



GIT

@miguelcampos

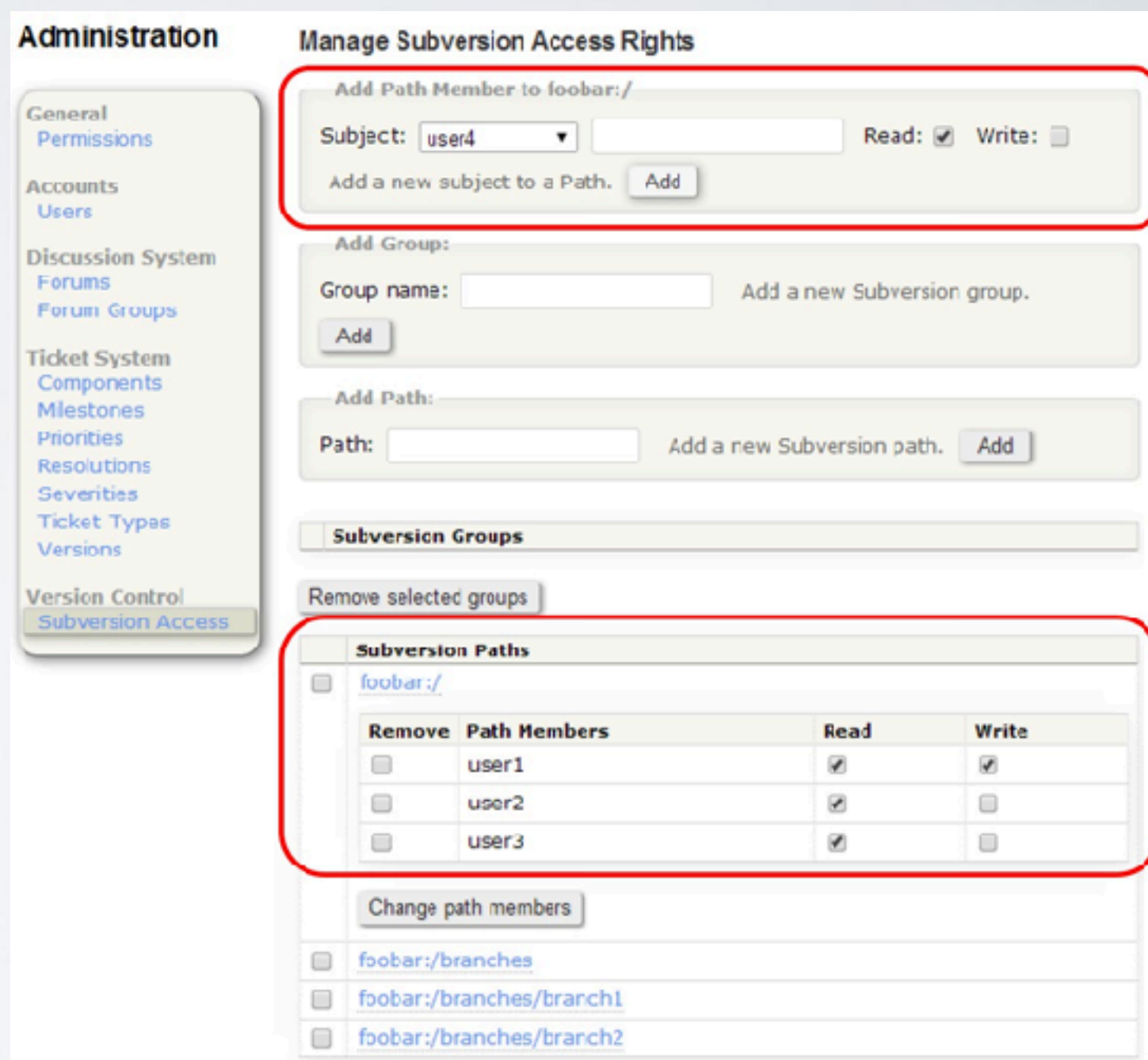
- **¿En qué consiste un control de versiones?**

- Un Control de Versiones no es más que una herramienta para facilitar el desarrollo de Software tanto entre equipos como individuales.

