Sistemas de control de versiones
El ejemplo de Git

Rafael López García

(rafalg@cifprodolfoucha.es)

CIFP Rodolfo Ucha Piñeiro

GOBIERNO DE ESPAÑA
MINISTERIO DE EDUCACIÓN Y FORMACIÓN PROFESIONAL

Unión Europea-
NextGenerationEU

CIFP Rodolfo Ucha Piñeiro

Parte I: Sistemas de control de versiones

Definición

- Un *sistema de control de versiones (VCS)* es un software que nos permite gestionar un registro de **cambios que realizamos a un conjunto de ficheros a lo largo del tiempo**
  - Hace posible recuperar versiones específicas más adelante si nos es necesario
  - Especialmente útil en proyectos en los que varias personas trabajan de forma simultánea

Nombres alternativos

- También se le puede encontrar con otros nombres como:
  - Gestión del Código Fuente (SCM)
  - Sistema de Control de Revisiones (RCS)

Características

- Los VCS suelen presentar las siguientes características:
  - Revertir cambios a una versión anterior del fichero
  - Resolver conflictos si varias personas trabajan sobre un mismo fichero al mismo tiempo
  - Funcionamiento distribuido
    - Hacer copias de seguridad remotas
  - Desarrollo no lineal
    - Posibilidad de abrir ramas (*branches*) de un repositiorio

Ejemplos de VCS

- Git (https://git-scm.com/)
- CVS (https://www.nongnu.org/cvs/)
- Apache Subversion (SVN) (https://subversion.apache.org/)
- Mercurial (https://www.mercurial-scm.org/)
- Monotone (https://www.monotone.ca/)
- Bazaar (https://bazaar.canonical.com/en/

Sistemas de almacenamiento en la nube que usan VCS

- Gran parte de sistemas de almacenamiento en la nube emplean VCS
  - Dropbox
  - Google Drive
  - Microsoft OneDrive
  - …

Parte II: Git

Git

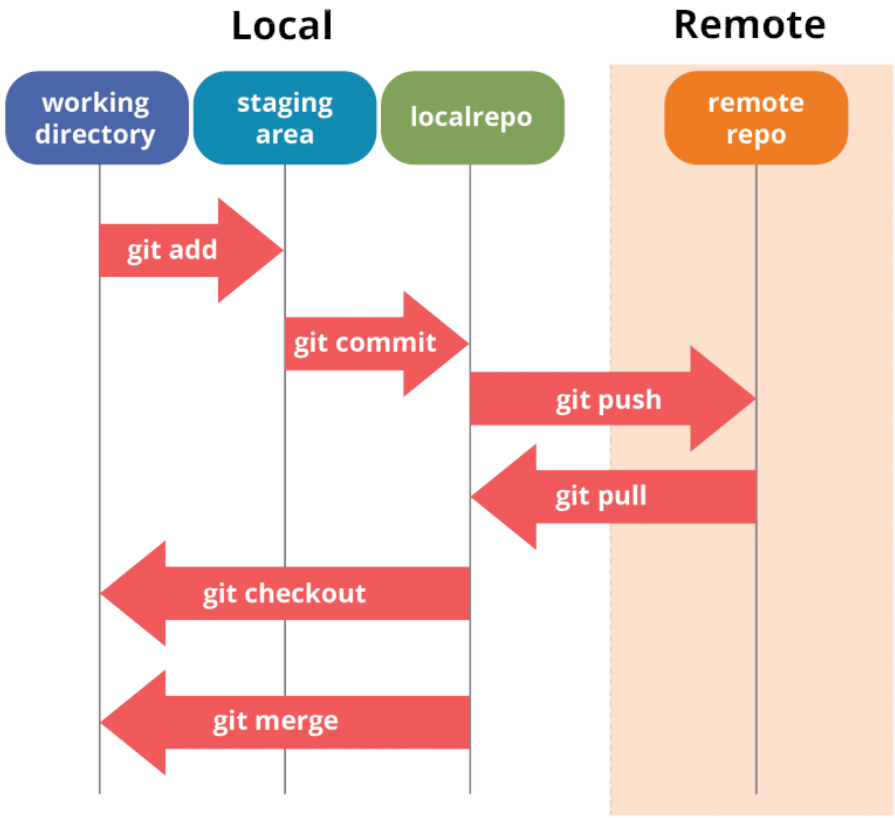
- **Como ejemplo de VCS emplearemos *Git***, ya que es uno de los mejores y más usados
- ***GitHub* es un repositorio central** muy conocido
  - https://github.com/
- Desde 2021 los comandos ya no se pueden emplear con usuario y contraseña, sino con token
  - https://stackoverflow.com/questions/68775869/message-support-for-password-authentication-was-removed-please-use-a-personal

Funcionamiento de Git (I)

- Git funciona con repositorios locales y remotos
- Dentro del equipo local, trabajamos en un directorio (*working directory*)
- Dentro de dicho directorio de trabajo, no todos los ficheros están siendo gestionados (*tracked*) por Git, sólo aquellos que se meten en el escenario (*stage*)

