Assignment 3.2 Version Control Guidelines

**Introduction**

Version Control Guidelines are a way to create a standardized method for making changes to a system. Having guidelines ensures that all commits or changes to a system are easily traceable not only in impact but also by who published or created the change (Perforce, n.d.). This allows for low friction collaboration across teams, and maintains a level of integrity of the work as it evolves and changes over time (Ernst, M.D. n.d.).

**Comparing Guidelines**

#### **1. Perforce Software**

Focuses on software development and provides practical rules for managing codebases (Perforce, n.d.):

* Use a consistent branching strategy.
* Protect the main branch (e.g., main or master) from unstable changes.
* Keep workflows simple and consistent across teams.
* Commit often but meaningfully.
* Review changes before merging.

#### **2. University of Washington**

Targets student developers (Ernst, M.D. n.d.):

* Commit logical units of work rather than partial or unrelated changes .
* Write clear, descriptive commit messages.
* Pull updates often to stay in sync with collaborators.
* Avoid committing generated or build files.
* Use version control early and frequently in projects.

#### **3. NCCIH**

Geared toward versioning in research documentation, not code (NCCIH, 2015):

* Assign version numbers manually (e.g., Draft v0.1, Final v1.0).
* Include version history and authorship in document headers.
* Maintain a change log to describe updates.
* Store documents in a consistent folder structure by version.
* Archive outdated versions for traceability.

**Relevance of Older Guidelines**

The guidelines offered by the NCCIH article are largely outdated in a world dominated by version control software like Git. For example, guidelines like manually writing version numbers (NCCIH, 2015). However, the principles behind Git are largely here in the sense of maintaining clear version history with authorship, clear change log to describe updates, and storing documents in a consistent folder structure (NCCIH, 2015). These principles are largely the underlying ideas behind how Git works especially in a team setting.

**My Version Control Guidelines**

Based on the research and personal development experience, I consider these six version control guidelines the most essential:

1. **Commit Logical Changes** Each commit should represent a complete, understandable change. This simplifies debugging and code review (Perforce, n.d.).
2. **Write Descriptive Commit Messages** A good message explains what was changed and why. This is invaluable for teammates and also your future self (NCCIH, 2015).
3. **Use a Clear Branching Strategy** Whether using Git Flow, trunk-based development, or a simple feature-branch model, a shared structure avoids confusion (Ernst, M.D. n.d.).
4. **Pull and Merge Frequently** Staying updated with teammates’ changes reduces the risk of painful merge conflicts and keeps your code relevant (Perforce, n.d.).
5. **Don’t Commit Build or Generated Files** This keeps the repository clean and ensures portability across systems. Also reduces size of repository when cloning (Ernst, M.D. n.d.).
6. **Protect the Main Branch** Use pull requests and code reviews before merging to main. It keeps the production ready code stable and tested (Perforce, n.d.).

**Conclusion**

Version control is not just a technical requirement for software teams it is a communication tool that connects teams together. Furthermore, version control documents the intent and reasons behind changes, and ensures quality both in changes and documentation. Modern version control software like what is mentioned in the Perforce article and the University of Washington empowers teams to create scalable and collaborative workflows.

**Reference :**

Ernst, M. D. (2024, April 8). *Version control concepts and best practices*. University of Washington. Retrieved April 6, 2025, from<https://homes.cs.washington.edu/~mernst/advice/version-control.html>

NCCIH. (2015, July). *Version control guidelines* (Version 2). U.S. Department of Health & Human Services. Retrieved April 6, 2025, from <https://files.nccih.nih.gov/s3fs-public/CR-Toolbox/Version_Control_Guidelines_ver2_07-17-2015.pdf>

Perforce Software. (n.d.). *8 version control best practices*. Perforce. Retrieved April 6, 2025, from<https://www.perforce.com/blog/vcs/8-version-control-best-practices>