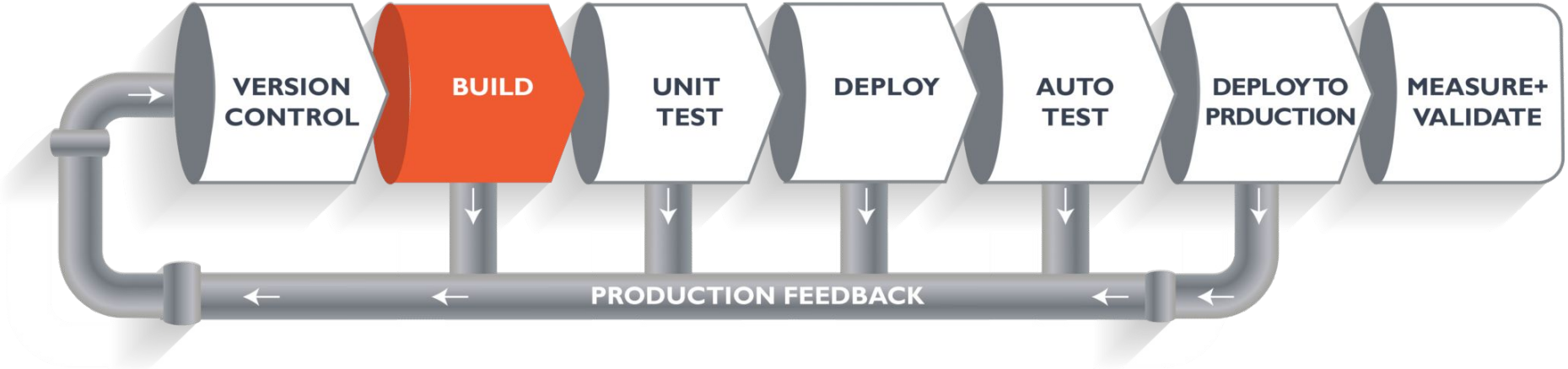
With CI vs Without CI

Development without CI	Development with CI
Lots of Bugs	Fewer bugs
Infrequent and slow releases	Regular working releases
Difficult integration	Easy and Effective Integration
Late bug finding(days,weeks)	Early bug finding(minutes,hours)
Issue raised are harder to fix	Find and fix problems faster and more efficiently.
Poor project visibility	Better project visibility

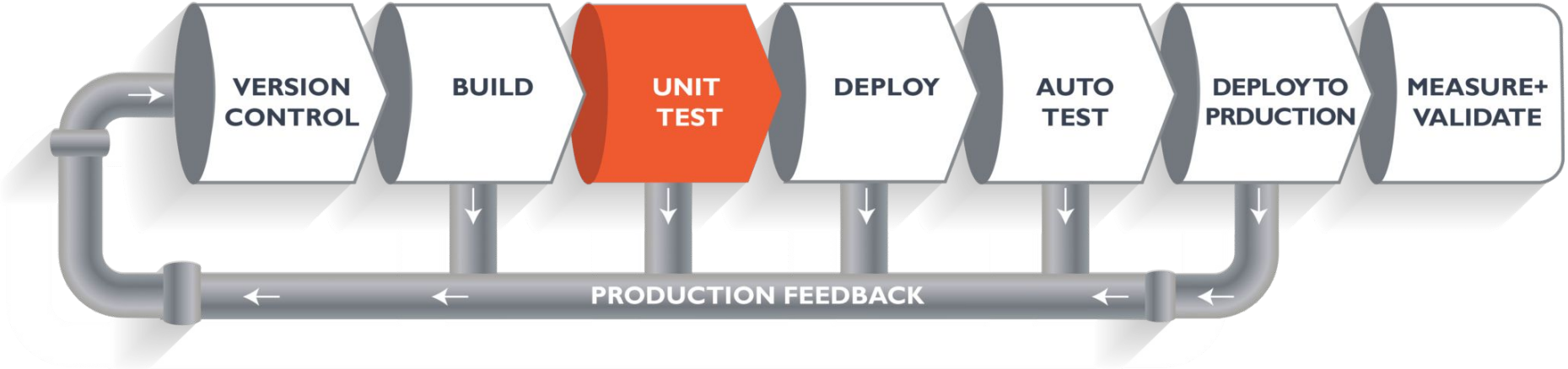
What is Continuous Integration?



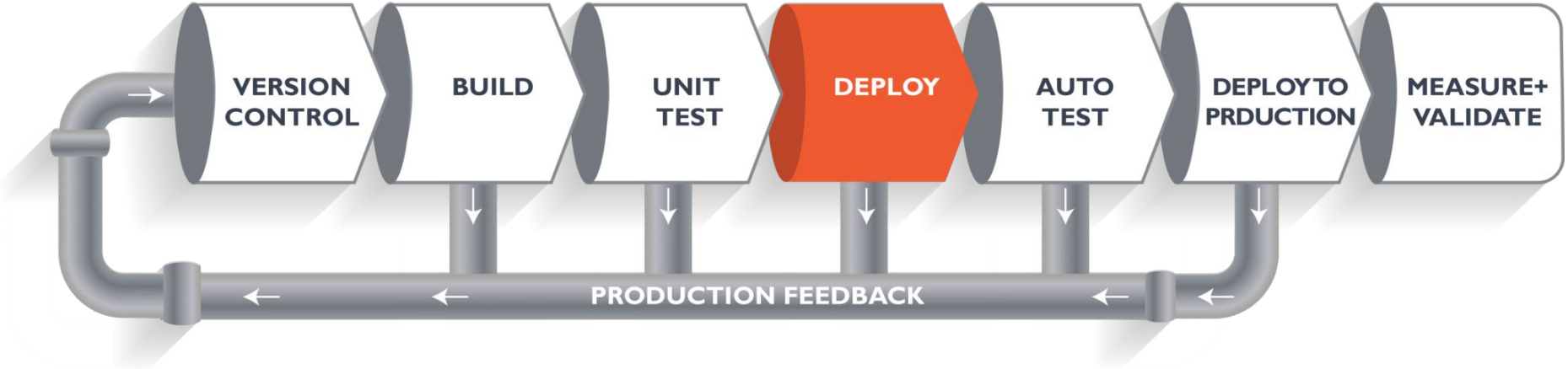
What is Continuous Integration?



