Universidad de El Salvador en línea.

Facultad de Ingeniería y arquitectura.

Escuela de Ingeniería de Sistemas Informáticos

Programación y manejo de datos.

Tema: Herramientas de control de versiones.

Alumno: Gerbert Alexis Hernández Morales HM22051

Andy Steven Rolin Reyes RR23056

Harold Daniel Miranda Zelaya MZ14012

Liliana Abigail Martínez Orellana MO23015

Asignatura: Programación I D.

Grupo teórico 8.

Tutor:

Robinson Vladimir Ruiz Ramírez.

Ciclo II – 2023

Índice

Introducción

En esta investigación, se explora el uso de herramientas de control de versiones en la creación de códigos de programación o códigos fuente y las ventajas y desventajas la utilización de estas. Actualmente existen diferentes herramientas de control de versiones como Git, SVN, Mercurial y CVS por mencionar algunos.

En muchas ocasiones un pequeño cambio en el código puede representar problemas en el funcionamiento del software que se esté desarrollando y encontrar el error en ocasiones se vuelve trabajo de muchas horas. Se propone una la utilización de un código para un programa en el que se demuestre la utilización de las diferentes herramientas de control de versiones.

Palabras clave: Control de versiones, herramientas, código fuente

Objetivos generales

Conocer las diferentes herramientas de control de versiones y las ventajas y desventajas de cada una.

Objetivos Específicos.

- Conocer las opciones de control de versiones que hay en el mercado.
- Adquirir las habilidades para el manejo de herramientas de control de versiones.
- Mostrar las ventajas y desventajas en el manejo de las herramientas

¿Qué es control de versiones?

El control de versiones, también conocido como gestión de código fuente o VCS, usa herramientas para dar seguimiento de las actualizaciones o modificaciones que se han realizado en el código fuente a lo largo del tiempo. El control de versiones permite la colaboración rápida y eficiente entre los desarrolladores y mantiene la integridad del código lo que produce que no exista conflictos en el código.

¿Por qué son importante el control de versiones?

Los sistemas de control de versiones son un pilar esencial de DevOps, ya que permite que los equipos colaboren e iteren el código fuente rápidamente.

Trabajar con un sistema de control de versiones significara que tendrás una copia de seguridad en caso pierdas el código original, las actualizaciones estarán siempre disponible para revisión en caso esta fallara

¿Cómo funciona el control de versiones?

Al crear un nuevo repositorio en tu sistema de control de versiones, se abre lo que se conoce como rama principal o tronco maestro. Aquí es donde la base de código ingresa al canal en el que se compila. Las ramas permiten que los desarrolladores introduzcan los cambios que deseen en su rama del código sin confirmarlos en la rama principal.

A lo largo del tiempo los controles de versión han ido evolucionando, clasificándolos en tres tipos: Sistema de control de versiones Locales, Centralizados y Distribuidos.

Sistemas de control de versiones centralizados.

Los sistemas de control de versiones centralizados utilizan un flujo de trabajo de registro/inserción para conectarse al servidor principal. Los cambios que se realiza se guardan automáticamente en el servidor. Dado que los equipos están vinculados a una sola versión del proyecto almacenado en el servidor, no obstante, aunque esto genera una gran ventaja las interrupciones en el servicio pueden generar demoras importantes.

Sistemas de control de versiones locales.

Esta es la forma mas sencilla de tener un control de versiones y la mas utilizada por desarrolladores independientes ya que se almacena como revisiones dentro de una computadora. El desarrollo de código

fuente en colaboración es un desafío y restaurar información perdida resulta complicado o imposible. Es una buena opción al iniciar como desarrollador, pero a medida se va creciendo lo mejor es optar por otro sistema que permita la colaboración de forma simultánea.

Sistemas de control de versiones distribuido.

Los sistemas de control distribuidos permiten subir el código, crear ramas y fusionarlas sin necesidad de conectarse al servidor principal. Cada desarrollador trabaja desde un repositorio clonado almacenado en la nube. Los sistemas de control de versiones distribuidos pueden presentar largos tiempos de espera en caso necesites descargar todo el historial del proyecto.

Las herramientas de control de versiones tienen varias ventajas generales que las hacen esenciales para el trabajo en equipo y la gestión de proyectos. A continuación, se presentan algunas de las ventajas más destacadas.

Historial completo de cambios: Los sistemas de control de versiones permiten mantener un historial completo de todos los cambios realizados en los archivos, lo que facilita la recuperación de versiones anteriores y la comparación de cambios a lo largo del tiempo.

Protección contra errores humanos: El control de versiones protege el código fuente contra errores humanos y consecuencias accidentales, lo que ayuda a prevenir la pérdida de datos ya mantener la integridad del proyecto.

Mejora en la colaboración y comunicación interna: Al tener total trazabilidad y control de las versiones de los archivos, es posible trabajar de manera más colaborativa y mejorar la comunicación interna del equipo.

Trazabilidad y responsabilidad: Los sistemas de control de versiones identifican al usuario responsable de cada modificación, lo que añade trazabilidad y responsabilidad al desarrollo del proyecto.

Facilita el trabajo en equipo: Los sistemas de control de versiones permiten el acceso compartido a los archivos y el desarrollo a la vez de varias bifurcaciones o ramas, lo que facilita el trabajo en equipo.

Añade trazabilidad al desarrollo de software: Los sistemas de control de versiones permiten ver qué cambios se han hecho en el código en cada versión, lo que añade trazabilidad al desarrollo de software.

Muestra información estadística del proyecto: Los sistemas de control de versiones muestran mucha información estadística de cómo se está desarrollando el proyecto, como los principales autores, número de versiones, cambios, etc.

Softwares de controles de versiones (Top 5)

- 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.

Conclusión

Podemos concluir que las herramientas de control de versiones se han establecido como parte importante en el desarrollo de software y la gestión de proyectos tecnológicos. A medida que los equipos buscan aumentar la colaboración en la creación de código fuente y optimizar los tiempos de desarrollo es esencial implementar herramientas de control de versiones que mejor se adapten al o los proyectos que se desarrollan. La elección de la herramienta adecuada debe basarse en la naturaleza del proyecto, la familiaridad del equipo con la tecnología y las características específicas que se requieren. En última instancia, el camino hacia el éxito en el desarrollo de software se ve influenciado de manera significativa por la elección acertada de una herramienta de control de versiones.

Recomendaciones.

En el panorama actual del desarrollo de software, las herramientas de control de versiones desempeñan un papel esencial en la gestión eficiente y colaborativa del código fuente. A través de una revisión exhaustiva de las ventajas y desventajas de las diversas opciones disponibles en el mercado, emerge un panorama complejo en el que cada herramienta presenta atributos distintivos que se alinean con las necesidades específicas de los equipos y proyectos. Si bien cada opción tiene sus propias características notables, no existe una solución universal que satisfaga todas las demandas de manera uniforme. Por lo tanto, la selección informada de una herramienta de control de versiones debe basarse en una evaluación minuciosa de los requisitos y la estructura del proyecto en cuestión.

Referencias.

- **What is version control?**
  https://unity.com/es/solutions/what-is-version-control

- **¿Qué es el control de versiones?**
  https://www.atlassian.com/es/git/tutorials/what-is-version-control

- **En resumen: control de versiones.**
  https://www.encora.com/es/blog/en-resumen-control-de-versiones

- **5 sistemas de control de versiones.**
  https://impulsate.between.tech/5-sistemas-control-versiones

- **Herramientas de control de versiones ¿Por qué debes usarlas?**
  https://dinahosting.com/blog/herramientas-de-control-de-versiones/