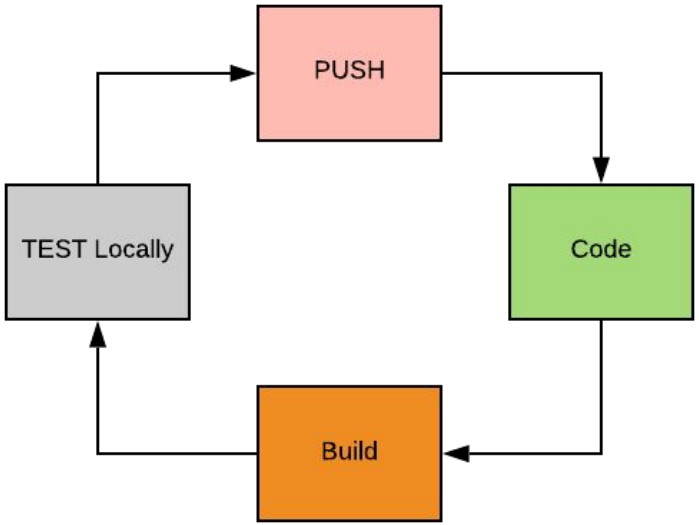
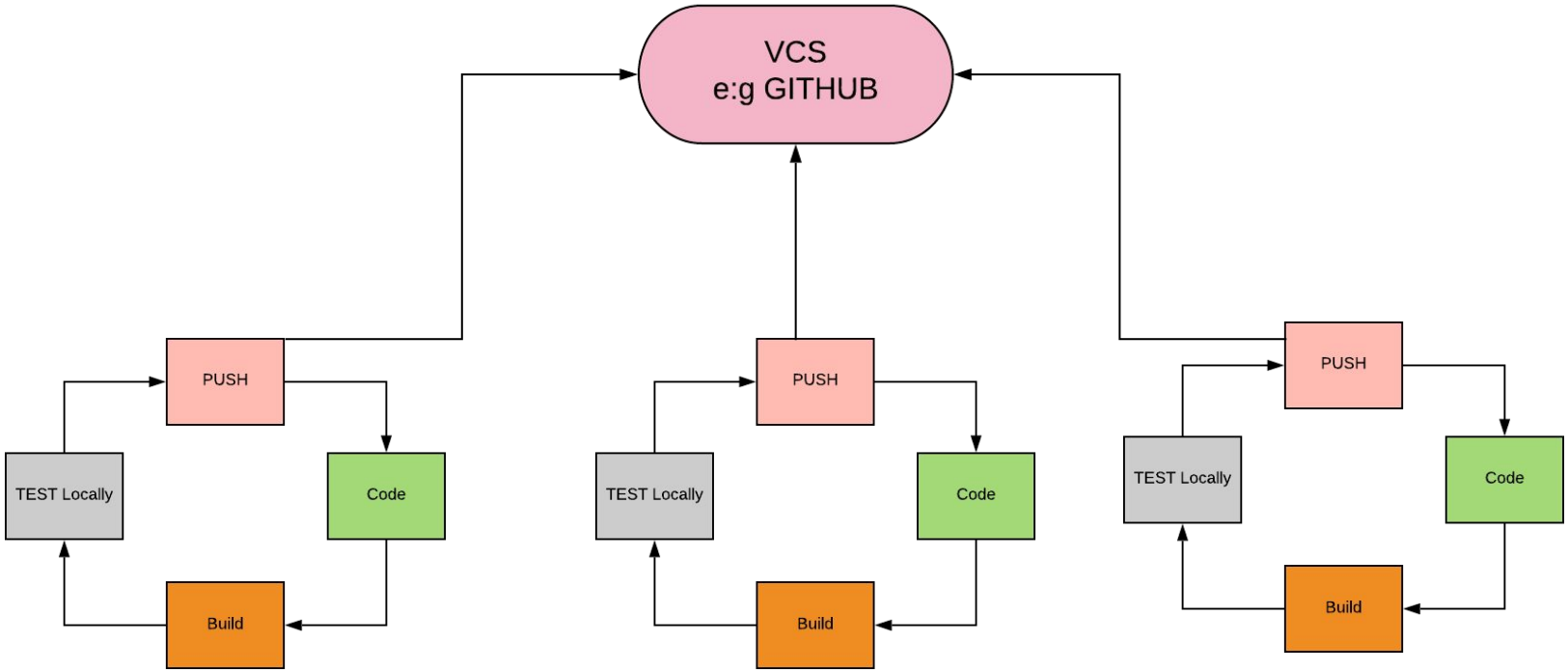
What is Continuous Integration?

Code, Build, Test & Push

Code, Build, Test & Push



Code, Build, Test & Push



PROBLEM?

Merged But **Not Integrated**

Developers keep merging code to VCS several times in a day. All the code collected from different developers would have generated conflicts and bugs.

Code Merged from a long time when built throws Conflict, Bugs and errors.
All these conflict, bugs & errors need to be resolved, which takes a very long time and halts development.