Funcionamiento de Git (II)

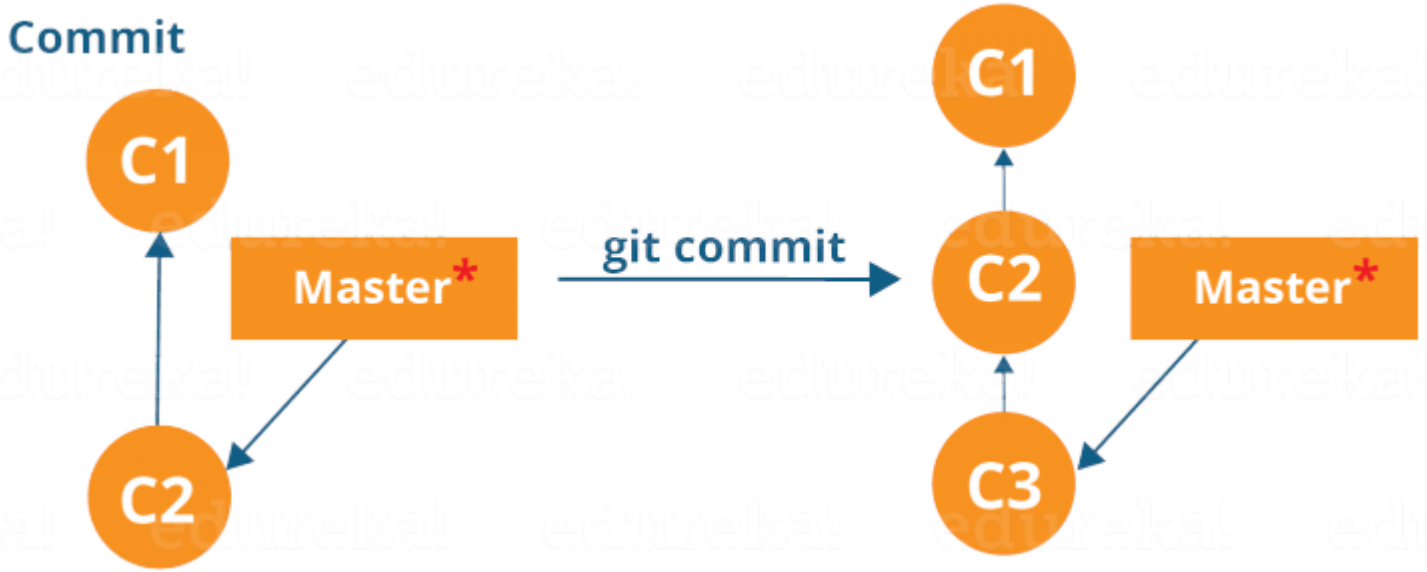
- Una vez metidos los ficheros en el escenario, se van haciendo versiones en un repositorio local, y, si se desea, en el remoto



Fuente de la imagen: https://www.edureka.co/blog/git-tutorial/

Funcionamiento de Git (III)

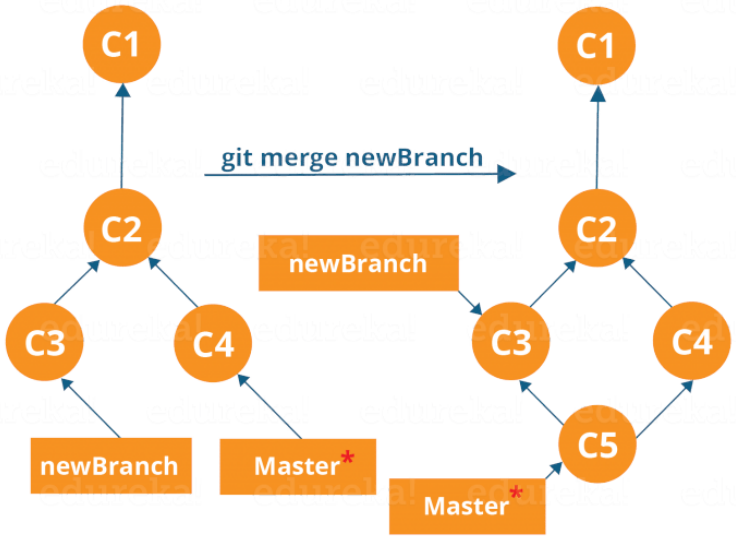
- Un *commit* es una imagen de los cambios hechos al proyecto en un cierto momento
  - Estas copias no se modificarán a menos que se haga explícitamente