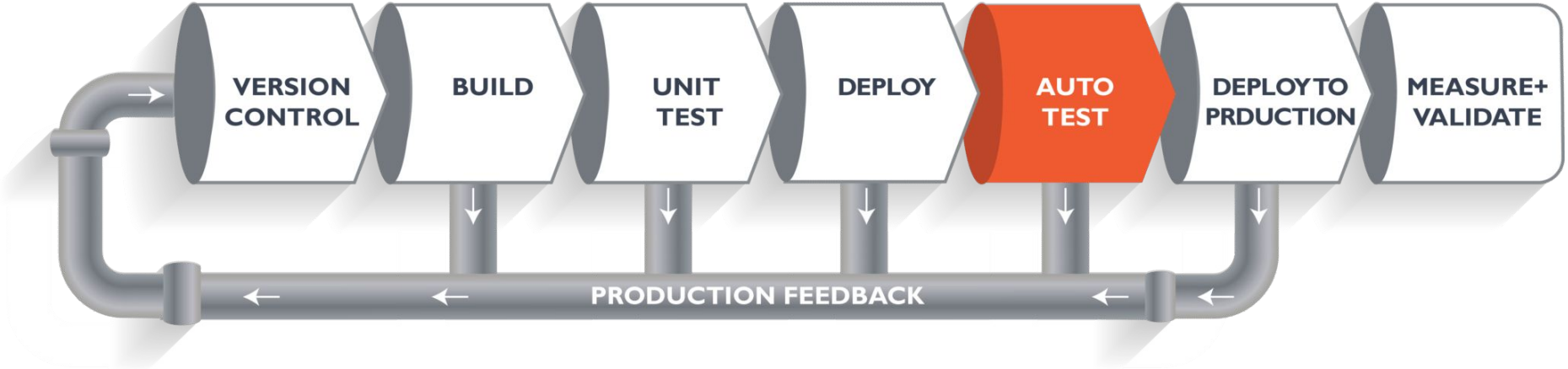
What is Continuous Integration?



What is Continuous Integration?



What is Continuous Integration?



User Acceptance Tests

Functional Tests

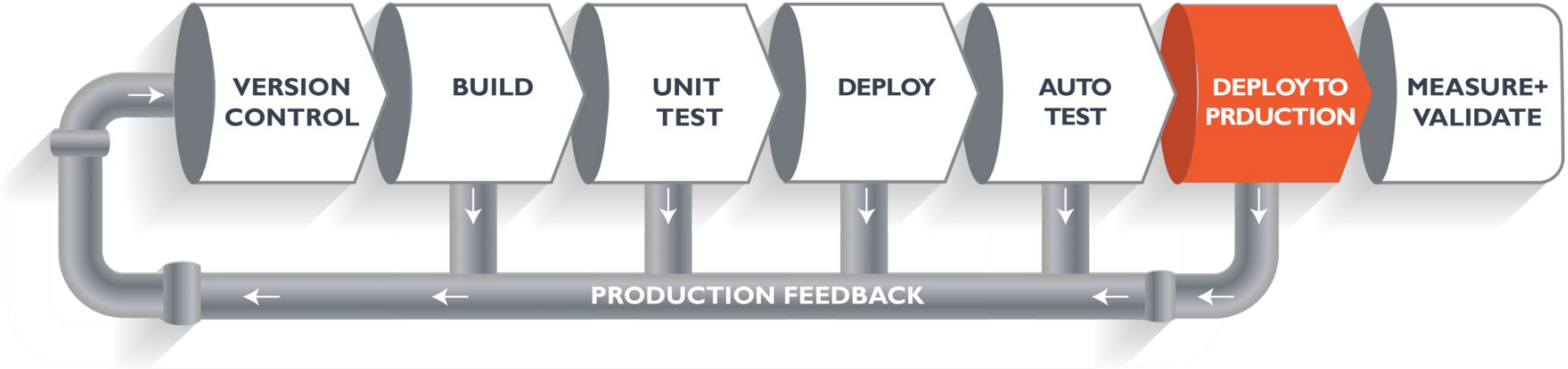
Load Tests

Security Tests

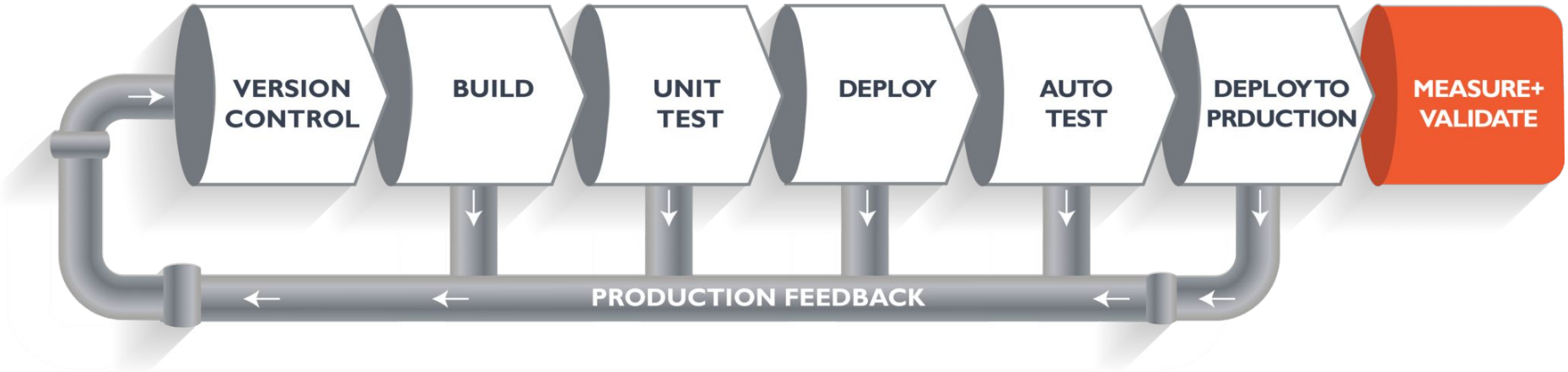
3

What is Continuous Delivery/Deployment?

