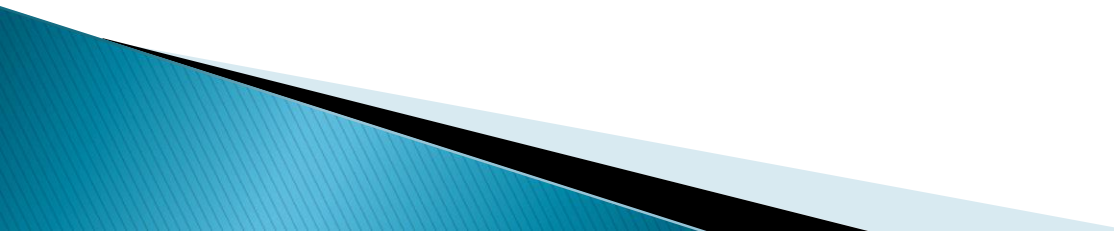




¿Qué es `git`?

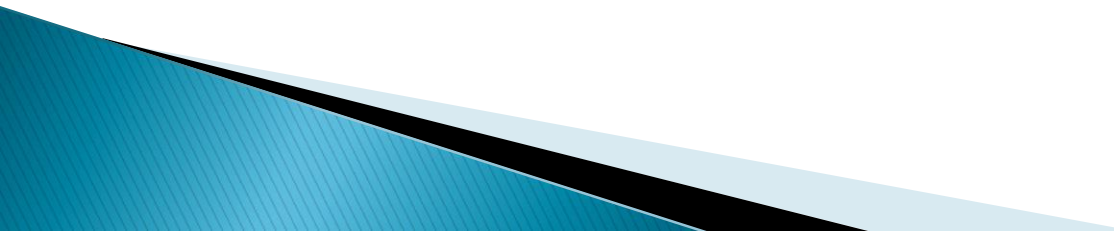
`git` es un sistema distribuido de control de versiones, orientado a la velocidad, con chequeo de integridad y ramas livianas, creado por Linus Torvalds en 2005 ante la revocación de la licencia de BitKeeper, el sistema de control de versiones propietario que se usaba en el desarrollo del kernel Linux, luego de que algunos desarrolladores hubieran aplicado ingeniería inversa al código de BitKeeper.



¿Para qué nos sirve esto del `git`?

- ✓ Tener todos la última versión del código.
- ✓ No tener infinitos .zip y no saber cuál es el que tiene el código que anda.

Ventajas con respecto al TFS

- ✓ Mayor nivel de aislamiento al ser distribuido.
 - ✓ Capacidad de ramificación del desarrollo.
 - ✓ Mayor nivel de control a través de los pull-requests a una rama específica.
- 

TFS VS Git

TFS

Check-In

Get Latest Version

'Map Local Path'

Shelve

Label

'Compare Local to Server'

Checkin and get Latest

Git

Commit + Push

Pull

Clone

Stash (only local though)

