Insight Tech Consulting Group

PROVIDING IT SOLUTIONS…

Jenkins

Continuous Integration

- ❑ Continuous Integration, in its simplest form, involves a tool that monitors your version control system and automatically compiles and tests your application whenever a change is detected.

- ❑ Continuous Integration is the most important part of DevOps that is used to integrate various ***DevOps stages***. Jenkins is the most famous Continuous Integration tool, I know you are curious to know the reason behind the popularity of Jenkins, and if Jenkins is easy to learn.

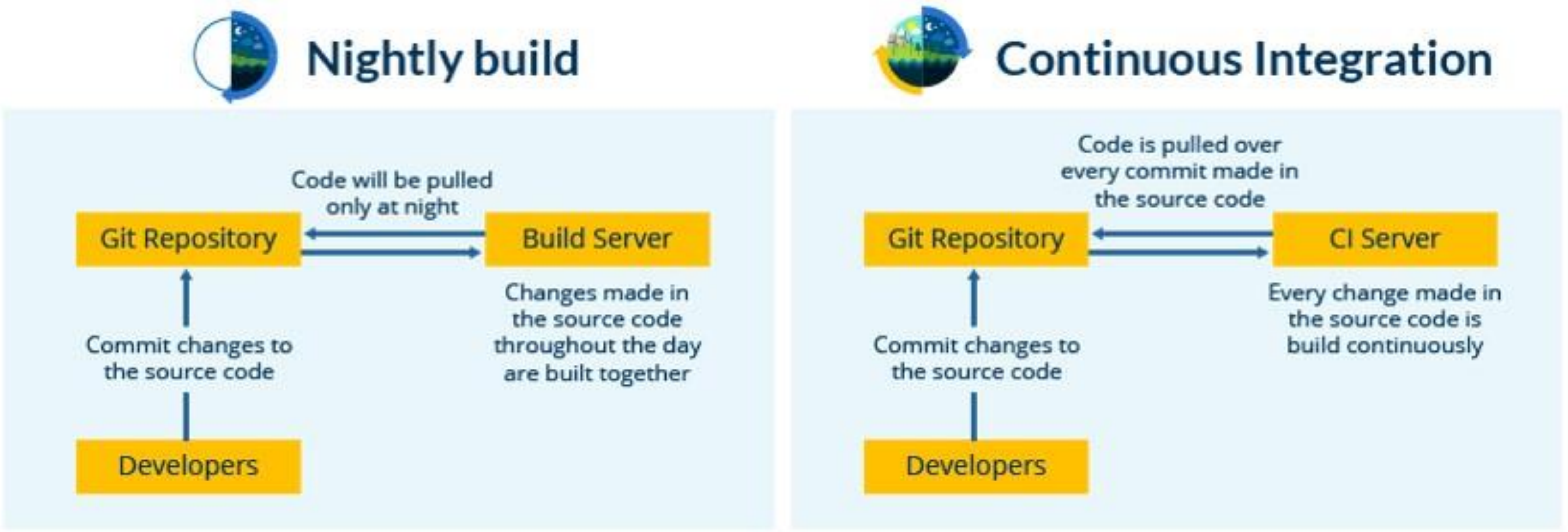
Advantages of CI

- ❑ Continuous Integration automatically monitors the health of your codebase, code quality, and code coverage metrics.
- ❑ Continuous Integration reduces risk by providing faster feedback.
- ❑ CI tools are designed to help identify and fix integration and regression issues faster, resulting in fewer bugs and quicker delivery.
- ❑ CI helps simplify and accelerate delivery by automating the deployment process.
- ❑ Automating the deployment process helps get your software into the hands of the testers and end users faster.

Continuous Integration Example: Nokia

In a software product development project at Nokia, there was a process called **Nightly builds**. Nightly builds can be thought of as a predecessor to Continuous Integration. It means that every night an automated system pulls the code added to the shared repository throughout the day and builds that code. The idea is quite similar to Continuous Integration, but since the code that was built at night was quite large, locating and fixing of bugs was a real pain. Due to this, Nokia adopted Continuous Integration (CI). As a result, every commit made to the source code in the repository was built. If the build result shows that there is a bug in the code, then the developers only need to check that particular commit. This significantly reduced the time required to release new software.