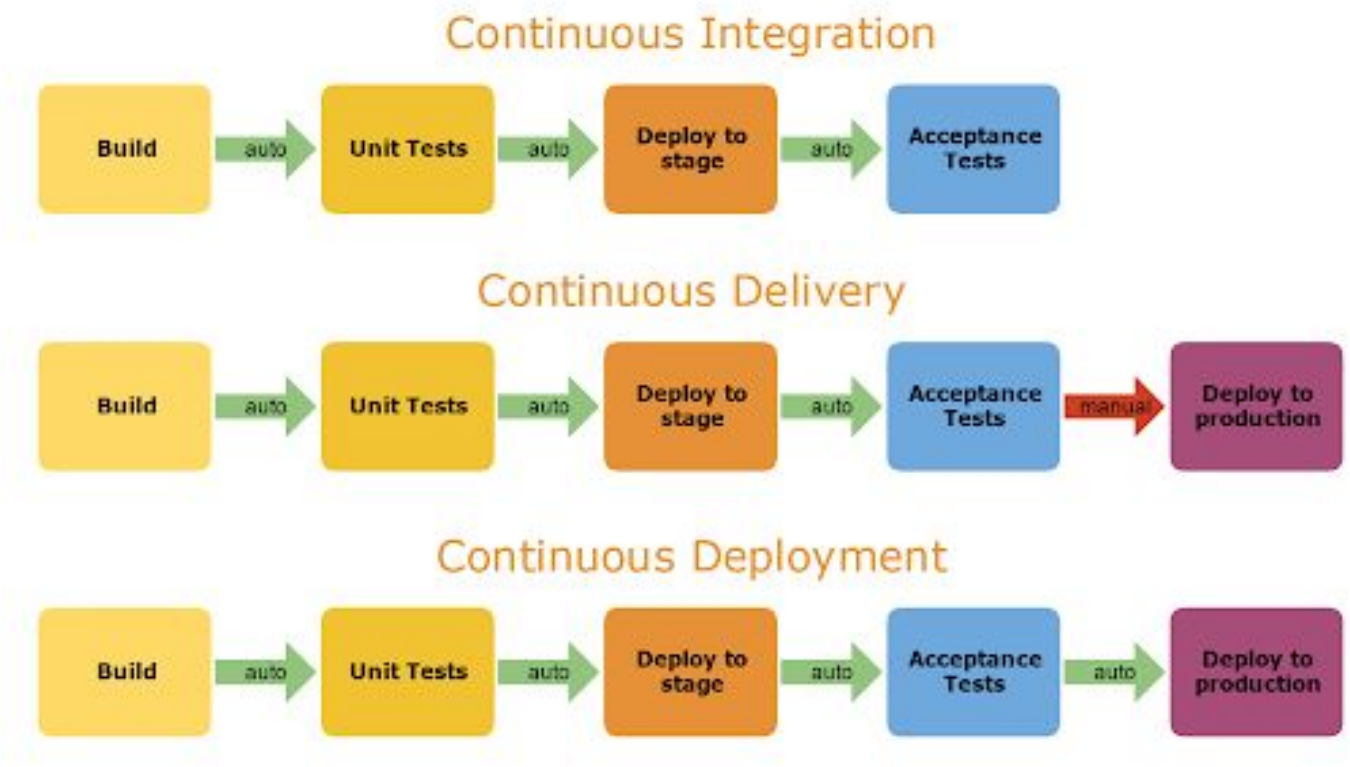
What is Continuous Delivery/Deployment?



What is Continuous Delivery/Deployment?

