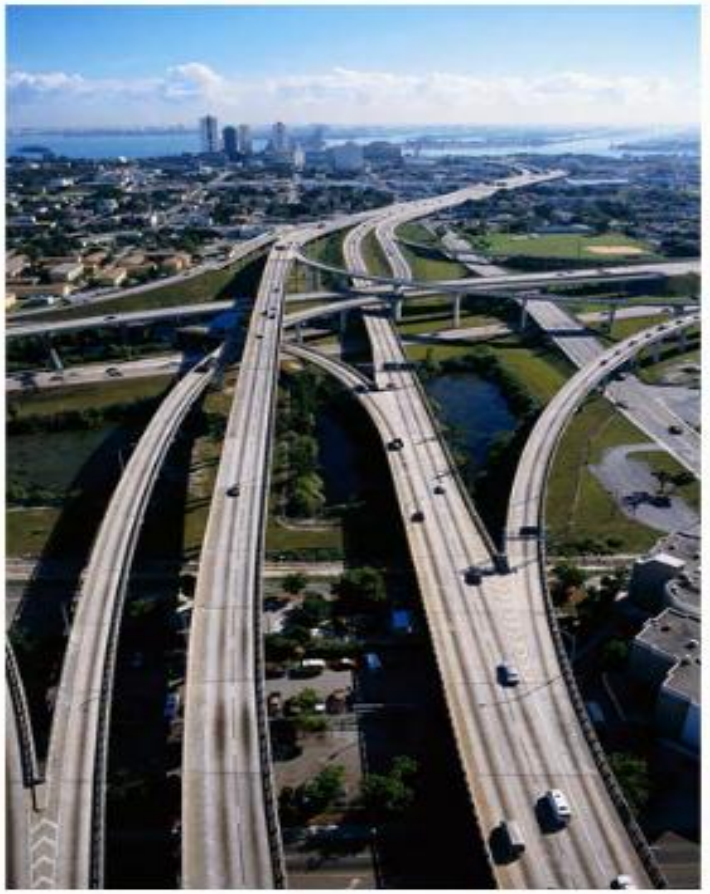
Capgemini

Jenkins Overview

Lesson Objectives

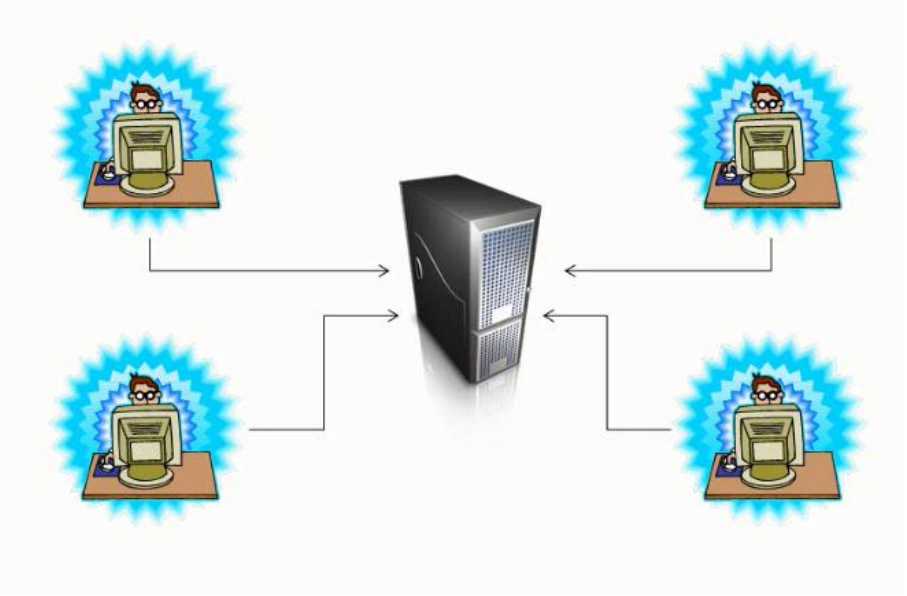
- **Module 1**
  - Understanding Jenkins
- **Modules 2-3**
  - Getting up and running
- **Module 4**
  - Extensibility (Plugins)
- **Module 5**
  - The big picture

