ASSIGNMENT

TOPIC: Version control system and its analysis using git

Submitted to:

Mr Jinson Devis

Submitted by

Divya Antony

Roll No:23

Introduction

In the dynamic landscape of software development, Version Control Systems (VCS) have emerged as fundamental tools, revolutionizing the way teams collaborate, manage code, and track project evolution. Version control systems are software tools that help software teams manage changes to source code over time. As development environments have accelerated, version control systems help software teams work faster and smarter.

Git is a version control system used for tracking changes in computer files. It is generally used for source code management in software development.

1.VERSION CONTROL SYSTEM

Version control, also known as source control, is the practice of tracking and managing changes to software code. Version control systems are software tools that help software teams manage changes to source code over time. As development environments have accelerated, version control systems help software teams work faster and smarter. They are especially useful for DevOps teams since they help them to reduce development time and increase successful deployments.
Version control software keeps track of every modification to the code in a special kind of database. If a mistake is made, developers can turn back the clock and compare earlier versions of the code to help fix the mistake while minimizing disruption to all team members.

Benefits of the version control system:

- Enhances the project development speed by providing efficient collaboration,
- Leverages the productivity, expedites product delivery, and skills of the employees through better communication and assistance,
- Reduce possibilities of errors and conflicts meanwhile project development through traceability to every small change,
- Employees or contributors of the project can contribute from anywhere irrespective of the different geographical locations through this VCS,

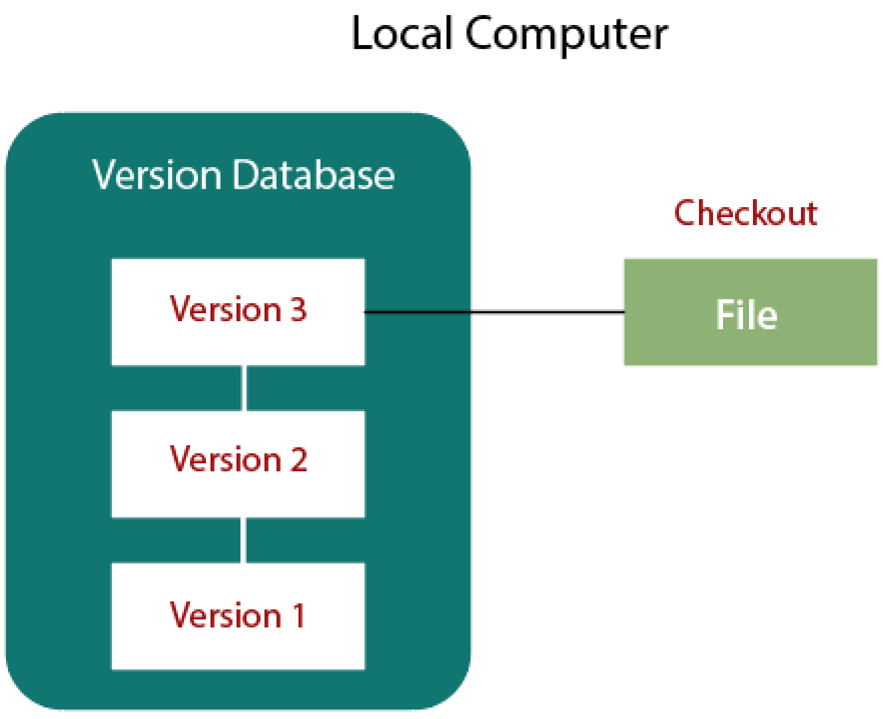
- For each different contributor to the project, a different working copy is maintained and not merged to the main file unless the working copy is validated. The most popular example is Git, Helix core, Microsoft TFS,
- Helps in recovery in case of any disaster or contingent situation,
- Informs us about Who, What, When, Why changes have been made.

1.1 Types of version control systems

- Localized version Control System
- Centralized version control systems
- Distributed version control systems

Localized version control system

The localized version control method is a common approach because of its simplicity. But this approach leads to a higher chance of error. In this approach, you may forget which directory you're in and accidentally write to the wrong file or copy over files you don't want to .To deal with this issue, programmers developed local VCSs that had a simple database. Such databases kept all the changes to files under revision control. A local version control system keeps local copies of the files. The major drawback of Local VCS is that it has a single point of failure.

