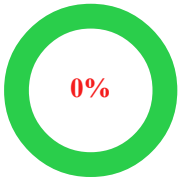
SmallSEOTools

Plagiarism Detection Report by SmallSEOTOOLS



Plagiarism	0%	Partial Match	0%
Exact Match	0%	Unique	100%

Scan details

Total Words	Total Characters	Plagiarized Sentences	Unique Sentences
1025	**7464**	**0**	**66 (100%)**

#1 100% Unique

Document and Software Configuration Management

Comparative Study of SVN and Git/GitHub with Version 1 and Version 2 Release Analysis

1. Introduction

Software Configuration Management (SCM) is a set of processes, policies, and tools used to manage interim and final artifacts of software development, including source code, documentation, testing files, builds, and release-related materials. Throughout the software development lifecycle, several modifications and enhancements are introduced. Without proper control mechanisms, these changes may lead to confusion, inconsistencies, and defects in the final product. SCM ensures that all artifacts are systematically controlled, tracked, and documented.

SCM mainly focuses on four important activities: version control, build management, release management, and change management. These activities help maintain the integrity, traceability, and reliability of software systems. The primary goal of configuration management is to minimize confusion and maximize productivity by reducing mistakes and ensuring that every team member works with the correct and updated version of the software.

Various tools are available to implement SCM practices. Among them, Apache Subversion (SVN) and Git/GitHub are widely used. SVN follows a centralized approach, whereas Git/GitHub uses a distributed model with strong collaboration and automation support. This report presents a detailed study and comparison of these tools and explains their usage through two software releases, namely Version 1 and Version 2.

2. Project Scenario

To understand the role of SCM clearly, a Student Management System (SMS) is considered. The system is developed for managing student information such as registration, attendance, marks, and reports.

The project was first developed using SVN and released as Version 1. After facing several development and maintenance issues, the team migrated to Git/GitHub and released an improved Version 2. The differences between these two versions are analyzed based on the four SCM components.

3. Version Control

Version control is the foundation of SCM. It manages modifications made to software artifacts and keeps a complete history of changes. It enables multiple developers to work simultaneously while preventing conflicts and loss of data.

In Version 1, SVN was used as the version control system. Since SVN follows a centralized repository structure, all developers depended on a single server for committing and retrieving files. Every change required an active network connection. If the server was unavailable, development activities stopped. Branching and merging were complicated processes, making parallel development difficult. As the team size increased, conflicts became frequent and productivity decreased.

In Version 2, Git/GitHub replaced SVN with a distributed version control approach. Each developer maintained a complete copy of the repository locally. Changes could be committed offline and later synchronized with the central repository. Branching became lightweight, allowing developers to create separate branches for each feature. GitHub also provided pull requests for reviewing and merging changes safely. This approach significantly improved collaboration, reduced conflicts, and increased development speed.

Thus, Version 2 demonstrated better reliability, flexibility, and performance compared to Version 1 in terms of version control.

4. Build Management

Build management refers to the process of compiling source code, resolving dependencies, testing functionality, and generating executable software. Proper build management ensures consistency and reduces integration problems.

During Version 1, builds were performed manually. Developers compiled code on their local machines and created executable files separately. There was no standardized or automated build process. As a result, builds often differed across systems, and missing dependencies caused failures. Manual testing also delayed defect detection, increasing debugging time.

With Version 2, automated build systems were introduced using GitHub's CI/CD tools. Every time code was committed or merged, the system automatically compiled the project and executed tests. Errors were identified immediately, and developers could fix them quickly. Automated builds ensured consistency and reliability across environments. This reduced manual effort and improved overall software quality.

Therefore, Version 2 achieved faster, more stable, and consistent builds compared to Version 1.

5. Release Management

Release management involves packaging, scheduling, and deploying stable versions of the software to users. It ensures that releases are reliable and properly documented.

In Version 1, releases were prepared manually. Developers created build packages, copied files, and deployed them without automation. Tagging of versions was limited, and rollback to previous releases was difficult. This process consumed time and was prone to errors.

In Version 2, release management became structured and automated. Git/GitHub enabled tagging of versions, automatic packaging, and deployment pipelines. Each release was documented with release notes and version labels. If any issue occurred, the team could easily revert to earlier stable versions. Automation improved speed and reliability while reducing human errors.

Hence, Version 2 provided a smoother and more efficient release process.

6. Change Management

Change management controls and monitors modifications to software artifacts. It ensures that every change is requested, reviewed, tested, and approved before integration.

In Version 1, change tracking relied only on commit messages. There was no structured review process or integrated issue tracking system. As a result, identifying the cause of defects was difficult. Lack of transparency reduced accountability among team members.

Version 2 introduced GitHub's issue tracking and pull request mechanisms. Every change was linked to a specific issue and underwent review before merging. Discussions, comments, and approvals were recorded, providing a complete audit trail. This improved accountability and ensured higher code quality.

Thus, change management became more controlled and transparent in Version 2.

7. Version 1 vs Version 2 Release Comparison

Version 1 delivered the initial functional system with basic capabilities such as student registration, attendance entry, and marks management. However, it suffered from manual processes, slower development, and limited traceability.

Version 2 introduced additional features including login authentication, role-based access, automated calculations, dashboards, and advanced reporting. More importantly, it improved the development workflow through modern SCM practices. Automation, structured reviews, and better collaboration significantly enhanced system quality and maintainability.

8. Comparison Tables

SCM Tool Comparison (SVN vs Git/GitHub)

Version 1 vs Version 2 Comparison

9. Conclusion

Software Configuration Management plays an essential role in maintaining the integrity and reliability of software systems. Proper control of versions, builds, releases, and changes ensures smooth development and minimizes errors.

The comparison between Version 1 and Version 2 clearly shows the impact of adopting modern SCM practices. Version 1, managed using SVN and manual processes, experienced several limitations such as slower
Lorem ipsum dolor sit amet consectetur. Ut enim mauris at vel mi mauris sagittis. Arcu fames lectus habitasse feugiat suspendisse. Ipsum volutpat ornare placerat sit quis semper dui pharetra. Vestibulum a ipsum aenean nisi dictum tempor. Lacinia pharetra donec aliquam egestas lectus ut turpis. Sapien quam urna in quis vivamus pretium ultrices ac hac. Elementum sit nisl elit tincidunt tortor. Adipiscing aenean mattis sit enim nibh imperdiet