

# PHPVIGO



# git





Ingredients:  
Pork with Ham,  
Salt, Water,  
Modified Potato  
Starch, Sugar,  
Sodium Nitrite

# SPAM<sup>®</sup>

Classic

U.S.  
INSPECTED  
AND PASSED BY  
DEPARTMENT OF  
AGRICULTURE

Serving  
Suggestion

NET WT  
12 OZ  
(340g)



jesus@jesusamieiro.com







# QUADRALIA



# Contenido

- Introducción
- Ramas, fusión y conflictos
- Repositorios remotos
- Flujos de trabajo
- Git flow



¿Qué es?

# Problemas



Presupuesto\_v2.doc



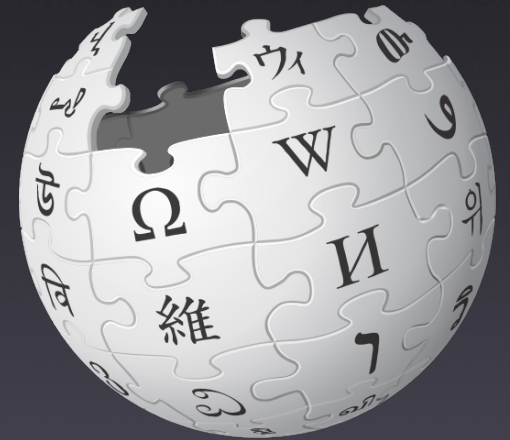
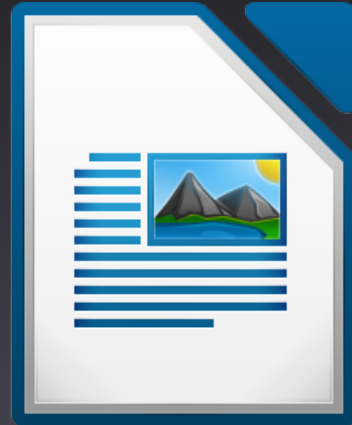
Cartel\_v5.jpg



