**Continuous Integration / C I :** is a **development practice that requires developers to integrate code into a shared repository several times a day**.

Each **Check-in is then verified by an automated build allowing team to detect problems early**.

**By integrating regularly. We can detect errors quickly and locate them more easily.**

**Because integrating frequently there is significant less back tracking to discover where things went wrong, so we spend more time building features.**

**Advantages:**

* Increase visibility enabling greater communication.
* Detect issues early.
* Spend less time debugging and more time adding features.
* Stop waiting to find out if code’s going to work.
* Reduce integration problems allowing us to deliver S/W more rapidly.

Note: CI doesn’t get rid of the bugs, but it does work them dramatically easier to find and remove.

**CI is backed by several important principles and practices:**

**The Practices:**

* Maintain a **single source repository.**
* **Automate the build.**
* **Make you build self - testing.**
* **Every commit should build on an Integration machine.**
* **Keep the build fast.**
* **Make it easy for anyone to get latest executable version**
* **Everyone can see what’s happening.**
* **Automate development.**

**How to do it:**

* **Developers check-out code** into their private workspaces.
* When done, **commit the changes** to the **repository**.
* The **CI Server monitors** the **repository** and **checks out changes when they occur.**
* The **CI Server builds** the **system** and **runs unit** and **integration tests.**
* The **CI Server releases deployable artefacts** for **testing.**
* The **CI Server assigns a build label** to the **version of the code it just built.**
* The **CI Server informs** the **team** **of the successful build.**
* **If the build or tests fail, the CI Server alerts the team.**
* The **team fixes the issue at the earliest opportunity.**
* Continue to **continually integrate** and **test** and **throughout the project.**

**Team responsibility:**

* **Check-in** in frequently.
* Don’t check in broken code.
* Don’t check in untested code.
* Don’t check in when build is broken.
* Don’t go home after checking in until the system build.

**Continuous Delivery/ CD:** is closely related to Continuous Integration and refers to the r**elease into production of the Software that passed the automated tests**.

* Essentially, it is the practice of releasing every good build to users.

By adopting both CI and CD, we not only reduce risks and catch bugs quicky, but also nore rapidly to working software.

**Top CI tools:**

Jenkins

Teamcity

Travis CI

GoCD

Bamboo

GitLab CI

Circe CI

CodeShip