Tutorial sobre GIT & GitHub

Acerca del control de versiones

El control de versiones es un sistema que registra los cambios realizados sobre un archivo o conjunto de archivos a lo largo del tiempo, de modo que puedas recuperar versiones específicas más adelante.

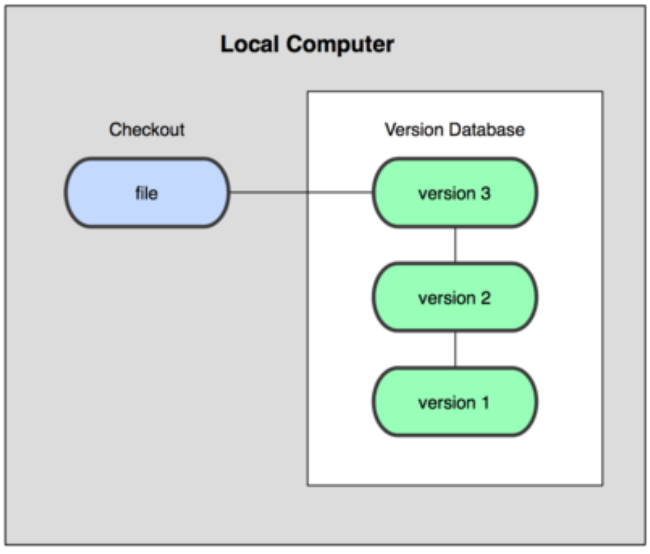
Te permite revertir archivos a un estado anterior, revertir el proyecto entero a un estado anterior, comparar cambios a lo largo del tiempo, ver quién modificó por última vez algo que puede estar causando un problema, quién introdujo un error y cuándo, y mucho más.

Hay diferentes sistemas de control de versiones que pueden clasificarse en:

Locales

Un método de control de versiones usado por mucha gente es copiar los archivos a otro directorio (quizás indicando la fecha y hora en que lo hicieron). Este enfoque es muy común porque es muy simple, pero también tremendamente propenso a errores.

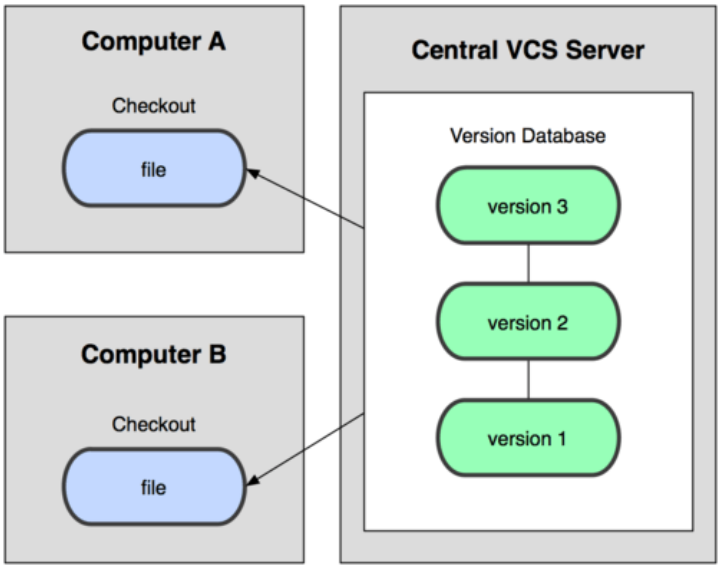
Para hacer frente a este problema, existen los VCSs locales que contienen una simple base de datos en la que se lleva registro de todos los cambios realizados sobre los archivos.



Centralizados

El siguiente gran problema que se encuentra la gente es que necesitan colaborar con desarrolladores en otros sistemas. Para solventar este problema, se desarrollaron los sistemas de control de versiones centralizados (Centralized Version Control Systems o CVCSs en inglés).

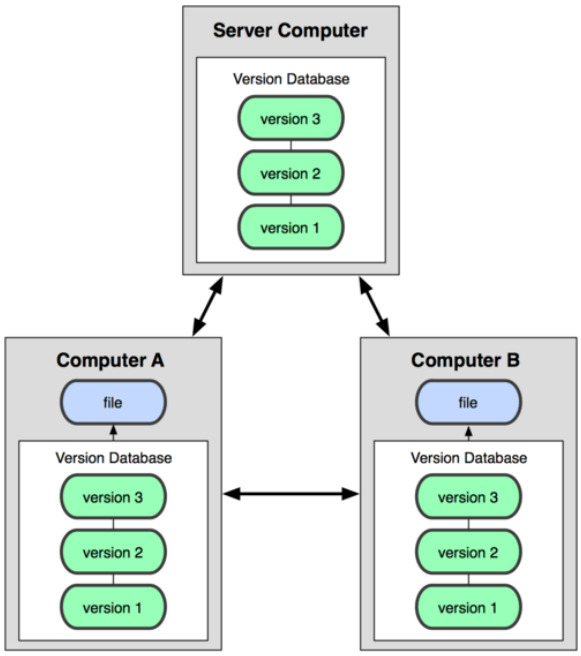
Estos sistemas, como CVS, Subversion, y Perforce, tienen un único servidor que contiene todos los archivos versionados, y varios clientes que descargan los archivos desde ese lugar central. Durante muchos años éste ha sido el estándar para el control de versiones.



Esta configuración también tiene serias desventajas. La más obvia es el punto único de fallo que representa el servidor centralizado. Si ese servidor se cae durante una hora, entonces durante esa hora nadie puede colaborar o guardar cambios versionados de aquello en que están trabajando. Si el disco duro en el que se encuentra la base de datos central se corrompe, y no se han llevado copias de seguridad adecuadamente, pierdes absolutamente todo —toda la historia del proyecto salvo aquellas instantáneas que la gente pueda tener en sus máquinas locales. Los VCSs locales sufren de este mismo problema— cuando tienes toda la historia del proyecto en un único lugar, te arriesgas a perderlo todo.

Distribuidos

Los sistemas de control de versiones distribuidos (Distributed Version Control Systems o DVCSs en inglés) resuelven este problema. En un DVCS (como Git, Mercurial, Bazaar o Darcs), los clientes no sólo descargan la última instantánea de los archivos: replican completamente el repositorio. Así, si un servidor muere, y estos sistemas estaban colaborando a través de él, cualquiera de los repositorios de los clientes puede copiarse en el servidor para restaurarlo. Cada vez que se descarga una instantánea, en realidad se hace una copia de seguridad completa de todos los datos.



Git entra dentro de esta categoría.

Instalando GIT

La instalación está muy bien explicada en

http://git-scm.com/book/es/Empezando-Instalando-Git

Comandos de GIT

Para trabajar con un repositorio Git existen un conjunto de comandos. Los más comunes son:

- INIT: Incializar un repositorio git
- STATUS: Ver estado de los archivos
- ADD: Agregar archivos para el próximo commit
- COMMIT: Confirmar localmente alguna modificación del repositorio.
- PULL: Bajar los cambios del repositorio remoto.
- PUSH: Envíar de cambios a repositorio remoto.

Referencias

Git Book (en español) http://git-scm.com/book/es/

GitHub http://github.com

http://rogerdudler.github.io/git-guide/index.es.html

http://nvie.com/posts/a-successful-git-branching-model/