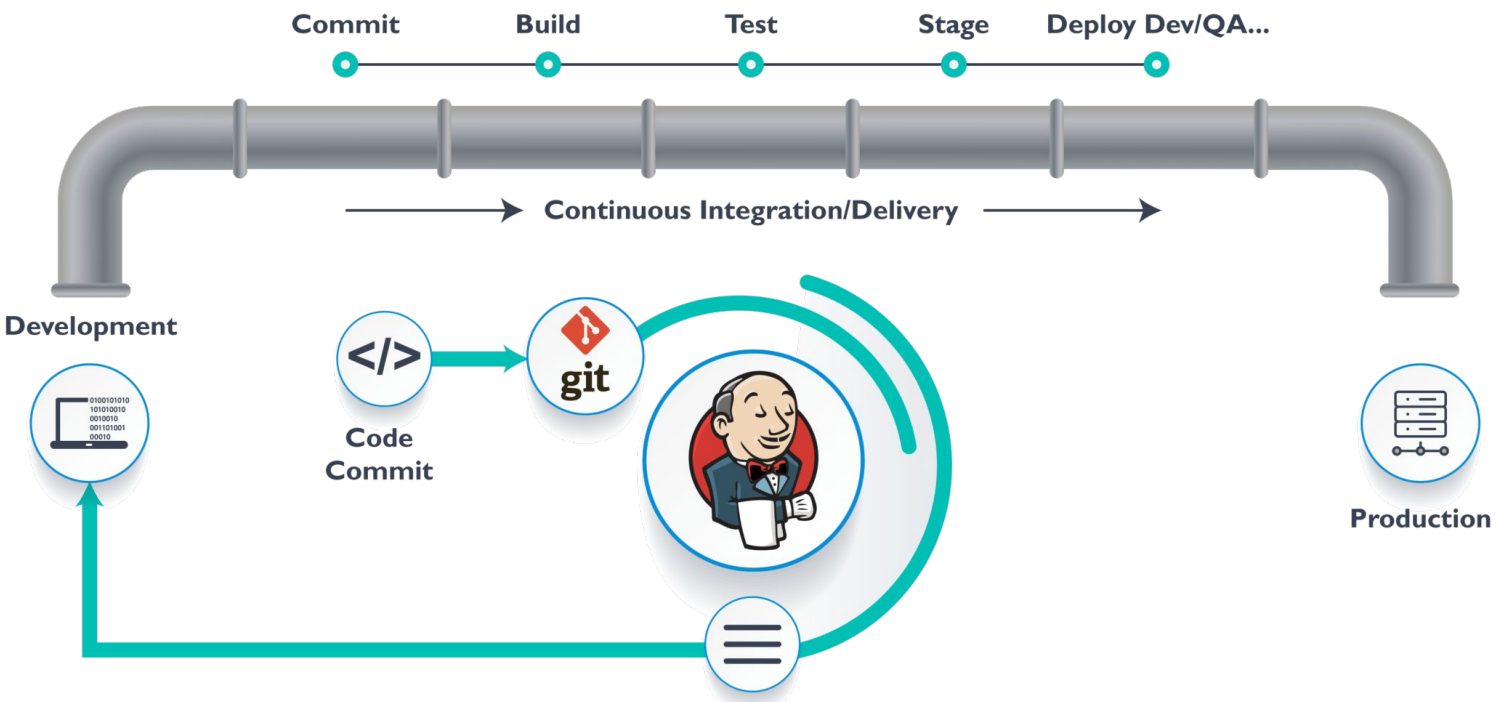


Continuous Delivery vs Continuous Deployment



4 DevOps Setup

Continuous Integration Server



Continuous Integration Server

Gitlab

Teamcity

Version Control Stage

1. Select repository host
2. Repository strategy single vs multi
3. Branching strategy
4. User roles and rights
5. User groups
6. Pull Request policy

- How to setup and manage repos on ….
- How to create branch(s) and manage rights
- What is gitflow?(dev,master,feature/XXX...)
- How to manage users,groups,roles
- PR policies?

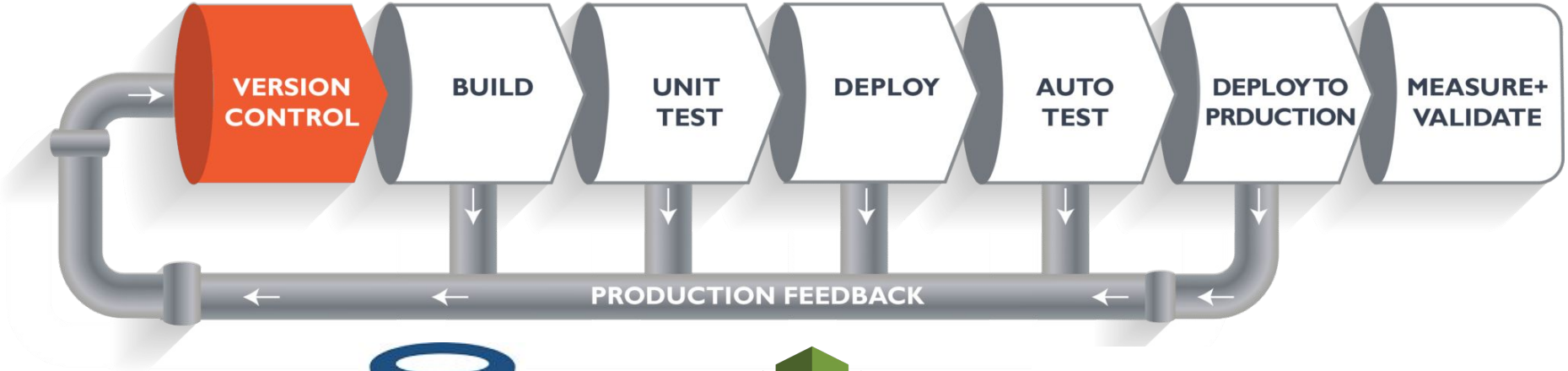
CodeCommit

AWS Tasks

- CodeCommit repository configuration

Version Control Stage

EC2
CodeCommit



GitHub

Bitbucket
Atlassian

Codecommit
AWS

Gitlab
Gitlab

Build Stage

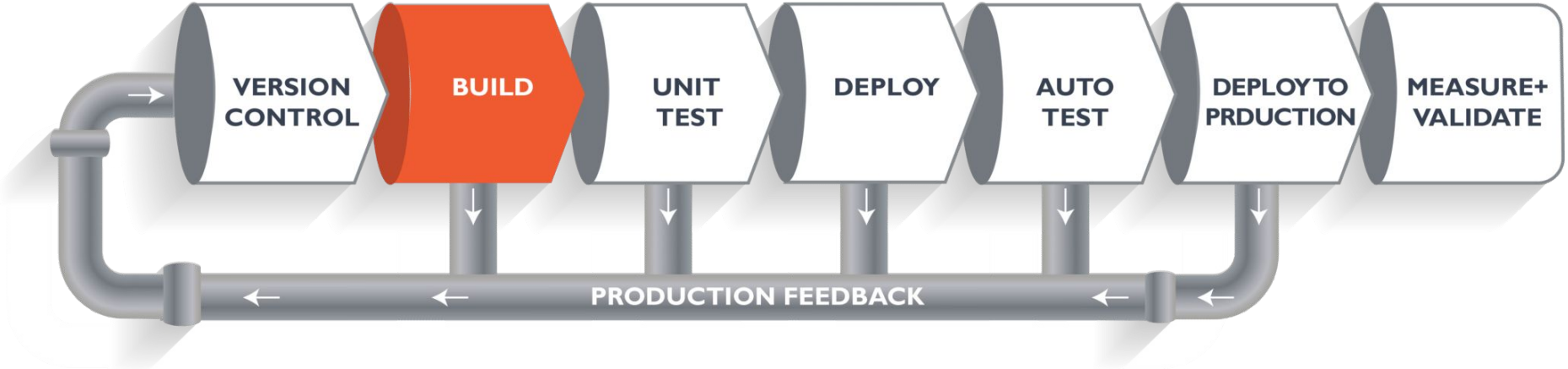
1. Local development setup for developers
   a. Which IDE?
   b. Which build tool?package manager?
2. Public/private package dependencies? Module structure?
3. Preparing local build scripts
4. CI server installation and configuration
5. Docker registry setup
6. Preparing build agents/env
   a. Docker images/AMI(s)
7. CI server build configurations
   a. Pipeline(s)
   b. Build scripts

EC2
CodeCommit,CodeBuild
ECR
ECS(Elastic Container Service)

- Preparing build instance AMI(s)
- CI server EC2 integration
- CodeBuild configuration
- ECR registry setup, CI Server integration
- ECS configuration to run build containers

Build Stage

EC2
CodeCommit,CodeBuild
ECS(Elastic Container Service)



