Universidad Europea del Atlántico

www.uneatlantico.es

# Introducción a la Gestión de Proyectos de Software (IGPS)

# Presentación 05 –GIT (1)

**Pablo Herrero García**

# "FINAL".doc



# "FINAL".doc

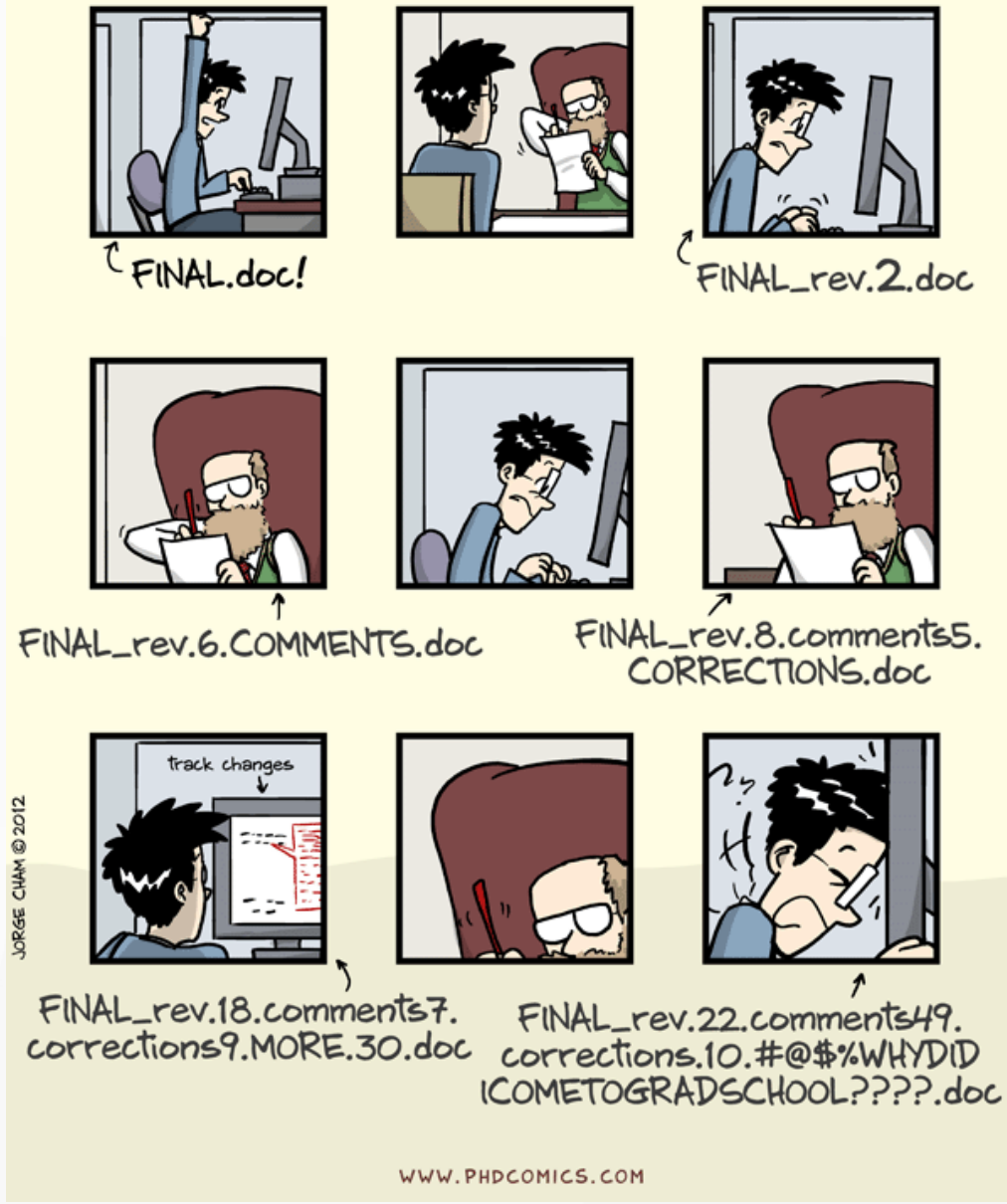
# "FINAL".doc

# "FINAL".doc

# "FINAL".doc

# "FINAL".doc

# "FINAL".doc



# GIT

# GIT

-

## ¿Qué es?

Git es un sistema de control de versiones distribuido gratuito y de código abierto diseñado para manejar todo, desde proyectos pequeños hasta proyectos muy grandes con velocidad y eficiencia.