Tag

Fetch

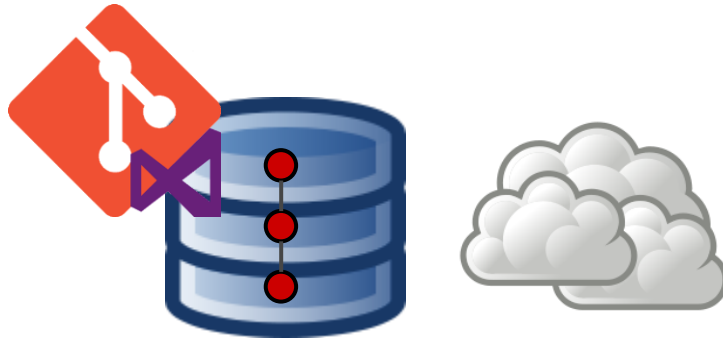
Sync

Usando git

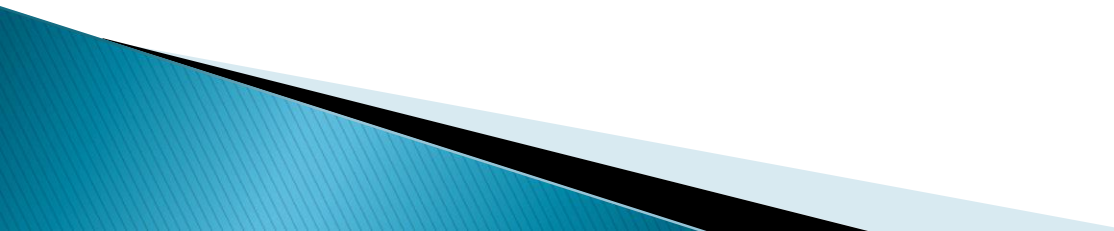
- ✓ Tenemos un repositorio remoto (GitHub / TFS / otros).
- ✓ Clonamos el repositorio localmente.
- ✓ Modificamos el directorio de trabajo.
- ✓ Stageamos/Indexamos cambios.
- ✓ **Commi**teamos.
- ✓ **pull**eamos nuevos commits.
 - ✓ Resolvemos conflictos.
- ✓ **pushe**amos nuestros commits.

Tenemos un repositorio Remoto

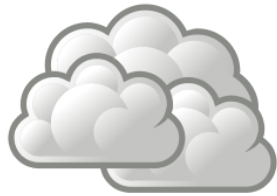
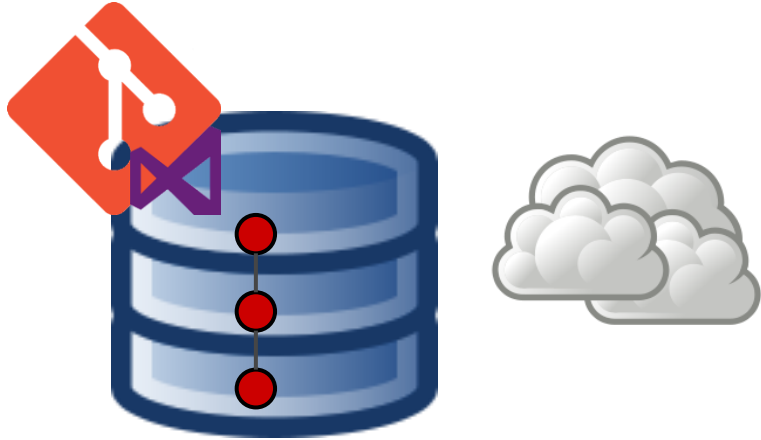
- ✓ Repositorio: directorio en el que `git` guarda toda la información que necesita
- ✓ Proveedor del repositorio: TFS a partir de la versión 2015, GitHub, entre otros.



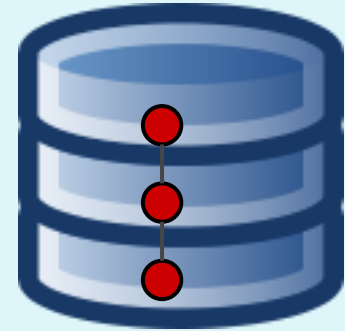
Clonamos el repositorio localmente

- ✓ **clone**: crea un repositorio con el mismo contenido que el original de Repositorio
 - ✓ Además, nos crea una Copia de Trabajo (Working Copy), un snapshot de la última versión del código
- 