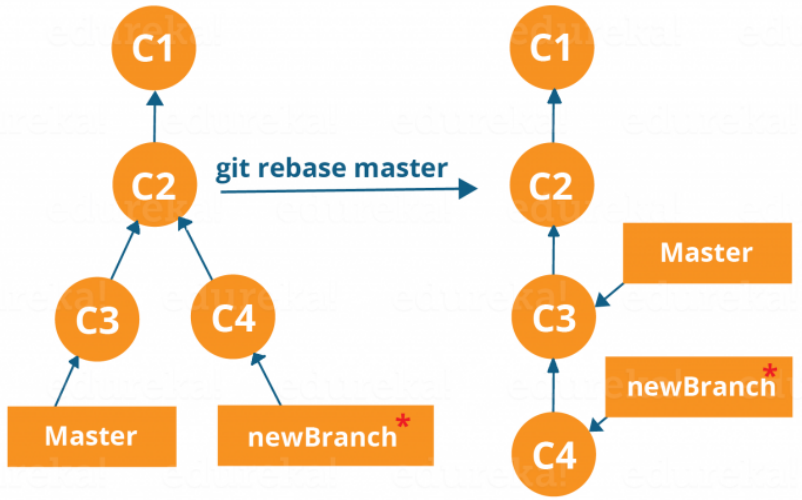
Fuente de la imagen: https://www.edureka.co/blog/git-tutorial/

Funcionamiento de Git (IV)

- En un momento dado, se pueden crear distintas ramas de un *commit*, y más tarde se pueden mezclar (*merge*) o colocar una tras otra (*rebase*)



Fuente de la imagen: https://www.edureka.co/blog/git-tutorial/



Fuente de la imagen: https://www.edureka.co/blog/git-tutorial/

Comandos básicos de Git (I)

- **git config**: permite configurar algunas variables de entorno de git
  - git config --global user.name "usuario"
  - git config --global user.email "ejemplo@test.com"
- **git clone URL**: obtiene una copia de un repositorio a partir de una URL
  - git clone https://github.com/microsoft/avml.git

Comandos básicos de Git (II)

- **git init [REPOSITORIO]**: crea un nuevo repositorio de git o reinicia uno existente
  - git init /home/usuario/proyecto
- **git add [-A | FICHERO]**: añade un nuevo fichero o directorio al escenario (*stage*) de trabajo actual
  - git add fichero
  - git add *
  - git add -A

Comandos básicos de Git (III)

- **git commit [-a] [-m MENSAJE]**: crea un *commit* y con -a añade ficheros nuevos y modificados
  - git commit -m "Primer commit"
  - git commit -a
- **git reset [FICHERO | [--hard] COMMIT]**: quita un fichero del entorno, vuelve a un *commit* anterior, etc.
  - git reset Clase.java
  - git reset --hard b01557d80d5f53…20d8b8c16

Comandos básicos de Git (IV)

- **git diff**: muestra las diferencias entre dos grupos de ficheros
  - los actuales y los del entorno:
    - git diff
  - los del entorno y los que se ha hecho *commit*:
    - git diff --staged
  - los de dos ramas distintas:
    - git diff rama_1 rama_2

Comandos básicos de Git (V)

- **git status**: muestra los ficheros de los que se ha hecho un *commit*
  - git status
- **git rm [FICHERO]**: borra un fichero del entorno y también del disco duro
  - git rm Clase.java
- **git show [COMMIT]**: muestra los metadatos y contenidos de un cierto *commit*
  - git show b01557d80d5f53…20d8b8c16

Comandos básicos de Git (VI)

- **git log [--follow FICHERO]**: muestra el historial de versiones de la rama actual, o de un cierto fichero
  - git log
  - git log --follow Clase.java
- **git tag [COMMIT_ID]**: le establece un tag a un cierto *commit*
  - git tag b01557d80d5f53…20d8b8c16

Comandos básicos de Git (VII)

- **git branch [[-d] BRANCH_ID]**: lista, crea y borra *ramas (branches)* de un proyecto
  - git branch
  - git branch rama_1
  - git branch -d rama_1
- **git checkout [-b] NOMBRE_RAMA**: cambia a un cierto *branch*
  - git checkout rama_2
  - git checkout -b rama_2

Comandos básicos de Git (VIII)

- **git merge [NOMBRE_RAMA]**: mezcla los cambios del *branch* especificado con el actual
  - git merge rama_2
- **git remote add NOMBRE URL_REPOSITORIO_REMOTO**: conecta el repositorio local con el servidor remoto
  - git remote add origin https://github.com/usuario/proyecto.git

Comandos básicos de Git (IX)

- **git push [-all] [NOMBRE_RAMA] [NOMBRE_RAMA]**: envía uno o varios branches al repositorio remoto
  - git push origin master
  - git push --all origin
- **git pull URL_REPO_REMOTO**: copia el repositorio remoto a local y mezcla los cambios
  - git pull https://github.com/usuario/proyecto.git

Comandos básicos de Git (X)

- **git stash [COMANDO]**
  - **save**: mete los ficheros en un almacenamiento temporal
  - **pop**: Saca los ficheros del almacenamiento
  - **list**: Lista los ficheros del almacenamiento
  - **drop**: Descarta los ficheros del almacenamiento

Referencias

Referencias (I)

- Listado comparativo de sistemas VCS:
  - https://spa.myservername.com/15-best-version-control-software
- Manuales y comandos de Git:
  - https://git-scm.com/book/es/v2
  - https://dzone.com/articles/top-20-git-commands-with-examples
  - https://www.edureka.co/blog/git-tutorial/