La característica de Git que realmente lo distingue de casi todos los demás SCM es su modelo de ramificación.

Git te permite y te anima a tener múltiples sucursales locales que pueden ser completamente independientes entre sí. La creación, fusión y eliminación de esas líneas de desarrollo lleva unos segundos.

# GIT

-

## Commit

- Los Commits son las unidades centrales de la línea de tiempo de un proyecto.
- Los Commits pueden considerarse instantáneas o hitos a lo largo de la línea de tiempo de un proyecto Git.
- Los Commits se crean con git commit comando para capturar el estado de un proyecto en ese momento.
- Los Commits de Git siempre se envían al repositorio local.

# GIT

-

## Ramas

Las ramas en GIT son una división del estado del código, esto permite crear nuevos caminos a favor de la evolución del código.

Normalmente la creación de ramas en otros sistemas de control de versiones puede tomar mucho tiempo y ocupar demasiado espacio en el almacenamiento. Por el contrario, las ramas en Git son parte diaria del desarrollo, son una guía instantánea para los cambios realizados.

# GIT

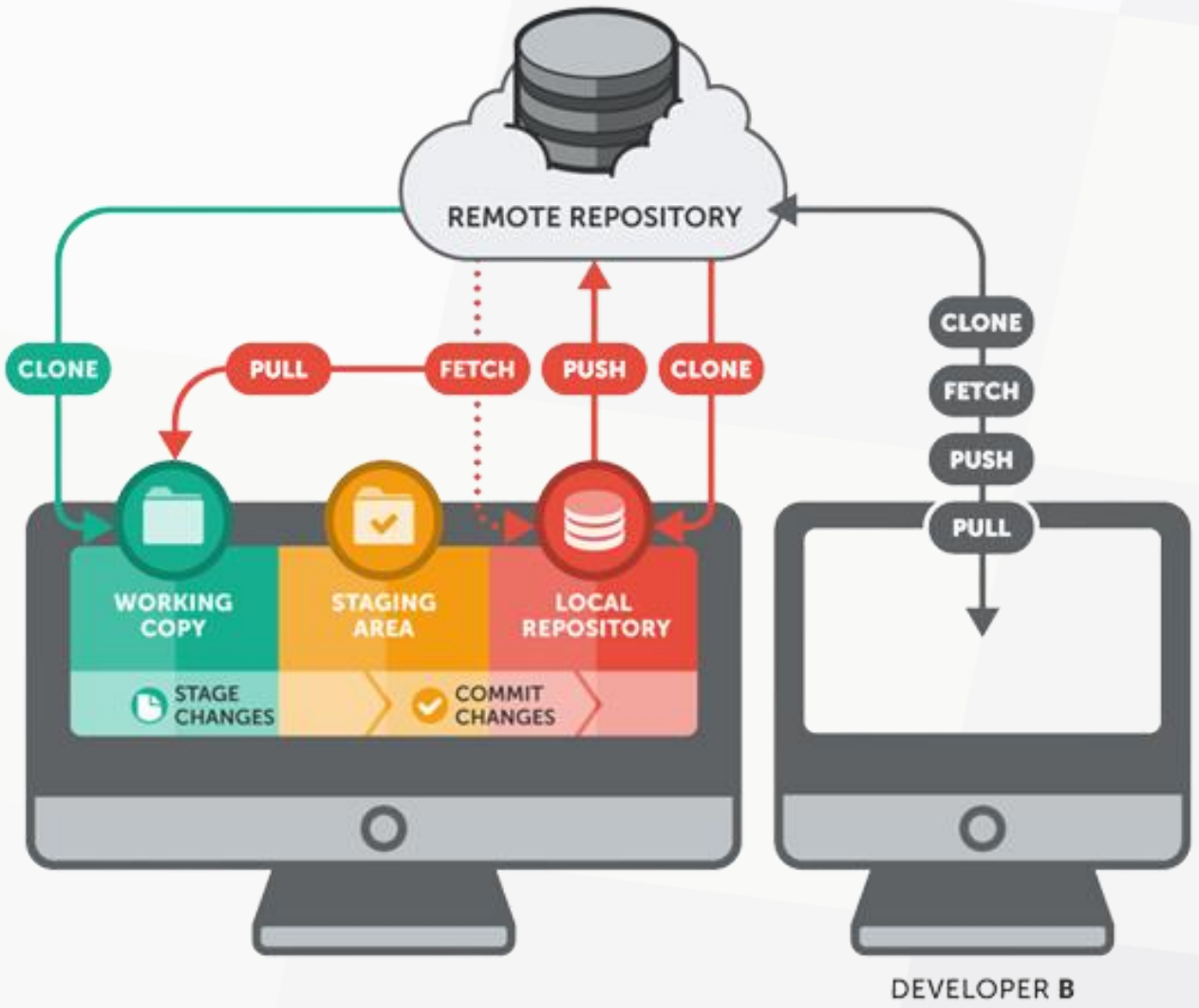
-

## Ramas

# GIT

-

## Repositorios



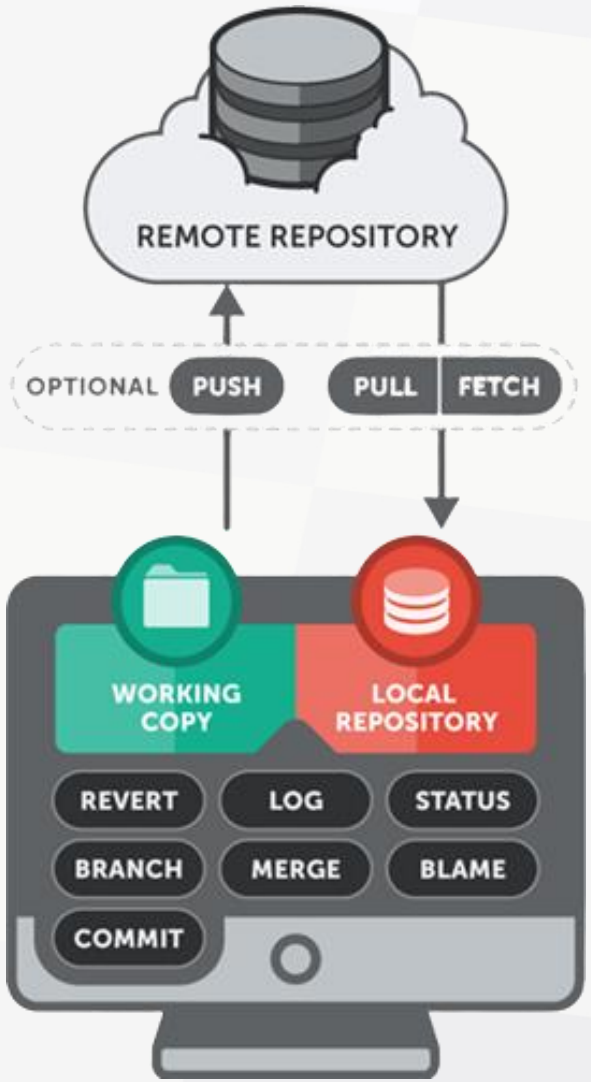
# GIT - Repositorios

- **Repositorio local:** Este repositorio es el que está en nuestra máquina local, y es sobre el cual haremos la mayor cantidad de operaciones. Además de que es el que siempre deberá existir de forma obligatoria, ya que puedo tener un repositorio Git solamente en mi PC sin uno remoto (por ende nunca podré compartir el código asociado).

- **Repositorio remoto:** Este repositorio es el que está en el servidor, permitiendo sincronizar los cambios de los distintos repositorios locales asociados al mismo.

Este esquema implica que todos los cambios y modificaciones que hagamos en el repositorio local no tendrán impacto en el servidor remoto hasta que ejecutemos los comandos de sincronización.