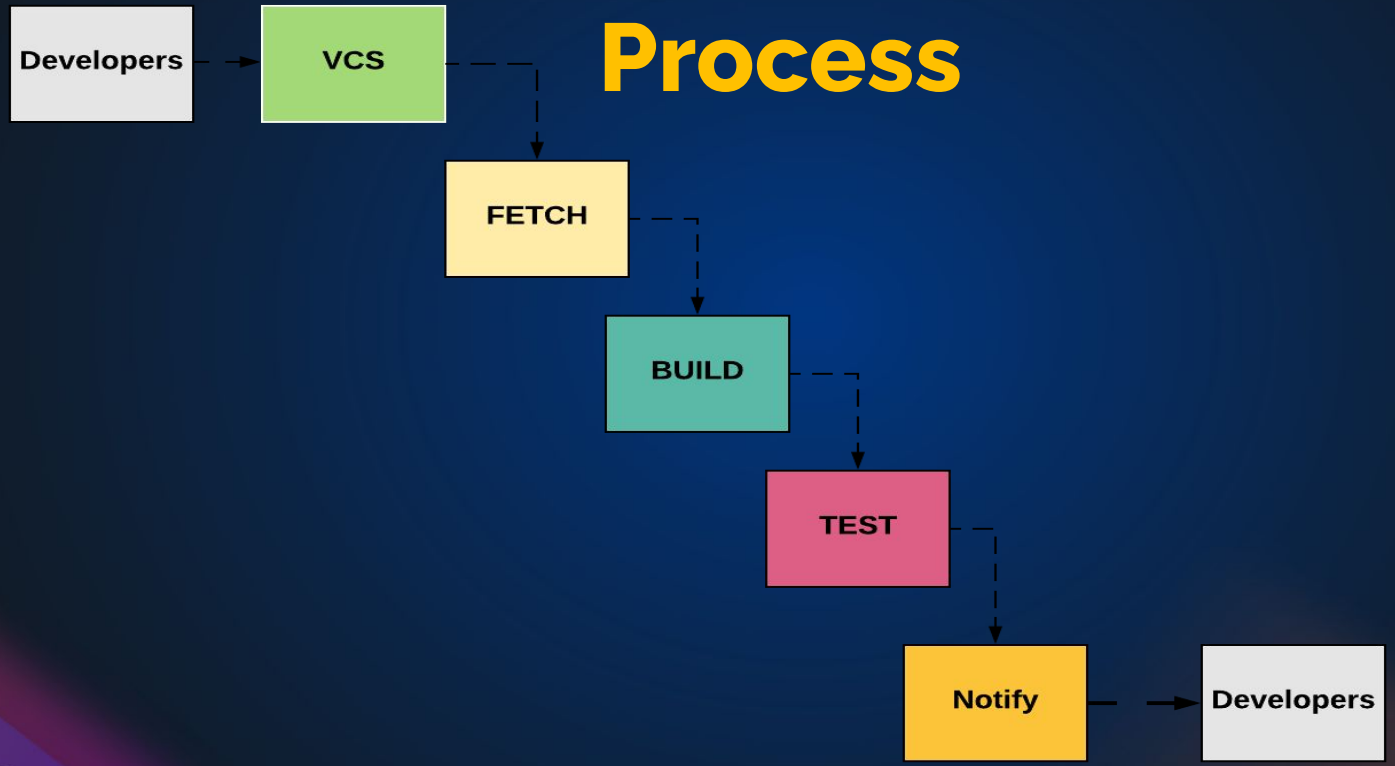
Integration is Painful

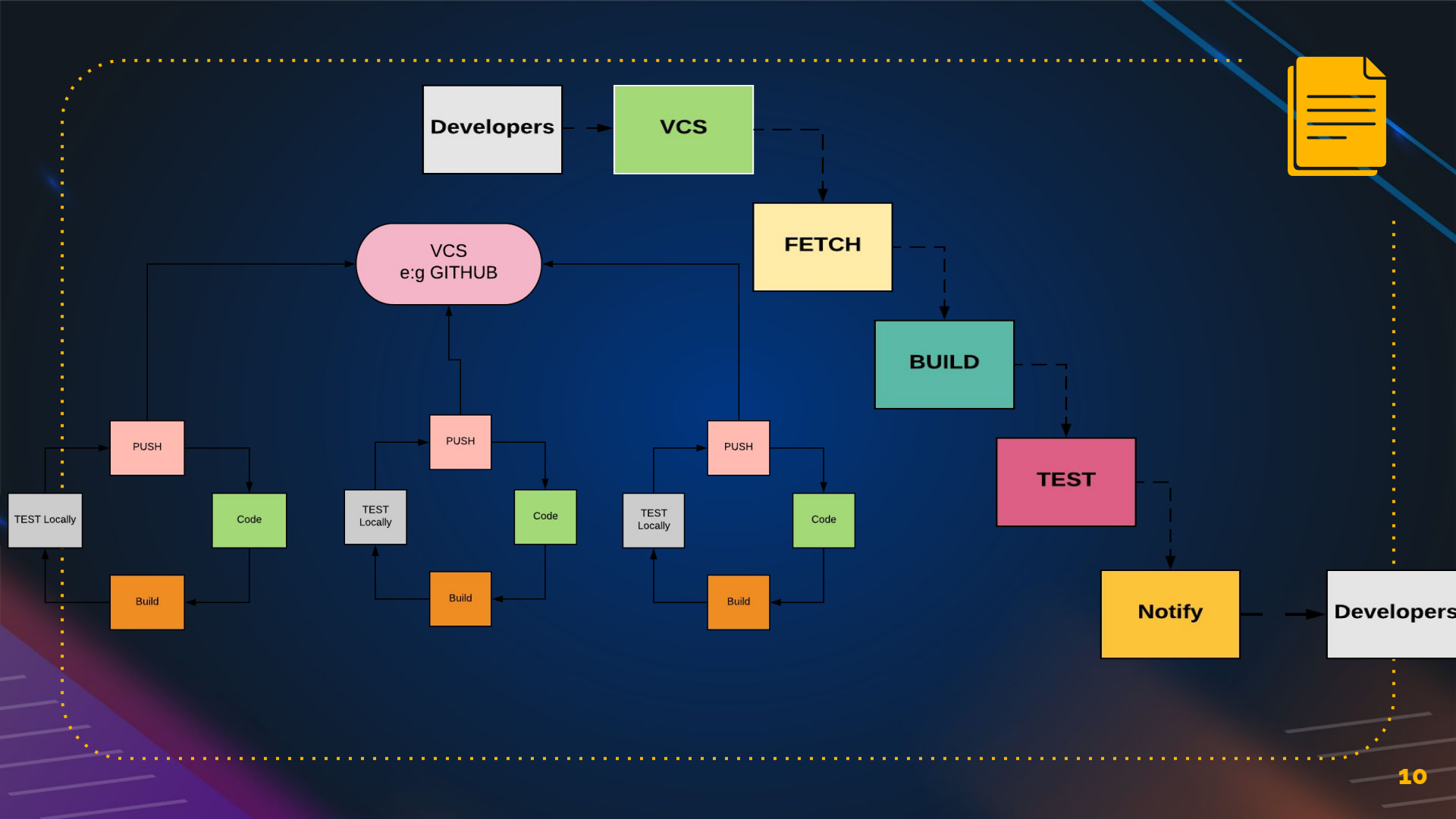
Solution

Build code from VCS after every commit.

