Jenkins Continuous Build System

Amir Dirin

Fall2024

Why Jenkins

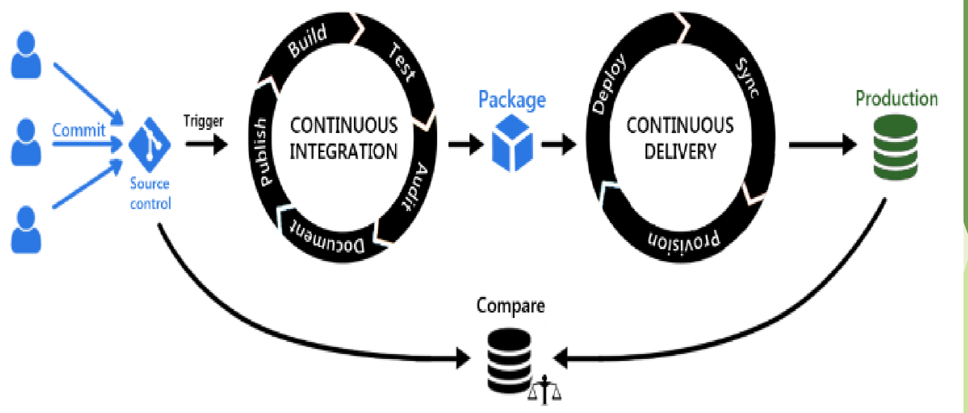
- Continuous integration systems are a vital part of any Agile team because they help enforce the ideals of Agile development
- Jenkins, a continuous build tool, enables teams to focus on their work by automating the build, artifact management, and deployment processes
- Jenkins’ core functionality and flexibility allow it to fit in a variety of environments and can help streamline the development process for all stakeholders involved

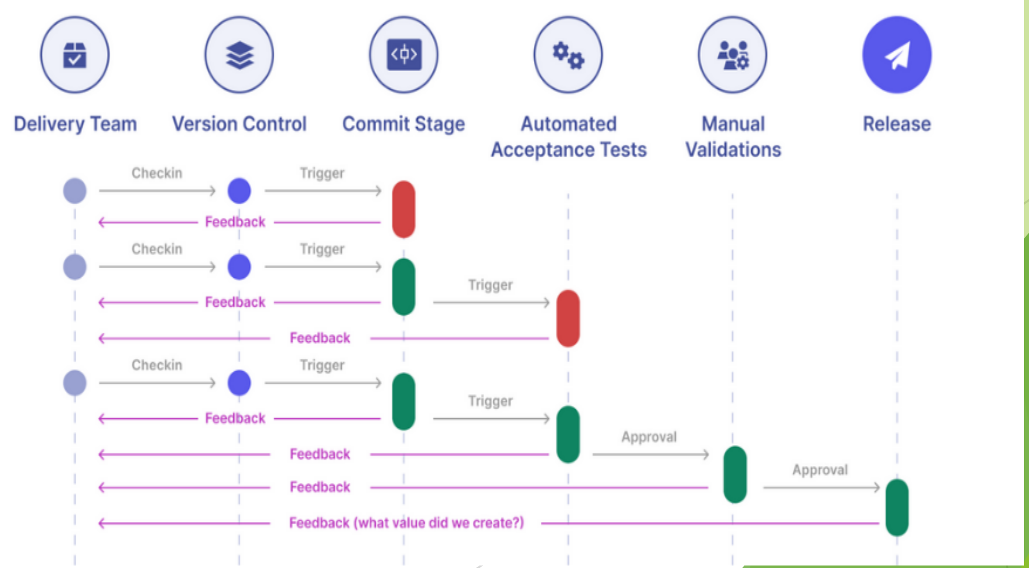
Continuous Integration (CI)

- "Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily - leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible" – Martin Fowler
- At every commit, the system is:
  - Integrated
  - Built
  - Tested
  - Archived
  - Deployed

CI-Workflow

- Deployment of packages goes through continuous delivery workflow …
- CI helps ensure that software components work together.
  - Developers build, run, and test code on their workstations before committing code to the version control repository.
  - After changes are made to the repo a chain of events is put into motion.
    1. Build the latest version of the source code
    2. Then unit tests are executed
    3. Then build is deployed to test environments (usually automated tests)
    4. The team is notified about the status
    5. Report on the delivery such as build number, defects, and the number of tests.