# GIT – Recomendaciones generales

•Hacer commits de forma periódica cuando tengamos una unidad lógica terminada (es algo que deberíamos hacer siempre, independientemente de la plataforma de versionado que utilicemos).

•Trabajar con ramas para separar las distintas features o funcionalidades que desarrollemos.

•Considerar que todo lo que hacemos en nuestro repositorio (excepto push y pull) es 100% local. Eso nos da flexibilidad para hacer determinadas acciones sin consecuencias.

# GIT – Recomendaciones generales

•Tener el entorno de trabajo limpio ante acciones que tienen impacto. Antes de cambiar de rama, de hacer un merge, de hacer un pull debo tener mi entorno de trabajo sin cambios pendientes. Antes debería haber hecho un commit o un stash.

•Cuando trabajas con otras ramas, generalmente para un merge, debes tener actualizado el contenido de la misma localmente con los últimos cambios del servidor, ya que el merge es una operación local.

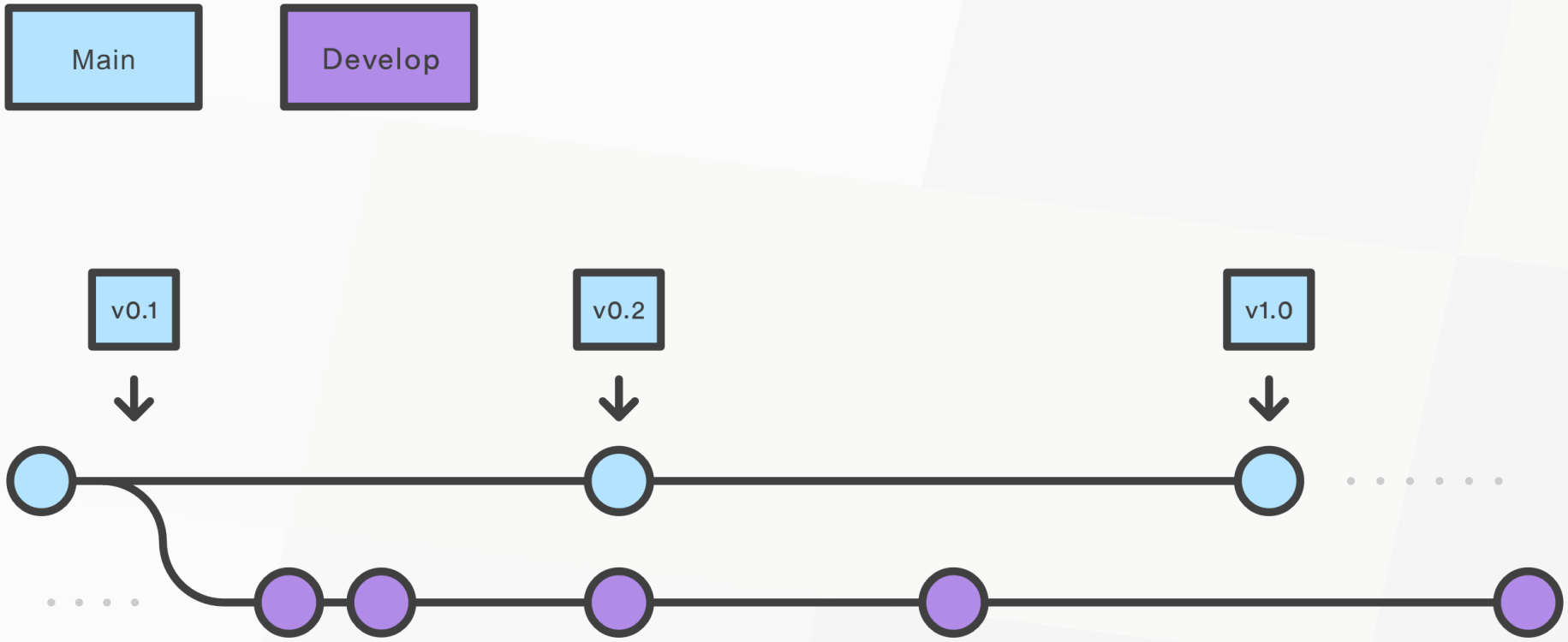
•El estado stage es de cambios, no de archivos. Esto implica que si hice un stage de un archivo y luego continuo editándolo, deberé hacer stage nuevamente si es que quiero hacer un commit de los mismos.

