Centro de enseñanza técnica industrial

Kevin Isaac Huerta Montero	22310411
Yael Imanol González Contreras	22310410
Román Armando Cruz Melendrez	22310347
Jorge Fernando Petterson Aguilar	22310359
Angel Daniel Brambila García	22310408

IDS 3er Parcial

Mtra. GUADALUPE ORTEGA TIRADO

Lunes 28 de noviembre de 2022

Introducción Desarrollo

3ER PARCIAL

Unidad 3

Presentar la Implementación de un sistema de control de versiones usando GIT
Aplicar el Control de versiones en una práctica de desarrollo de software con GIT

Contenido

Estrategia de aprendizaje

La herramienta Git nos permite tener un sistema de control de versiones distribuido que se diferencia del resto en el modo en que modela sus datos, en los últimos años se ha convertido en una herramienta necesaria para el buen desarrollo de un sistema, por lo que su aprendizaje es importante si quieres estar al mismo nivel de un desarrollador buscado por las empresas. Utilizado principalmente por ingenieros de software para hacer un seguimiento de las modificaciones realizadas en el código fuente, el sistema de control de versiones les permite analizar todos los cambios y revertirlos sin repercusiones si se comete un error.

En otras palabras, el control de versiones permite a los desarrolladores trabajar en proyectos simultáneamente. Les permite hacer tantos cambios como necesiten sin infringir o retrasar el trabajo de sus colegas.

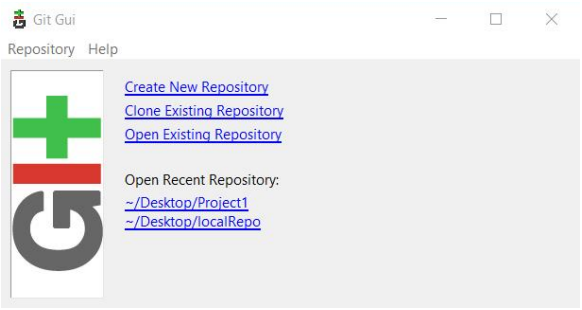
Explicando la estrategia utilizada para aprender GitHub fue reunir la información proporcionada por paginas oficiales de GitHub y de compañeros que previamente han utilizado esta herramienta, y en base a esta información hacer una lluvia de ideas, además de apoyarse viendo algunos videos de YouTube para ver la práctica y el uso que tiene en algunas funciones básicas que tiene git, aunque el ambiente de comandos resulto ser el mas efectivo de todos también es el más difícil de entender y llevar en práctica, aunque algunas fuentes explicaban que el ambiente de comandos es el mas utilizado en las condiciones laborales, entonces como buen escritor doy mi recomendación a aprender Git en vez de los otros entornos que ofrece este sistema de versiones.

Al ver que en el ambiente de comandos no la armábamos, optamos por cambiarnos a GitHub en su pagina oficial de internet, es decir utilizamos el ambiente de interfaz grafica para trabajar y desarrollar nuestra práctica, siendo más fácil de implementar nuestro proyecto. En base a esto partimos con nuestra organización de equipo en base a un proyecto recién creado, ayudándonos a ver la organización y el flujo de trabajo de una mejor manera.

Instalación de GIT

Como se mencionó anteriormente, primero se instaló la versión GIT con ambiente de comandos, cabe mencionar que esta versión cuenta con 3 maneras de utilizarse, una de ellas es GIT GUI, el cual se enfoca en permitir que los usuarios realicen cambios en su repositorio al realizar nuevos compromisos, modificar los existentes, crear ramas, realizar fusiones locales y obtener/empujar a repositorios remotos.

Todo esto con ayuda de una mini interfaz que hace mas ameno los procesos anteriores, sin duda alguna es muy recomendable utilizar este programa si recién estas empezando a utilizar GIT y quieres utilizar el ambiente de comandos. Ahora están las 2 maneras de utilizar GIT en línea de comandos, en forma de CMD y BASH, todos las guías y videos que consulte antes de comenzar a utilizar GIT decían que era mucho mas recomendable utilizar BASH en vez de CMD, ya que BASH (comandos linux) fue hecho originalmente para GIT, además en las empresas se utiliza siempre BASH, por lo que al principio se utilizó este modo.