Jenkins



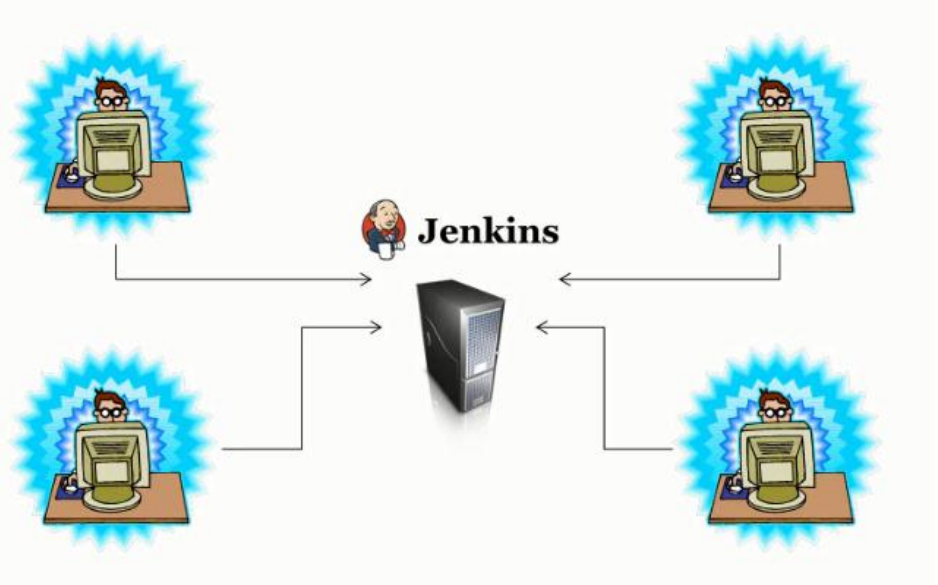
Introduction to CI

- Continuous Integration involves a tool that monitors your version control system for any changes and automates application building.
- CI system must be executed under configuration management.
- Developers are notified automatically if any build action failed.
- CI brings a practice to integrate work frequently in software development.
- Monitoring of Code Quality and Code coverage metrics is automated.



Need of CI in Software development

- Helps to locate code based defects in a centralized location.
- Tools can be used to automate deployment.
- Minimizes integration errors in SVN during build process(Errors are uncovered during Manual Build) by invoking automation.
- Increase amount of quality code and improve development standards.