Maven
npm
nuget
BASH
THE BOURNE-AGAIN SHELL
Xcode
Gradle
<APACHE ANT>
CMake

Unit Test Stage

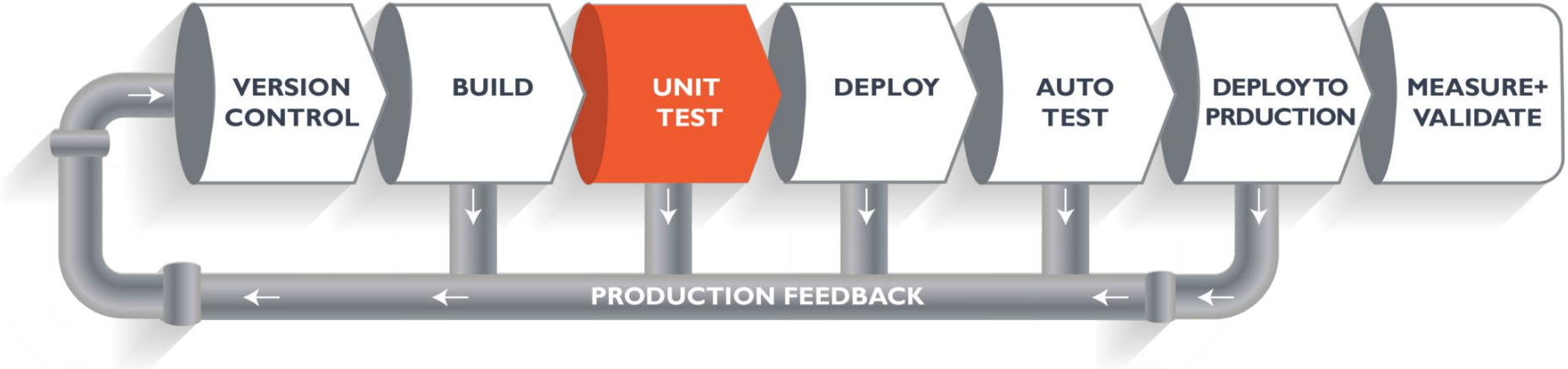
1. Choose UT framework implementation unit tests(Engineering)
2. Choose IT framework and implement Integration tests(Engineering)
3. Choose and setup static code analysis tool
4. Updating build scripts to run UT(s)
5. Updating build scripts to run Integration tests
6. Update repository configuration based on testing requirements
7. Choose code coverage tool
8. Update build configurations to generate code coverage reports
9. Define code coverage policies for code acceptance

EC2
CodeCommit,CodeBuild
ECS(Elastic Container Service)

- Update CodeBuild pipeline
- Update build scripts

Unit Test Stage

EC2
CodeCommit,CodeBuild
ECS(Elastic Container Service)



JUnit

JACOCO
Java Code Coverage

sonarqube

Deploy Stage

DEPLOYMENT PREP

1. Setup docker registry
2. Setup artifactory server(Nexus,JFrog)
3. Prepare provisioning(Cloudformation,Terraform) templates for qa,staging infrastructure
4. Docker orchestrator setup(swarm OR kubernetes) for dev,qa and staging
   a. Networking
   b. Storage

EC2
CodeCommit,CodeBuild, CodeDeploy
ECS(Elastic Container Service)
ECR,S3,RDS,DynamoDB
Cloudformation,Beanstalk
ELB, AutoScaling

- Cloudformation Templates
- ECR setup
- ECS setup
- EKS configuration
- Beanstalk configuration

docker kubernetes ANSIBLE

Deploy Stage

DEPLOYMENT

1. Update CI pipeline configurations
2. Prepare Dockerfile(s)
3. Prepare Ansible scripts
4. Update build scripts to build docker images,AMI(s)
5. Prepare dev,qa and staging deployment scripts
   a. Push to registry
   b. Automated provisioning & deployment of each env

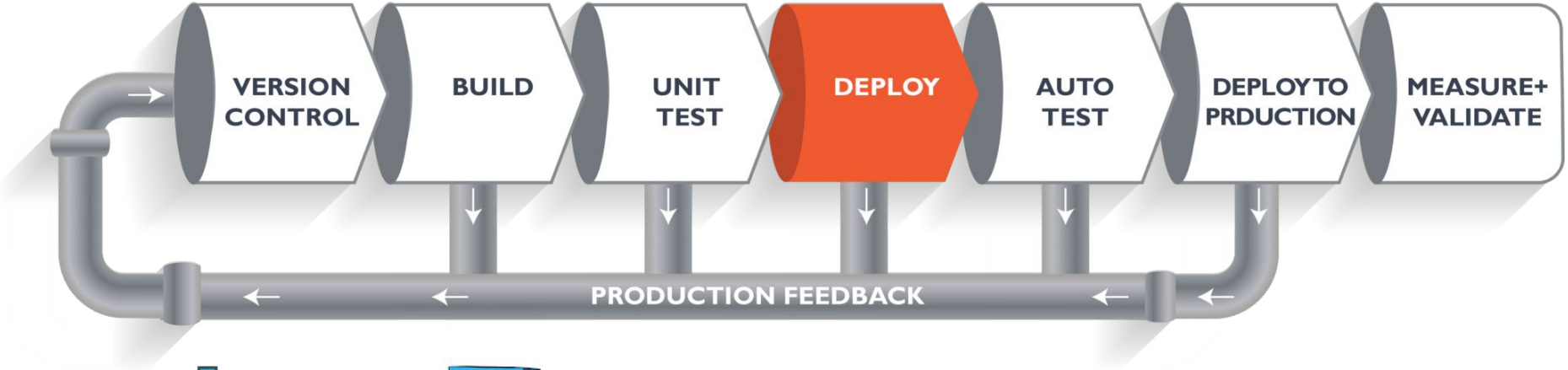
EC2
CodeCommit,CodeBuild, CodeDeploy
ECS(Elastic Container Service)
ECR,S3,RDS,DynamoDB
Cloudformation,Beanstalk
ELB, AutoScaling

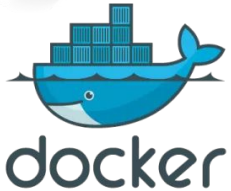
- Update CodeDeploy
- Update ECR&ECS
- Configure AMI creation process
- Create/Update Cloudformation templates
- Automate Cloudformation stack create/update/delete

