Sistema de control de versiones

Gamaliel Silva López

9 de mayo de 2022

Resumen

1. Introducción

Durante el desarrollo de proyectos siempre se tienen diferentes opciones al momento de como poder tener algún respaldo de versiones tanto localmente hasta de forma que tengamos un software que nos ayude a gestionar nuestras versiones como lo son GitHub en este documento se hablara sobres las diferentes características con las que cuenta, su arquitectura y tipos de sistemas de control de versiones. El control de versiones hacer que nuestro de desarrollo de software sea mas eficiente al momento de poder gestionar las diferentes versiones y en el peor de las situaciones puede ayudarnos a poder regresar a una versión la cual se encuentre libre de errores antes de poder modificar algún componente esto por lo regular es de gran ayuda en proyectos de gran esca labilidad. También es eficiente al momento de querer unir diferentes partes de un mismo proyecto de forma que si se tiene diferentes colaboradores se puede unir de manera fácil y sin tener ningún tipo de complicaciones.

2. Desarrollo

2.1. Características

El control de versiones que tiene algunas características como lo son la práctica de rastrear y gestionar los cambios en el código de software, ayudan a los equipos de software a gestionar los cambios en el código fuente a lo largo del tiempo, realiza un seguimiento de todas las modificaciones en el código en un tipo especial de base de datos.

2.2. Tipos de sistemas de control de versiones

Es una herramienta que permite la automatización de compilación de código abierto. Los scripts de compilación de Gradle se escriben utilizando Groovy o Kotlin DSL (Domain Specific Language). Algunas características de Gradle que podemos destacar son las siguientes:

2.2.1. Sistemas de Control de Versiones Locales

En vez de mantener las versiones como archivos independientes, los almacenaban en una base de datos. Cuando era necesario revisar una versión anterior del proyecto se usaba el sistema de control de versiones en vez de acceder directamente al archivo, de esta manera en cualquier momento solo se tenía una copia del proyecto, eliminando la posibilidad de confundir o eliminar versiones.

2.2.2. Sistemas de Control de Versiones Centralizados

Los sistemas de control de versiones centralizados abordan el problema que impedía que los usuarios invalidaran el trabajo de los demás. Si dos personas editaban el mismo archivo y se presentaba un conflicto alguien debía solucionar este problema de manera manual y el desarrollo no podía continuar hasta que todos los conflictos fueran resueltos y puestos a disposición del resto del equipo.

Esta solución funcionó en proyectos que tenían relativamente pocas actualizaciones y por ende pocos conflictos, pero resulto muy engorroso para proyectos con docenas de contribuyentes activos que realizaban actualizaciones a diario.

2.2.3. Sistemas de Control de Versiones Distribuidos

Tiene la idea de un solo repositorio centralizado y optó por darle a cada desarrollador una copia local de todo el proyecto, de esta manera se construyó una red distribuida de repositorios, en la que cada desarrollador podía trabajar de manera aislada pero teniendo un mecanismo de resolución de conflictos mucho más elegante que un su versión anterior.

2.3. Repositorio de código

Es el lugar en el que se almacena y se puede realizar la distribución del código de una aplicación o un programa. Este debe ser un servidor seguro que utiliza sistemas de control de versiones. Debe contener las diferentes versiones de la aplicación o programa, disponiendo de un historial con los cambios realizados sobre el original y sobre cada nueva versión. Además, debe permitir poder revertir esos cambios. Y permitir que la aplicación o programa pueda ser utilizado en paralelo por diferentes usuarios al mismo tiempo, en la misma o en sus diferentes versiones.

2.3.1. Ventajas de utilizar un repositorio de código.

A continuaciónn se muestra algunas ventajas que son:

- Permite el trabajo en paralelo de dos o más usuarios de una aplicación..
- La seguridad de un repositorio de código en un servidor es máxima.
- disponer y acceder a un historial de cambios.

2.4. Ejemplo de sistema de control de verciones

2.4.1. Git

- Es muy potente.
- Es un software libre..
- Permite el acceso para la totalidad del directorio.

2.4.2. SVN

- Dispone de control de versiones centralizado.
- Se basa en un repositorio central en el que se generan copias de trabajo para los programadores.
- Permite el bloqueo de archivos si es voluntad de su desarrollador.

2.5. Conclusión.

Durante el proceso de elaboración de software se debe elegir la mejor herramienta al momento de trabajar en equipo esto por las diferentes cosas que considera el sistema a desarrollar, pero actualmente surgieron diferentes herramientas que nos facilitan la vida de modo que podemos guardar las diferentes versiones de un proyecto el cual este siendo modificando constantemente esto nos ayudara a poder tener los diferentes versiones de forma que si en futuras modificaciones las cuales ocasión algún tipo de modificación, se podrá acceder a las diferentes versiones anteriores las cuales no cuentan con problemas.

3. Referencias

Referencias

¿Sistemas de Control de Versiones, qué son y por qué amarlos. (2022). Recuperado 9 mayo 2022, de https://medium.com/@jointdeveloper/sistemas-de-control-de-versiones-qu

Qué son los repositorios de código y cuáles son sus beneficios - The Black Box Lab. (2022). Recuperado 9 mayo 2022, de https://theblackboxlab.com/2021/02/22/que-son-los-repositorios-de-codigo-y-cuales-son-sus-beneficios/

Documentation (2.5.0-rc1) — Apache Ivy™ Documentation. (2022). Recuperado 18 abril 2022, de https://ant.apache.org/ivy/history/latest-milestone/index.html