CONTROL DE VERSIONES

Un producto software se modifica infinitas veces durante su tiempo de vida y, por lo tanto, se necesita una herramienta que permita llevar un control de los distintos cambios que puede sufrir a lo largo de su desarrollo o vida útil.
Git es una de las herramientas de control de versiones más utilizadas. Entre sus características se encuentran:

a) Es un SCV gratuito y de código abierto.
b) Puede utilizarse para proyectos pequeños y grandes
c) Es fácil de aprender y tiene un buen rendimiento.

En la mayoría de productos software, muchas personas intervienen en su desarrollo y, por lo tanto, es imprescindible que el control de versiones no se haga de forma manual, puesto que pueden cometerse muchos errores y el coste sería muy alto.

Se puede definir control de versiones como la capacidad de recordar todos los cambios que se realizan tanto en la estructura de directorios como en el contenido de los archivos. Esto es de mucha utilidad cuando se desea recupera un documento, o una carpeta o un proyecto en un momento concreto del desarrollo.

Almacenamiento de las distintas versiones

Los sistemas de control de versiones pueden clasificar según cómo se almacena el código. Existen dos tipos de sistemas (centralizados y distribuidos)

A) Sistemas centralizados

Su gestión es mucho más sencilla. El repositorio es único y en él se almacenará todo el código del software.
Generalmente, en muchos software, se generan varias **ramas** y, en ese caso, un responsable deberá aprobar el almacenamiento de las distintas ramas de software.
En estos sistemas, el control es mayor, y, generalmente, existe un único número de versión lo que hace que su gestión sea más sencilla, Esa rigidez implica que, muchas veces, se desee tener más flexibilidad y optan por todos tipos de configuración como los sistemas distribuidos.

B) Sistemas distribuidos

En este tipo de sistemas, cada programador mantiene una copia del repositorio. Todos los usuarios tienen una copia del software. La ventaja es que , en caso de fallo, habrá muchas copias de las cuales podrá recuperarse la última versión. en el servidor principal, residirá la copia principal.

Cuando quiere trabajarse con versiones inestables o versiones no oficiales, mucho desarrolladores y empresas de software optan por utilizar este tipo de sistemas. En el caso de que una versión inestable o de prueba supere las pruebas pertinentes, el responsable las subirá al repositorio común y formará parte del proyecto.

Además, la ventaja de que el usuario tenga una copia del proyecto en su equipo hace que el trabajo sea mucho más ágil. El servidor también está más descargado y no se necesitará una máquina tan potente ni tantos recursos.

Tipos de colaboración en un SCV

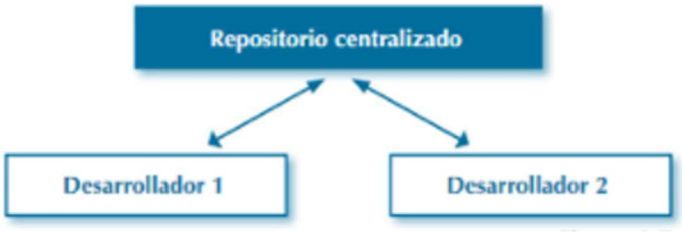
Generalmente, se colabora de dos formas en un sistema de control de versiones (SCV).

A) **Trabajar con el sistema en exclusiva**: Cuando un usuario es responsable de una parte o módulo de un sistema o desarrollo de software. Cuando está modificándolo, notifica al repositorio que esa parte está pendiente de modificación y el mismo repositorio bloquea esa parte para que no sea modificada. Cuando finaliza el trabajo, lo comunica al repositorio y sube el software modificado. El repositorio desbloquea esa parte y queda disponible para los demás usuarios.

B) **Trabajar en forma colaborativa**. En este caso, cada usuario trabaja con su cuenta en su copia local y, cuando terminan los cambios que estaba realizando, los sube al repositorio común. Los problemas más comunes al trabajar de esta forma es que pueden aparecer conflictos con las distintas modificaciones. La coordinación en este tipo de trabajo es fundamental para evitar dichos problemas.

La forma en la que se trabaja con un sistema de control de versiones se llama **workflow o flujo de trabajo.** Existen diferentes formas de trabajar con un SCV y por lo tanto, diferentes tipos de flujo de trabajo. Las más frecuentes son:

1- Flujo de trabajo centralizado o workflow centralizado- Suele utilizarse un servidor centralizado y los desarrolladores van publicando sus actualizaciones de código en él.
Estos sistemas funcionan de forma exclusiva, lo que implica que, si alguien está modificando un elemento ningún otro desarrollador, puede realizar actualizaciones de este.



2- Flujo de trabajo con gestor de integración - En ocasiones para un mayor control sobre las versiones del proyecto, se utiliza un gestor de integración, que suele ser una persona del equipo de desarrollo que actualiza los trabajos en el repositorio común. Los desarrolladores actualizan sus repositorios públicos y es el gestor de integración el encargado de actualizar el repositorio común con los trabajos públicos independientes de los desarrolladores. Este sistema suele ser muy utilizado en SCV distribuidos.

C) **Flujo de trabajo con dictador y tenientes**. - Este tipo de flujo de trabajo se utiliza solamente en caso de proyectos masivos (ej- kenel de linux). Es una evolución de modelo anterior.

En un proyecto muy grande, existen persona supervisores de varias partes del proyecto llamadas **tenientes** y un persona llamada **dictador** que puede actualizar los cambios que sus tienes le envían. En este sistema, todos los desarrolladores tienen acceso al repositorio para poder clonarlo y crearse su propia copia.