

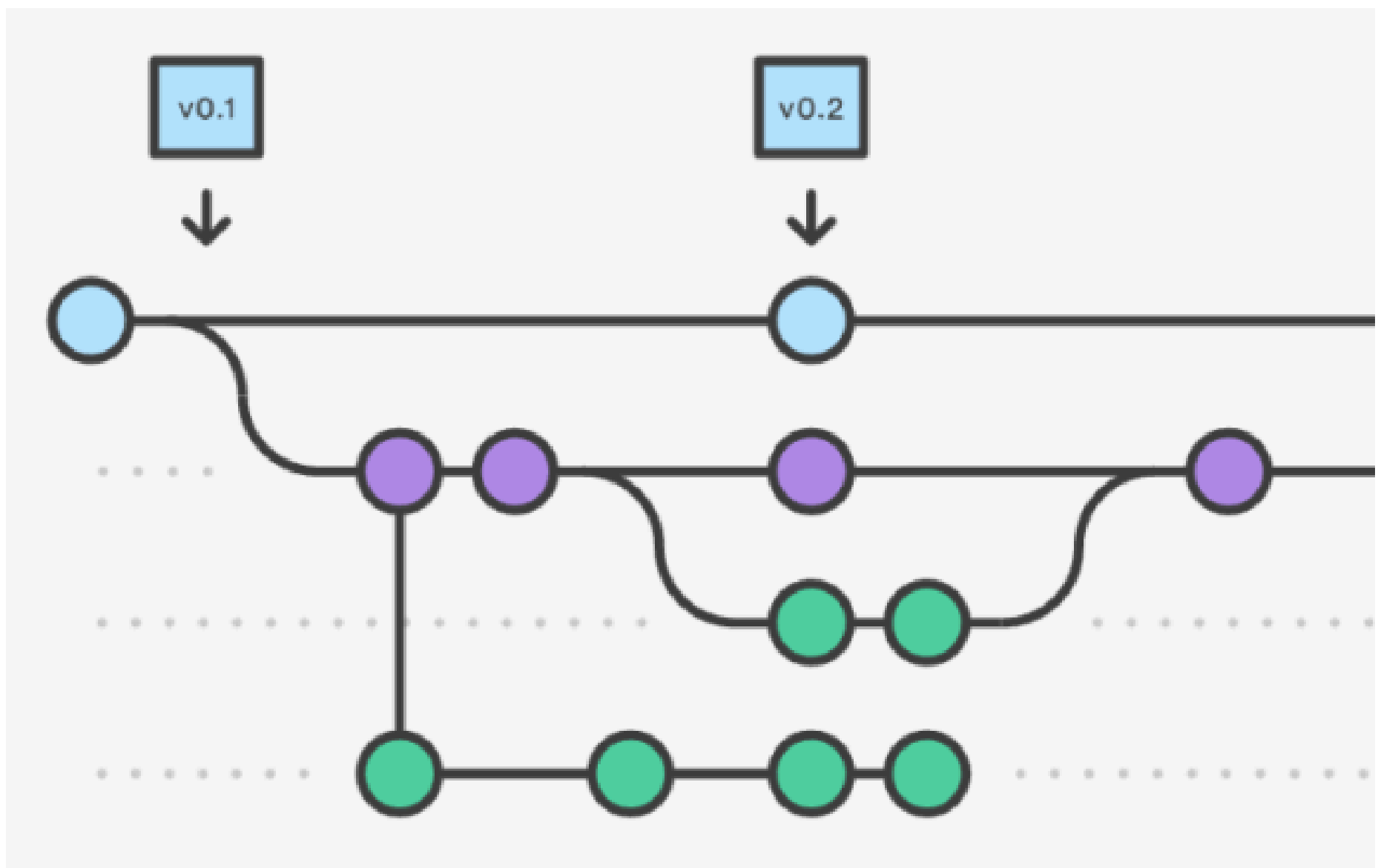
6.1 Why Version Controlling?

6.1 Why Version Controlling?

Version control (also known as source control) means **tracking** and **maintaining** changes to software code. Version control systems(VCS) are software tools that help software teams, manage changes to source code over time. As development environments have improved, version control systems help software teams work faster and smarter.

Version control software keeps track of every change of the code in a special kind of database. If a mistake is made, developers can go back and check earlier versions of the code to fix the mistake.