- **¿Qué podemos hacer con un control de versiones?**

1. Llevar un seguimiento de la evolución de nuestro Software
2. Poder versionar nuestro Software.
3. Colaborar sobre una misma base de código, sin tener que copiar y pegar entre los desarrolladores del proyecto.

- En un control de versiones centralizado (CVCS), como por ejemplo puede ser **Subversion**, todo el versionado se hace en una máquina que digamos que contiene la única “fuente de verdad”.
- El servidor que contiene el control de versiones establece el grado de acceso que tiene cada usuario, y las acciones que puede realizar (sólo lectura / lectura y escritura).



Administration

- General
- Permissions**
- Accounts
- Users
- Discussion System
- Forums
- Forum Groups
- Ticket System
- Components
- Milestones
- Priorities
- Resolutions
- Severities
- Ticket Types
- Versions
- Version Control
- Subversion Access**

Manage Subversion Access Rights

Add Path Member to foobar: /

Subject: Read: ☒ Write: ☐

Add a new subject to a Path.

Add Group:

Group name: Add a new Subversion group.

Add Path:

Path: Add a new Subversion path.

Subversion Groups

Subversion Paths

	Remove	Path Members	Read	Write
<input type="checkbox"/>	<input type="checkbox"/>	user1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/>	<input type="checkbox"/>	user2	<input checked="" type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	<input type="checkbox"/>	user3	<input checked="" type="checkbox"/>	<input type="checkbox"/>

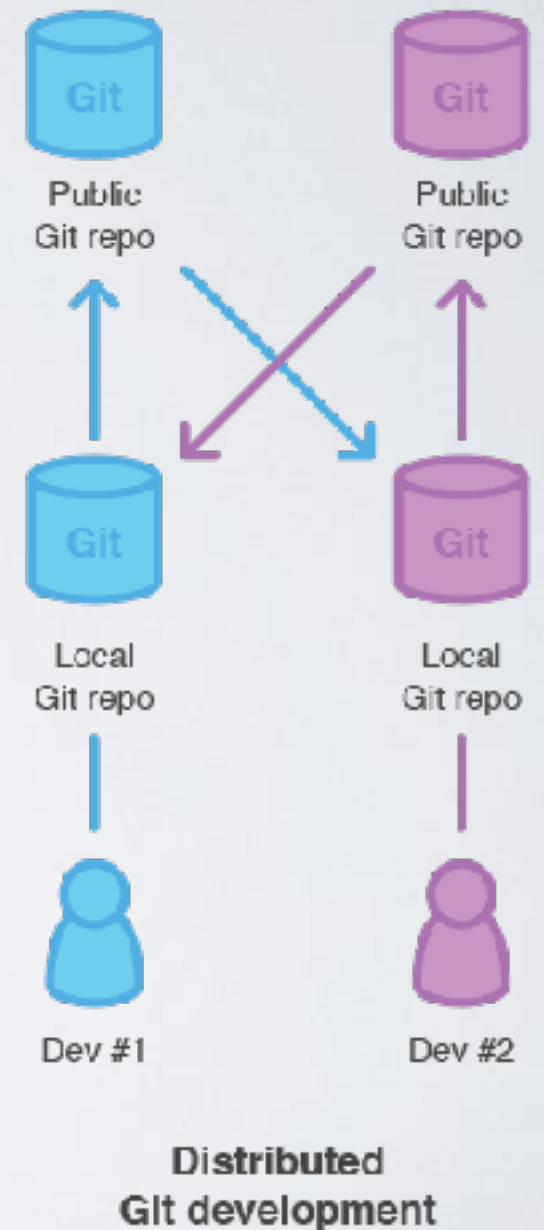
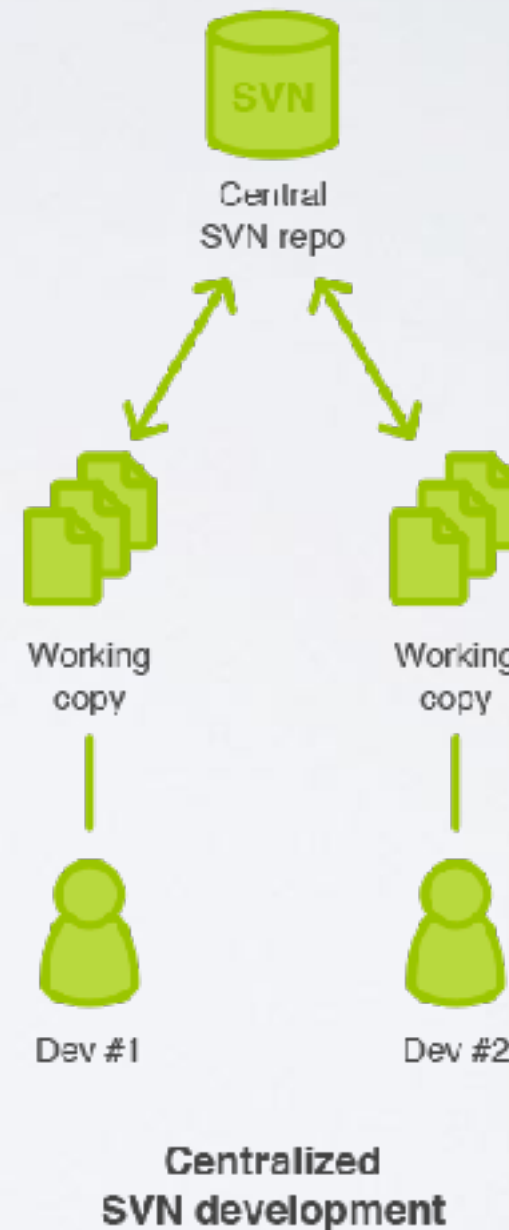
☐ foobar:/