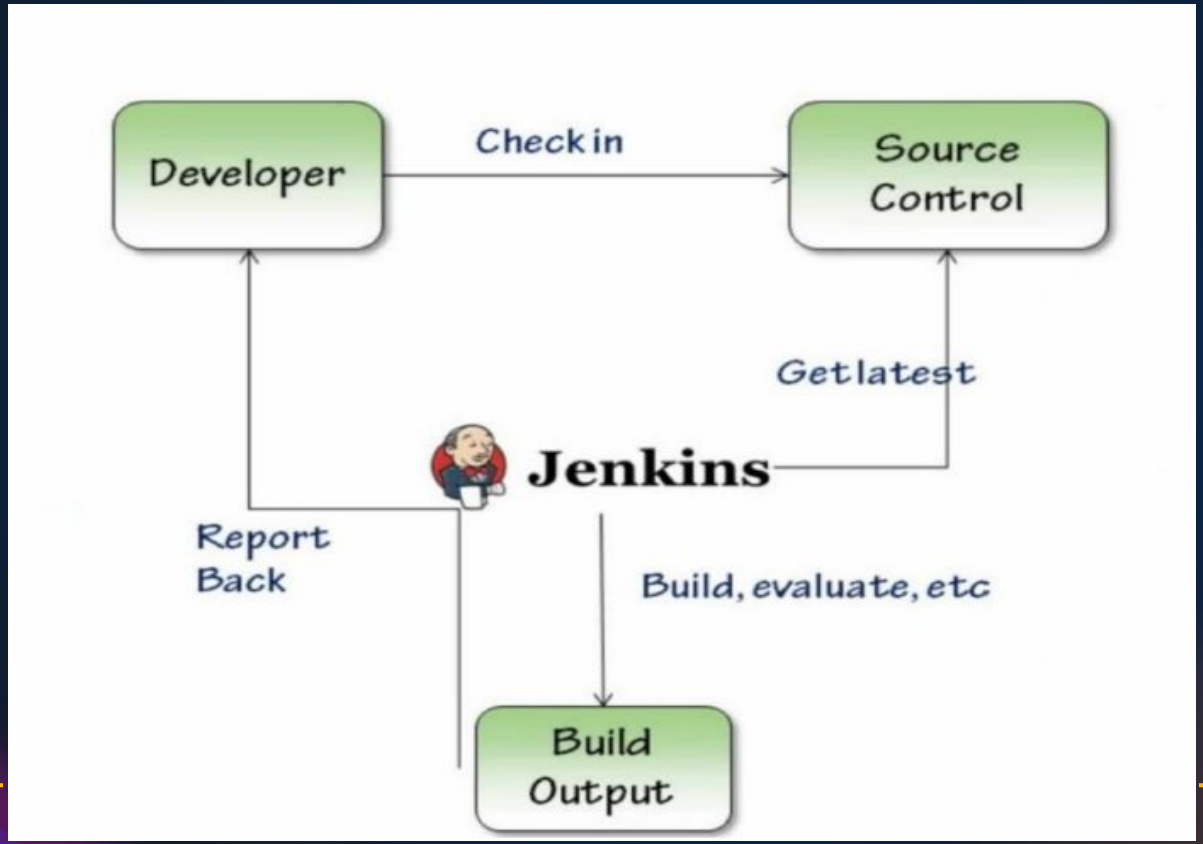
CONTINUOUS INTEGRATION.

Continuous Integration Process




Developers
VCS
FETCH
BUILD
TEST
Notify
Developers
VCS
e:g GITHUB
PUSH
TEST Locally
Code
Build
PUSH
TEST Locally
Code
Build
PUSH
TEST Locally
Code
Build

CI with Jenkins



Jenkins Features

OpenSource

Extensible

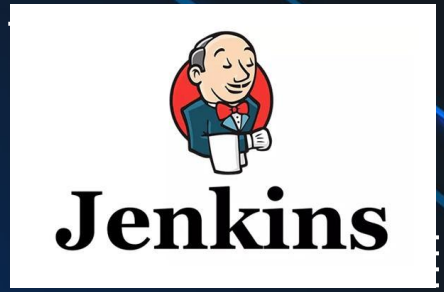
Plugins:

- VCS Plugin
- Build Plugin
- Cloud Plugin
- Testing Plugin
- Etc etc etc

Jenkins Installation

Prerequisite

1. Java- JRE, JDK
2. Any OS



Pipeline As Code

Jenkinsfile declares stages in Pipeline

Introduction

- Automate pipeline setup with Jenkinsfile
- Jenkinsfile defines Stages in CI/CD Pipeline
- Jenkinsfile is a text file with Pipeline DSL Syntax
- Similar to groovy
- Two Syntax
  - Scripted
  - Declarative

