What is a version control system?

A version control system or VCS, also know as revision control or source control system, is a software utility that tracks and manages changes to a filesystem. A VCS also offers collaborative utilities to share and integrate these filesystem changes to other VCS users. When operating at the filesystem level, a VCS will track the addition, deletion, and modification actions applied to files and directories. A repository is a VCS term which describes when VCS is tracking a filesystem. In the scope of individual source code files, a VCS will track additions, deletions, modifications of the lines of text within that file. Popular software industry VCS options include Git, Mercurial, SVN and preforce.

## Why do I need version control software?

VCS is an invaluable tool with many benefits to a collaborative software team workflow. Any software project that has more than one developer maintaining source code files should absolutely use a VCS. Additionally, sole-maintainer projects will also greatly benefit from utilizing a VCS. There is arguably no valid reason to forgo the use of a VCS in any modern software development project.

#### Conflict resolution

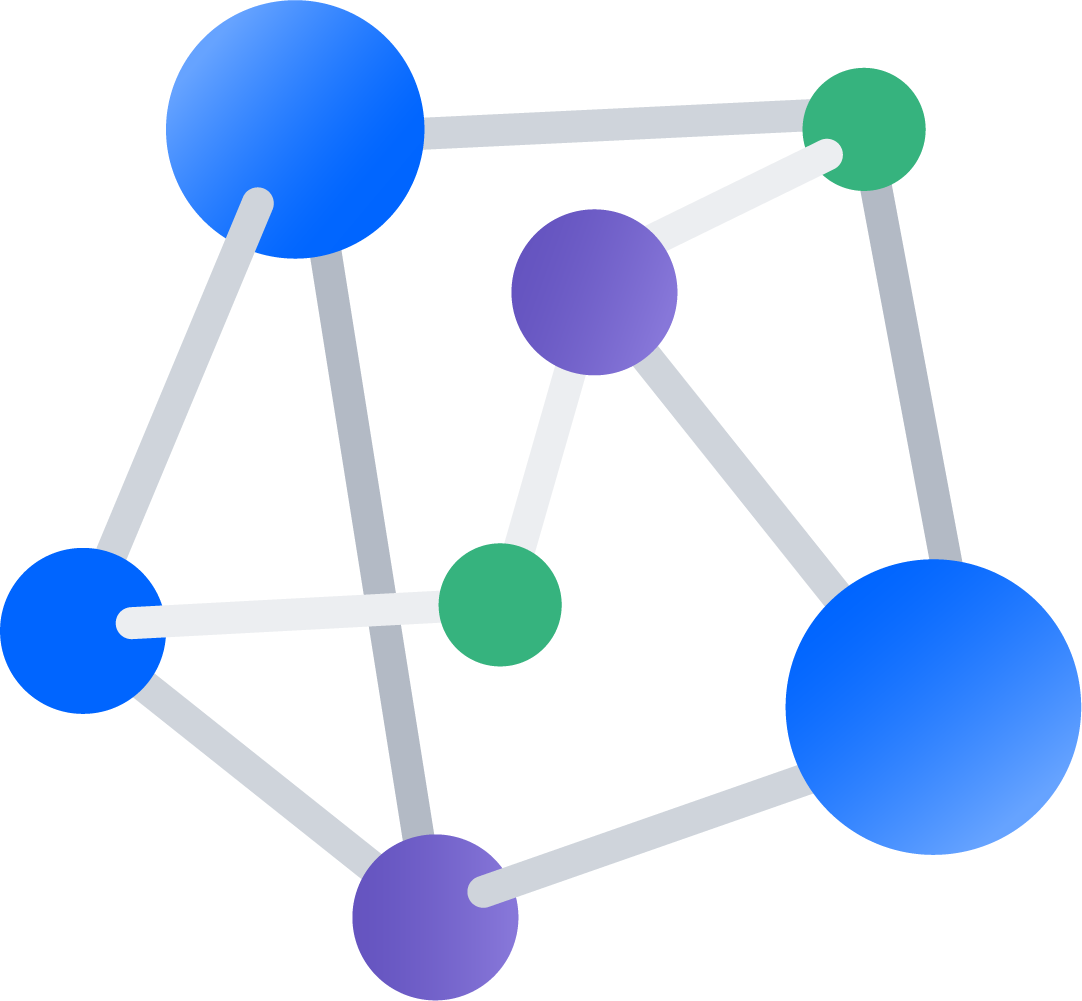
During the life-cycle of a team driven software project, it is most likely that multiple team members will have the need to make changes to the same source code file at the same time. A VCS will tracks and aids in conflicts between multiple developers. These conflict resolution operations leave an audit trail that provides insight into the history of a project.