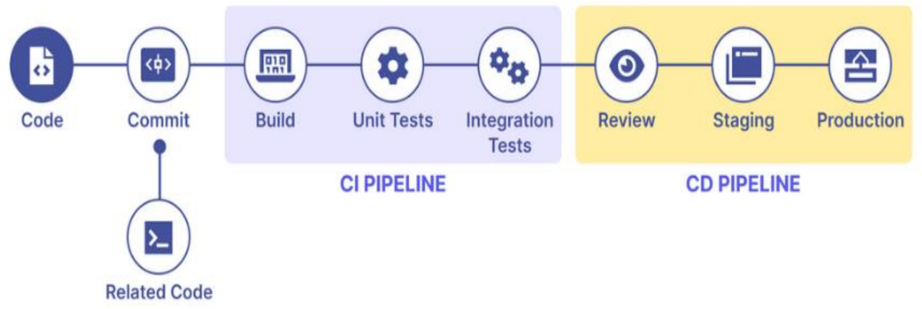
A typical CI pipeline

- Detect changes in the source code repository
- Source code quality analysis
- Build
- Execute all unit tests
- Execute all integration tests
- Generate deployable artifacts
- Report status

If one of the steps above fails:

- Integration may stop or continue depending on defect severity and configuration
- Results are notified to the team via email or chat system
- The team fixes defects and commits again
- Tasks are performed again

CI/CD workflow pipeline



https://katalon.com/resources-center/blog/ci-cd-introduction

Why Jenkins?

- Jenkins is a highly configurable system by itself ▫
- The additional community-developed plugins provide even more flexibility ▫
- By combining Jenkins with Ant, Gradle, or other Build Automation tools, the possibilities are limitless.
- Released under the MIT license

Jenkins Features:

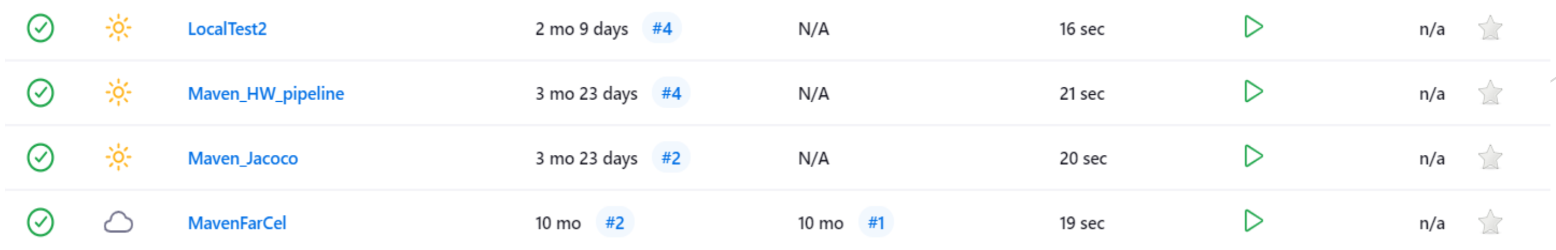
- Generate Test Reports
- Integrate with many different VCS
- Push to various artifact repositories
- Deploys directly to production or test environments
- Notify stakeholders of the build status
- --- and much more

Jenkins Setup

- When setting up a project in Jenkins, out of the box you have the following general options:
  - Associating with a version control server
  - Triggering builds
    - Polling, Periodic, Building based on other projects
  - Execution of shell scripts, bash scripts, Ant targets, and Maven targets
  - Artifact archival
  - Publish JUnit test results and Javadocs
  - Email notifications
- As stated earlier, plugins expand the functionality even further

Jenkins: Reporting

- Jenkins comes with basic reporting features
  - Keeping track of build status
    - Last success and failure
    - "Weather" – Build trend
- These can be greatly enhanced with the use of pre-build plugins
  - Unit test coverage
  - Test result trending
  - Findbugs, Checkstyle, PMD



✓	☼	LocalTest2	2 mo 9 days #4	N/A	16 sec	▷	n/a	☆
✓	☼	Maven_HW_pipeline	3 mo 23 days #4	N/A	21 sec	▷	n/a	☆
✓	☼	Maven_Jacoco	3 mo 23 days #2	N/A	20 sec	▷	n/a	☆
✓	☁	MavenFarCel	10 mo #2	10 mo #1	19 sec	▷	n/a	☆

Tying it into Agile

- For an Agile team, Jenkins provides everything needed for a robust continuous build system
- Jenkins supports Agile principles by constantly providing access to working copies of software
- Jenkins' extensibility allows the system to adapt to many different pre-existing environments

Practical example

- Demo