☐ foobar:/branches

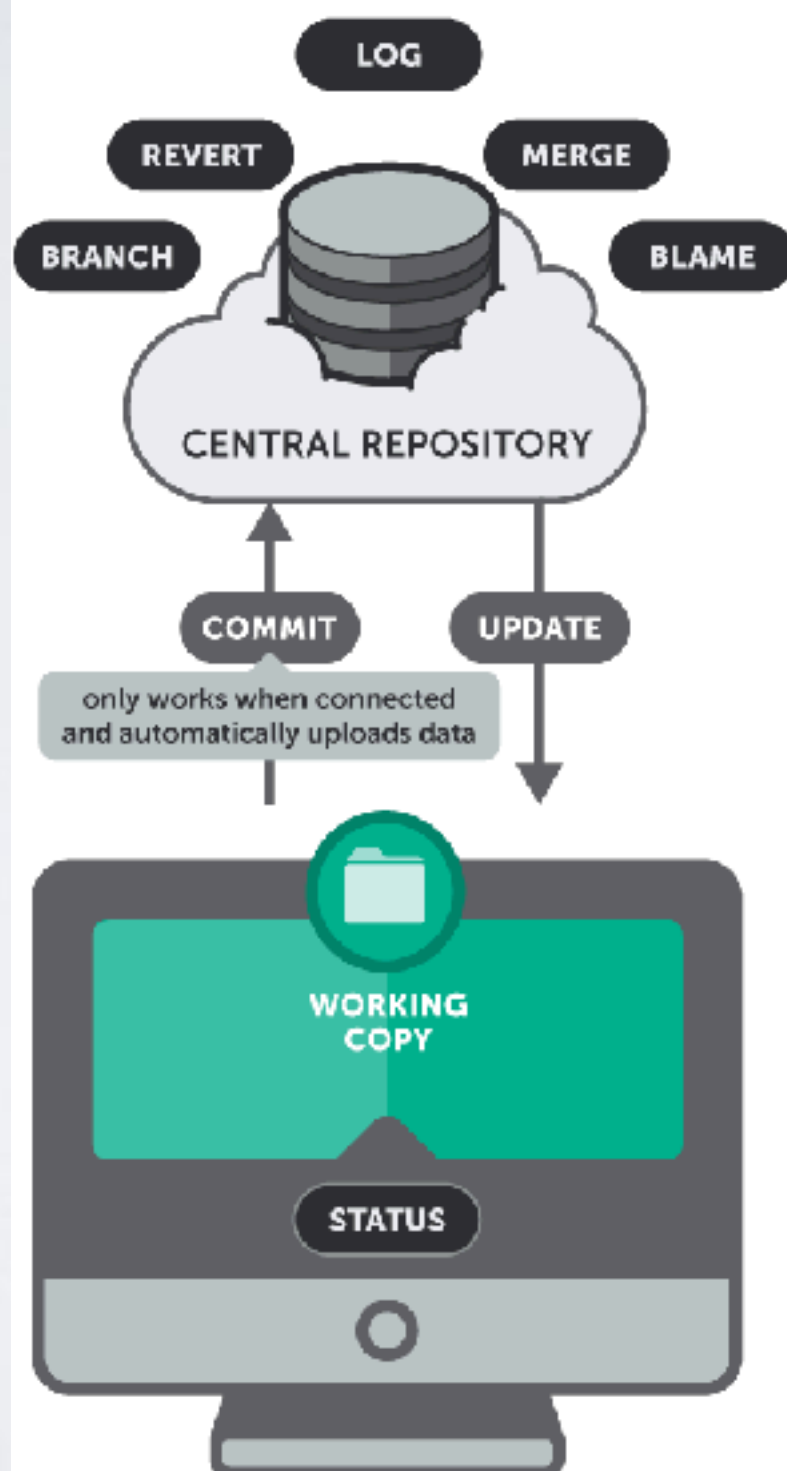
☐ foobar:/branches/branch1

☐ foobar:/branches/branch2

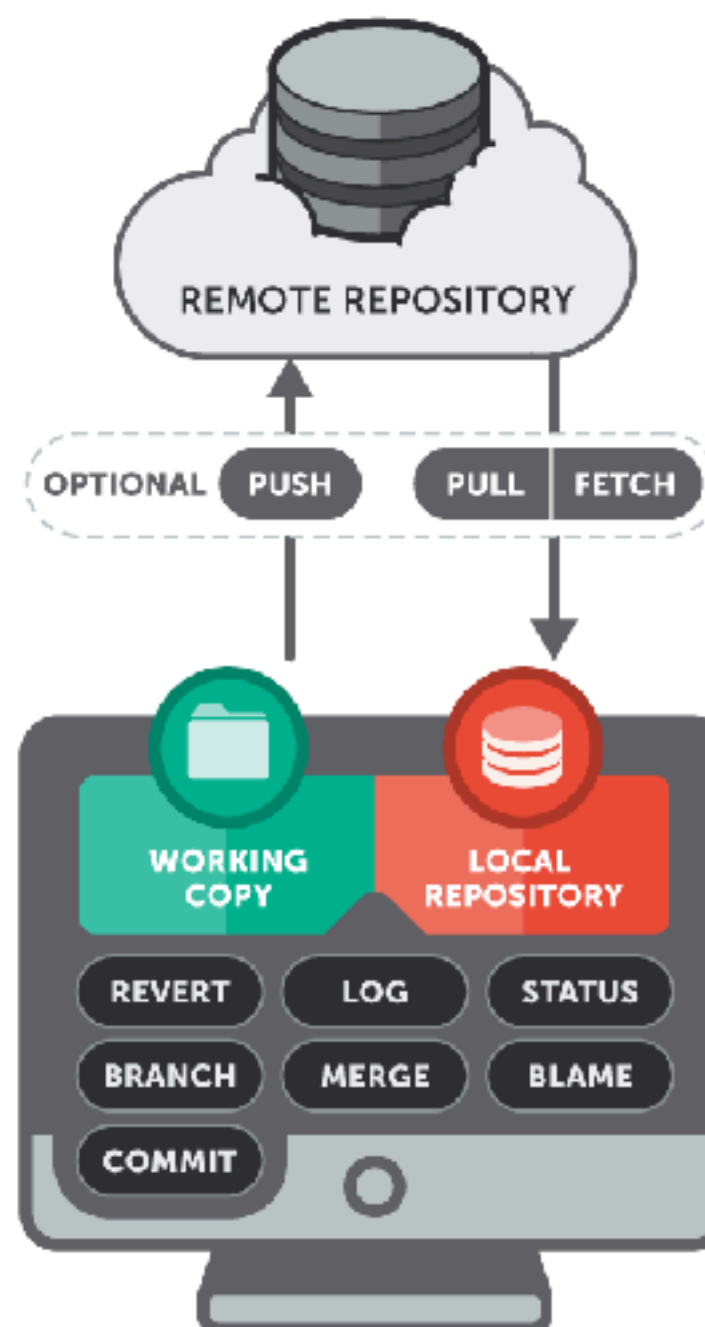
- La clave para diferenciar un CVCS respecto a un control de versiones distribuido como Git, es que toda la historia y todo el versionado se hace en el servidor. Si dicho servidor fallara , y no se tuviera respaldo de todo lo anterior, se perdería todo el histórico de cambios.
- En cambio , en Git , cuando accedes a un repositorio mediante **clone** , te estás descargando un **snapshot** (una instantánea) de todo el repositorio con toda su historia hasta el momento en el que lo estás descargando. Podríamos verlo como que el servidor es una copia de respaldo , y toda la gente que trabaja en ese repositorio , va “sincronizando” esos cambios poco a poco, además de poder trabajar sin necesidad de conexión , ya que toda la actividad sobre la copia que tengamos es local, y no se perdería nada.