Clonamos el repositorio localmente

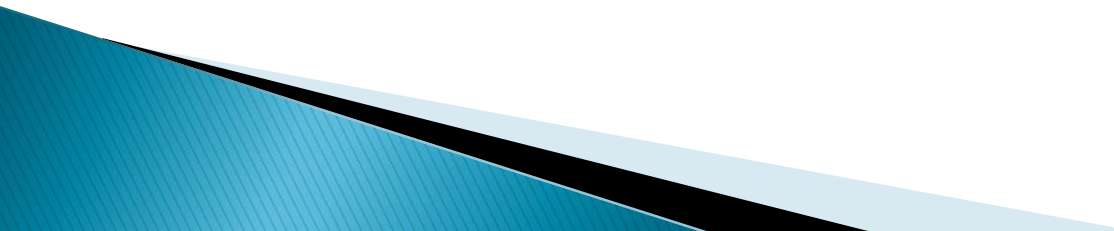


Copia de
Trabajo

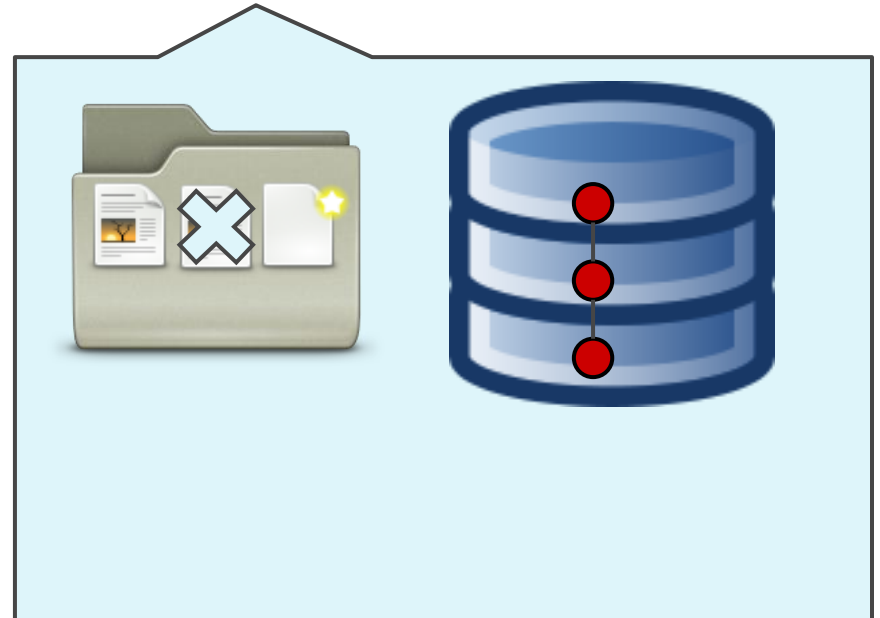
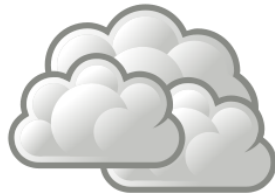
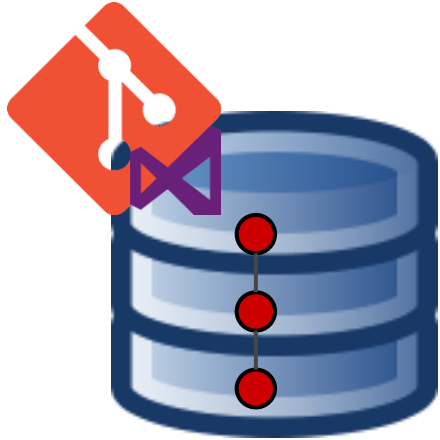


Repositorio
local

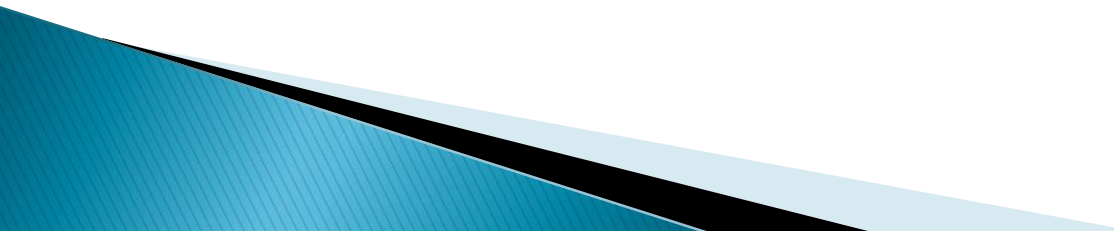
Modificamos el directorio de trabajo

- ✓ Creamos, eliminamos o editamos archivos
 - ✓ La Copia de Trabajo queda alterada
 - ✓ El Repositorio sigue intacto
- 

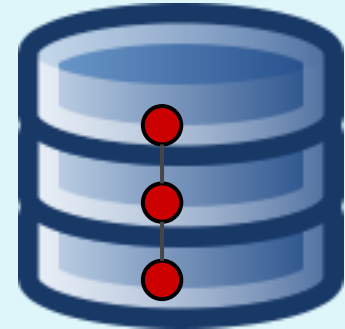
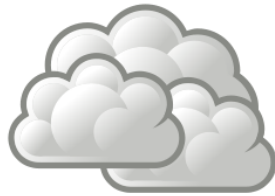
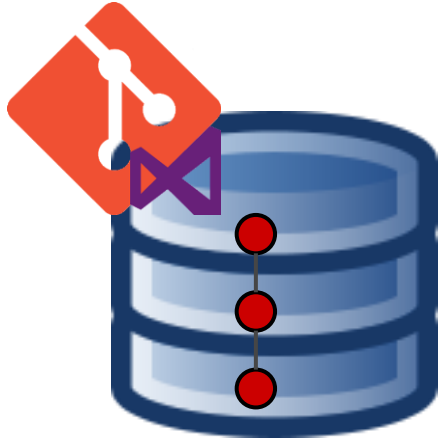
Modificamos el directorio de trabajo