Pipeline Concepts

- ★ Pipeline
- ★ Agent
- ★ Stage
- ★ Step

```
pipeline {
    agent any
    stages {
        stage('Build') {
            steps {
                //
            }
        }
        stage('Test') {
            steps {
                //
            }
        }
        stage('Deploy') {
            steps {
                //
            }
        }
    }
}
```

```
pipeline {
    ..
    ...
    .....
}
```

```
pipeline {
    agent {
    }
    tools {
    }
    environment {
    }
    stages {
    }
}
```

```
pipeline {

    agent {
      label "master"
    }


    tools {
      maven "Maven"
    }


}
```

```
pipeline {

    environment {
     NEXUS_VERSION = "nexus3"
     NEXUS_PROTOCOL = "http"
     NEXUS_URL = "you-ip-addr-here:8081"
     NEXUS_REPOSITORY = "maven-nexus-repo"
     NEXUS_CREDENTIAL_ID = "nexus-user-credentials"
     ARTVERSION = "${env.BUILD_ID}"
     TIME = "${BUILD_TIMESTAMP}"
    }



}
```

```
pipeline {

    stages {
        stage("Clone code from VCS") {

        }

        stage("Maven Build") {

        }

        stage("Publish to Nexus Repository Manager") {

        }
    }

}
```

```
pipeline {

  stages {
     stage("Clone code from VCS") {
        steps {

        }
        post {

        }
     }


}
```

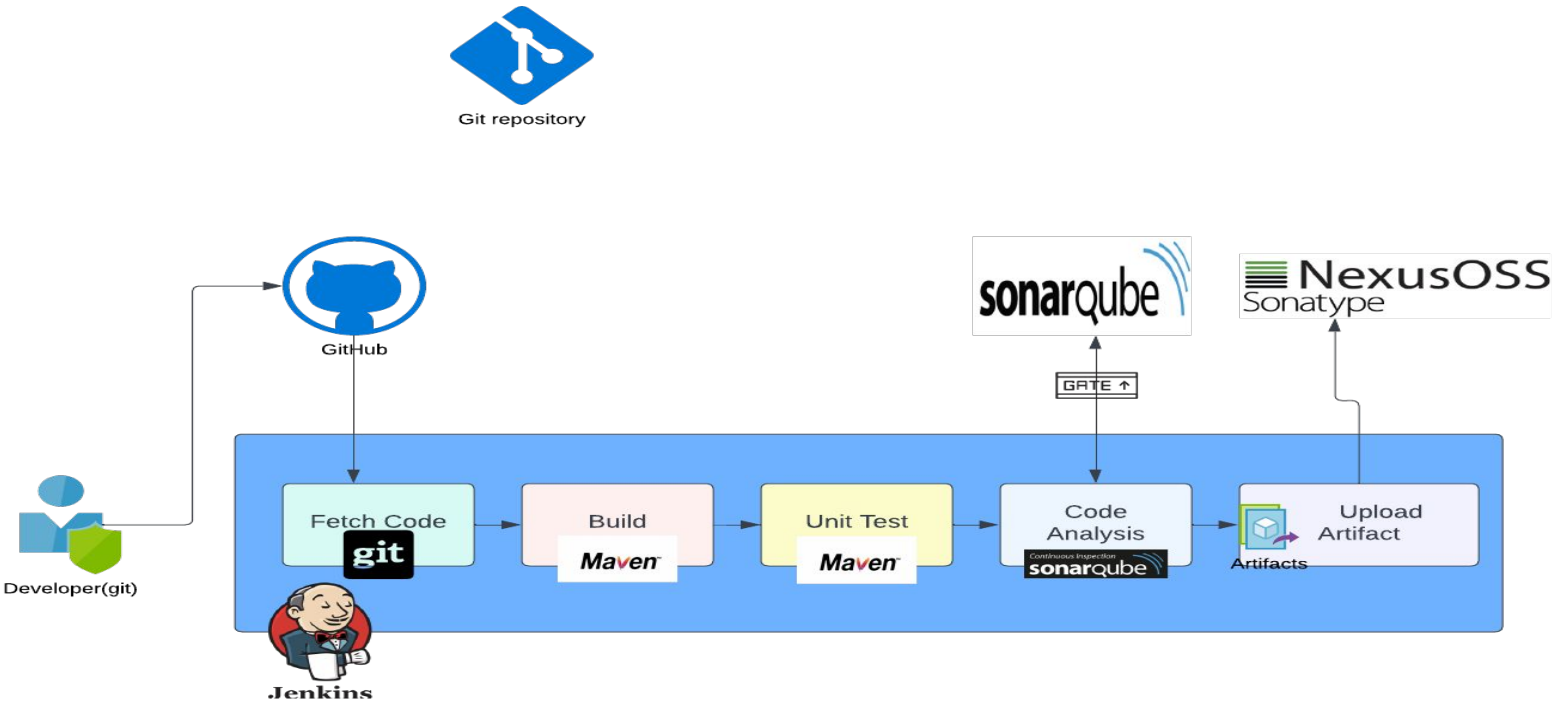
```
pipeline {

  stage('BuildAndTest'){
        steps {
           sh 'mvn install'
        }
        post {
           success {
              echo 'Now Archiving...'
              archiveArtifacts artifacts: '**/target/*.war'
           }
        }
     }



}
```

Everything Needs Automation

PIPELINE AS A CODE

Code Analysis

Detects Vulnerability and functional errors


Git repository
GitHub
sonarqube
NexusOSS
Sonatype
GATE
Developer(git)
Fetch Code
git
Build
Maven
Unit Test
Maven
Code Analysis
sonarqube
Upload Artifact
Artifacts
Jenkins

Why Code Analysis?

- Best Practices
- Vulnerabilities in code
- Functional Errors before deployment

Variety of test performed on the CODE.