# GIT

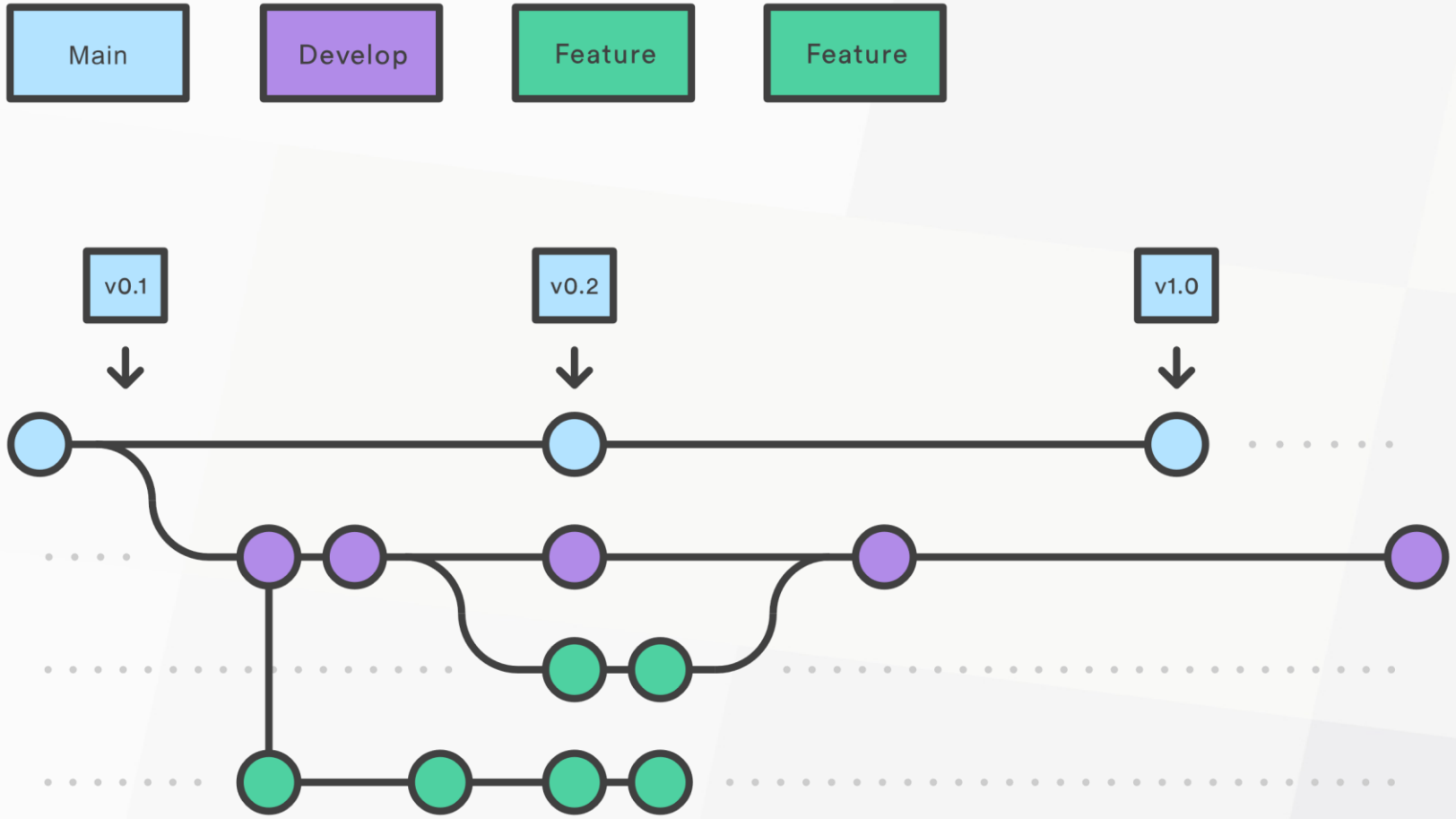
-

## Flujo de trabajo

# GIT — Flujo