Stageamos/Indexamos cambios

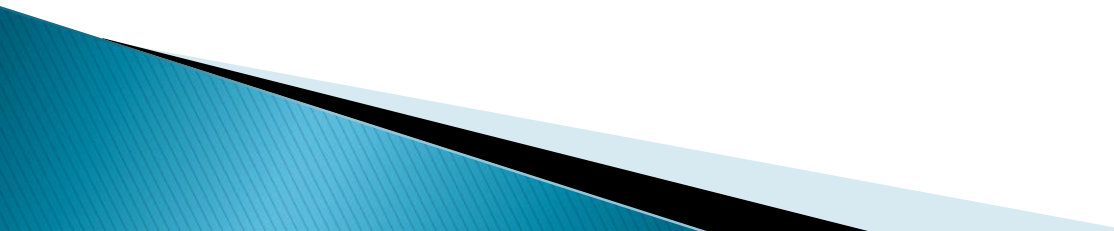
- ✓ Agregamos al Índice los cambios que entrarán en el próximo commit
 - ✓ Podemos elegir indexar archivos completos, o solo partes
 - ✓ El Repositorio sigue intacto
- 

Stageamos/Indexamos cambios

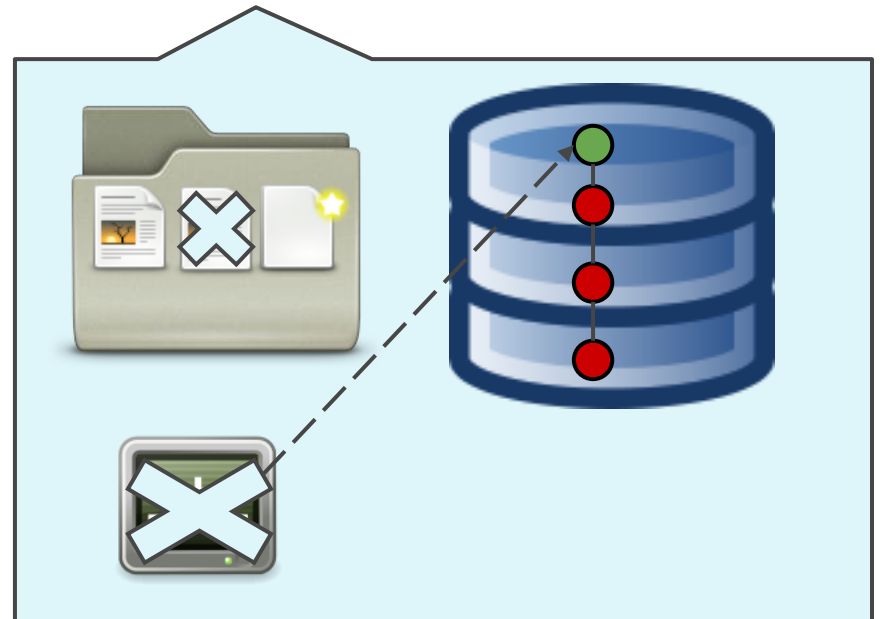
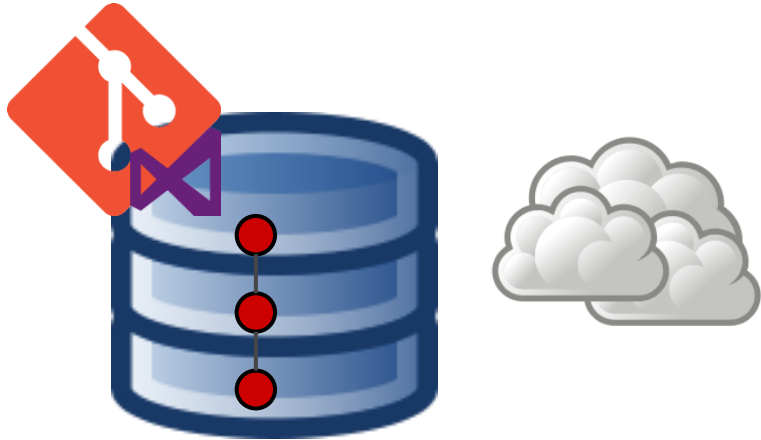