2010\_05\_17\_web



# Soluciones

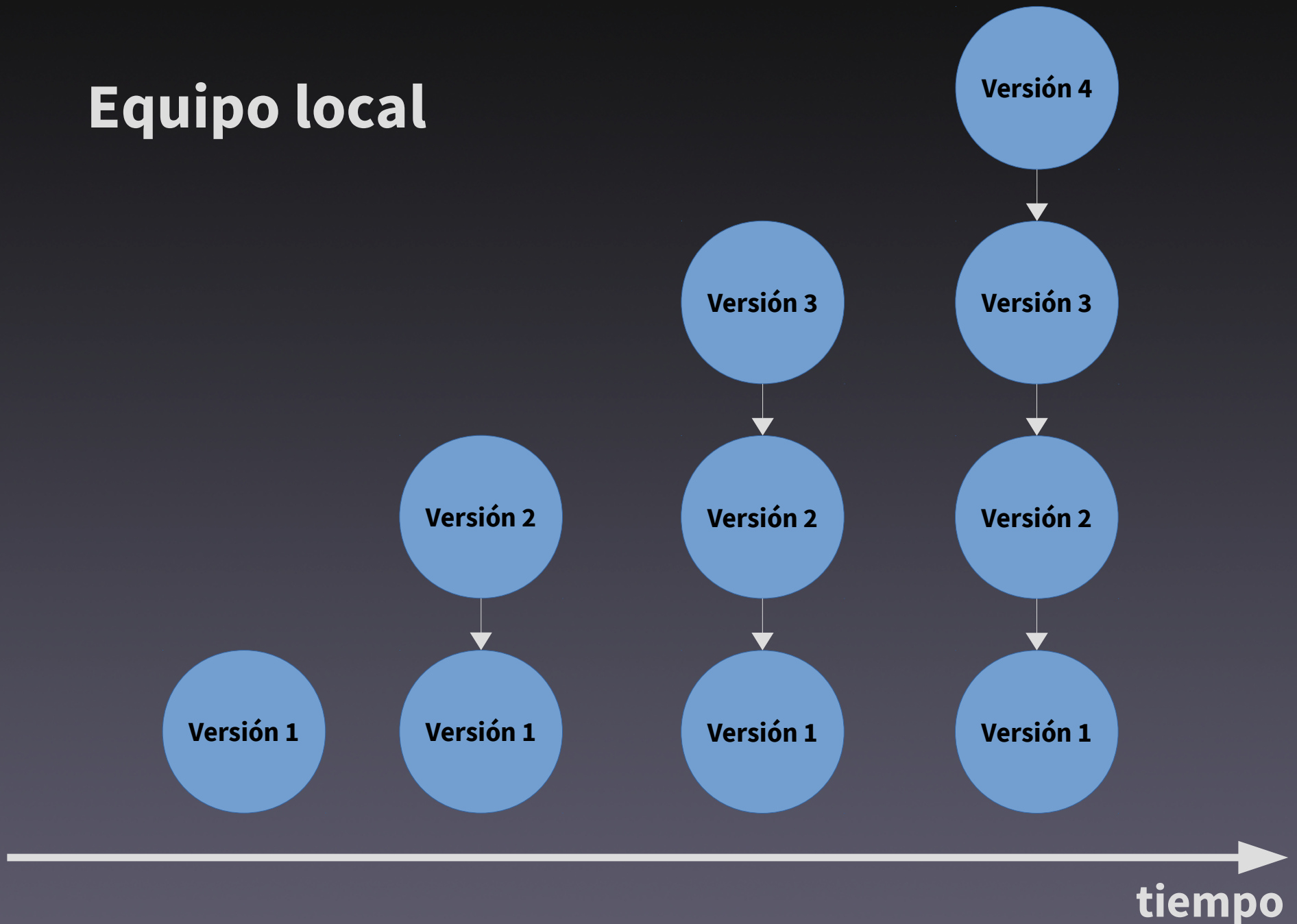


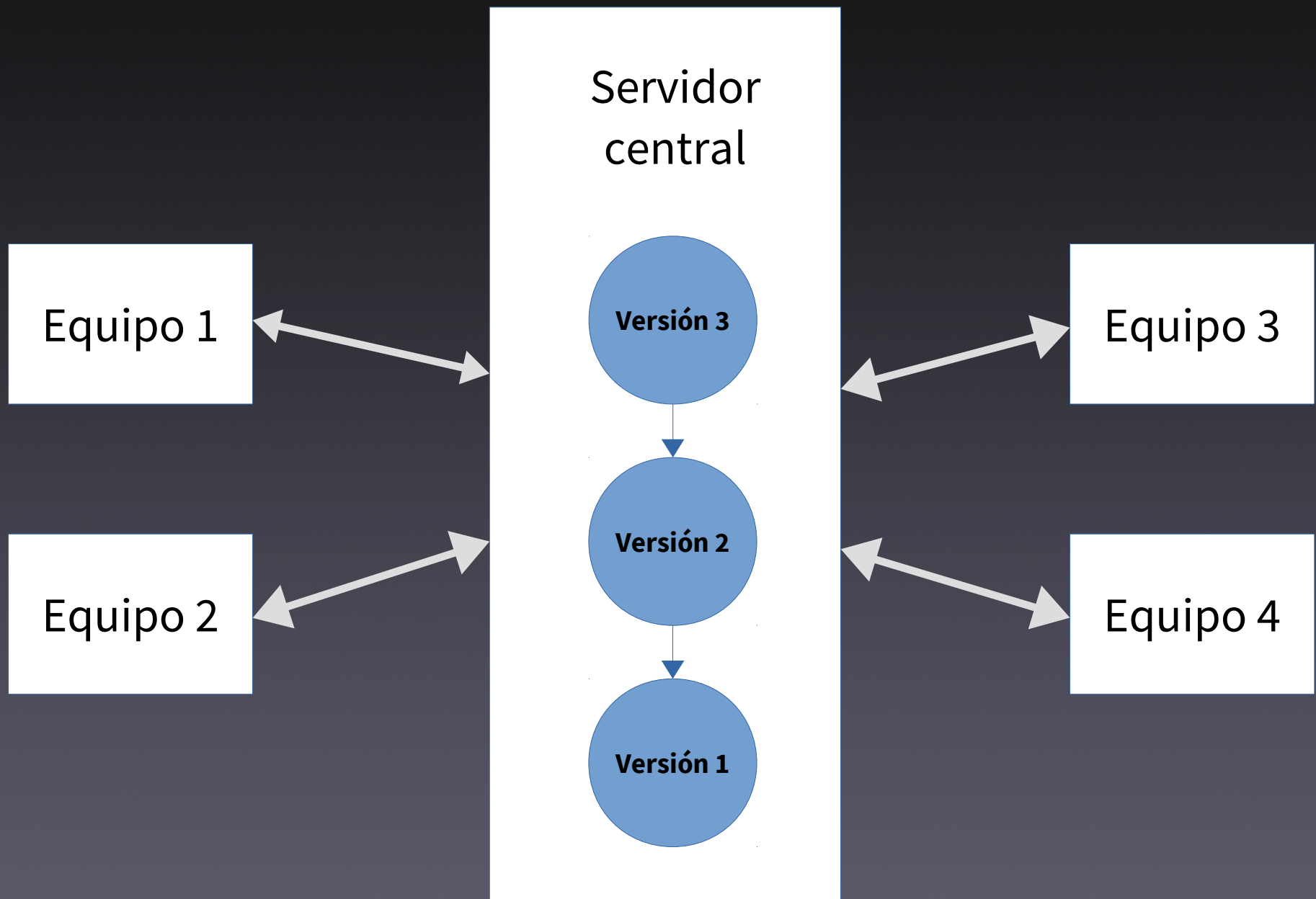
VCS / SCM



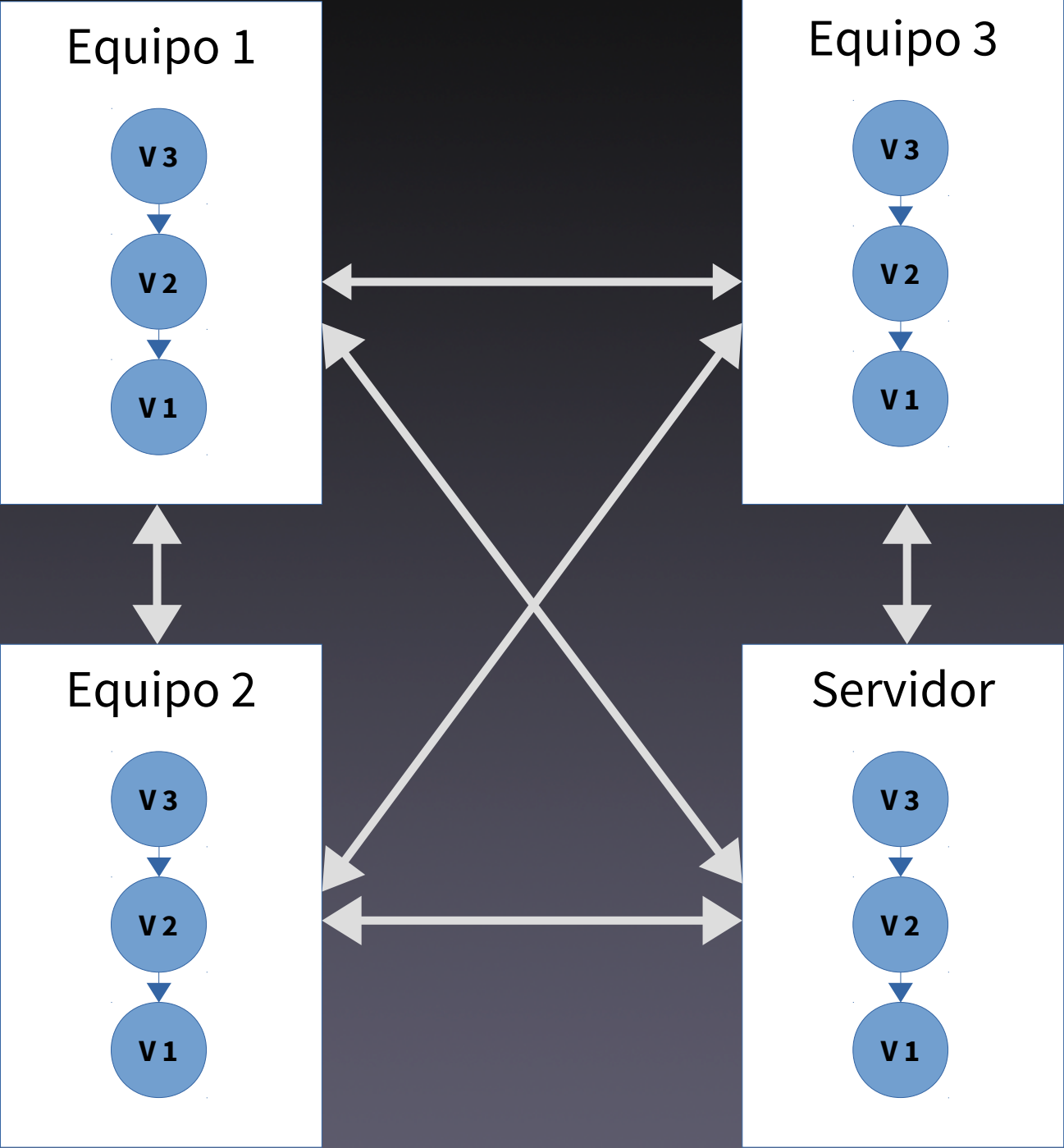
# Tipos de SCM

# Equipo local









# Git. Características

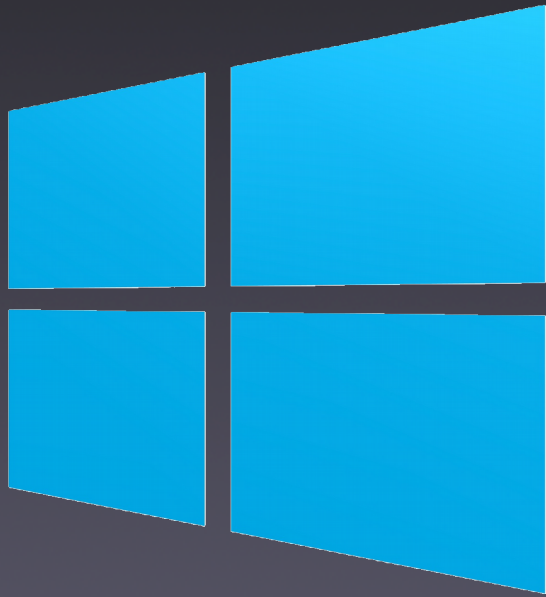
- Rápido y escalable
- Copia completa
- Desarrollo distribuido
- Trabajo local
- Alienta las ramas
- Instantáneas

# Git. Características (II)

- Múltiples protocolos
- Robustez: SHA-1
- Libre
- Gratuito



# Versiones



# Cliente

- Consola
- GUI
- IDE

# Servidor

- SaaS
  - GitHub
  - Bitbucket
  - GitLab
- Servidor
  - GitHub Enterprise
  - Bitbucket Server
  - GitLab
  - Gitolite