SUBVERSION



GIT



- Veamos cómo instalar git en los distintos sistemas operativos.

Linux

Desde la línea de comandos, y dependiendo de tu distribución (tu gestor de paquetes):

- Debian/Ubuntu

```
apt-get install git-core
```

- RedHat/CentOS

```
yum install git-core
```

Mac (OSX)

Para mac, vamos a emplear *Homebrew* que es un gestor de paquetes muy útil, para tener acceso a paquetes de distribuciones linux.

Si no tenemos instalado aún *homebrew*, lo instalaremos con la siguiente línea de comandos (**no hacerlo desde root**):

```
/usr/bin/ruby -e "$(curl -fsSL https://  
raw.githubusercontent.com/Homebrew/install/master/  
install)"
```

Una vez instalado *Homebrew* , podemos instalar git con esta línea :

```
brew install git
```


Windows

- En windows, vamos a instalar un terminal mejorado llamado **Cmdr** , que nos incorpora git de serie, así como otros comandos básicos de linux.
- Para instalar *Cmdr* con git incluido , debemos descargarnos el siguiente enlace :

<https://github.com/cmdrdev/cmdr/releases/>

- **Git** surgió como solución a los desarrolladores del *Kernel de Linux* en 2005. Necesitaban una herramienta en la que multitud de desarrolladores pudieran colaborar con la misma base de código. Anteriormente se trabajaba con parches de código que se pasaban en una lista de distribución y que hacía todo el proceso muy complejo y con una alta probabilidad de error.
- Antes de desarrollar Git , desde 2002 , empezaron a utilizar una herramienta llamada *BitKeeper* que en sus inicios era gratuita, pero dejó de serla , y la comunidad decidió el crear su propia herramienta : *Git* .