Continuous Integration Example: Nokia



Nightly build

Code will be pulled only at night

Git Repository

Build Server

Changes made in the source code throughout the day are built together

Commit changes to the source code

Developers

Continuous Integration

Code is pulled over every commit made in the source code

Git Repository

CI Server

Every change made in the source code is build continuously

Commit changes to the source code

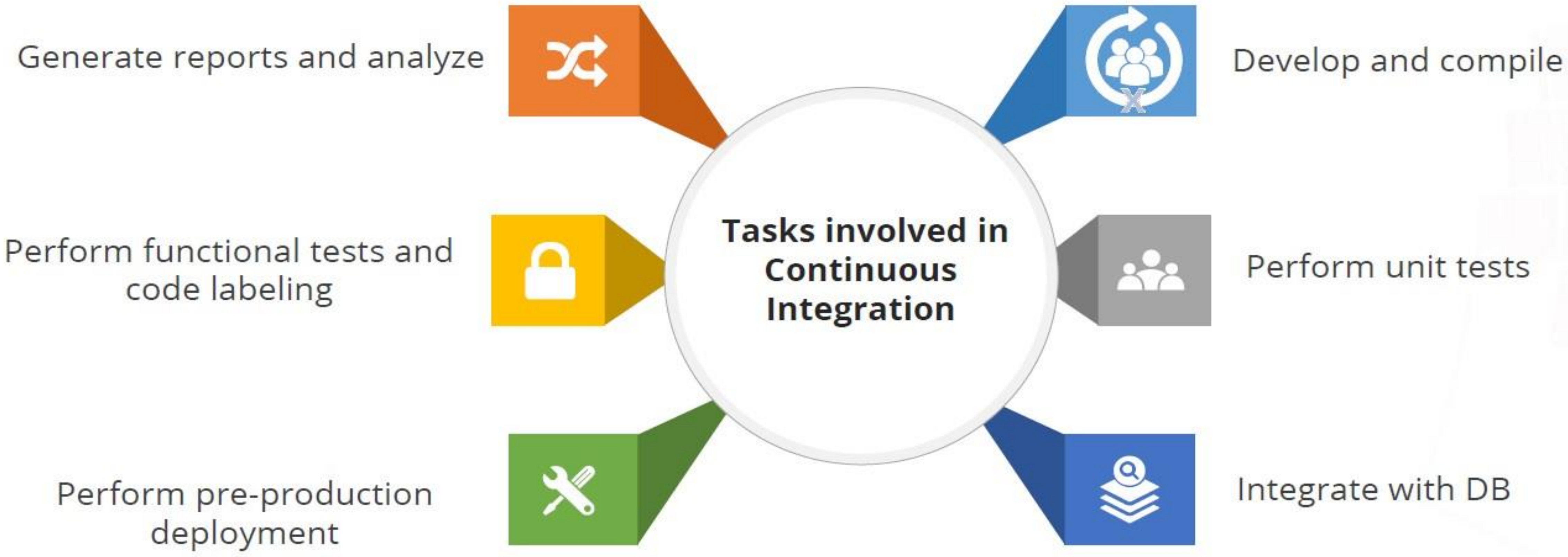
Developers

Continuous Integration

- Continuous Integration can be defined as a development practice of code integration into a shared repository.

- Each integration is verified by an automated build and automated tests.

This figure shows the tasks involved in Continuous Integration.


