Software version control

Zolkifli et al. (2018) classifies Version control system into two subgroups- CVCS (Centralized version control system) and DVCS (Distributed version control system). The main difference between CVCS and DVCS is that CVCS has a centralized repository whereas DVCS has centralized repositories along with local repository for each user. CVS and Subversion are some of the most common tools used for CVCS. Some of the tools which has moved to DVCS are GIT, Mercurial, Bazaar and Bitkeeper. Zolkifli et al. (2018) also mentions that other reason of software/tools moving to DVCS is that it allows users to work offline unlike CVCS where users' has to be connected to a network in order to access any repository.

DVCS also requires a network connection but only when sharing the repository with other team members.Distributed Version Control Systems also provides accessibility to work from different locations and timezone whereas Centralized Version Control Systems only provides you access from a single site. Apart from all the differences, the most interesting characteristic DVCS offers is that it is suitable for single user or multiple users working on a project while CVCS is only compatible for team members or multiple users.

Software version control, according to Ruparelia (2010), also known as Revision control system (RCS) allows developers to obtain and maintain all the historic source codes and files at the backend which can be retrievable when required. It can be described as a software which can keep track of all the steps taken towards building a code, which can be altered/retrieved by the developer if changes are required. According to Loeliger and McCullough (2012), one of the best feature version control offers is tracking the changes made by all the users working on on a

shared project entailing what changes were made and why. All these changes are later saved under the changes log as all the version control software does not allow changes to be made without administrative rights/permissions, ensuring accountability to the changes made to any file as Tichy (1985) also identifies maintaining the software with ongoing/latest updates while working on a project as a continuous challenge for any software developer. One of the best examples of such software is GIT. Some of the common features version control has been as follows-

1. Changes can be reversed.
2. On the go modifications can be synchronized and updated to the working directory/file.

Advantages-

1. VCS (Version control software) provides freedom of working on any project by allowing users to work simultaneously on a same project without overlapping each other's work.
2. Version control software merges all the different changes made to a file,
3. It also triggers a warning if the changes are similar or conflicting in nature, allowing users to review before committing those changes.
4. Improves efficiency by auto saving changes and updating the latest version of the software.

5. A new file is saved every time changes are committed to a working file allowing users to keep the previous version with a brief message about the changes made, which can provide a thorough explanation to the team members.
6. Provides accessibility to always go back to the previous version of the code.
7. Because of it's centralized nature, VCS also helps saving on some disk space on the local and server level.

According to Spinellis (2005) some of the best practices for VCS are as follows-

1. Using the VCS as personal project by backing up favorite projects with your homepage.
2. Version control can be used more than just as a source code. It can be further utilized for storing help files, documents, codes,GUI elements, etc,.
3. It's better to name your files using start file names and VCS can often get confused with similar file names, which can result in either losing file's revision history or older versions.
4. When all releases are labelled, as having an unique label can have releases identified during bug fixes.
5. Following policies and procedures is a safe practice when it comes to VCS as it affects all developers.

References

Loeliger, J., & McCullough, M. (2012). *Version control with git: Powerful tools and techniques for collaborative software development*. " O'Reilly Media, Inc.".

Ruparelia, N. B. (2010). The history of version control. *SIGSOFT Softw. Eng. Notes*, *35*(1), 5–9. https://doi.org/10.1145/1668862.1668876

Spinellis, D. (2005). Version control systems. *IEEE Software*, *22* (5), 108–109. https://doi.org/10.1109/MS.2005.140

Tichy, W. F. (1985). RCS—a system for version control. *Software: Practice and Experience*, *15*(7), 637–654.

Zolkifli, N. N., Ngah, A., & Deraman, A. (2018). Version control system: A review. *Procedia Computer Science*, *135*, 408–415.