Continuous integration

- ***Continuous integration (CI)*** *is the practice of regularly* integrating and testing our solution to incorporate changes made to its definition;
- Changes include updating the source code, changing a database schema, or updating a configuration file.
- When one or more changes are checked into our **configuration management system**, the solution should be rebuilt (recompiled), retested, and any code or schema analysis performed on it.
- Failing that, we should strive to do so at least once if not several times a day
- On a developer’s workstation, the integration job could run at specific times, perhaps once an hour, or better every time that she checks in something that is part of the build.
- This whole process of continuously integrating a developer’s code with the rest of a team’s code in and then running automated **test regressions** in an integration environment is a critical part of agile done right.
- Continuous Integration ensures high-quality working software at all times
- Code is integrated and tested many times a day: one set of changes at a time
- There will be a machine dedicated to integration
- A pair with code ready to integrate
    - Sits when the machine is free.
    - Loads the current release.
    - Loads their changes (checking for and resolving any collisions).
    - Runs the tests until they pass (100% correct)

➢ If a test fails during integration → current pair should fix it

❖ **<u>Continuous Integration – Advantages:</u>**

– Reduces the duration, which is otherwise lengthy.

– Enables the short releases practice as the time required before release is minimal.

AutomAted build tools for CI

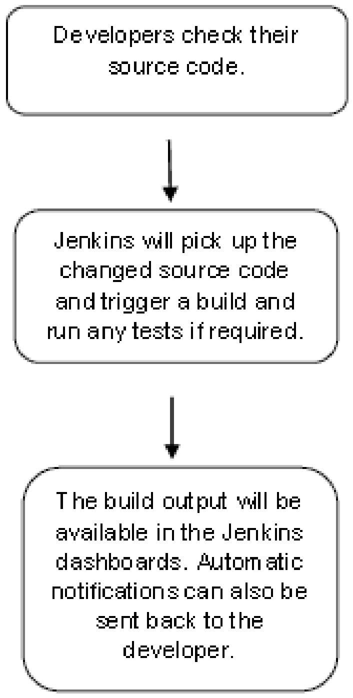
1. **AnthillPro,** build automation with pipeline support for deployment automation and testing. Cross-platform, cross-language.
2. **Apache Continuum** - discontinued
3. **Bamboo,** continuous integration software
4. **Buildbot,** a Python-based software development continuous integration tool which automates the compile/test cycle
5. **BuildIT,** a free graphical build or task tool for Windows with an emphasis on simplicity and ease of use
6. **CABIE** - Continuous Automated Build and Integration Environment, open source, written in Perl
7. **CruiseControl,** for Java and .NET
8. **Go continuous delivery,** open source, cross-platform
9. **GitLab,** continuous integration and git server
10. **Hudson,** an extensible continuous integration engine, **forked from Jenkins**
11. **Jenkins,** an extensible continuous integration engine, **forked from Hudson**

Jenkins

Jenkins is an open source automation server written in Java. Jenkins helps to automate the non-human part of the software development process, with **continuous integration** and facilitating technical aspects of **continuous delivery.**

It is a server-based system that runs in servlet containers such as **Apache Tomcat.** It supports **version control tools**, including AccuRev, CVS, Subversion, Git, Mercurial, Perforce, ClearCase and RTC, and can execute Apache Ant, Apache Maven and sbt based projects as well as arbitrary shell scripts Linux and Windows batch commands.

The following flowchart demonstrates a very simple workflow of how Jenkins works.


Developers check their source code.
Jenkins will pick up the changed source code and trigger a build and run any tests if required.
The build output will be available in the Jenkins dashboards. Automatic notifications can also be sent back to the developer.

Along with Jenkins, sometimes, one might also see the association of **Hudson.** Hudson is a very popular open-source Java-based continuous integration tool developed by Sun Microsystems which was later acquired by Oracle. After the acquisition of Sun by Oracle, a fork was created from the Hudson source code, which brought about the introduction of Jenkins.