SISTEMA DE CONTROL DE VERSIONES

¿QUÉ ES UN SISTEMA CONTROL DE VERSIONES?

Un sistema de control de versiones es una herramienta utilizada en el desarrollo de software para evitar el riesgo de conflictos que puedan surgir al trabajar en colaboración con otros equipos de desarrollo.

ARQUITECTURA

Locales: Los cambios son guardados localmente y no se comparten con nadie. Esta arquitectura es la antecesora de las dos siguientes.
Centralizados: Existe un repositorio centralizado de todo el código, del cual es responsable un único usuario (o conjunto de ellos). Se facilitan las tareas administrativas a cambio de reducir flexibilidad, pues todas las decisiones fuertes (como crear una nueva rama) necesitan la aprobación del responsable. Algunos ejemplos son CVS y Subversion.
Distribuidos: Cada usuario tiene su propio repositorio. Los distintos repositorios pueden intercambiar y mezclar revisiones entre ellos. Es frecuente el uso de un repositorio, que está normalmente disponible, que sirve de punto de sincronización de los distintos repositorios locales. Ejemplos: Git y Mercurial.

VENTAJAS Y DESVENTAJAS

Permite que varias personas trabajen simultáneamente en un solo proyecto. Cada persona edita su propia copia de los archivos y elige cuándo compartir esos cambios con el resto del equipo. Por lo tanto, las ediciones temporales o parciales de una persona no interfieren con el trabajo de otra.
Permite a una persona usar varios ordenadores para trabajar en un proyecto, por lo que es valioso incluso si se está trabajando solo.
Integra el trabajo realizado simultáneamente por diferentes miembros del equipo. En la mayoría de los casos, las ediciones de diferentes archivos o incluso el mismo archivo se pueden combinar sin perder trabajo.
Da acceso a versiones históricas de su proyecto. Este es un seguro contra fallos informáticos o pérdida de datos. Si se comete un error, se puede volver a una versión anterior.
Cuando hablamos de los pros y los contras de cada arquitectura, la función de copia de seguridad externa es el principal punto de debate. Un VCS centralizado cuenta con un solo punto de error, que es la instancia remota del VCS central. Si se pierde dicha instancia, puede producir la pérdida de datos y productividad, y se deberá sustituir por otra copia del código fuente. Si se vuelve temporalmente no disponible, evitará que los desarrolladores envíen, fusionen o reviertan código. Un modelo distribuido de arquitectura evita estos obstáculos manteniendo una copia total del código fuente en cada instancia de VCS. Si se produce en el modelo distribuido cualquiera de los casos de error centralizados antes mencionados, se puede introducir una instancia de VCS al principal mitigando cualquier caída grave de productividad.

TIPOS DE SISTEMAS DE VERSIONES

Sistema de Control de Versiones Local: El sistema de control de versiones local mantiene un seguimiento de los archivos dentro del sistema local. Este enfoque es muy común y simple. Este tipo también es propenso a errores, lo que significa que las posibilidades de escribir accidentalmente en el archivo incorrecto son mayores.
Sistema de Control de Versiones Centralizados: En este enfoque, todos los cambios en los archivos se rastrean en el servidor centralizado. El servidor centralizado incluye toda la información de los archivos versionados y la lista de clientes que extraen archivos desde ese lugar central.
Sistema de Control de Versiones Distribuido: Los sistemas de control de versiones distribuidos aparecen para superar el inconveniente del sistema de control de versiones centralizado. Los clientes clonan completamente el repositorio, incluido su historial completo. Si algún servidor está inactivo o desaparece, cualquiera de los repositorios del cliente se puede copiar en el servidor para restaurarlo. Cada clon se considera una copia de seguridad completa de todos los datos.

REPOSITORIOS

Git: Es un sistema de control de versiones distribuido escrito en una combinación de Perl, C y varios scripts de shell, estuvo diseñado por Linus Torvalds según las necesidades del proyecto del núcleo Linux; con los requisitos de descentralización, rápido, flexible y robusto. Subversion.
Apache Subversion (SVN): Es un sistema de control de versiones centralizado de código abierto bajo la licencia de Apache. Sus características clave incluyen administración de inventario, administración

de seguridad, seguimiento del historial, controles de acceso de usuarios, recuperación de datos y administración del flujo de trabajo. SVN es fácil de implementar con cualquier lenguaje de programación. Además, ofrece almacenamiento uniforme para manejar archivos de texto y binarios.
Mercurial: Es un sistema de control de versiones distribuido escrito en Python como un recambio en software libre de Bitkeeper; descentralizado, que pretende ser rápido, ligero, portable y fácil de usar. Además, posee una interfaz web integrada. Monotone
Monotone: Es un sistema de control de versiones distribuido escrito en C ++ . Los sistemas operativos que admite incluye Unix, Linux, BSD, Mac OS X y Windows. Brinda un buen apoyo para la internacionalización y localización. Además de funcionar con protocolo P2P.

ENTORNOS GRÁFICOS VENTAJAS Y DESVENTAJAS

Git: es una de las mejores herramientas de control de versiones disponible en el mercado actual. Es un modelo de repositorio distribuido compatible con sistemas y protocolos existentes como HTTP, FTP, SSH y es capaz de manejar eficientemente proyectos pequeños a grandes.
CVS: es otro sistema de control de versiones muy popular. Es un modelo de repositorio cliente-servidor donde varios desarrolladores pueden trabajar en el mismo proyecto en paralelo. El cliente CVS mantendrá actualizada la copia de trabajo del archivo y requiere intervención manual sólo cuando ocurre un conflicto de edición.
Apache Subversion (SVN): abreviado como SVN, apunta a ser el sucesor más adecuado. Es un modelo de repositorio cliente-servidor donde los directorios están versionados junto con las operaciones de copia, eliminación, movimiento y cambio de nombre.
Mercurial: es una herramienta distribuida de control de versiones que está escrita en Python y destinada a desarrolladores de software. Los sistemas operativos que admite son similares a Unix, Windows y macOS. Tiene un alto rendimiento y escalabilidad con capacidades avanzadas de ramificación y fusión y un desarrollo colaborativo totalmente distribuido. Además, posee una interfaz web integrada.
Monotone: está escrito en C ++ y es una herramienta para el control de versiones distribuido. El sistema operativo que admite incluye Unix, Linux, BSD, Mac OS X y Windows. Brinda un buen apoyo para la internacionalización y localización. Además, utiliza un protocolo personalizado muy eficiente y robusto llamado Netsync

Bibliografía:

https://www.campusmvp.es/recursos/post/que-es-git-ventajas-e-inconvenientes-y-por-que-deberias-aprenderlo-bien.aspx
https://aulasoftwarelibre.github.io/taller-de-git/cvs/
https://bitbucket.org/product/es/version-control-software
https://www.drauta.com/5-softwares-de-control-de-versiones