Version control helps developer teams by, tracking every individual change by each developer and preventing simultaneous work from conflicting. Changes made in one part of the software can be incompatible with those made by another developer working at the same time. This problem should be found and solved without blocking the work of the rest of the team. In all software development, any change can cause new bugs and new software can't be trusted until it's tested. Therefore testing and development proceed together until a new version is ready.



Features of a good Version Control Software

A good version control software supports a developer's preferences without specifying a way of working. In ideal conditions, it also supports any platform, rather than specify what operating system or toolchain developers must use. Great version control systems facilitate a smooth and continuous flow of changes to the code rather than file locking(giving the green light to one developer at the expense of blocking the progress of others).

Importance of Version Control

Software teams that don't use any version control usually run into problems like not knowing which changes that have been made are available to users or the creation of incompatible changes between two unrelated pieces of work that must then be unscrambled and reconstructed. If you are a developer who has never used version control you'll have created versions of your files with suffixes like "final" or "latest" then had to handle a new final version. You may have commented out code blocks because you want to disable a certain function without deleting the code, assuming that there might be a use for it later. Version control is a way to handle these problems.

Version control is an essential part of the modern software team's work. Individual software developers who are familiar with a capable version control system in their teams typically recognize the incredible value of version control, even on small projects. Once familiar with the powerful benefits of version control systems, many developers wouldn't consider working without it even for non-software projects.

Using VCS is the best practice for high-performing software and DevOps teams*. VCS also helps developers act faster and enables

software teams to secure efficiency as the team grows to include more developers.

Benefits of a VCS

The primary benefits you should expect from version control are as follows.

Complete long-term change history of every file

This means every change over the years. Those changes include the creation and deletion of files as well as edits to their contents. Different VCS tools differ on how well they handle the renaming and moving of files. This history should also include the author, date, and written notes on the aim of every change. Having the whole history facilitates going back to previous versions for root cause analysis for bugs and it's critical when required to fix problems in older versions of software. If the software is being actively developed, almost everything will be considered an "older version" of the software.

Branching and merging

Having team members work concurrently could be a no-brainer, but even individuals engaged on their own can have the benefit of the flexibility to work on independent streams of change. Creating a "branch" in VCS tools keeps multiple streams of work independent from one another while also providing the ability to merge that employment back together, enabling developers to verify that the changes on each branch don't conflict. Many software teams adopt a practice of branching for every feature or perhaps branching for every release, or both. There are many various workflows that teams can choose between once they decide the way to make use of branching and merging facilities in VCS.

Traceability

Being able to trace each change made to the software and connect it to project management and bug tracking software like Jira, and having the ability to annotate each change with a message describing the aim and intent of the change can help not only with root cause analysis and other forensics. Having the annotated history of the code at your fingertips once you are reading the code, trying to know what it's doing and why it's so designed can enable developers to create correct and harmonious changes that are in unison with the intended long-term design of the system. this will be especially important for working effectively with legacy code* and is crucial in enabling developers to estimate future work with any accuracy.



Some popular Version Control Systems

Git

Version Control Systems (VCS) have seen great improvements over the past few decades and a few are better than others. VCS has sometimes been referred to as SCM (Source Code Management) tools or RCS (Revision Control System).

One of the foremost popular VCS tools in use today is Git. Git may be a Distributed VCS, a category referred to as DVCS. Like many of the foremost popular VCS systems available today, Git is free and open source.

[Git website](https://git-scm.com/)

Note:

***A DevOps** team includes developers and IT operations working collaboratively throughout the merchandise lifecycle, so as to improve the speed and quality of software deployment. it is a new way of working, a cultural shift, that has significant implications for teams and the organizations they work for.

***Legacy code** is old computer source code. It could simply check with an organization's existing code base which has been written over a few years, or it could imply a codebase that's in some respect obsolete or supporting something obsolete.

Further Reading:
[What is version control? - Atlassian](#)

[About Version Control - Pro Git eBook: 2nd edition \(2014\).](#)


 **Lecture Note**


2 / 4  


◀ 5.3 Quiz

Jump to...

GET IN TOUCH

 University of Moratuwa
Centre for Open & Distance Learning
CODL

 011 308 2787/8

 011 265 0301 ext. 3850,3851

 open@uom.lk

 [University Website](#)

 [CODL Website](#)

[Data retention summary.](#)

 mora_logo.png Sponsored By  logo.png