Tasks involved in Continuous Integration
Generate reports and analyze
Develop and compile
Perform functional tests and code labeling
Perform unit tests
Perform pre-production deployment
Integrate with DB

Continuous Integrations Tools

While there are many DevOps tools for Continuous Integration, there are some that are more widely used. Selecting a best appropriate CI tool can be a bit challenging, more so if one is going to use it for the first time. Below are few of the popular CI tools:

- **Bamboo**
- **CircleCI**
- **GitLab CI**
- **Travis CI**
- **TeamCity**
- **Jenkins**

Jenkins

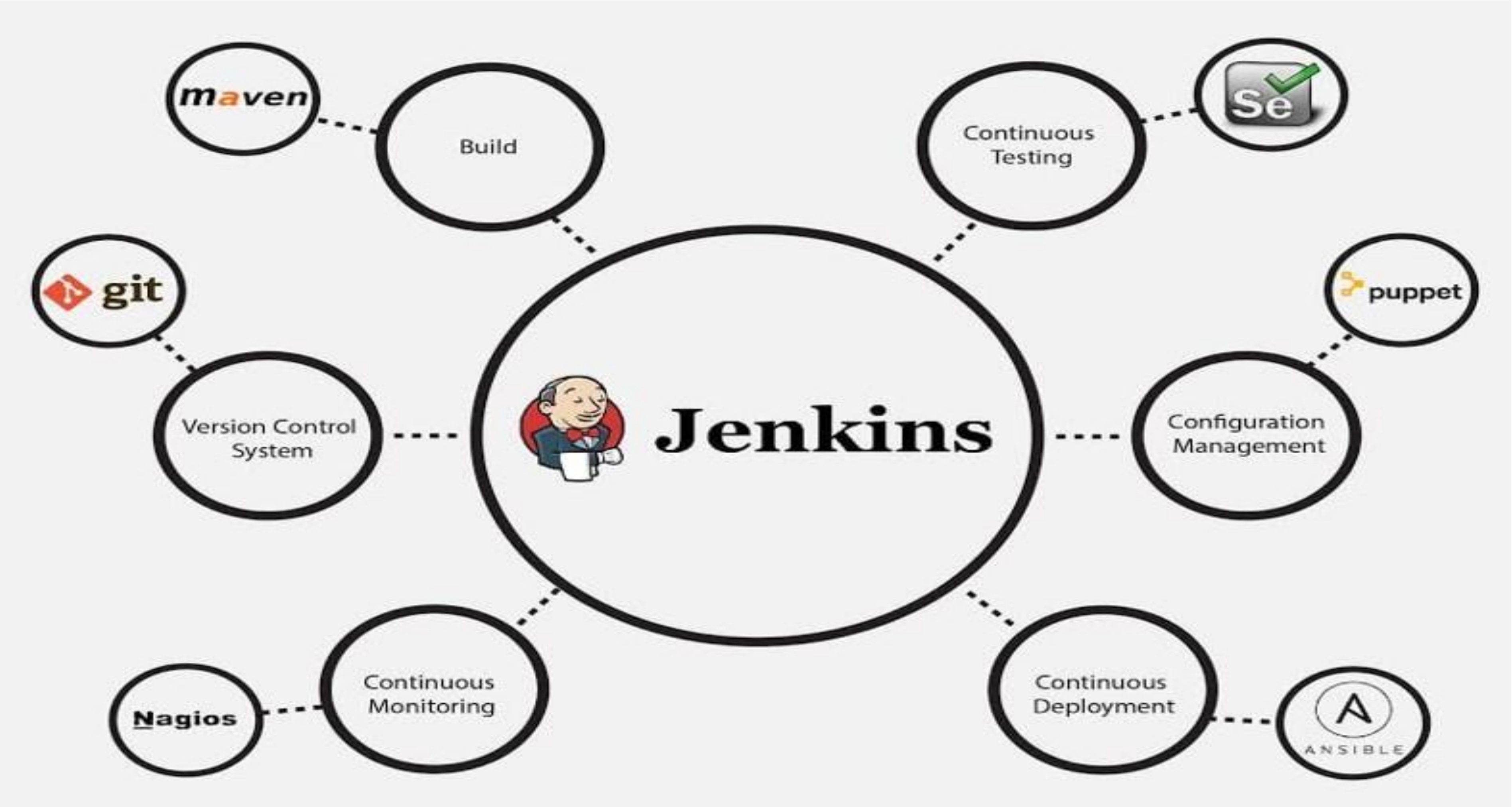
Let us understand what is Jenkins in simple words.