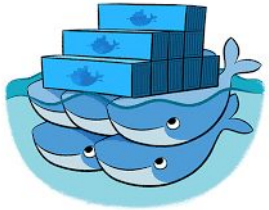
docker kubernetes ANSIBLE

Deploy Stage

EC2
CodeCommit,CodeBuild, CodeDeploy
ECS(Elastic Container Service)
ECR,S3,RDS,DynamoDB
Cloudformation,Beanstalk
ELB, AutoScaling







Auto Test Stage

1. Implementation of test suites(FT,UAT,...)(Eng)
2. Update project configuration to use related test framework
3. Create test automation run scripts
4. Update CI pipeline to add new steps
5. Update CI pipeline configurations to run test automation scripts nightly for dev branch
6. Update CI pipeline to trigger tests for each merge on release branch
7. Create/update build scripts to generate test reports

EC2
CodeCommit,CodeBuild, CodeDeploy
ECS(Elastic Container Service)
ECR,S3,RDS,DynamoDB
Cloudformation,Beanstalk
ELB, AutoScaling

- Update CodeBuild pipeline
- Create/Update Cloudformation templates

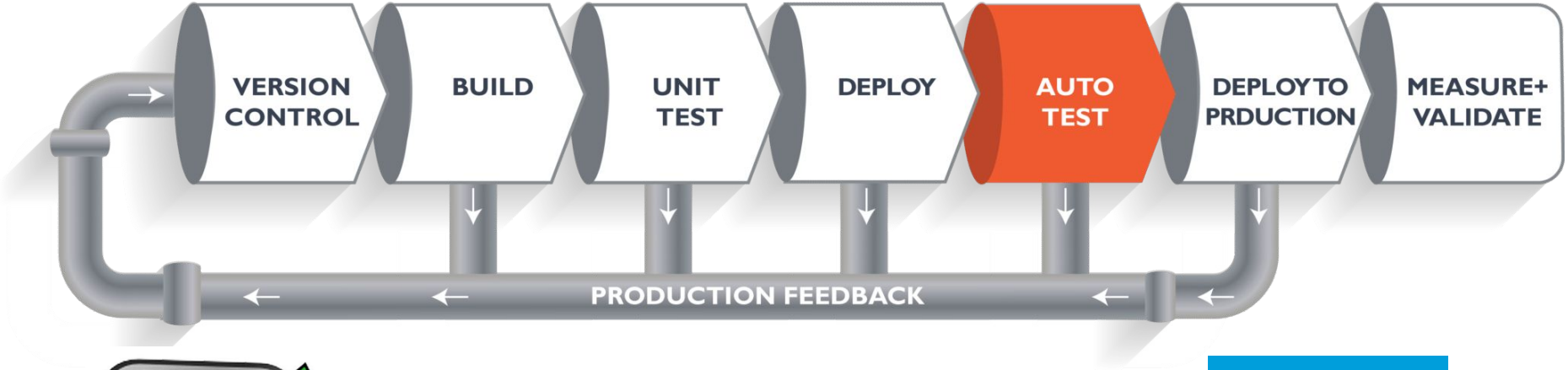
Protractor end to end testing for AngularJS

cypress.io

Visual Studio Coded UI

Auto Test Stage

EC2
CodeCommit,CodeBuild, CodeDeploy
ECS(Elastic Container Service)
ECR,S3,RDS,DynamoDB
Cloudformation,Beanstalk
ELB, AutoScaling



Protractor
end to end testing for AngularJS

cypress.io

Visual Studio
Coded UI

Deploy to Production

1. Create production env docker deployment scripts
2. Create docker templates for orchestrator
3. Update docker registry for production
4. Prepare deployment process monitoring setup
5. Update CI Pipeline for manual/automated deployment

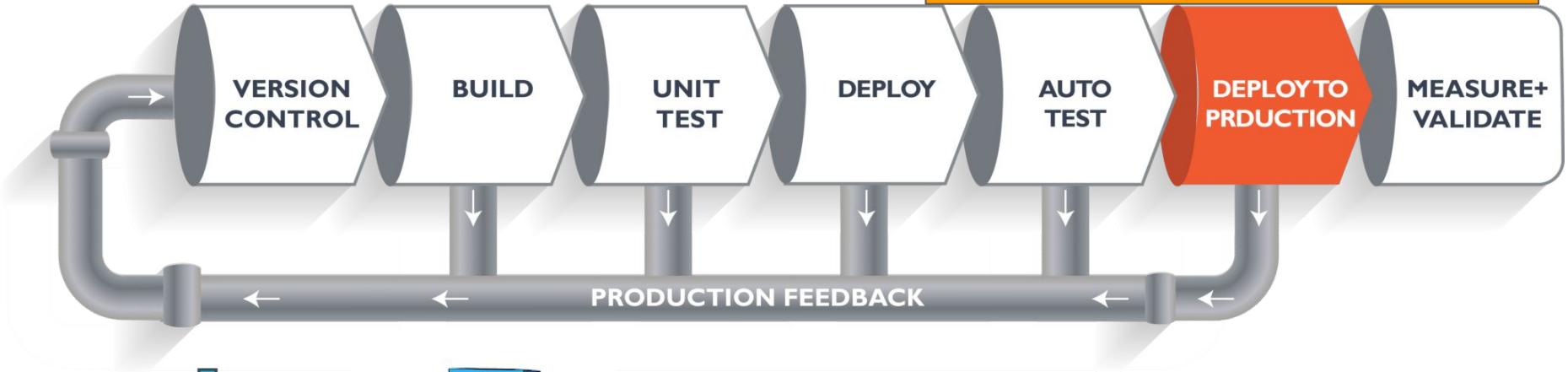
EC2
CodeCommit,CodeBuild, CodeDeploy
ECS(Elastic Container Service)
ECR,S3,RDS,DynamoDB
Cloudformation,Beanstalk
ELB, AutoScaling
Cloudfront,Route53

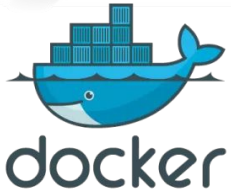
- Update CodeBuild pipeline
- Create/Update Cloudformation templates
- Create/update Beanstalk
- Create/update ECS
- Create update EKS