#### Rollback and undo changes to source code

Once a VCS has begun tracking a source code file system, it keeps a history of changes and the state of the source code throughout a project’s history. This enables the possibility to “undo” or rollback a source code project to a last well-known state. If a bug is discovered in a live application, the code can be quickly reverted to a known stable version.

#### Offsite source code backup

When using a VCS collaboratively, a remote instance of the VCS needs to be created to share changes between developers. This remote VCS instance could be hosted off-site with a trusted third-party like Bitbucket. It then becomes a secure offsite backup. In an unfortunate scenario like a stolen laptop, the remote VCS instance will still retain a copy of the source code.



##### DISTRIBUTED

##### CENTRALIZED

## VCS tools come in two primary types of remote architecture. These architecture types are centralized and distributed.

When discussing the pros and cons of each architecture, the off-site backup feature is the primary discussion point. A centralized VCS has a single point of failure, which is the remote central VCS instance. If this instance is lost, it can cause productivity and data loss, and it will need to be replaced with another copy of the source code. If it temporarily becomes unavailable, it will prevent developers from pushing, merging or rolling-back code. A distributed model architecture avoids these pitfalls by keeping a full copy of the source code at each VCS instance. If any of the previously mentioned centralized failure scenarios happen within the distributed model, a new VCS instance can be swapped in to lead development mitigating any serious drop in productivity.

## The benefits of version control tools

Integrating a VCS into a software development project enables a variety of organizational and management benefits.  By default, a VCS on its own offers the previously discussed technical benefits of team conflict resolution and collaboration aides. A hosted VCS service wraps a default VCS and gives enhanced features. This 'enhanced VCS' is incredibly powerful and gives a transparent view into the software development process, which can traditionally be an opaque creative endeavor. The following points are some high-level benefits a hosted VCS offers.

#### Extended 3rd party integrations

A VCS can be further enhanced when integrated with a secure 3rd party hosted solution like Bitbucket. A hosted VCS offers its own further extensions to other apps and services, through integrations like the Atlassian Marketplace. For example, a VCS hosted with Bitbucket will have the ability to share VCS data with Jira task tracking software, and the Confluence knowledge base documentation platform. These are only two examples, from a world of integrations that a hosted VCS can offer.

#### Team communication

A hosted VCS provides extended messaging and forum-like communication tools, which can be invaluable aides in team collaboration. These messaging tools enable clear threaded discussion of line-by-line source code development. The messaging tools are not restricted to source code discussion and can be invaluable tools to promote cross team discussion on high-level business goals related to source code development.

#### Insight, measurement and accountability

An engineering based organization can use a hosted VCS to track and measure key performance indicators (KPIs). A hosted VCS will give insight into the speed and efficiency a software development project is operating within. A hosted VCS provides tools to capture measurements like: new features deployed, code creation velocity, time in development, bugs created and fixed, and more.

#### CI/CD pipeline automation

Continuous delivery workflows can be constructed through the use of hosted VCS external integrations. A developer initiates a transaction to integrate their source code edits into the larger code base. During this process the hosted VCS can be configured with many extensions to help streamline the code review and delivery process. Things like automated builds, tests, and deployments can be orchestrated in a simple config-as-code YAML file.