Universidad de San Carlos
Facultad de Ingeniería
Escuela de Estudios de Postgrado
Maestría en Ingeniería para la Industria con Especialización en Ciencias de la Computación
Fundamentos de Programación y Scripting

24/02/2023

Pablo Roberto Roche Palacios

Registro: 999012369

Tarea #1 - GIT

Empecemos con una pregunta: ¿Cuál es la mejor forma de mantener controlado a un archivo, una carpeta o un proyecto? Cuando se habla de control de archivos se refiere al registro preciso de creación, modificación, eliminación, renombramiento, entre otras acciones, que se realiza a un archivo a lo largo del tiempo.

Por ejemplo, a un archivo de texto de Word se le puede empezar el lunes y escribir una breve introducción de una investigación para guardarlo; el martes se escribe otra parte de la investigación y se guarda; el miércoles se vuelve a leer la introducción y se decide cambiarla por algo más descriptivo; el jueves se escribe más partes de la investigación; el viernes, día de entrega de la investigación, se piensa que la mejor introducción fue la del lunes, ¿cómo se puede volver a ese punto en tan poco tiempo?

Por sí mismo Word no puede registrar los cambios, por lo que necesitaríamos haber implementado un sistema de control de versiones para ver los cambios. Un método sencillo de haber hecho esto es que por cada vez que hayamos guardado el archivo guardar una copia indicando que cambio hicimos. Tal vez los archivos se terminarían llamando “Investigación-Introducción”, “Investigación-MarcoTeórico”, “Investigación-NuevaIntroducción” e “Investigación-Final”. Y a ese último archivo copiar-pegar el contenido del primer archivo. Esto no solo es tedioso de hacer, sino que se vuelve una mala práctica al tener tantas versiones de un mismo archivo, ocupando más espacio en memoria, también se puede perder la referencia del archivo más actualizado. Esto tiene varias soluciones, entre ellas GIT.

GIT es una implementación del versionamiento de control distribuido en la que los usuarios no solo tienen una copia de los archivos también tienen una copia de los cambios realizados durante el tiempo. De esta manera, si se quiere realizar una copia del archivo, también se tendrá que copiar el registro. GIT nace de la necesidad de la comunidad de mantenimiento del núcleo de Linux de poder mantener el registro de los archivos del proyecto, dado que antes se usaba un versionamiento de control distribuido propietario llamado BitKeeper. Este programa ya no daba el soporte que necesitaba la comunidad, por lo que Linus Torvalds, también creador de Linux, impulsó al desarrollo de una herramienta veloz, sencilla, con múltiples líneas de desarrollo, completamente distribuida y capaz de manejar proyectos grandes. Ahora GIT se usa por ser un estándar en la programación, siendo parte fundamental del desarrollo de lenguajes, software, páginas web y otros.

Comandos[1]

Con GIT se puede hacer el rastreo completo de archivos pertenecientes a un directorio. Para iniciar el rastreo se utiliza el comando en una terminal de comandos:

$$git\ init$$

Con este comando se le creará una carpeta llamada ". git" al directorio, en la que se guardarán los cambios de los archivos. A este directorio desde ahora se le conocerá como repositorio. Este repositorio se hace localmente, por lo que para compartirlo con otras personas se podrá copiar el repositorio o subirlo en páginas especializadas, tales como GitHub, GitLab o BitBucket. Al querer copiar un repositorio que esté en alguna de estas páginas, y con permiso del usuario original, se puede crear una copia, a lo que se le conoce como clon, con el comando:

$$git\ clone\ [url\ del\ repositorio]$$

Ahora se puede trabajar en los archivos que van a ser rastreados, siempre que existan en el repositorio o se agreguen a él. Entre los cambios que podrán ser rastreados se encuentran:

- Archivos agregados: Estos archivos no existen en el repositorio y su rastreo quiere ser agregado.
- Archivos modificados: Estos archivos existen en el repositorio y su contenido es modificado.
- Archivos renombrados: Estos archivos existen en el repositorio y quieren ser nombrados de otra forma, por lo que se le aplica un cambio solo en el nombre, más no en el contenido.
- Archivos eliminados: Estos archivos existen en el repositorio y su rastreo quiere ser eliminado.

Para agregar un archivo al repositorio para empezar a rastrearlo, con el comando:

$$git\ add\ [direccion\ del\ archivo]$$

O bien, rastrear a todo el directorio, con el comando

$$git\ add\ .$$

Con esto se agregarán al rastreo del repositorio todos los archivos, pasando a una fase llamada preparada, que indica que está lista para guardar el registro. Al modificar los archivos estos pasan a una fase llamada modificada, que se ha modificado el archivo y está lista para guardar el registro. Al guardar el estado actual del repositorio se pasa a una fase llamada confirmada. Esto se realiza con el comando:

$$git\ commit - m\ "[mensaje\ descriptivo]"$$

Esto realizará un guardado automático de los cambios de nuestros archivos rastreados, es decir una versión. A esta versión se le guarda con un identificador alfanumérico, que ayudará a rastrear los cambios hechos. También se guardará el mensaje, que describirá lo realizado hasta ese punto del tiempo. Los cambios, mensaje e identificador se guardarán en la carpeta ". git". Y con esto se regresan todos los archivos al estado inicial del repositorio, que se conoce como directorio de trabajo.

---

[1] Comandos extraídos de GitKraken, Learn Git https://www.gitkraken.com/learn/git. Puede descargarse una página de comandos en: https://www.gitkraken.com/pdfs/git-basics-cheat-sheet

Con las modificaciones que se hacen, puede que en algún punto no se vea el estado actual de los archivos que hemos modificado, para ver los cambios en el repositorio se utiliza el comando:

$$git\ status$$

Para mostrar toda la información de los estados del repositorio, se utiliza el comando:

$$git\ log$$

O en una versión corta de los estados con el comando:

$$git\ \log --oneline$$

Si se quiere regresar a cambios hechos a cierta versión, realizando una nueva versión regresando todos los archivos hasta la versión, se utiliza el comando:

$$git\ revert\ [commit]$$

O regresar a los cambios a una versión, destruyendo las modificaciones, más no los archivos no existentes, se utiliza con el comando:

$$git\ reset\ [commit]$$

O descartar completamente los archivos hasta una versión

$$git\ reset --hard$$

Estos dos últimos comandos no se recomiendan por ser destructivos y que puedan crear conflictos en el futuro.

Un repositorio local puede conectarse a uno o más repositorios para poder realizar cambios en ellos. Si se clona un repositorio, automáticamente se realiza una conexión al repositorio remoto u origen. Para realizar la conexión, se hace con el comando:

$$git\ remote\ add\ [nombre]\ [url\ del\ repositorio]$$

Para sincronizar los cambios, es decir, colocar los cambios hechos localmente en el repositorio remoto, se utiliza el comando:

$$git\ push$$

Para descargar el historial y archivos desde el repositorio remoto, se utiliza el comando:

$$git\ pull$$

Para descargar y combinar los cambios del repositorio remoto y los cambios locales, se utiliza el comando:

$$git\ merge$$

Para actualizar el directorio con los últimos cambios hechos en el repositorio remoto (siendo una combinación de lo que se hace con fetch y merge), se utiliza el comando:

$$git\ pull$$