Tools

Checkstyle

Cobertura

mstest

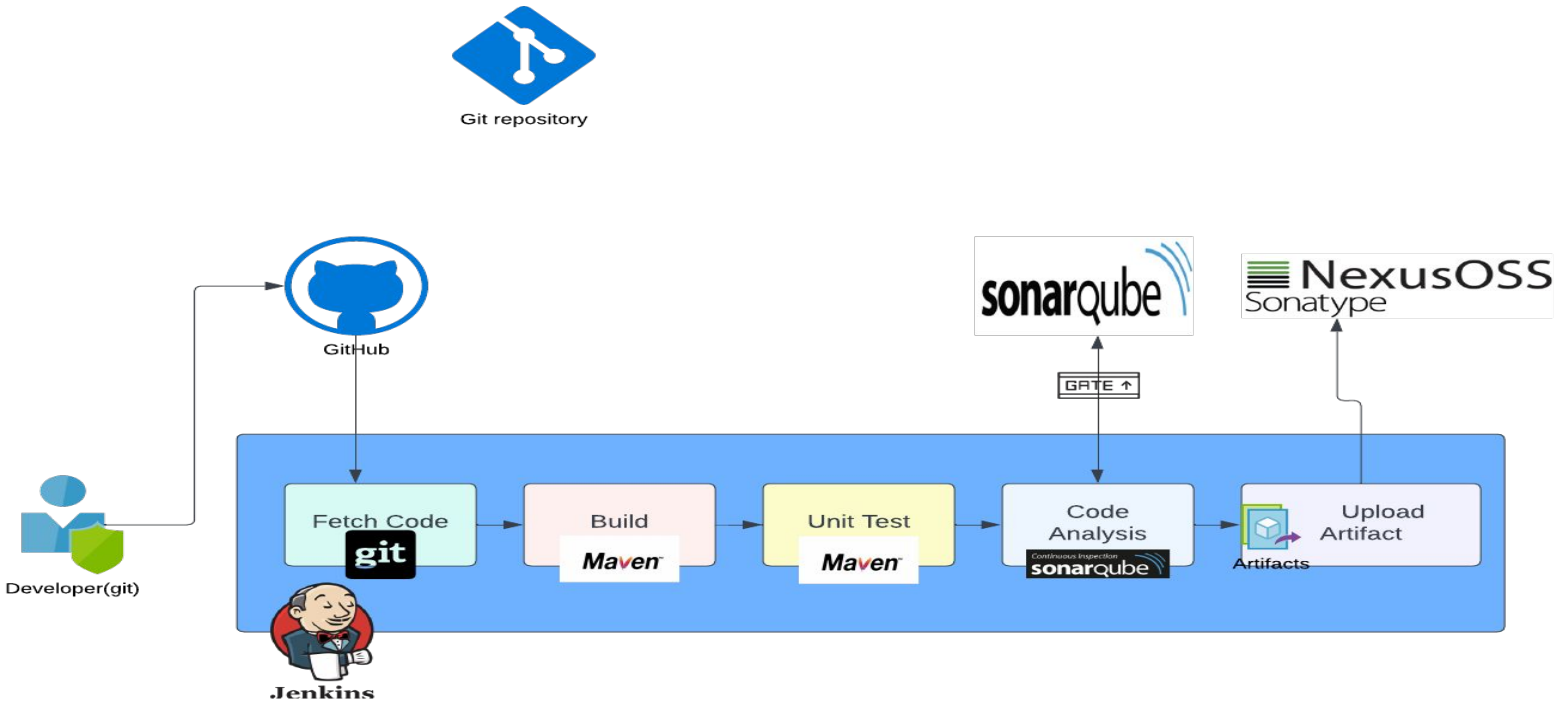
owasp

SonarQube Scanner

Etc etc etc ….

Software Repositories

Storage location for software packages


Git repository
GitHub
sonarqube
NexusOSS
Sonatype
GATE
Developer(git)
Fetch Code
git
Build
Maven
Unit Test
Maven
Code Analysis
sonarqube
Upload Artifact
Artifacts
Jenkins

Your own Repo for Softwares/Packages

Maven

Maven dependencies

apt

Packages for debian based systems.

yum

Packages for RedHat based systems.

nuget

package manager for .NET.