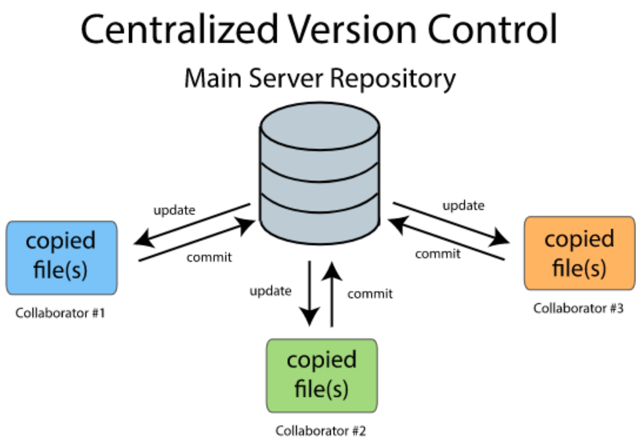


Centralized Version Control Systems (CVCS):

In a CVCS, there is a single, centralized server that stores the entire version history and the files. Developers check out files from this central repository to work on them, and they commit changes back to the central server. Examples of CVCS include Concurrent Versions System (CVS) and Apache Subversion (SVN). Centralized version control systems have many benefits

- Everyone on the system has information about the work what others are doing on the project.
- Administrators have control over other developers.
- It is easier to deal with a centralized version control system than a localized version control system.
- A local version control system facilitates with a server software component which stores and manages the different versions of the files.

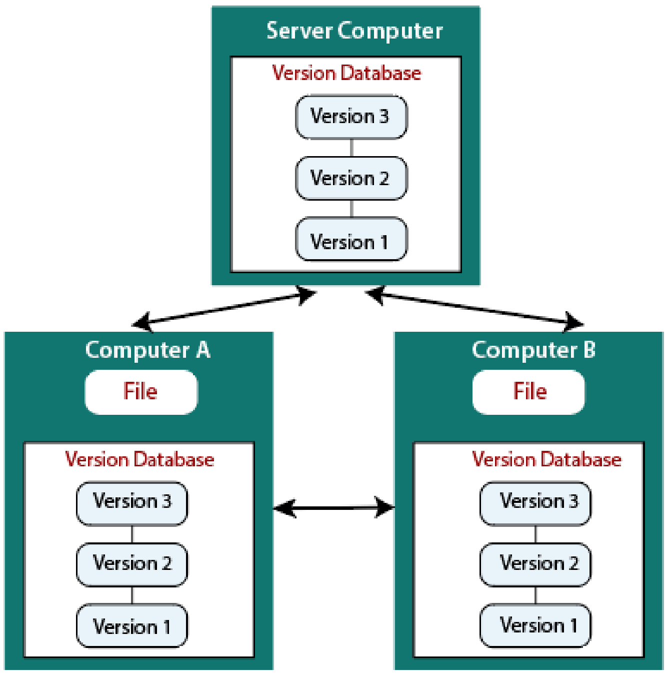
It also has the same drawback as in local version control system that it also has a single point of failure.



A Centralized Version Control System. Users could check out files they wanted to work on, then commit them once they made their changes.

Distributed Version Control Systems (DVCS):

In a DVCS, each user has a complete copy of the entire repository, including its full history, on their local machine. Users can work independently and commit changes to their local repository. They can also synchronize their changes with other repositories. This decentralized nature allows for more flexibility, offline work, and easier branching and merging. Examples of DVCS include Git, Mercurial, and Bazaar. Git is the most widely used distributed version control system. It has become the standard for many open-source and private projects due to its speed, flexibility, and strong branching and merging capabilities.



Benefits of version control

- Quality
- Acceleration
- Visibility

1.2 Version Control Tools

1.Git

Git is among the most powerful version control programs now on the market. The creator of Linux, Linus Torvalds, created the distributed version control system known as Git. Its memory footprint is minimal and can follow changes in any file. When you add this to its extensive feature set, you get a full-featured version control system that can handle any project. Due to its simple workflow, it is employed by Google, Facebook, and Microsoft.

2.Apache Subversion

A version control system called Apache Subversion, which is free and open-source, enables programmers to manage both the most recent and previous iterations of crucial files. It can track modifications to source code, web pages, and documentation for large-scale projects. Subversion’s main features are workflow management, user access limits, and cheap local branching. Both commercial products and individual projects can be managed using Subversion, a centralized system with many powerful features. It is one of Apache’s many open-source solutions, like Apache Cassandra.

3.Mercurial

Developers and businesses adore Mercurial for its search capabilities, backup system, data import and export, project tracking and management, and data migration tool. The free source control management program Mercurial supports all popular operating systems. It is a distributed versioning solution and can easily manage projects of any size. Through extensions, programmers can quickly expand the built-in functionality. For software engineers, source revisioning is made simpler by its user-friendly and intuitive interface.

