Brian Preston

CSD-380/ Module 3.26

6/15/2025

**Version Control Guidelines**

When I first started learning about version control, I started to realize just how essential it would become, not just for teams but for beginner developers like myself. After researching multiple sources on the topic, I have certainly started to appreciate that great version control is less about the tool and more about the habits you develop around it. In this paper, I am going to compare version control guidelines from three expert sources and share my own take on the most important ones.

**Comparing the Guidelines**

The first article I reviewed, *Version Control Concepts and Best Practices* by Michael Ernst (2024), emphasizes the foundational and philosophical aspects of version control, especially for distributed systems like Git. Ernst outlines best practices such as using clear commit messages, making each commit a logical unit, and not committing generated files. One quote that stood out to me was:

“It only takes a moment to write a good commit message... useful when someone is looking for changes related to a given concept” (Ernst, 2024).  
I think this is such an overlooked habit, and I’ve already started changing how I write my own commit messages because of it.

The second article, *Document Version Control Made Easy* by Elizabeth Harrin (2024), focuses more on project documentation rather than code, but the principles still apply. Harrin talks about the importance of having a version control strategy and naming conventions. She says,

“Version control provides a way of tracking changes so you know who changed what and when” (Harrin, 2024).  
I feel like this quote really hits home, especially when I’m working on shared documents or trying to remember what I changed last week.

The third source, Brent Schiestl’s blog post *8 Version Control Best Practices* (2020), covers a broad range of technical do’s and don’ts. He recommends atomic commits, traceability, and pre-commit reviews. A quote I appreciated from Schiestl was:

“Good-quality commits will improve your project, making you more productive and successful” (Schiestl, 2020).  
I think this reinforces how much version control is tied to productivity—it’s not just about backup, it’s about working smarter.

**Less Relevant Guidelines I have found**

While all three sources provide solid advice, I did notice that centralized version control (like SVN) was still mentioned in some detail, especially in Ernst’s article. In my opinion, centralized systems are rarely used in modern development environments. Most teams—including the one I hope to be part of someday—use Git or another distributed system. So I think guidelines specifically for SVN or older systems might be less relevant today.

**My “Top” Version Control Guidelines**

After comparing these sources, here’s my own personal list of version control guidelines that I think are the most important, and why

1. **Write Clear Commit Messages**  
   I think this is one of the simplest yet most impactful practices. It saves so much time when revisiting your code later or trying to understand a teammate’s work.
2. **Commit Changes with a Single Purpose**  
   I feel this helps keep the history clean and logical. Schiestl really drove this point home, and I’ve already seen how messy things get when a commit tries to do too many things at once.
3. **Pull and Push Frequently**  
   Ernst emphasized how frequent sharing reduces conflicts. I think this is especially important in collaborative projects. I have been in situations where I forgot to pull, and resolving the conflicts later was frustrating.
4. **Do not Commit Generated Files**  
   I feel this keeps the repository clean and easier to manage. It also avoids unnecessary merge conflicts.
5. **Use Branches for Features and Fixes**  
   From Schiestl’s article, I learned how structured branching policies can reduce confusion. I think this is something I’ll lean into more as I work on bigger projects.
6. **Review Before Committing to Main**  
   I think peer reviews—or even self-reviews—help catch errors and improve overall quality. This aligns with the professional workflow I want to build for myself.
7. **Coordinate With Your Team**  
   Ernst suggests checking in before making major changes, and I feel this is respectful and practical. Communication saves everyone time.

Learning about version control has changed how I approach not just code, but all types of collaborative work. I think it’s one of those skills that pays off the more intentionally you practice it. After reading these sources, I feel more confident in my ability to manage projects responsibly and efficiently. Whether it is through clear commits, good coordination, or consistent branching, version control helps bring structure and clarity to the development process overall.

**References**

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

Harrin, E. (2024, August 31). *Document version control made easy*. Rebel’s Guide to Project Management. <https://rebelsguidetopm.com/how-to-do-document-version-control/#why-is-version-control-important>

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