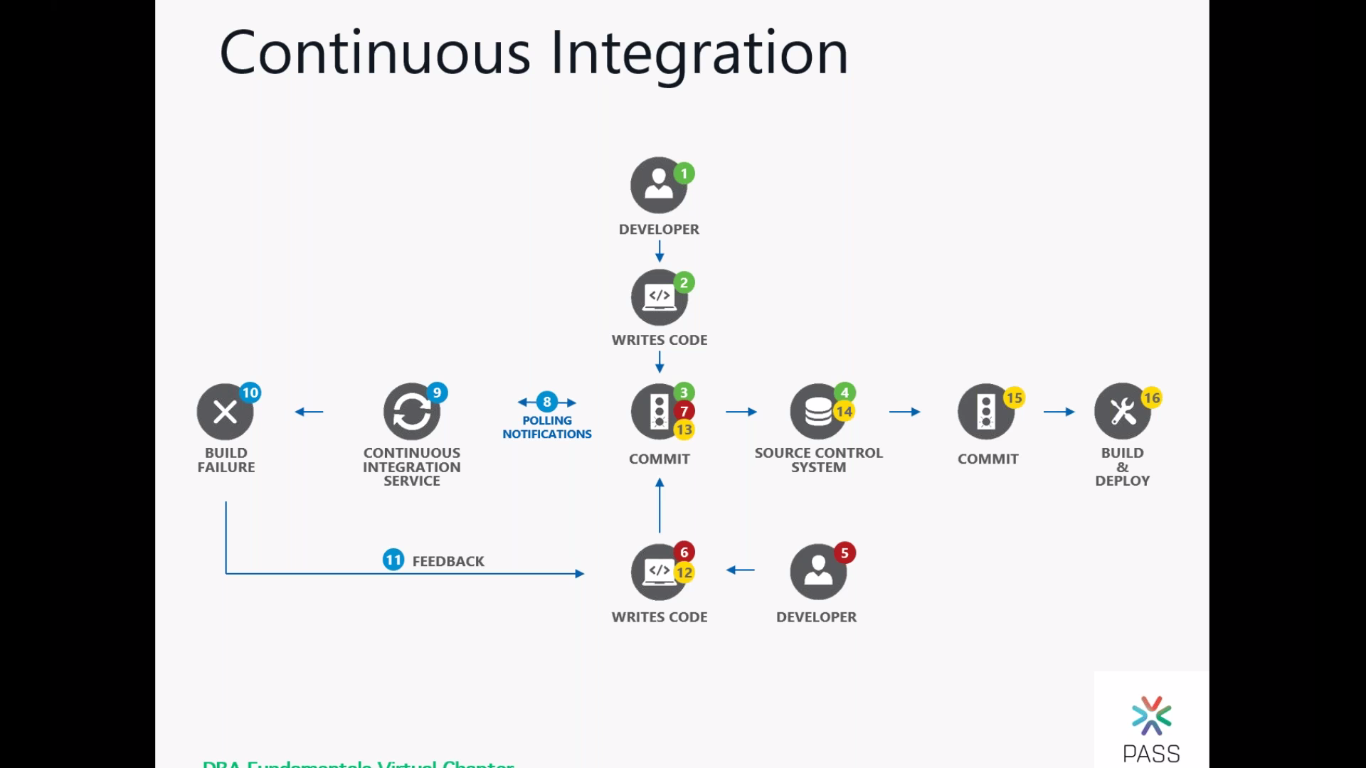
**Agile**

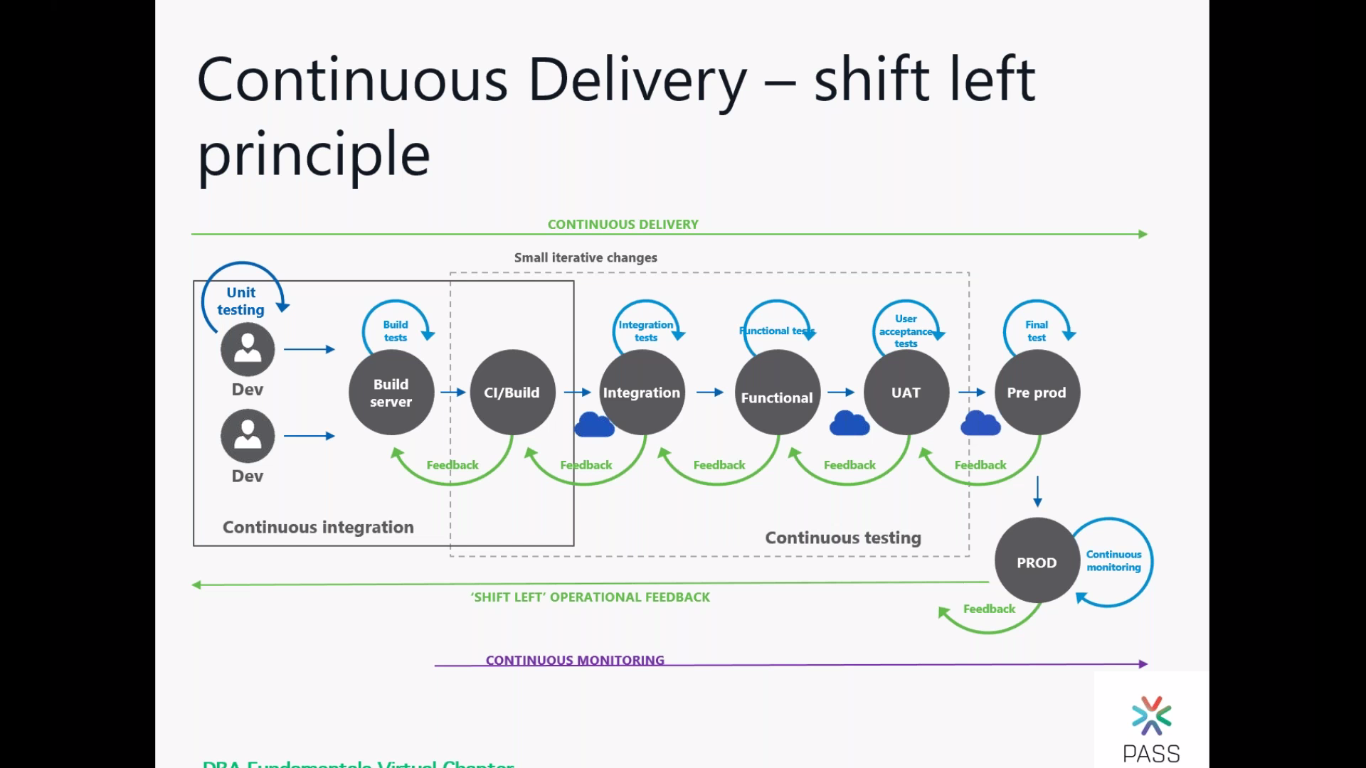
Agile is a set of values and principles about how to produce i.e. develop software. Example: if you have some ideas and you want to turn those ideas into working software, you can use the Agile values and principles as a way to do that. But, that software might only be working on a developer’s laptop or in a test environment. **You want a way to quickly, easily and repeatably move that software into production infrastructure, in a safe and simple way**. To do that you need DevOps tools and technique

Agile software development methodology focuses on the development of software but DevOps on the other hand is responsible for development as well as deployment of the software in the safest and most reliable way possible

**DevOps**

* Continuous integration
* Continuous testing
* Continuous delivery
* Continuous monitoring
* Continuous security
* Continuous collaboration
* Continuous communication on chat





companies are required to:

1. Increase deployment frequency
2. Lower failure rate of new releases
3. Shortened lead time between fixes
4. Faster mean time to recovery in the event of new release crashing

DevOps fulfils all these requirements and helps in achieving seamless software delivery