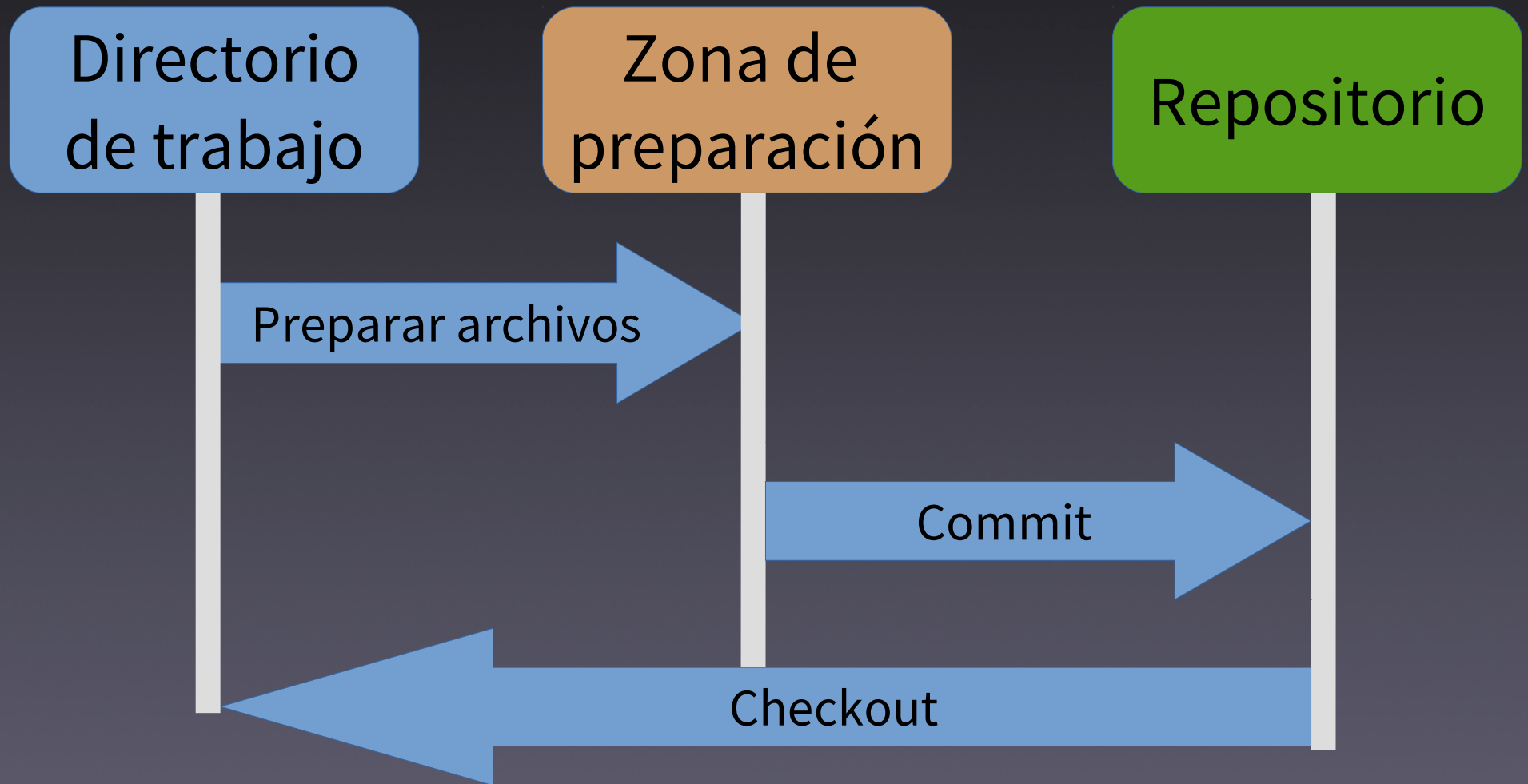


# Conceptos básicos

# Repositorio

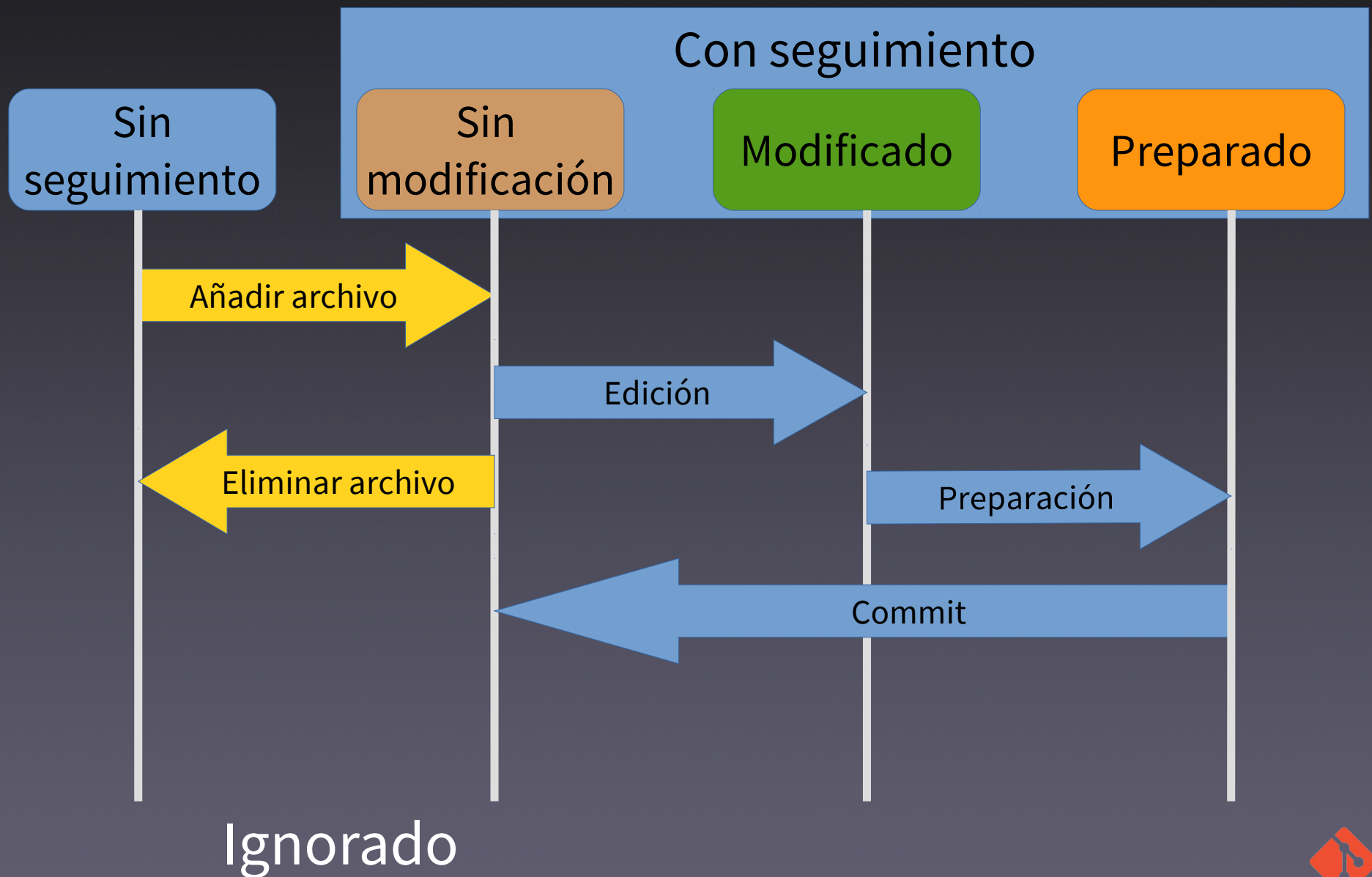
# Commit

# Zonas en Git

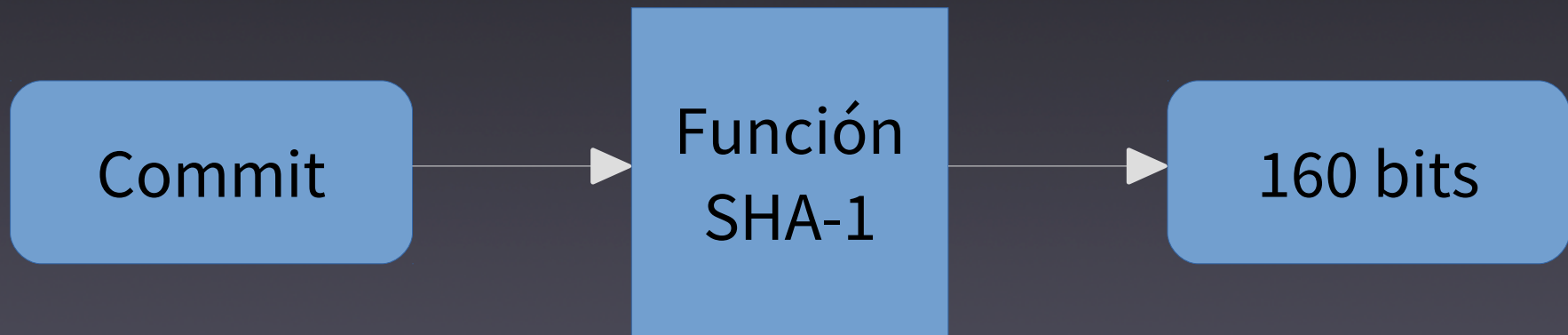




# Estados y flujo



# SHA-1



# HEAD

Commit 1



Commit 1



Commit 2



Commit 1



Commit 2



Commit 3



# Comandos básicos

- git config
  - git config --global user.name "Jesús Amieiro"
- git init
- git clone
- git status
- git diff
- git log