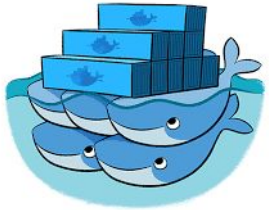
docker kubernetes ANSIBLE

Deploy to Production

EC2
CodeCommit,CodeBuild, CodeDeploy
ECS(Elastic Container Service)
ECR,S3,RDS,DynamoDB
Cloudformation,Beanstalk
ELB, AutoScaling
Cloudfront,Route53







Measure + Validate

1. Setup Prometheus to collect data from prod env
2. Setup Grafana to collect data from Prometheus and other resources
3. Automate monitoring system deployment
4. Update CI/CD pipeline configuration for monitoring
5. Automate monitoring system updates linked to automated deployment
6. Create/update alarms

Cloudwatch
SNS
SQS
SES

- Update CodeBuild pipeline
- Create/Update Cloudformation templates
- Create/update Beanstalk
- Create/update ECS
- Create update EKS

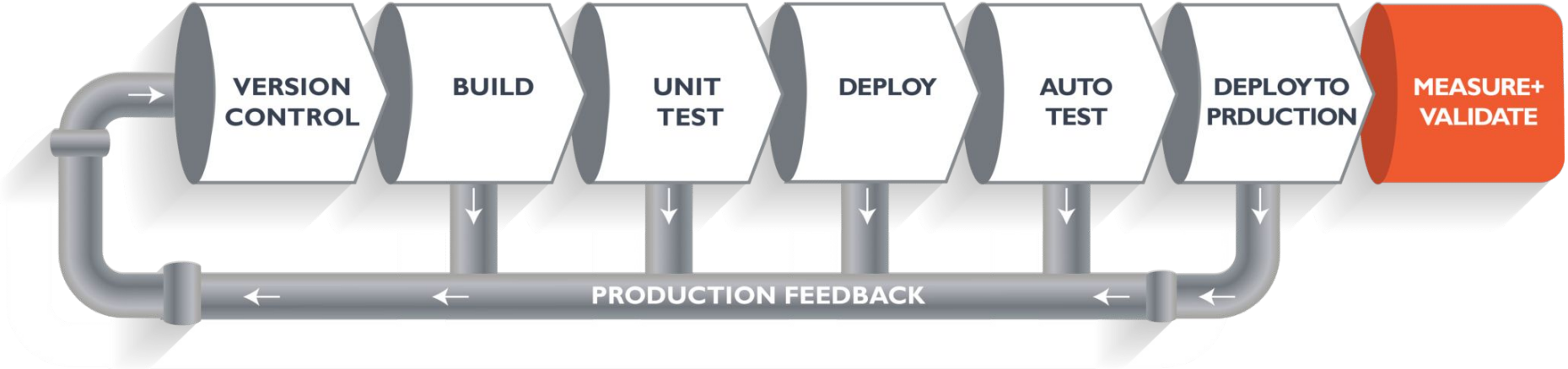
Grafana

ATLASSIAN Opsgenie

Prometheus

Measure + Validate

Cloudwatch
SNS
SQS
SES



Grafana

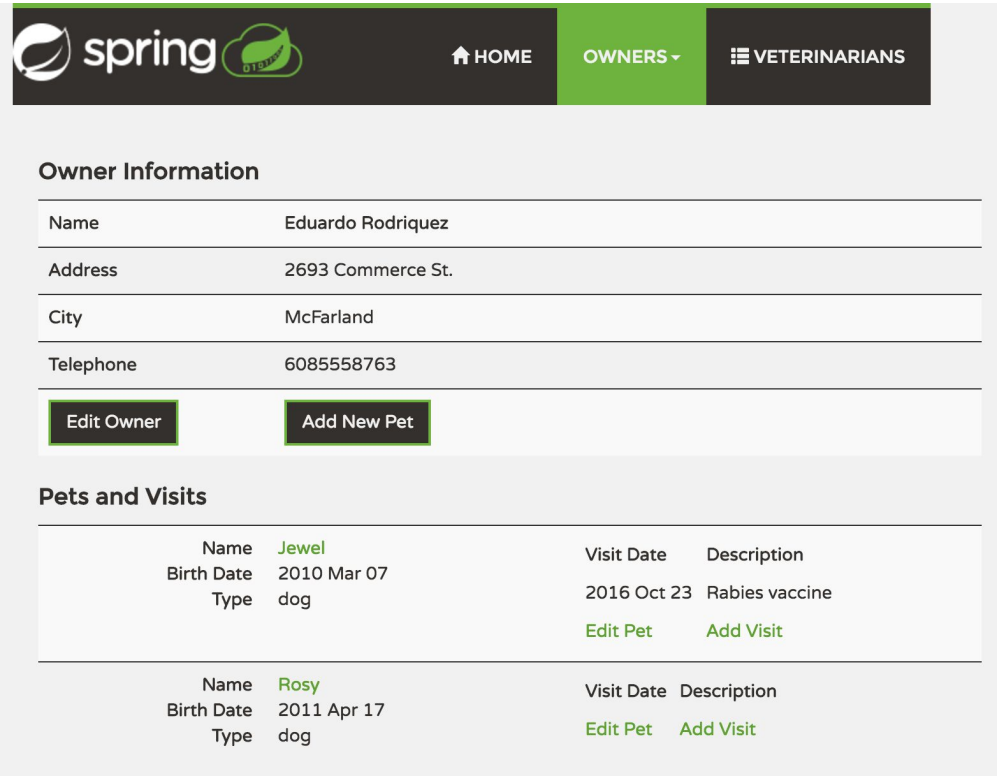
Nagios®

Prometheus

ATLASSIAN Opsgenie

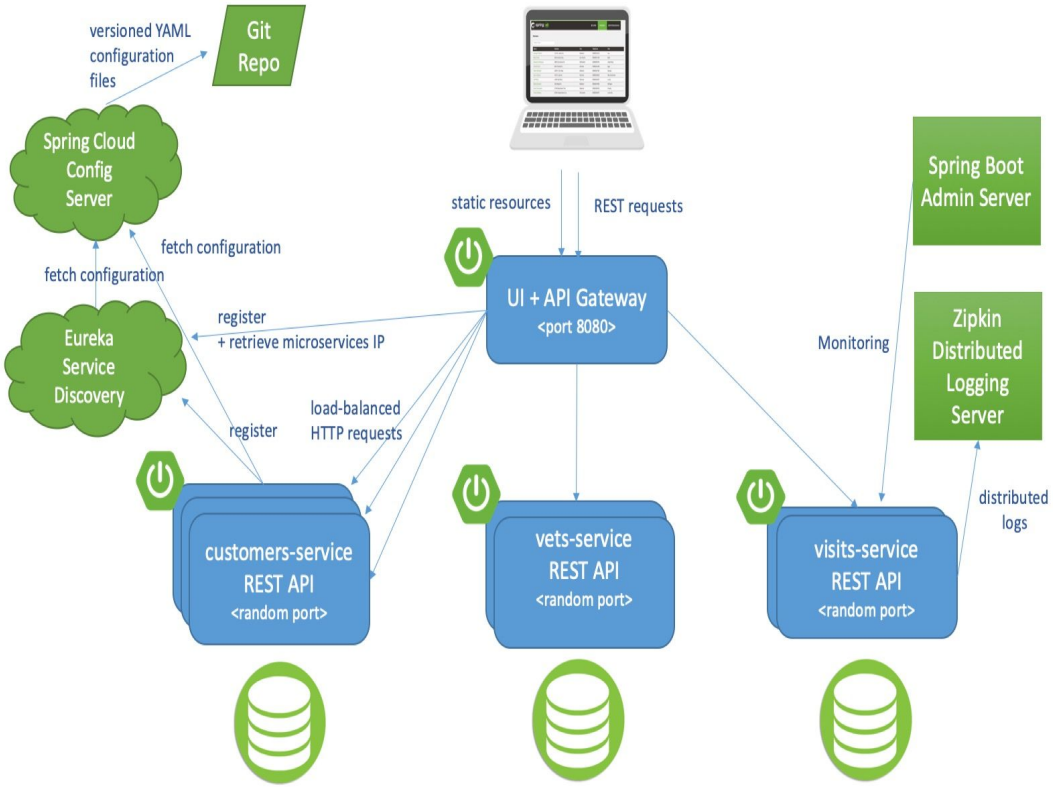
5 Microservice Petclinic

Application Demo



Let’s see the Demo

Deployment Diagram



Version Control Stage

1. Select repository host
2. Repository strategy single vs multi
3. Branching strategy
4. User roles and rights
5. User groups
6. Pull Request policy

- How to setup and manage repos on ….
- How to create branch(s) and manage rights
- What is gitflow?(dev,master,feature/XXX...)
- How to manage users,groups,roles
- PR policies?

GitHub

- Set up repository
- Create base branch(s)
  - master
  - dev
  - release

Build Stage

1. Local development setup for developers
   a. Which IDE?
   b. Which build tool?package manager?
2. Public/private package dependencies? Module structure?
3. Preparing local build scripts
4. CI server installation and configuration
5. Docker registry setup
6. Preparing build agents/env
   a. Docker images/AMI(s)
7. CI server build configurations
   a. Pipeline(s)
   b. Build scripts

Maven™

BASH THE BOURNE-AGAIN SHELL

- Configure maven
- Prepare local build scripts for developers
- Prepare Docker files
- Setup Jenkins server
- Configure Jenkins project
- Configure Jenkins pipelines
- Prepare Docker registry

Unit Test Stage

1. Choose UT framework implementation unit tests(Engineering)
2. Choose IT framework and implement Integration tests(Engineering)
3. Choose and setup static code analysis tool
4. Updating build scripts to run UT(s)
5. Updating build scripts to run Integration tests
6. Update repository configuration based on testing requirements
7. Choose code coverage tool