Índice


committeamos

- ✓ Registramos los cambios indexados con un mensaje, autor, fecha y commit[s] antecesor[es]
 - ✓ El commit se crea en nuestro repositorio
 - ✓ El Index se vacía
- 

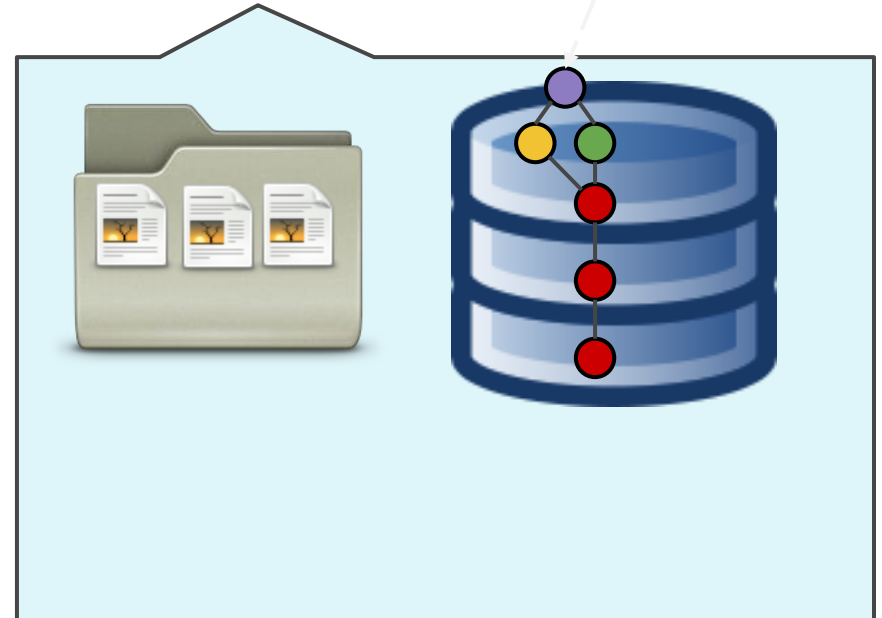
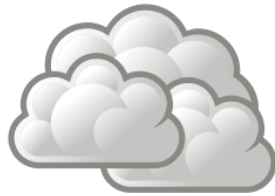
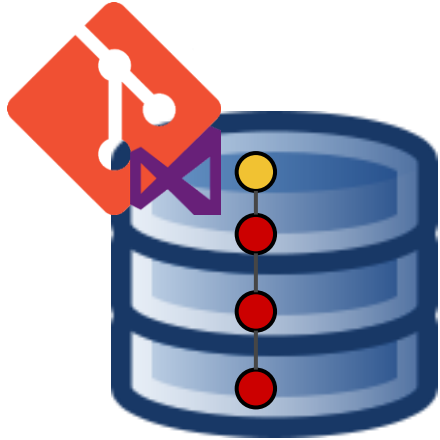
commitamos



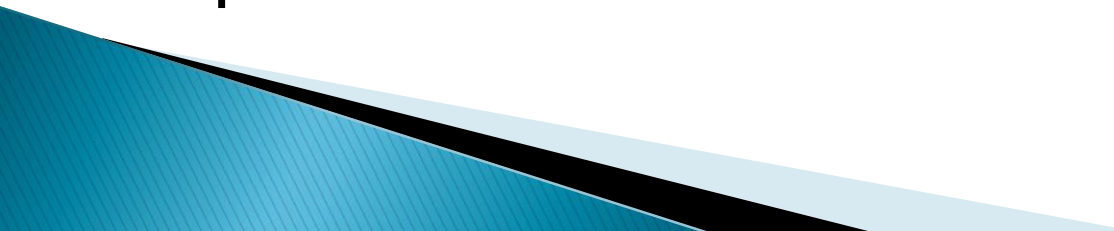
pullleamos commits del remoto

- ✓ Copiamos al repositorio local los nuevos commits del remoto.
 - ✓ Si teníamos nuevos commits nuestros, `git` intenta **merge**arlos en un nuevo commit.
 - ✓ Si hay conflictos (modificaciones superpuestas), editamos a mano los archivos y confirmamos el commit de merge.
- 

pullamos commits del remoto



pusheamos commits al remoto

- ✓ Enviamos al repositorio remoto nuestros nuevos commits
 - ✓ El Repositorio local, Index y Copia de Trabajo quedan intactos
 - ✓ El resto de colaboradores del repositorio pueden ver nuestros commits.
- 

pusheamos commits al remoto