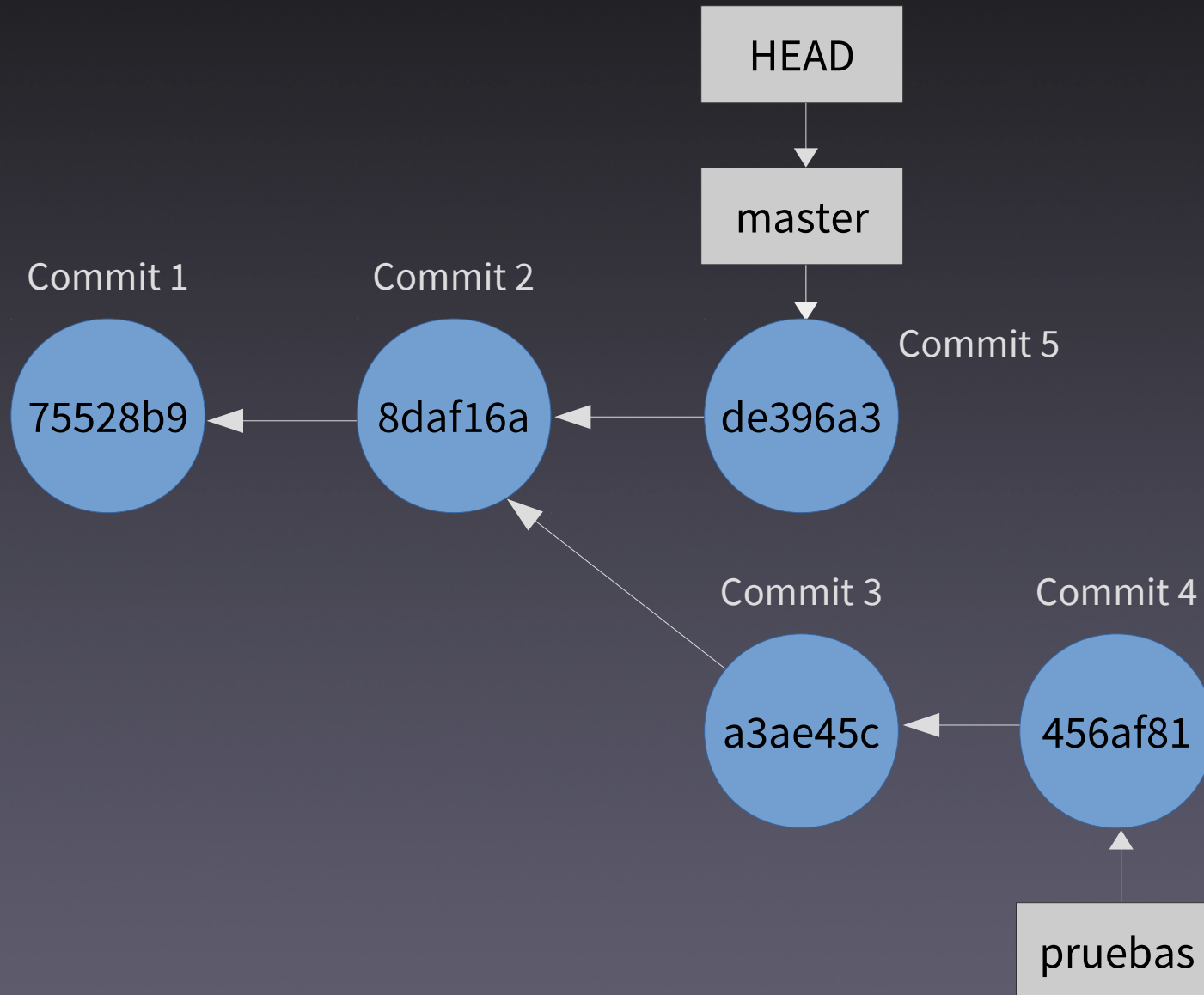


# Comandos básicos (II)

- `git add`
  - `git add .`
- `git commit`
  - `git commit -m "Commit inicial"`

# Ramas

# Rama



# Comandos ramas

- `git branch`
- `git branch [nombre-rama]`
- `git checkout [nombre-rama]`
- `git checkout -b [nombre-rama]`
- `git branch -d [nombre-rama]`
- `git merge [nombre-rama]`



# Conflictos fusión

- Abortar la fusión
  - `git merge --abort`
- Resolver manualmente
- Herramientas fusión
  - Meld
  - P4merge
  - KDiff3

# Conflictos fusión (II)

```
$ git merge pruebas
```

```
Auto-merging archivo_a.txt
```

```
CONFLICT (content): Merge conflict in archivo_a.txt
```

```
Automatic merge failed; fix conflicts and then  
commit the result.
```

# Conflictos fusión (III)

```
$ cat archivo_a.txt
```

```
<<<<<<< HEAD
```

Experimento añadiendo una nueva línea al archivo\_a.txt en la rama experimento

Añado una segunda línea al archivo\_a.txt en la rama master

```
=====
```

Inserto una línea en el archivo\_a.txt

```
>>>>>>> pruebas
```

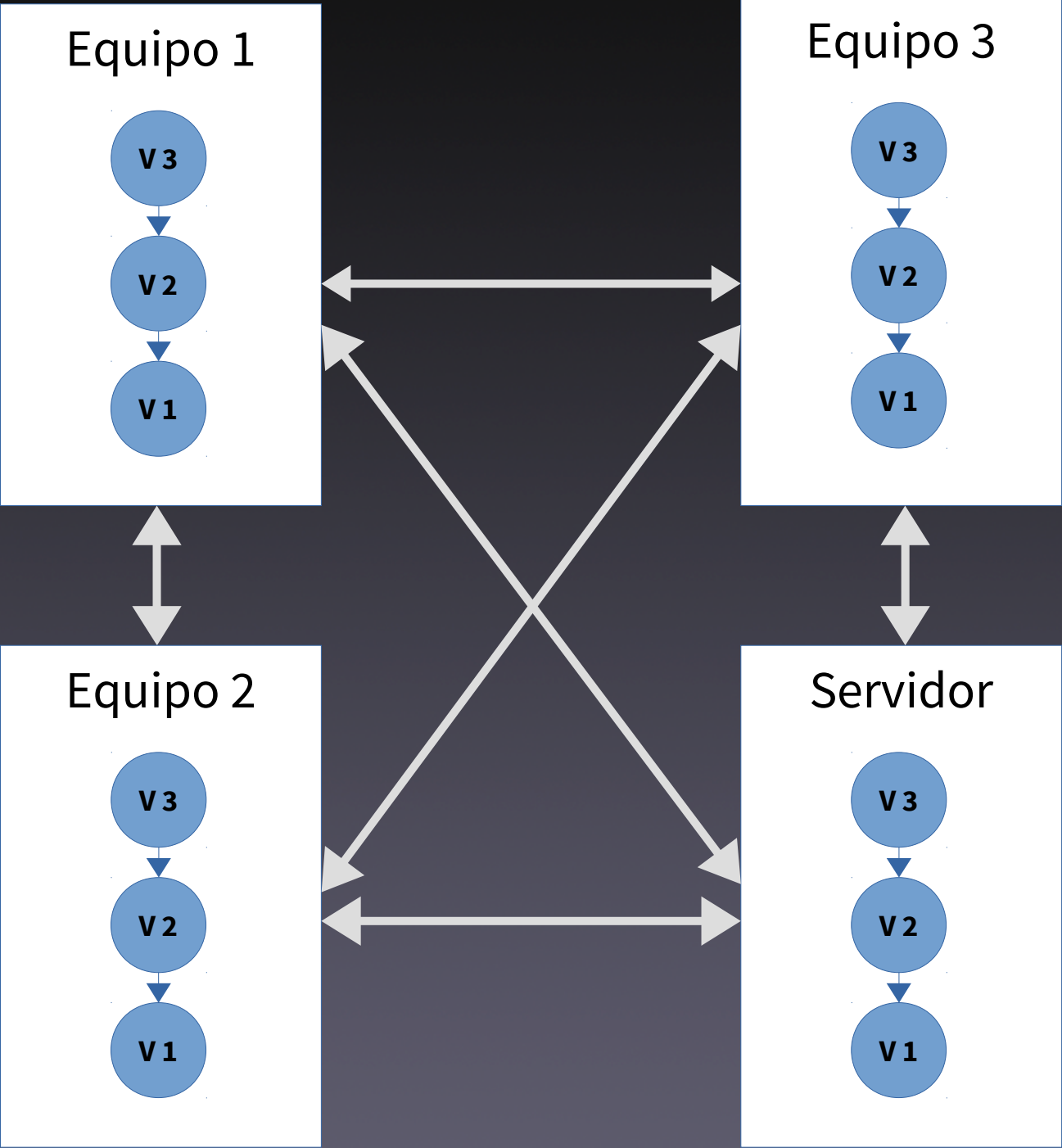
# Conflictos fusión (IV)

```
$ git add archivo_a.txt
```

```
$ git commit -m "Resuelto el conflicto en la línea 1 en  
el archivo_a.txt"
```