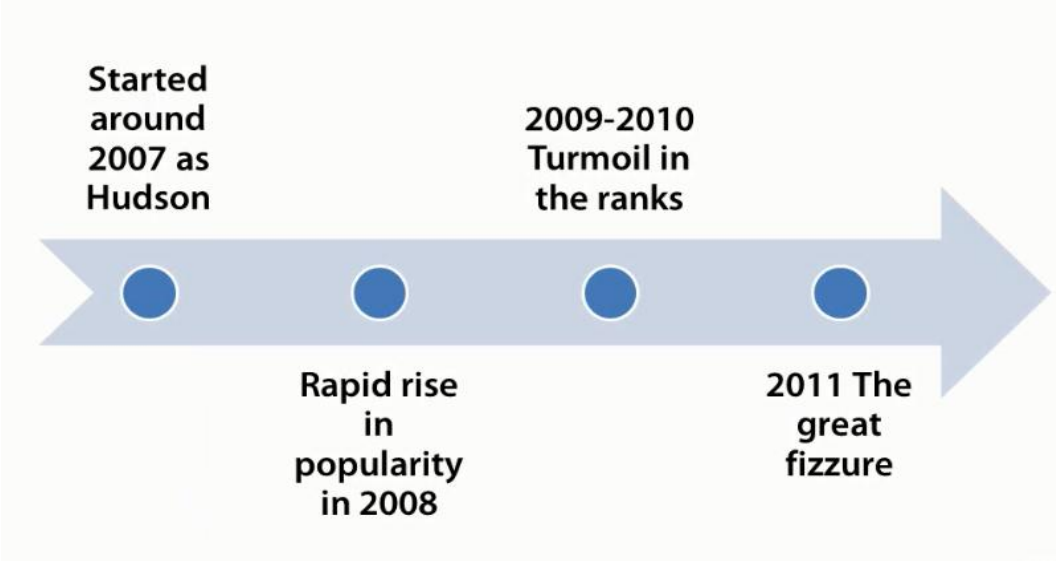
Benefits of CI

- Aims to eliminate code integration issues
- Minimizes project risk with notification of defects and code quality issues early in the process.
- Reduces cost of quality
- Early warning of conflicting changes code.
- Drives automation of build and testing of an application.

Introduction to Jenkins

Introduction to Jenkins

- Jenkins is an open source continuous integration tool written in java developed by Kohsuke Kawaguchi.
- Monitors the change in the source control systems like GIT, SVN, CVS, etc.
- Builds the application using various build tools like ANT, MAVEN, etc.
- Provides a fresh build whenever there is a change in the source control system
- Sends messages on the status of the build through Email, SMS, etc.
- Can support software releases, documentation, monitoring, and a number of use case secondary to continuous integration



Features of Jenkins

- Easy installation
- Dashboard support
- Supports a large number of build tools via plug-in
- Install plugins from UI
- Extensibility
- Self upgrade
- Real time console output
- JUnit test reporting
- Distributed builds

Continuous Integration and Continuous Delivery

As an extensible automation server, Jenkins can be used as a simple CI server or turned into the continuous delivery hub for any project.

Easy installation

Jenkins is a self-contained Java-based program, ready to run out-of-the-box, with packages for Windows, Mac OS X and other Unix-like operating systems.

Easy configuration

Jenkins can be easily set up and configured via its web interface, which includes on-the-fly error checks and built-in help.

Plugins

With hundreds of plugins in the Update Center, Jenkins integrates with practically every tool in the continuous integration and continuous delivery toolchain.

Extensible

Jenkins can be extended via its plugin architecture, providing nearly infinite possibilities for what Jenkins can do.

Distributed

Jenkins can easily distribute work across multiple machines, helping drive builds, tests and deployments across multiple platforms faster.

Jenkins Installation

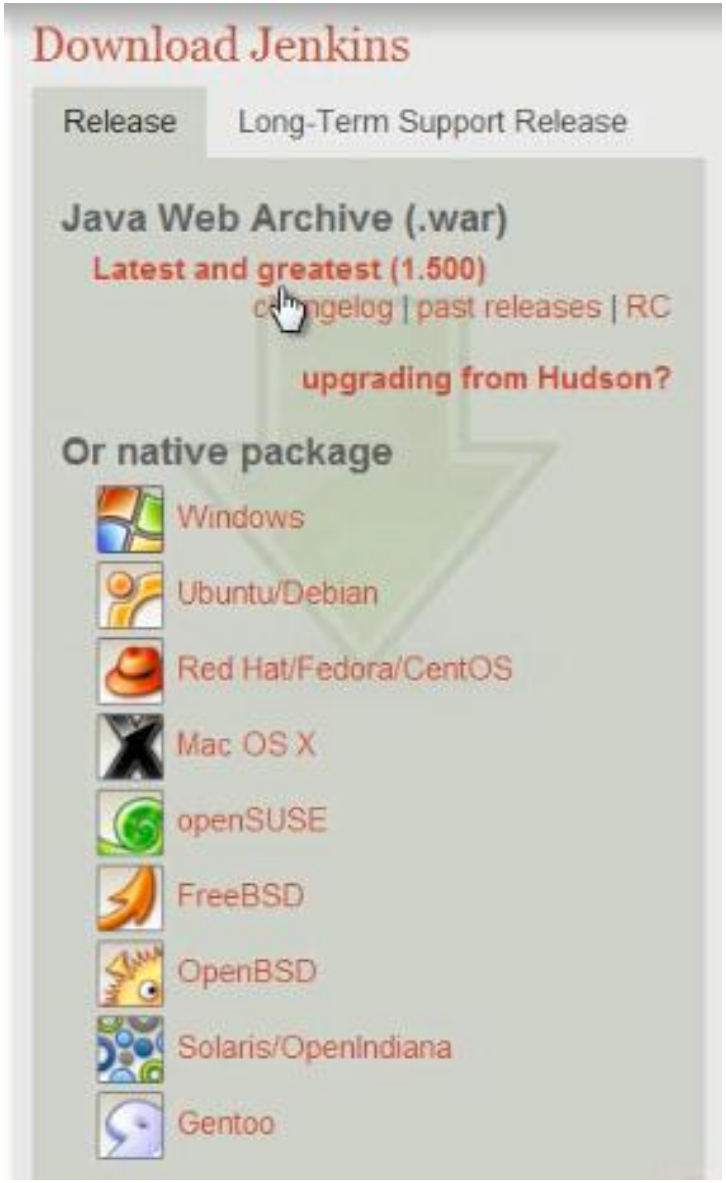
Jenkins is easy to install.