8. Update build configurations to generate code coverage reports
9. Define code coverage policies for code acceptance

- Update build scripts to run UT(s)
- Update build scripts to generate code coverage reports
- Jenkins pipeline configuration for UT(s)

Deploy Stage

DEPLOYMENT PREP

1. Setup docker registry
2. Setup artifactory server(Nexus,JFrog)
3. Prepare provisioning(Cloudformation,Terraform) templates for qa,staging infrastructure
4. Docker orchestrator setup(swarm OR kubernetes) for dev,qa and staging
   a. Networking
   b. Storage

- Cloudformation & Ansible for swarm cluster deployment automation
- Set up docker swarm cluster for qa-automation env
- Set up docker swarm cluster for qa env
- Docker deployment files (docker-compose.yml,...)
- Jenkins pipeline for deployment automation

docker kubernetes ANSIBLE

Auto Test Stage

1. Implementation of test suites(FT,UAT,...)(Eng)
2. Update project configuration to use related test framework
3. Create test automation run scripts
4. Update CI pipeline to add new steps
5. Update CI pipeline configurations to run test automation scripts nightly for dev branch
6. Update CI pipeline to trigger tests for each merge on release branch
7. Create/update build scripts to generate test reports

- Update CI pipeline configurations to run test automation scripts nightly for dev branch
- Update CI pipeline to trigger tests for each merge on release branch
- Create/update build scripts to generate test reports

Se Protractor end to end testing for AngularJS cypress.io Visual Studio Coded UI

Deploy to Production Stage

1. Create production env docker deployment scripts
2. Create docker templates for orchestrator
3. Update docker registry for production
4. Prepare deployment process monitoring setup
5. Update CI Pipeline for manual/automated deployment

- Automate k8 cluster deployment Cloudformation & Ansible
- Set up prod k8 cluster
- Jenkins pipeline for prod deployment

docker kubernetes ANSIBLE

Measure + Validate

1. Setup Prometheus to collect data from prod env
2. Setup Grafana to collect data from Prometheus and other resources
3. Automate monitoring system deployment
4. Update CI/CD pipeline configuration for monitoring
5. Automate monitoring system updates linked to automated deployment
6. Create/update alarms

- Automate monitoring system deployment Cloudformation & Ansible
- Setup Prometheus to collect data from prod env
- Setup Grafana to collect data from Prometheus and other resources

Grafana

ATLASSIAN Opsgenie

Prometheus

Development Diagram