4.AWS CodeCommit

Private Git repositories are hosted by the managed version control system AWS CodeCommit. It smoothly integrates with other Amazon Web Services (AWS) products, and the code is hosted in secure AWS settings. Therefore, it’s a suitable fit for AWS’s current users. Access to various helpful plugins from AWS partners is also made available through AWS integration, aiding in program development. You don’t have to worry about maintaining or scaling your source control system when you use CodeCommit.

1.3 Advantages of Version Control system

- Traceability
- Document History
- Branching and Merging
- Parallel Development
- Backup Recovery

1.4 Disadvanatges of Version Control System

- Learning Curve
- Infrastruture Dependency
- Complex Conflit Resolution
- Storage Requirements
- Network Dependency

<u>2.Version Control System Analysis using Git</u>

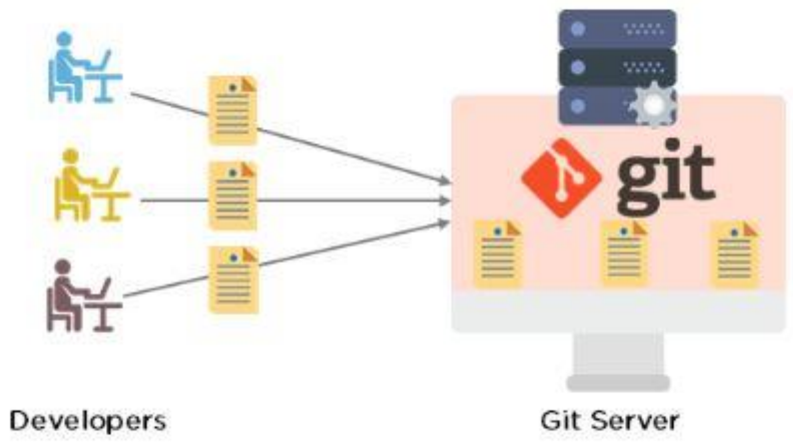
Git is a DevOps tool used for source code management. It is a free and open-source version control system used to handle small to very large projects efficiently. Git is used to tracking changes in the source code, enabling multiple developers to work together on non-linear development. Git is a version control system used for tracking changes in computer files. It is generally used for source code management in software development.

- Git is used to tracking changes in the source code
- The distributed version control tool is used for source code management
- It allows multiple developers to work together
- It supports non-linear development through its thousands of parallel branches

2.1 Features of Git

- Tracks history
- Free and open source
- Supports non-linear development

- Creates backups
- Scalable
- Supports collaboration
- Branching is easier
- Distributed development



2.2 Benefits of Git

A version control application allows us to **keep track** of all the changes that we make in the files of our project. Every time we make changes in files of an existing project, we can push those changes to a repository. Other developers are allowed to pull your changes from the repository and continue to work with the updates that you added to the project files. Some **significant benefits** of using Git are as follows:

- **Saves Time**: Git is lightning fast technology. Each command takes only a few seconds to execute so we can save a lot of time as compared to login to a GitHub account and find out its features.
- **Offline Working**: One of the most important benefits of Git is that it supports **offline working**. If we are facing internet connectivity issues, it will not affect our work. In Git, we can do almost everything locally. Comparatively, other CVS like SVN is limited and prefer the connection with the central repository.

- **Undo Mistakes**: One additional benefit of Git is we can **Undo** mistakes. Sometimes the undo can be a savior option for us. Git provides the undo option for almost everything.
- **Track the Changes**: Git facilitates with some exciting features such as **Diff, Log,** and **Status**, which allows us to track changes so we can **check the status, compare** our files or branches.

2.1Advantages of Git

- **Distributed Development**: Git allows each developer to have their own local copy of the entire repository. This distributed nature enables offline work and faster access to version history.
- **Branching and Merging**: Git provides robust branching and merging capabilities, making it easy for developers to work on isolated features or bug fixes in separate branches and merge changes seamlessly
- **Speed and Performance**: Git is designed to be fast and performant, even with large repositories. This speed is crucial for developers to maintain efficiency in their workflow.
- **Security Features**: Git includes built-in security features, such as cryptographic integrity checks and support for GPG signatures for commits. Access controls can be enforced through authentication mechanisms, adding a layer of security to code repositories.
- **Flexibility for Different Workflows**: Git is flexible and supports various workflows, accommodating the needs of different development teams. Whether it's feature branching, Gitflow, or GitHub flow, Git can adapt to different project structures.

Conclusion

Version control systems like Git play a pivotal role in modern software development. Git's distributed nature, efficient branching model, and strong community support make it an essential tool for maintaining code quality, facilitating collaboration, and ensuring project integrity. In summary, version control systems, exemplified by Git, are indispensable tools in software development. Their role in managing code changes, supporting collaboration, and ensuring project stability makes them a cornerstone in the software development lifecycle