Npm

package manager for JavaScript

Docker

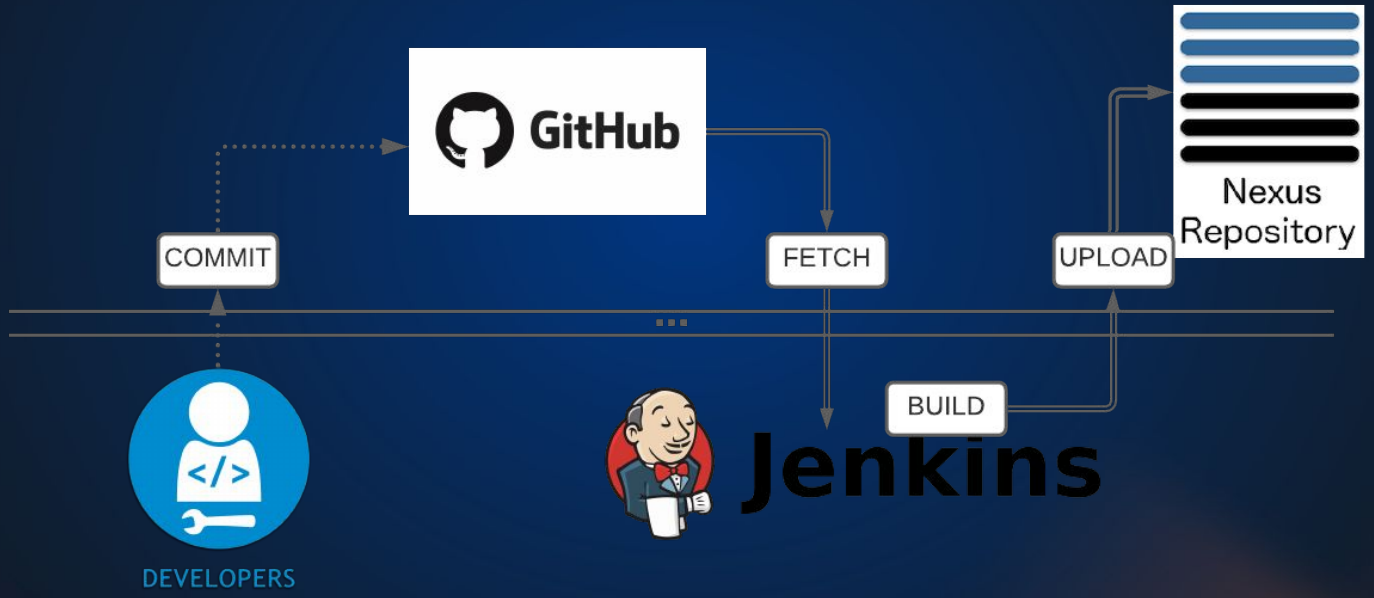
Registry to store Docker Images

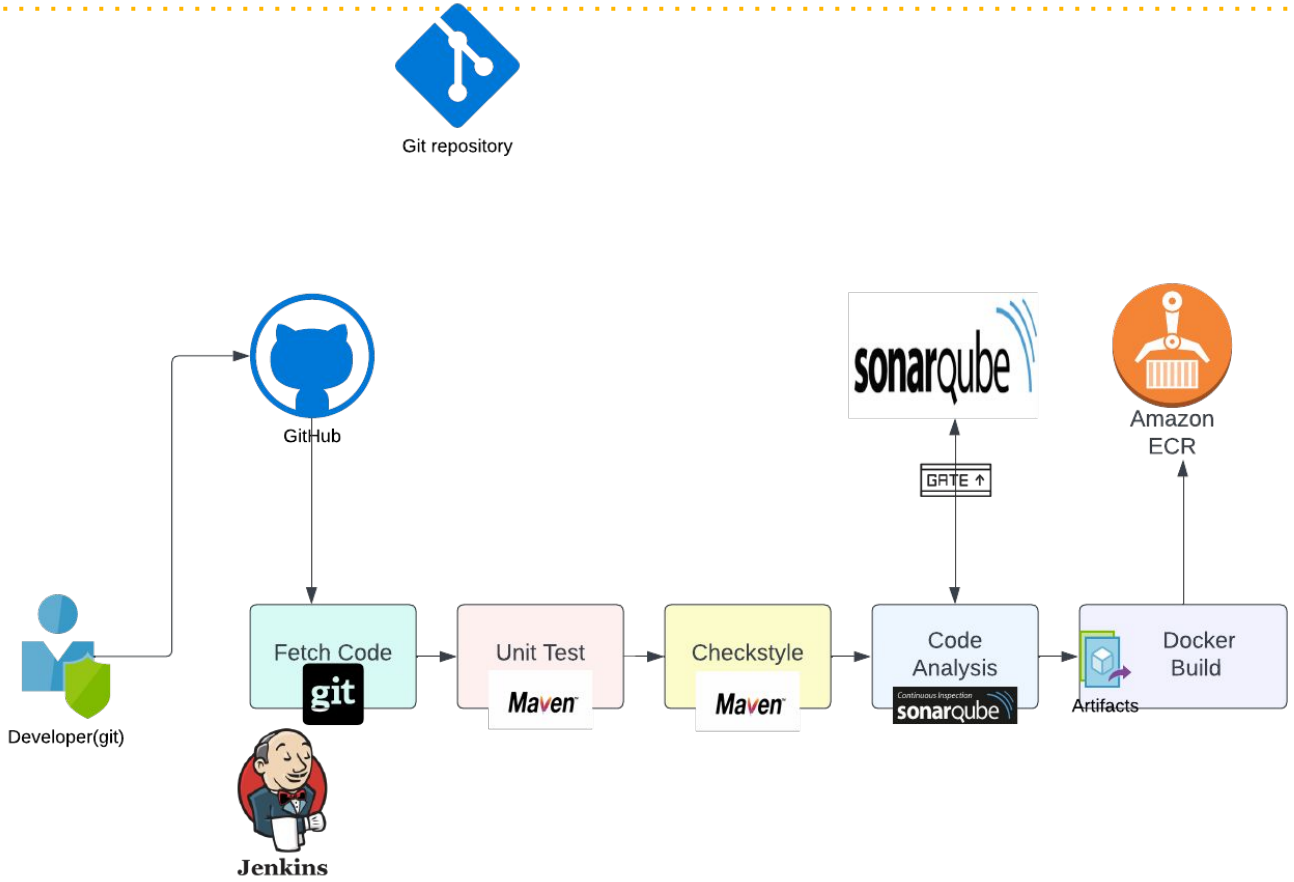
Nexus Software Repository Manager.