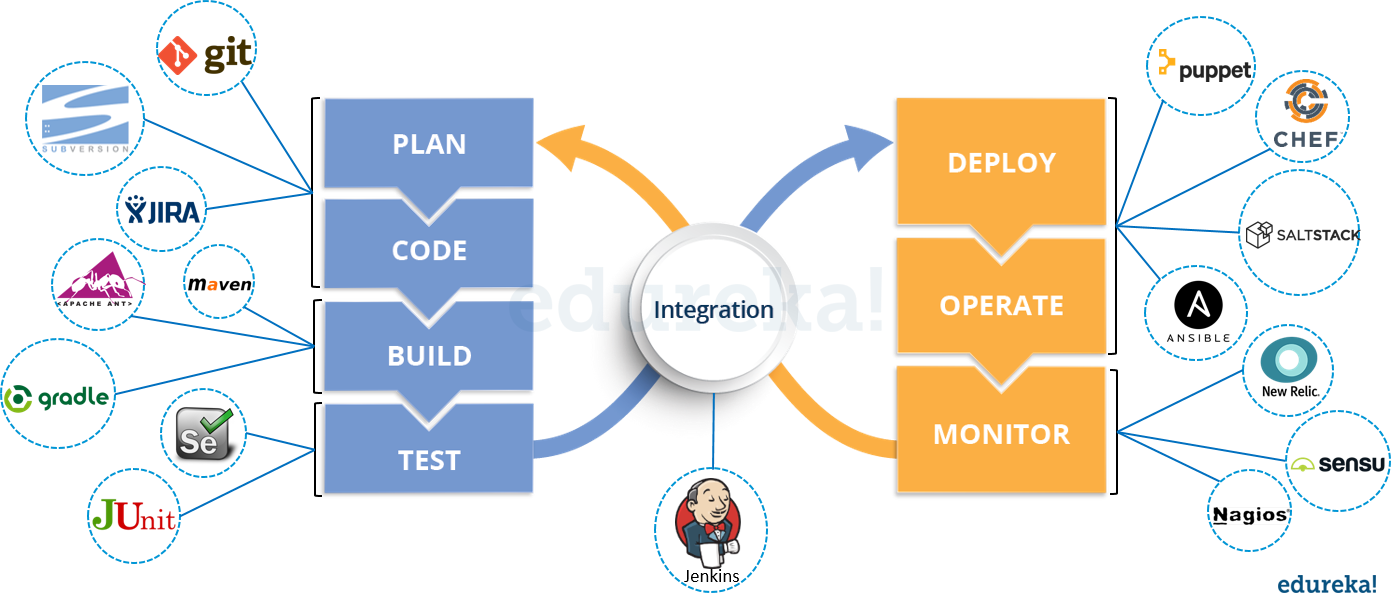
**The most popular DevOps tools**

* Git: Version Control System tool
* Jenkins: Continuous Integration tool
* Puppet, Chef, Ansible: Configuration Management and Deployment tools
* Nagios: Continuous Monitoring tool
* Docker: Containerization tool

**DevOps Work Flow**

where everything gets automated for seamless delivery. However, this flow may vary from organization to organization as per the requirement

1. Developers develop the code and this source code is managed by Version Control System tools like Git etc.
2. Developers send this code to the Git repository and any changes made in the code is committed to this Repository.
3. Jenkins pulls this code from the repository using the Git plugin and build it using tools like Ant or Maven.
4. Configuration management tools like puppet deploys & provisions testing environment and then Jenkins releases this code on the test environment on which testing is done using tools like selenium.
5. Once the code is tested, Jenkins send it for deployment on the production server (even production server is provisioned & maintained by tools like puppet).
6. After deployment, it is continuously monitored by tools like Nagios.
7. Docker containers provides testing environment to test the build features



**Advantages**

Technical benefits:

* Continuous software delivery
* Less complex problems to fix
* Faster resolution of problems

Business benefits:

* Faster delivery of features
* More stable operating environments
* More time available to add value (rather than fix/maintain)

------------

**Version control**

the uses of version control

* Revert files back to a previous state.
* Revert the entire project back to a previous state.
* Compare changes over time.
* See who last modified something that might be causing a problem.
* Who introduced an issue and swhen ?

**Benefits**

* With Version Control System (VCS), all the team members are allowed to work freely on any file at any time. VCS will later allow you to merge all the changes into a common version.
* All the past versions and variants are neatly packed up inside the VCS. When you need it, you can request any version at any time and you’ll have a snapshot of the complete project right at hand.
* Every time you save a new version of your project, your VCS requires you to provide a short description of what was changed. Additionally, you can see what exactly was changed in the file’s content. This allows you to know who has made what change in the project.

A distributed VCS like Git allows all the team members to have complete history of the project so if there is a breakdown in the central server you can use any of your teammate’s local Git repository.

**SVN Installation on windows (SVN\_Install\_Win.txt)**