# Repositorios remotos



# Comandos repositorios

- `git remote`
  - `git remote add origin git@bitbucket.org:amieiro/proyectocompartido.git`
- `git remote -v`
  - `origin git@bitbucket.org:amieiro/proyectocompartido. (fetch)`
  - `origin git@bitbucket.org:amieiro/proyectocompartido. (push)`

# Comandos repositorios (II)

- `git fetch`
  - `git fetch`
  - `git branch -a -v`
  - `git checkout dev`
  - `git merge origin/dev`



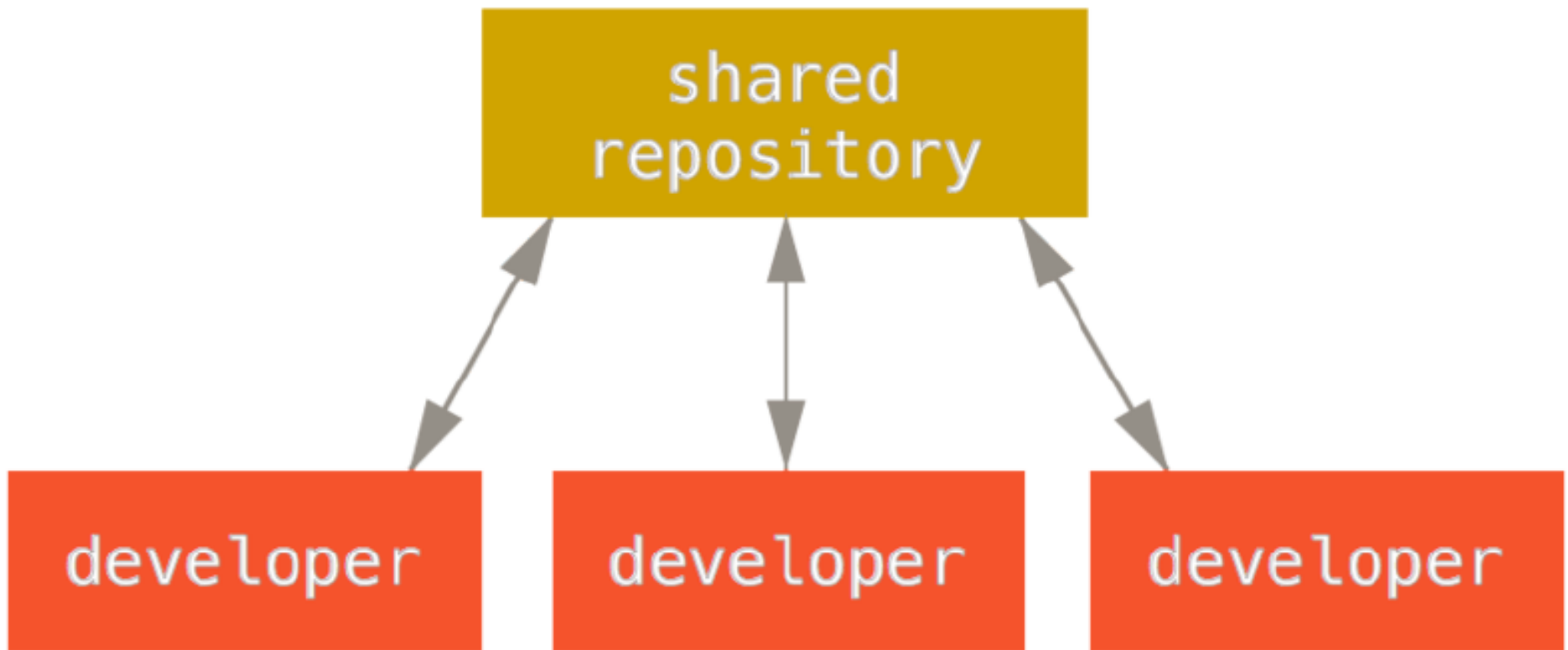
# Comandos repositorios (III)

- `git pull`
  - `git checkout master`
  - `git pull`
  - `git pull --all`

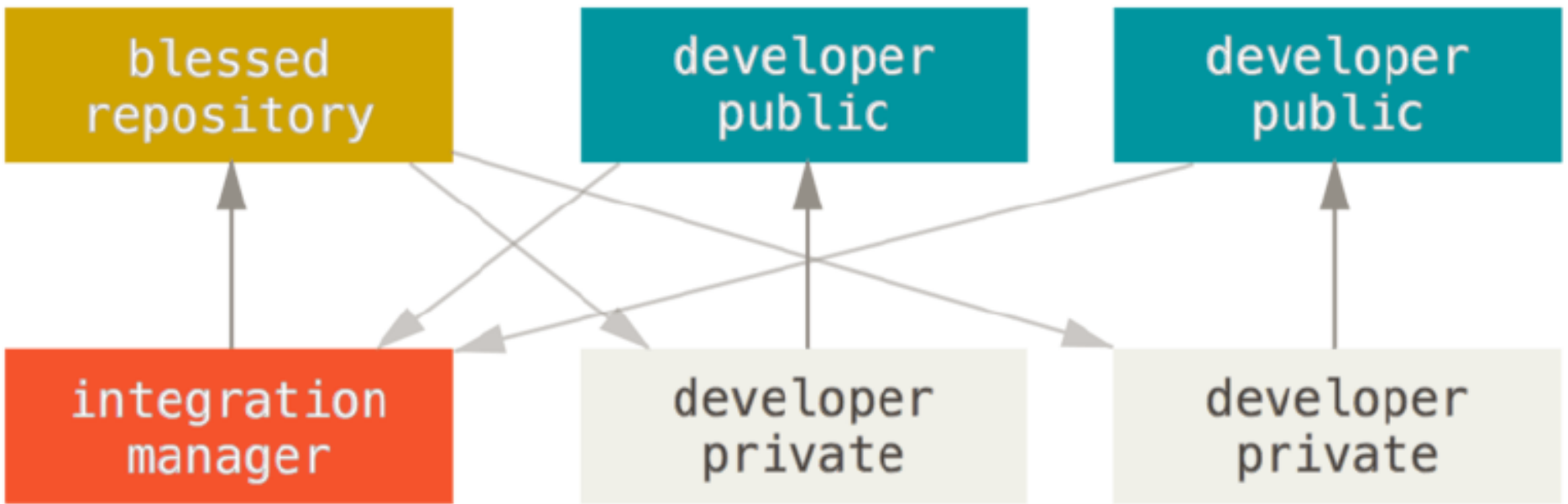
# Comandos repositorios (IV)