Linus Torvalds

Estados

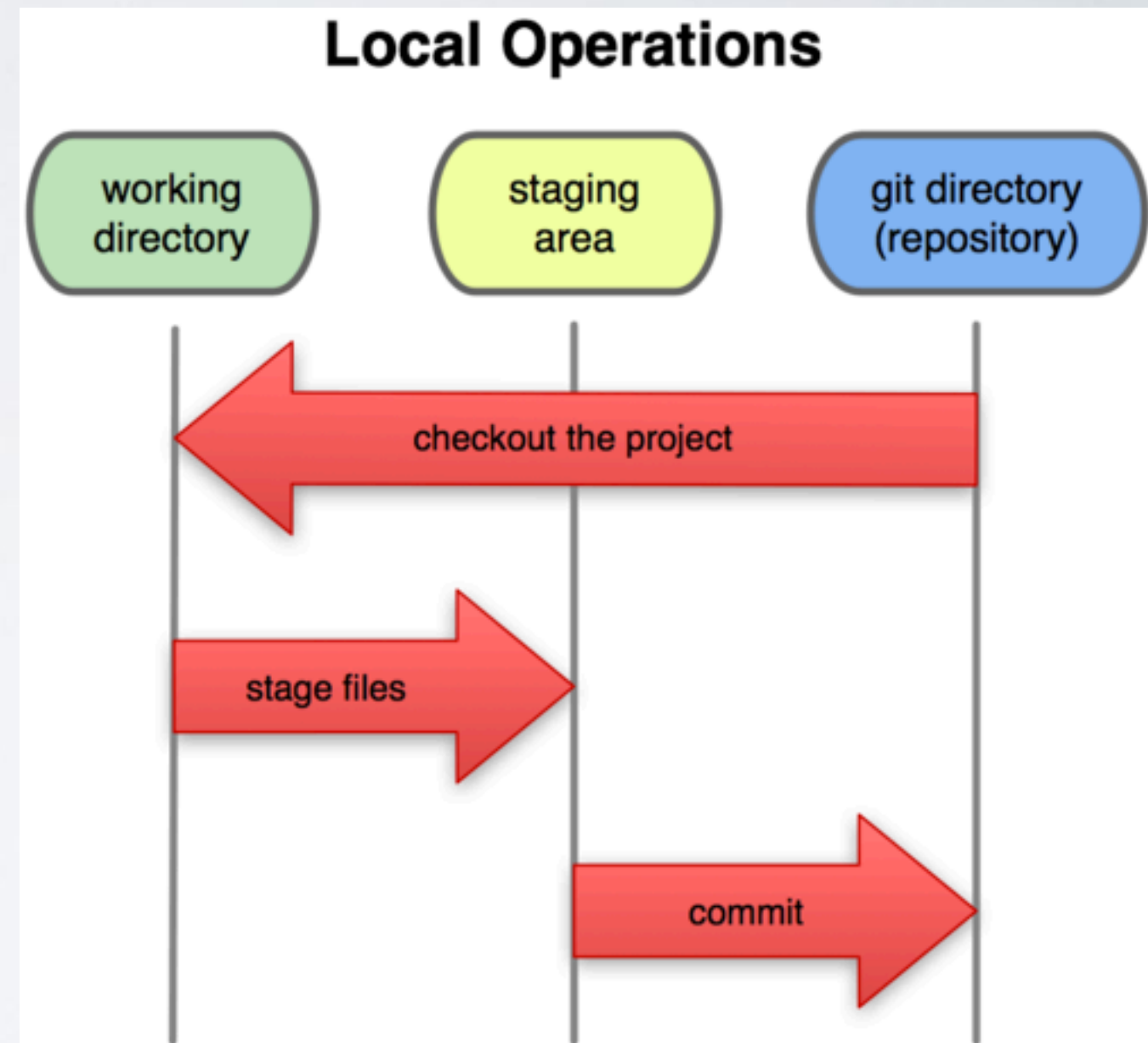
- Vamos a diferenciar 3 estados por los que pasa un fichero cuando está bajo un control de versiones git.
- **I. Git Directory**
 - Cuando nos traemos por primera vez un repositorio a nuestro ordenador (**git clone**) o bien iniciamos un nuevo repositorio (**git init**), lo que estamos haciendo es crear un directorio oculto .git donde se van a almacenar todos los metadatos y actividad de los cambios que haya en esa carpeta.
 - Dicho directorio contiene una serie de metadatos y una **“base de datos” local** eficiente y transparente para el usuario, con todo nuestro histórico de cambios.

