**Version Control Guidelines**

Liz Hinz

Professor Nathan Braun

CSD380-A339: DevOps

June 15, 2025

**Version Control Guidelines**

Version control is a pivotal part of software development, or at least should be. Version control means that all software changes are tracked and managed. We should implement version control for many reasons, since we can return to a previous working state if a system crashes, and identify who is making changes and why. Different sources have recommended guidelines for version control.

According to GeeksforGeeks (2025), version control manages and tracks changes to "code, documents, and other files". Because of version control, developers can work on a project at the same time as other developers (GeeksforGeeks, 2025). There are two types of version control systems: centralized and distributed. Centralized version control consists of one global repository where code changes are committed, and to view changes, the code is updated from the repository (GeeksforGeeks, 2025). Distributed version control has several repositories, so changes must be committed, pushed, pulled, and updated (GeeksforGeeks, 2025).

Ethan Lott on Filestage provides his recommended guidelines when working with version control. An additional benefit of version control, when properly used, means there is no confusion on which version is the final one (Lott, 2025). Lott (2025) provides four best practices for version control: "define naming conventions to make sure everyone knows which version is the latest, draw a clear line between versions so everyone knows when to stop giving feedback, collect feedback and collaborate in one place to avoid version duplication, and make it easy for reviewers to compare versions so they can easily see what has changed".

Last year, Nimrod Kramer on Daily.dev provided his best practices for version control. Kramer's (2024) best practices for version control are "set up a clear version control plan, make good commits with clear messages, use branches for different features or releases, implement regular reviews and teamwork, keep documentation in sync with code changes, user continuous integration for automated updates, implement security measures and access controls, and regularly back up and test recovery processes".

Brent Schiestl compiled eight version control best practices on Perforce. The best practices for version control include: "commit changes atomically, commit files with a single purpose-not as a backup, write good commit messages, don't break builds, do reviews before committing to a shared repository, make sure every commit is traceable, following branching best practices, and protect your assets" (Schiestl, 2020).

There were many similarities and differences when comparing the guidelines from Lott, Kramer, and Schiestl. All the sources agree that there should be a plan in place, all changes should be tracked, regular commits with clear commit messages, and collaboration and communication, like reviews and gathering feedback. All the guidelines appear to still be relevant today. Lott is the only source that suggested focusing on feedback and clarity in review cycles. Only Kramer mentioned continuous integration for automated updates, including access control and backup/recovery. Schiestl is the only source that emphasized atomic commits, single-purpose commits, and traceability.

Based on the guidelines mentioned in the articles and previous knowledge, some guidelines are more important than others. These guidelines are: defining naming conventions, collecting feedback, collaborating with others on new versions, and using branches for different features or releases. The correct version must be used when reviewing, working with, or implementing it, which proper naming conventions help with. Having feedback from other developers makes it easier to find and remediate any errors within the code. Collaboration can help identify issues and make improvements to the newest versions. Using branches for different features or releases helps keep everything separate and easier to work on, especially with others.

When developing code, version control should be used throughout the entire process. Implementing guidelines is the best practice to help maximize the benefits of version control. With proper version control guidelines, development runs smoother for everyone, and updating code becomes more manageable.

**References**

GeeksforGeeks. (2025, March 6). *Version Control Systems*. GeeksforGeeks. https://www.geeksforgeeks.org/version-control-systems/#

Kramer, N. (2024). *Documentation Version Control: Best Practices 2024*. Daily.dev. https://daily.dev/blog/documentation-version-control-best-practices-2024

Lott, E. (2025, June 5). *Document Version Control Best Practices to Make Your Life Easier*. Filestage.io. https://filestage.io/blog/document-version-control/

Schiestl, B. (2020, May 21). *8 Version Control Best Practices*. Perforce Software. https://www.perforce.com/blog/vcs/8-version-control-best-practices