Jenkins is an open-source automation tool written in Java with plugins built for Continuous Integration purposes. Jenkins is used to build and test your software projects continuously making it easier for developers to integrate changes to the project and making it easier for users to obtain a fresh build. It also allows you to continuously deliver your software by integrating with a large number of testing and deployment technologies.

With Jenkins, organizations can accelerate the software development process through automation. Jenkins integrates development life-cycle processes of all kinds, including build, document, test, package, stage, deploy, static analysis, and much more.

Jenkins achieves Continuous Integration with the help of plugins. Plugins allow the integration of Various DevOps stages. If you want to integrate a particular tool, you need to install the plugins for that tool.

For example: Git, Maven 2 project, Amazon EC2, HTML publisher etc.


maven
Build
Continuous Testing
Se
git
puppet
Version Control System
Jenkins
Configuration Management
Nagios
Continuous Monitoring
Continuous Deployment
ANSIBLE

Advantages of Jenkins

- It is an open-source tool with great community support.
- It is easy to install.
- It has 1000+ plugins to ease your work. If a plugin does not exist, you can code it and share it with the community.
- It is free of cost.
- It is built with Java and hence, it is portable to all the major platforms.

Features of Jenkins

There are certain things about Jenkins that separates it from other the Continuous Integration tool. Let us take a look on those points. The following are some facts about Jenkins that makes it better than other Continuous Integration tools:

- **Adoption:** Jenkins is widespread, with more than 147,000 active installations and over 1 million users around the world.
- **Plugins:** Jenkins is interconnected with well over 1,000 plugins that allow it to integrate with most of the development, testing and deployment tools.

Jenkins Console Components

- **Jenkins Jobs:** A job is a collection of steps that you can use to build your source code, test your code, run a shell script, or to run an Ansible role in a remote host. Ther are multiple job types available to support your workflow for continuous integration & continuous delivery.
- **Jenkins Plugins:** Plugins are community-developed modules that you can install in your Jenkins server. It lets you add more functionalities that are not natively available in Jenkins. You can also develop your custom plugins. Check out all plugins from the Jenkins Plugin Index
- **Jenkins User:** Jenkins has its own user database. It can be used for Jenkins's authentication.
- **Jenkins Global Security:** Jenkins has the following two primary authentication methods.
  - Jenkins's own user database:- Set of users maintained by Jenkins's own database.
  - LDAP Integration:- Jenkins authentication using corporate LDAP configuration.

Jenkins Console Components

- **Jenkins Credentials:** If you want to save any secret information that has to be used in the jobs, you can store it as a credential. All credentials are encrypted by Jenkins.
- **Jenkins Nodes/Clouds:** You can configure multiple slave nodes (Linux/Windows) or clouds (docker, kubernetes) for executing Jenkins jobs.
- **Jenkins Global Settings (Configure System):** Under Jenkins global configuration, you have all the configurations of installed plugins and native Jenkins global configurations. Also, you can configure global environment variables under this section.
- **Jenkins Logs:** Provides logging information on all Jenkins server actions including job logs, plugin logs, webhook logs, etc.

Jenkins Data

All the Jenkins data will be store in the following folder location. Data includes all jobs config files, plugins configs, secrets, node information, etc.

/var/lib/jenkins/

Jenkins Installations

The complete process to install Jenkins can be summarized in five steps:

1. **Install Java Version 8** – Jenkins is a Java based application; hence Java is a must.
2. **Install Apache Tomcat Version 9** – Tomcat is required to deploy Jenkins war file.
3. **Download Jenkins war File** – This war is required to install Jenkins.
4. **Deploy Jenkins war File** – Jenkins war file needs to be deployed using Tomcat to run Jenkins.
5. **Install Suggested Plugins** – Install a list of plugins suggested by Jenkins.