Download Jenkins.war file from the Jenkins site:

- http://jenkins-ci.org
- https://jenkins.io/

Jenkins can be installed in different ways:

- As a standalone application
- Windows Service
- Deploy it on any application server.



Jenkins Installation

To start Jenkins as a standalone application execute the below command in command prompt:

- **java –jar jenkins.war**
- Once Jenkins is started, the Jenkins dash board can be accessed by giving the following link in the browser

  http://localhost:8080/
- To stop Jenkins, press Ctrl+C

To start Jenkins as a windows service, follow the below given steps:

- First, start Jenkins as a standalone application and access Jenkins dash board.
- Click “Manage Jenkins” link available in Jenkins dash board.
- Select “Installation Directory” for Jenkins and click on Install.
- After installation, Jenkins will always run on portno 8080.

Configuring Jenkins

In Jenkins, all aspects of system configuration will be managed virtually in the "Manage Jenkins" screen.

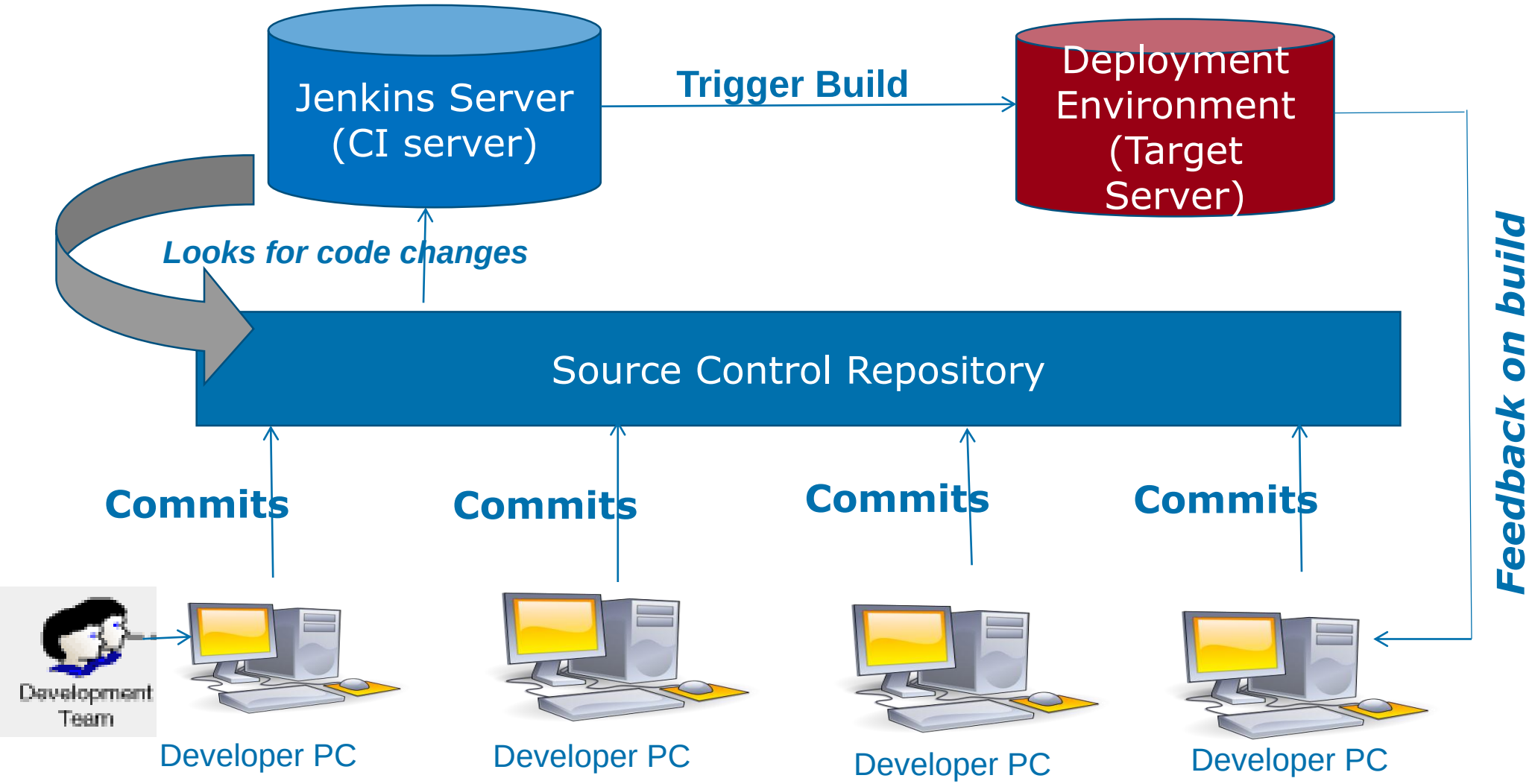
"Configure System" aspect in "Manage Jenkins" screen is mandatory to be configured.

Before using Jenkins, configure JDK and build system as below:

- In "Configure system" on manage Jenkins, enter JDK path
- Also enter build tool path.
- Click on save.
- (Jenkins can also install these for your automatically.)

Jenkins Plugins

How it works?


Jenkins Server
(CI server)
Trigger Build
Deployment
Environment
(Target
Server)
Looks for code changes
Source Control Repository
Commits
Commits
Commits
Commits
Feedback on build
Development
Team
Developer PC
Developer PC
Developer PC
Developer PC

Jenkins Plugins

Plugins Walk through

