Andres Melendez

CSD 380

1 / 24 / 2025

**Version Control Guidelines: Research and Analysis**

Version control is critical to managing projects, ensuring that documents, code, and assets are well-organized, trackable, and recoverable. By researching the version control guidelines from Filestage, GitLab, and Daily.dev, this paper compares and contrasts their recommendations, identifies outdated policies, and proposes a refined set of essential practices for modern usage.

**Comparison of Guidelines**

**Filestage**

Filestage emphasizes the importance of clear naming conventions, regular backups, and centralized document storage systems. Their approach minimizes confusion and prevents version conflicts, particularly in collaborative environments. A notable recommendation is using timestamps and descriptive names for file versions, such as project-name\_v1\_2025-01-01.docx.

**GitLab**

GitLab’s version control best practices are tailored for software development, emphasizing branching strategies, consistent commit messages, and regular reviews. The guidelines highlight the benefits of feature branches to isolate development work and the importance of merging smaller, incremental changes instead of large, monolithic updates. GitLab also recommends automated testing and Continuous Integration/Continuous Deployment (CI/CD) pipelines to ensure quality.

**Daily.dev**

Daily.dev’s guidelines focus on documentation-specific version control. Their recommendations include maintaining a single source of truth for documentation, establishing a clear version history, and creating robust access controls to prevent unauthorized changes. The guidelines stress the importance of testing recovery processes and using tools like Git to manage large-scale documentation projects.

**Contrast and Relevance**

While all three sources agree on the necessity of systematic organization and robust tracking, the approaches vary by domain:

* **Filestage** targets collaborative document management with an emphasis on human-readable file naming.
* **GitLab** prioritizes software development with an advanced focus on branching, CI/CD, and code reviews.
* **Daily.dev** bridges the gap between documentation and development by emphasizing accessibility and recovery testing.

Guidelines like manual file backups, commonly suggested in Filestage, may be less relevant today due to the widespread use of cloud-based solutions with automatic backups. Similarly, some legacy version control systems (e.g., Subversion) are less applicable in an era dominated by Git-based tools.

**Proposed Essential Guidelines**

Based on this research, the following are the most critical version control practices:

1. **Use a Centralized Version Control System (e.g., Git):** A centralized system ensures collaboration, version tracking, and rollback capabilities.
2. **Establish Clear Naming Conventions:** Descriptive and consistent file or branch names make it easier to identify versions and changes.
3. **Leverage Branching Strategies:** Feature or topic branches minimize conflicts and allow parallel development.
4. **Write Detailed Commit Messages:** Accurate and concise messages help trace changes and understand the evolution of a project.
5. **Implement Automated Testing and CI/CD Pipelines:** Automation ensures that changes are tested and deployed consistently.
6. **Create a Recovery Plan:** Regularly test backup and recovery processes to ensure data is retrievable in emergencies.
7. **Enforce Access Controls:** Limit editing permissions to authorized users to maintain security and accuracy.

**Rationale for Selection**

These guidelines were chosen because they address the needs of modern workflows, emphasizing collaboration, automation, and security. For instance, centralized systems like Git have become industry standards, rendering older manual processes obsolete. Automated testing and CI/CD pipelines reduce human error, while clear naming conventions and commit messages improve traceability. Recovery plans and access controls safeguard data integrity, ensuring a seamless workflow.

By combining insights from diverse sources and prioritizing adaptability, these guidelines are a robust foundation for version control in today’s dynamic environments.

**References**

(15 July 2024). *Documentation Version Control: Best Practices 2024*. Daily.dev. https://daily.dev/blog/documentation-version-control-best-practices-2024#:~:text=Set%20up%20a%20clear%20version,up%20and%20test%20recovery%20processes

(2 January 2025). *Documentation version control best practices and examples*. Filestage. https://filestage.io/blog/document-version-control/

GitLab. (n.d.). *What are Git version control best practices?* https://about.gitlab.com/topics/version-control/version-control-best-practices/