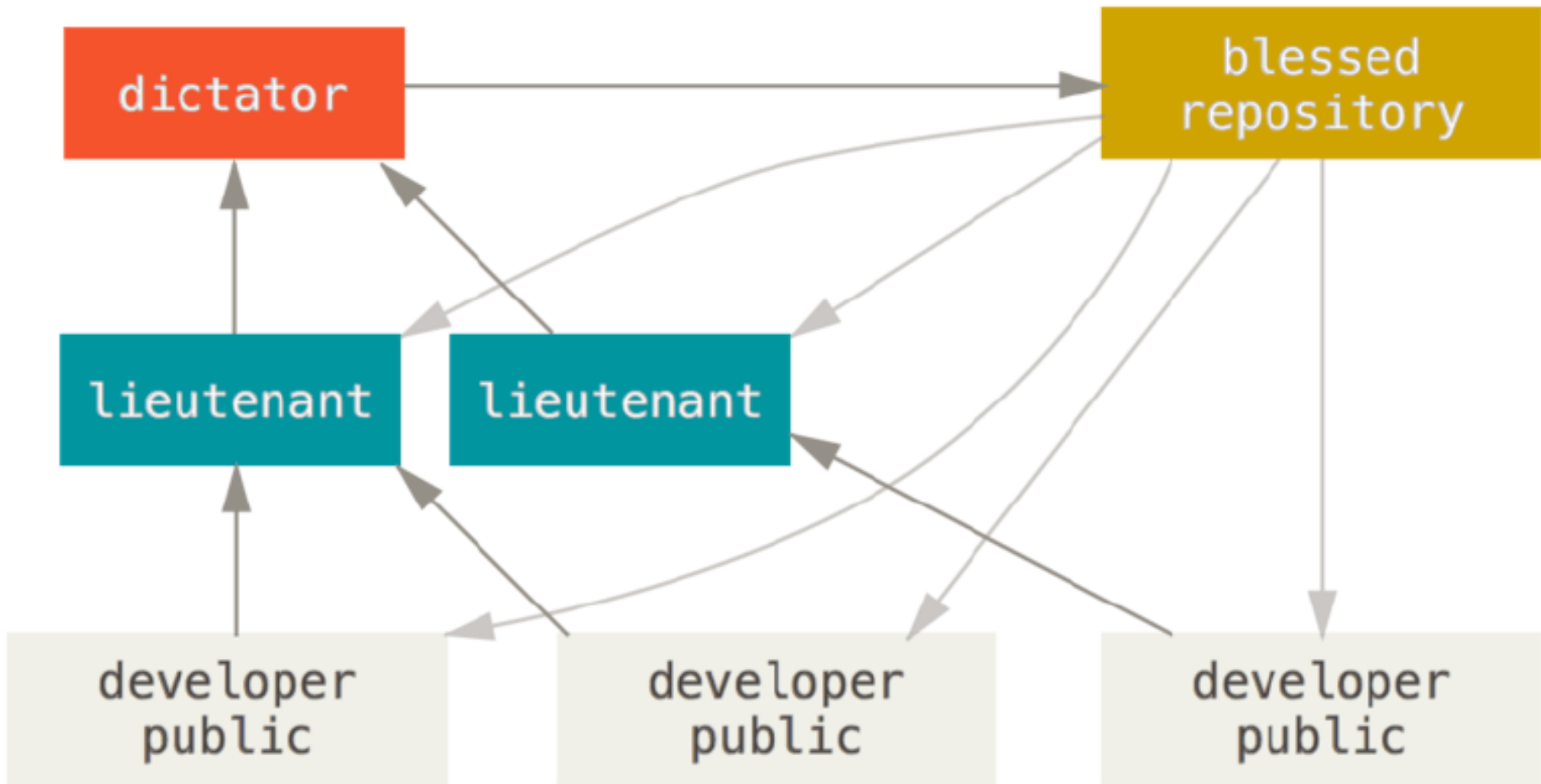
- `git push`
  - `git push -u origin master`
  - `git push`
  - `git push --all`
  - `git push -tags`
- `pull` → `push`
- `fetch` vs `pull`

# Flujos de trabajo habituales









# Git flow

# Git flow

- Repositorio organizado
- Procedimientos más claros
- Estructuras familiares entre proyectos



# Git flow (II)

- Rama de producción: master
- Rama de desarrollo: develop
- Rama de característica: feature/
- Rama de publicación: release/
- Rama de revisión: hotfix/

## Initialise repository for Git Flow

Create / use the following branches:

Production branch:

Development branch:

Use the following prefixes in future:

Feature branch prefix:

Release branch prefix:

Hotfix branch prefix:

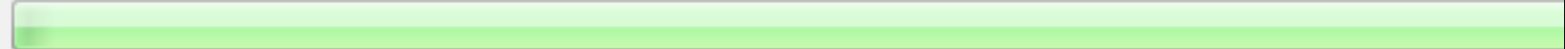
Version tag prefix:

Use Defaults

OK

Cancel

## Initialise repository for Git Flow



Show Full Output

```
sh.exe C:\Users\...\AppData\Local\Atlassian\SourceTree\gitflow_local\gitflow\git-flow init -d  
Using default branch names.
```

Which branch should be used for bringing forth production releases?

- master

Branch name for production releases: [master]

Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?

Feature branches? [feature/]

Release branches? [release/]

Hotfix branches? [hotfix/]

Support branches? [support/]

Version tag prefix? []

Completed successfully.

## Choose Next Flow Action

Recommended actions:

Start New Feature

Start New Release

Start New Hotfix

Other Action...