The Big Picture

Introduction to Jenkins Build Job

A Build Job is a way of compiling, testing, packaging and deploying project to an application server.

During creation of build job, project configuration should be done.

Execution of build job can be done either automatically (Scheduled) or manually.

Create build Job by following the below steps:

- Select “New Job” from menu item which avails in Jenkins Dashboard.
- Select an appropriate type of Build Jobs from the list offered by Jenkins.

Introduction to Jenkins Build Job
Different type of Build Jobs

Freestyle software project

- Freestyle build jobs are general-purpose build jobs, which provides a maximum of flexibility.

Maven project

- The “maven2/3 project” is a build job specially adapted to Maven projects.
- Jenkins understands Maven pom files and project structures, and can use the information gleaned from the pom file to reduce the work related to the set up of project.

Monitor an external job

- The “Monitor an external job” build job lets you keep an eye on non-interactive processes, such as cron jobs.

Multiconfiguration job

- The “multiconfiguration project” (also referred to as a “matrix project”) run the same build job in many different configurations.

Configuring Source Code Management

Jenkins tight integration with Version Control System is essential.

Jenkins supports CVS, GIT and SVN (Version Control System). Also it is easy to integrate with all other version control system via plug-in.

To configure Version Control System, SCM configuration options are offered in all Jenkins Build Jobs.

By default, Jenkins will check out the repository contents into a subdirectory of current Jenkins workspace.