Cuando empezamos a utilizar Git y al ver la dificultad de este, optamos por usar GitHub, ya que también algunos compañeros tuvieron problema con el uso de git por razones de sistema operativo y de versiones, así que el uso de GitHub fue muy importante a lo largo del desarrollo de la practica por razones de compatibilidad. Además, para la programación del proyecto utilizamos editores de texto que pudieran compilar en C++, por lo que utilizamos VSC, aunque otros compañeros utilizaron simples editores de texto.

problemas que se presentan habitualmente en desarrollo de proyectos de software al no contar con un Sistema de Control de Versiones

Utilizar un software de control de versiones es una práctica recomendada para los equipos de software y de DevOps de alto rendimiento. El control de versiones también ayuda a los desarrolladores a moverse más rápido y permite que los equipos de software mantengan la eficiencia y la agilidad a medida que el equipo se escala para incluir más desarrolladores.

Aunque se puede desarrollar software sin utilizar ningún control de versiones, hacerlo somete al proyecto a un gran riesgo que ningún equipo profesional debería aceptar. Así que la pregunta no es si utilizar el control de versiones, sino qué sistema de control de versiones usar.

Aun así, hemos detectado algunos problemas que derivan del no uso o mal uso del control de versiones. Uno de ellos es el posible riesgo de eliminar el proyecto y perder todos los datos al no contar con un respaldo, el control de versiones te ayuda a esto y más, teniendo un registro de cambios que aseguran la integridad del proyecto.

Tiempo

Cuando no tienes un control de versiones te puede afectar en el tiempo, ya que al surgir un error te será muy difícil encontrar de donde proviene o que lo provoca.

Ventajas

- Un completo historial de cambios a largo plazo
- Creación de ramas y fusiones.
- Trazabilidad. Ser capaz de trazar cada cambio que se hace en el software y conectarlo con un software de gestión de proyectos y seguimiento de errores como Jira

Conclusión

1

En lo personal mi experiencia con github ha sido poca pero con lo poco que le he usado, he tenido muchas buenas experiencias al usarlo ya que me gusta el hecho de que se pueden crear ramas para que en dado caso haya errores se puedan resolver sin afectar el trabajo main. Tiene muchas herramientas guthub que me gustan es muy útil al igual que la comunidad que usa este sitio pueden ayudarse entre todos o puedes encontrar códigos o soluciones que te sirvan en algún momento.

2

Con el desarrollo del proyecto integrador, pude volver a usar la plataforma de GitHub de una manera más centrada, ya que debido a la escuela no podía mantenerme siempre en solo usarlo. Logre recordar cómo usarlo y también como emplearlo de una manera más adecuada para evitar errores al momento de crear la extracion y combinacion (pull request).

3

A manera de conclusión, un SCV lo comparo con un diario, donde puedes anotar todas tus actividades diarias, eso es un control de versiones, además es un sistema de control de versiones distribuido, lo que significa que un clon local del proyecto es un repositorio de control de versiones completo. Estos repositorios locales plenamente funcionales permiten trabajar sin conexión o de forma remota con facilidad. Los desarrolladores confirman su trabajo localmente y, a continuación, sincronizan su copia del repositorio con la copia en el servidor. Este paradigma es distinto del control de versiones centralizado, donde los clientes deben sincronizar el código con un servidor antes de crear nuevas versiones. después habla sobre los términos que se usan en Git (lenguaje o palabras clave); repositorio: historial de versiones, Commit: cambios hechos al control de versiones, Rama: copia del proyecto aislado que sirve para trabajar con el proyecto, Clon: copia de todo el proyecto. Después habla sobre la historia

de Git, ve una entrevista a el dueño de Linux y le pregunta sobre Git. Habla sobre las características de Git, lo que ya había investigado antes, aunque lo explico de manera más funcional que teórica.

Referencias Bibliográficas

Atlassian. (s. f.). Qué es el control de versiones | Atlassian Git Tutorial. https://www.atlassian.com/es/git/tutorials/what-is-version-control

Git - GUI Clients. (s. f.-b). https://git-scm.com/download/gui/windows

Fazt Code. (2021, 30 noviembre). Github Desktop Tutorial - Interfaz Gráfica Oficial de Github [Vídeo]. YouTube. https://www.youtube.com/watch?v=TuOQBfhp-r0

A., D. (2022, 8 febrero). Mejores clientes Git GUI de 2022: todas las plataformas incluidas. Tutoriales Hostinger. https://www.hostinger.mx/tutoriales/mejores-clientes-git-gui

Kinsta (29 de diciembre 2020). "Git vs Github: ¿Cual es la Diferencia y como Empezar?". Recuperado el 27 de noviembre del 2022 del sitio web https://kinsta.com/es/base-de-conocimiento/git-vs-github/#una-introduccin-a-git-y-al-control-de-versiones

colaboradores de Wikipedia. (2022, 19 octubre). Git. Wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/Git