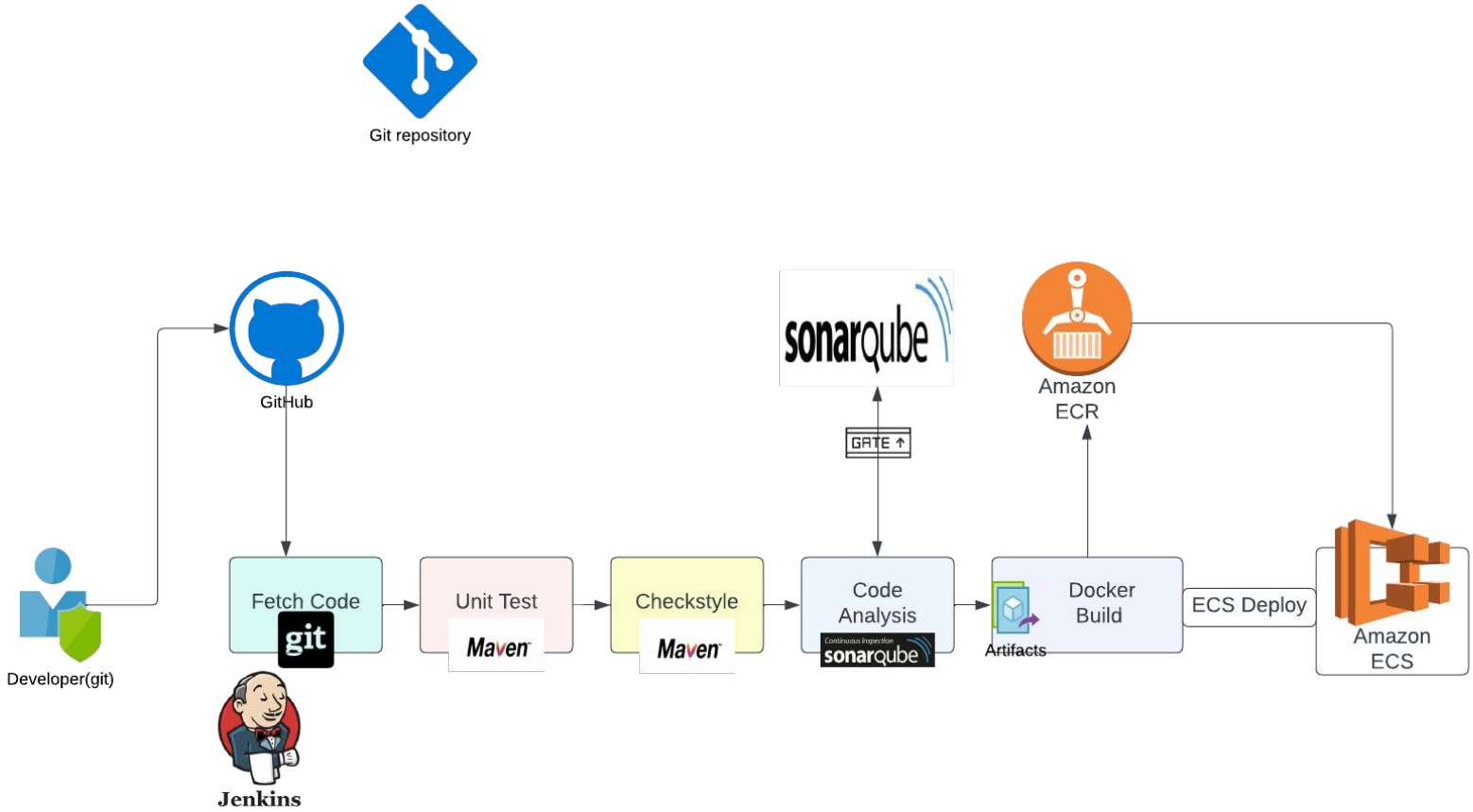
Key Points

- Runs on java
- Used to store artifacts
- Used as a Package manager for dependencies
- Opensource & Enterprise Versions
- Supports Variety of repo like maven,apt,docker, Ruby gems etc etc

Jenkins Integration with Nexus




Git repository
GitHub
Developer(git)
Fetch Code
git
Unit Test
Maven
Checkstyle
Maven
Code Analysis
sonarqube
GATE
sonarqube
Amazon ECR
Docker Build
Artifacts
Jenkins


Git repository
GitHub
sonarqube
Amazon
ECR
GATE
Developer(git)
Fetch Code
git
Unit Test
Maven
Checkstyle
Maven
Code
Analysis
sonarqube
Artifacts
Docker
Build
ECS Deploy
Amazon
ECS
Jenkins

Jenkins Master/Slave

Distributed Builds, Cross Platform builds and much more

Use Cases

Load Distribution

Jenkins Master Executes Build Job on Node it selected.