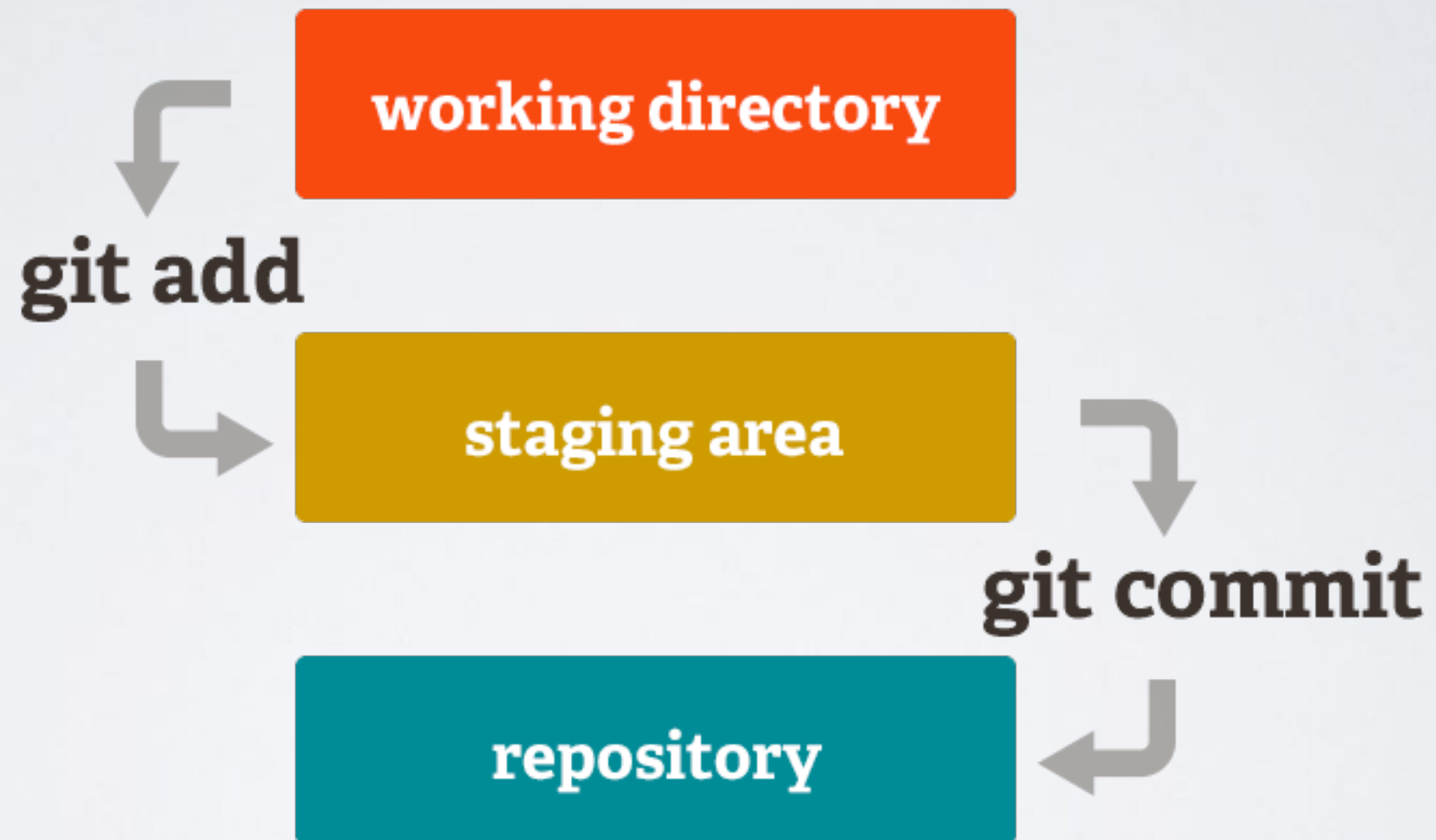
- **2. Working Directory**

- El directorio de trabajo no es más que una copia de la versión más reciente almacenada en la “base de datos de git” y expuesta en el sistema de ficheros para que podamos trabajar con ella de forma transparente.

- **3. Staging Area**

- El área de preparación es un archivo contenido en tu directorio *git* , en el que se almacena toda la información que va a llevar nuestra nueva versión (Qué ficheros se han añadido, modificado o borrado). El área de preparación es el preludio de un **commit** .







- Disponible en el aula virtual

GitHub

HOJA DE REFERENCIA PARA GITHUB GIT

HOSTING DE GIT

En este aspecto podemos destacar dos soluciones gratuitas:



- **Github** : gratuito para Repositorios públicos de código abierto.



- **BitBucket** : gratuito para repositorios públicos y privados (con un límite de invitación de usuarios en el caso de un repositorio privado).
- También tienes la posibilidad de montar tu propio hosting de repositorios git, pero te tocará a tí administrarlo (Disponibilidad , Notificaciones , etc). En este caso la opción más frecuente suele ser **GitLab** .



GIT



ME VOY A CREAR UNOS REPOS