Cancel



## Start New Feature

Feature Name:

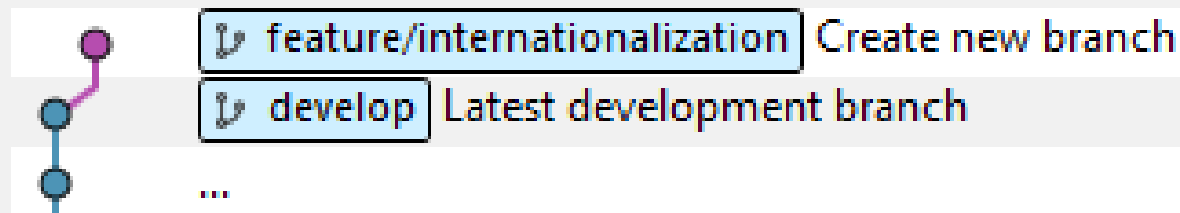
Start at:  Latest development branch

Working copy parent

Specified commit:

...

### Preview



OK

Cancel

## Start New Feature

Show Full Output

```
sh.exe C:\Users\████████\AppData\Local\Atlassian\SourceTree\gitflow_local\gitflow\git-flow feature start internationalization  
Switched to a new branch 'feature/internationalization'
```

Summary of actions:

- A new branch 'feature/internationalization' was created, based on 'develop'
- You are now on branch 'feature/internationalization'

Now, start committing on your feature. When done, use:

```
git flow feature finish internationalization
```

Completed successfully.

The screenshot displays a Git GUI interface with a sidebar on the left and a main graph area on the right. The sidebar contains the following sections:

- File Status**: Working Copy
- Branches**: develop, feature (subfolder), internationalization (selected), master
- Tags**
- Remotes**: origin (subfolder), master

The main graph area is titled "Graph" and includes the following controls at the top:

- Dropdown menu: All Branches
- Checkbox: Show Remote Branches (unchecked)
- Dropdown menu: Date Order

The graph shows a vertical sequence of commit nodes connected by a blue line. The top three nodes are highlighted with a blue background and labeled in tabs: feature/internationalization, master, and develop. Below these, several other commit nodes are visible, with a pink line indicating a branch point or merge point near the bottom of the graph.





## Choose Next Flow Action

Current state:

Feature: internationalization

Recommended actions:

Finish Feature

Other Action...

Start New Feature

Finish Feature

Start New Release

Finish Release

Start New Hotfix

Finish Hotfix

## Finish Feature

Feature Name: internationalization

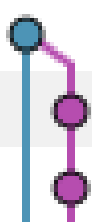
### Options

Rebase on development branch

After finishing:

Delete branch  Force deletion

### Preview



 **develop** Merge feature/internationalization into develop

Latest feature branch

...

OK

Cancel

## Finish Feature

Show Full Output

```
sh.exe C:\Users\... \AppData\Local\Atlassian\SourceTree\gitflow_local\gitflow\git-flow feature finish -D internationalization  
Switched to branch 'develop'
```

```
Updating 72784c0..2e12828  
Fast-forward
```

```
values/strings.xml | 0  
1 file changed, 0 insertions(+), 0 deletions(-)  
create mode 100644 values/strings.xml
```

```
Deleted branch feature/internationalization (was 2e12828).
```

### Summary of actions:

- The feature branch 'feature/internationalization' was merged into 'develop'
- Feature branch 'feature/internationalization' has been removed
- You are now on branch 'develop'

Completed successfully.

File Status

Working Copy

Branches

- develop
- master

Tags

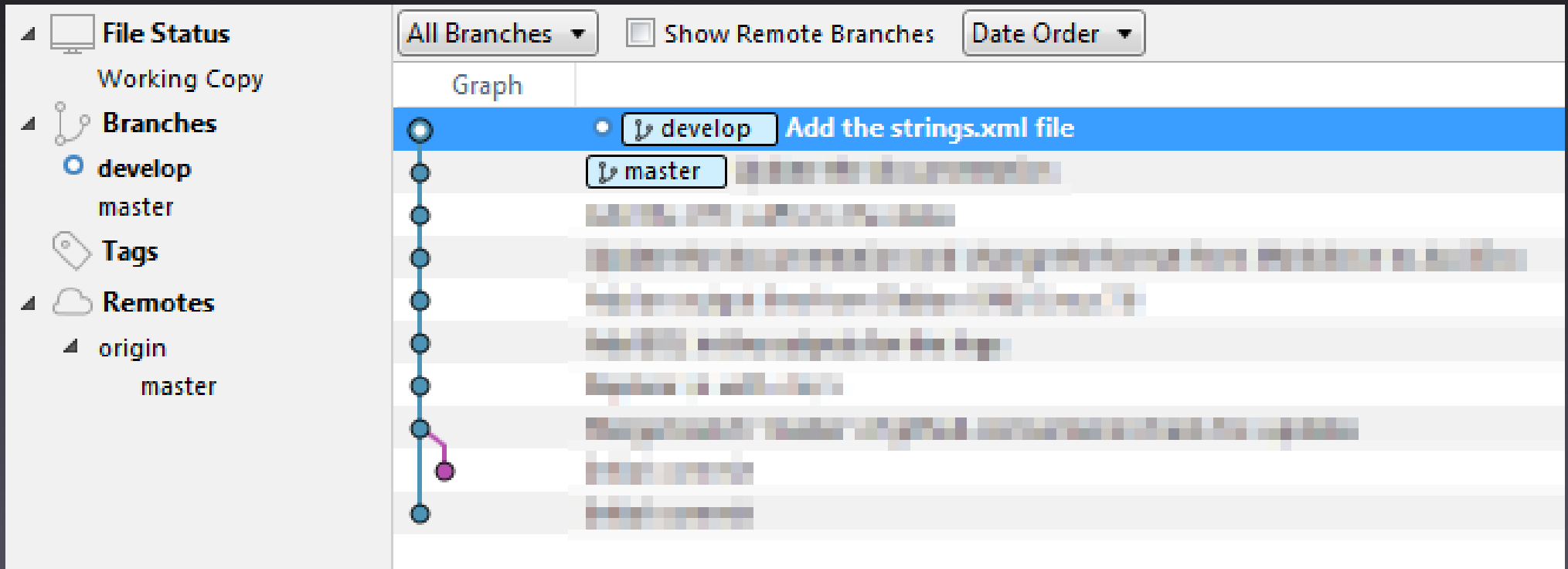
Remotes

- origin
  - master

All Branches  Show Remote Branches Date Order

Graph

- develop Add the strings.xml file
- master





# Git flow. Comandos

- <https://github.com/nvie/gitflow/wiki/Command-Line-Arguments>
- `git flow init`
- `git flow feature`
- `git flow feature start <name>`  
`[<base>]`
- `git flow feature finish <name>`

# Git flow. Comandos (II)

- `git flow feature publish <name>`
- `git flow feature pull <remote>`  
`<name>`



*That's all Folks!*

[www.jesusamieiro.com](http://www.jesusamieiro.com)

14/03/2016