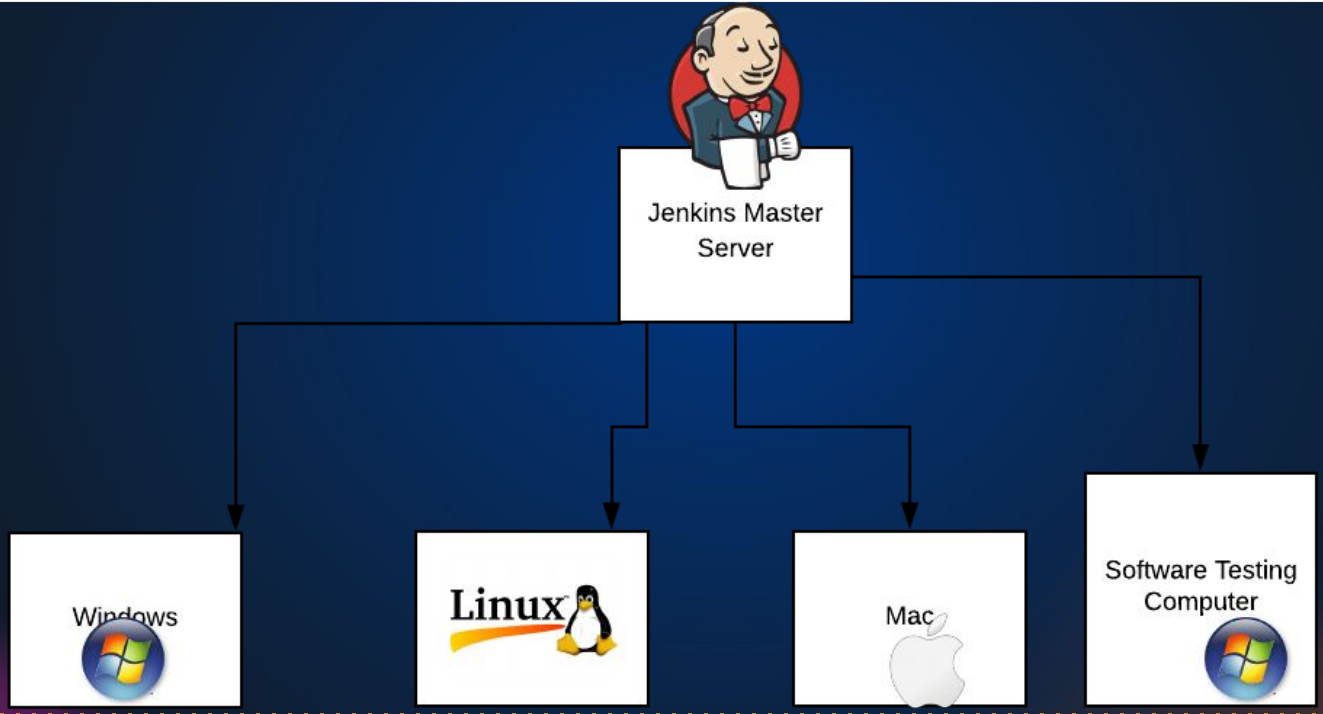
Cross Platform Builds

Executing Build of other platforms like .net(Windows), IOS(Mac OS)
from Jenkins Master(Linux)

Software Testing

Execute Testers Test Automation Scripts from Node.

Jenkins Master/Slave



Execute Anything

Can add any computer in network as **Jenkins Node** and execute commands on Nodes.

e:g scripts, commands, test scripts, etc

Prerequisites for Node Setup

1. Any OS
2. Network access from Master
   Note: Check Firewall rules
3. Java, JRE, JDK
4. User
5. Directory with User ownership
6. Tools as required by the Jenkins job
   e:g Maven, Ant, Git etc

Jenkins Security

user, permissions, roles, jobs permissions

Securing Jenkins

User Login

Jenkins own database.

Sign Up

LDAP Integration

Permissions on Jenkins

- Admin
- Read
- Jobs
- Credentials
- Plugins etc

Permissions on Jobs

- View
- Build
- Delete
- Configure
- etc

Jenkins Tools

Install & Integrate various tools from Jenkins

Jenkins integration with tools

Plugin

Install plugin related to the tool like Git, Ant, Docker, Maven etc

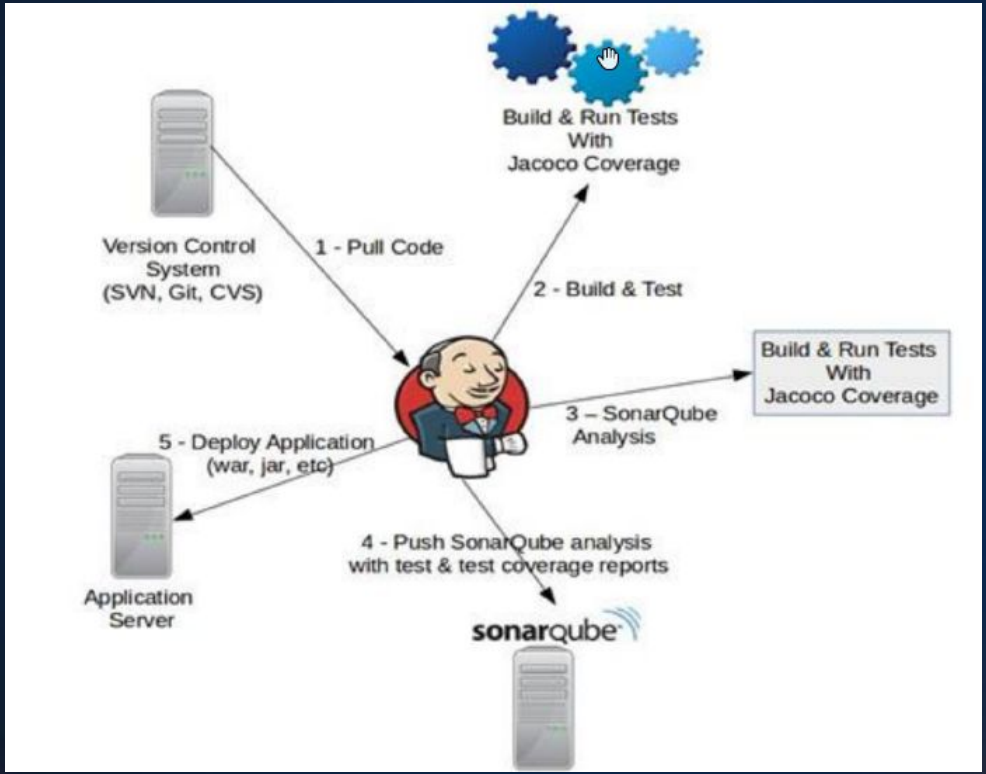
Install Tool in Jenkins for the plugin

Git, Ant, Docker, Maven etc

Sonarqube Integration

Code Analysis with SonarQube

Sonarqube Integration



Sonarqube Installation

Prerequisite

1. Any OS
2. JDK
3. Database
   like MySQL

Build Triggers

Run Jenkins Job Automatically

Build Job Remotely

- Generate Job URL
- Generate API Token
- Generate CRUMB
- Use URL,API Token &CRUMB in Curl URL to execute Job Remotly

Build After a Job

- Set as downstream job

Build Periodically

- Set Cronjob format schedule
- Job can get execute at a specified time
- OR Periodically execution like every 15 mins
  Cron Format
  Min Hour DOM Month DOW

  Min = 0-59, hour = 0 - 23, DOM = 1-31, Month = 1-12, DOW = 0-7

Poll SCM

- Cronjob to POLL SCM like Github
- Execute Job when new commit found.

Pipeline As Code

Jenkinsfile declares stages in Pipeline

Benefits

- Version control your Pipeline
- Use Groovy syntax for setting up logic

Sample

Jenkinsfile (Declarative Pipeline)

```
pipeline {
    agent any

    stages {
        stage('Build') {
            steps {
                echo 'Building..'
            }
        }
        stage('Test') {
            steps {
                echo 'Testing..'
            }
        }
        stage('Deploy') {
            steps {
                echo 'Deploying....'
            }
        }
    }
}
```