de trabajo

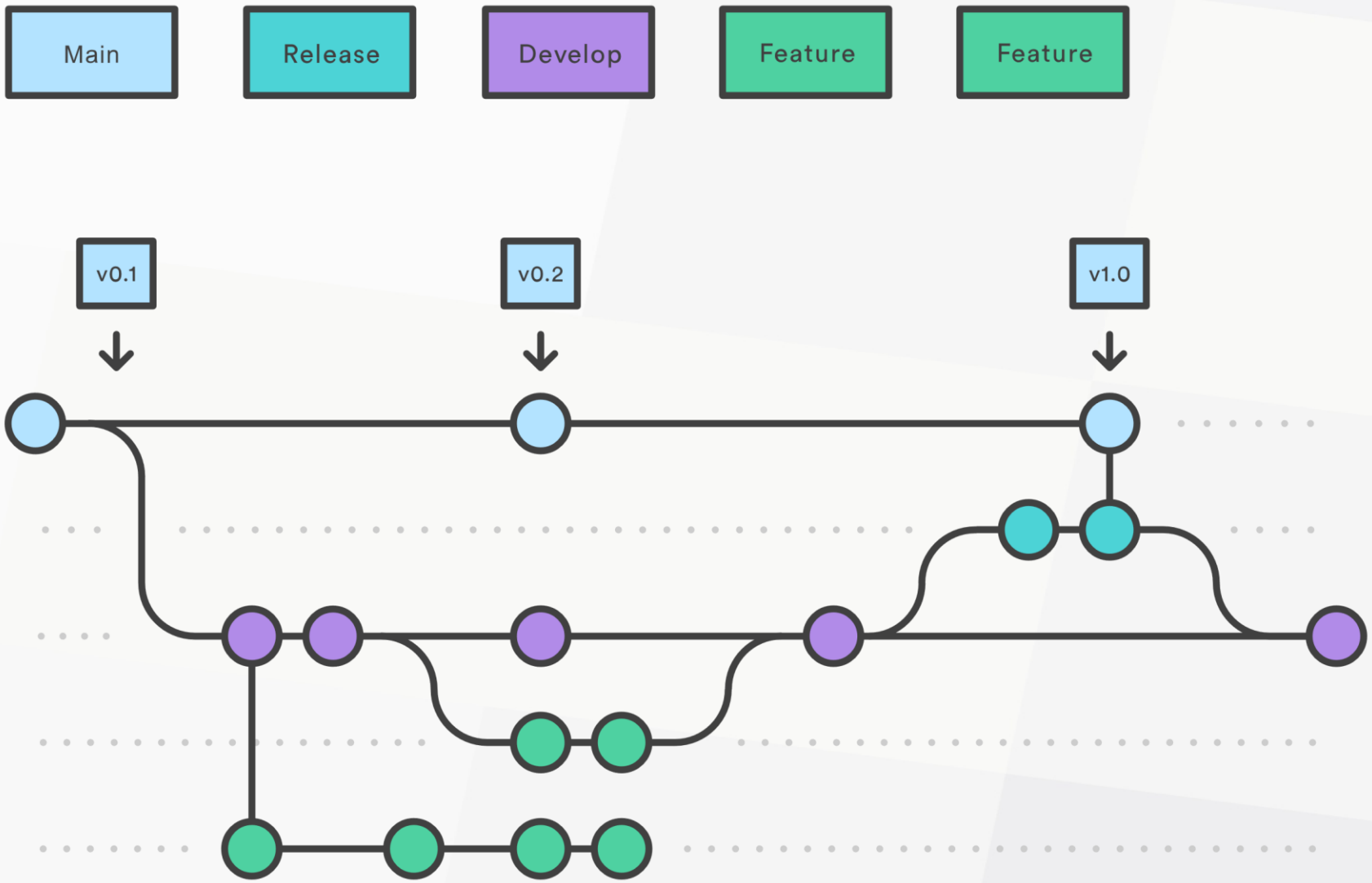


# GIT — Flujo de trabajo



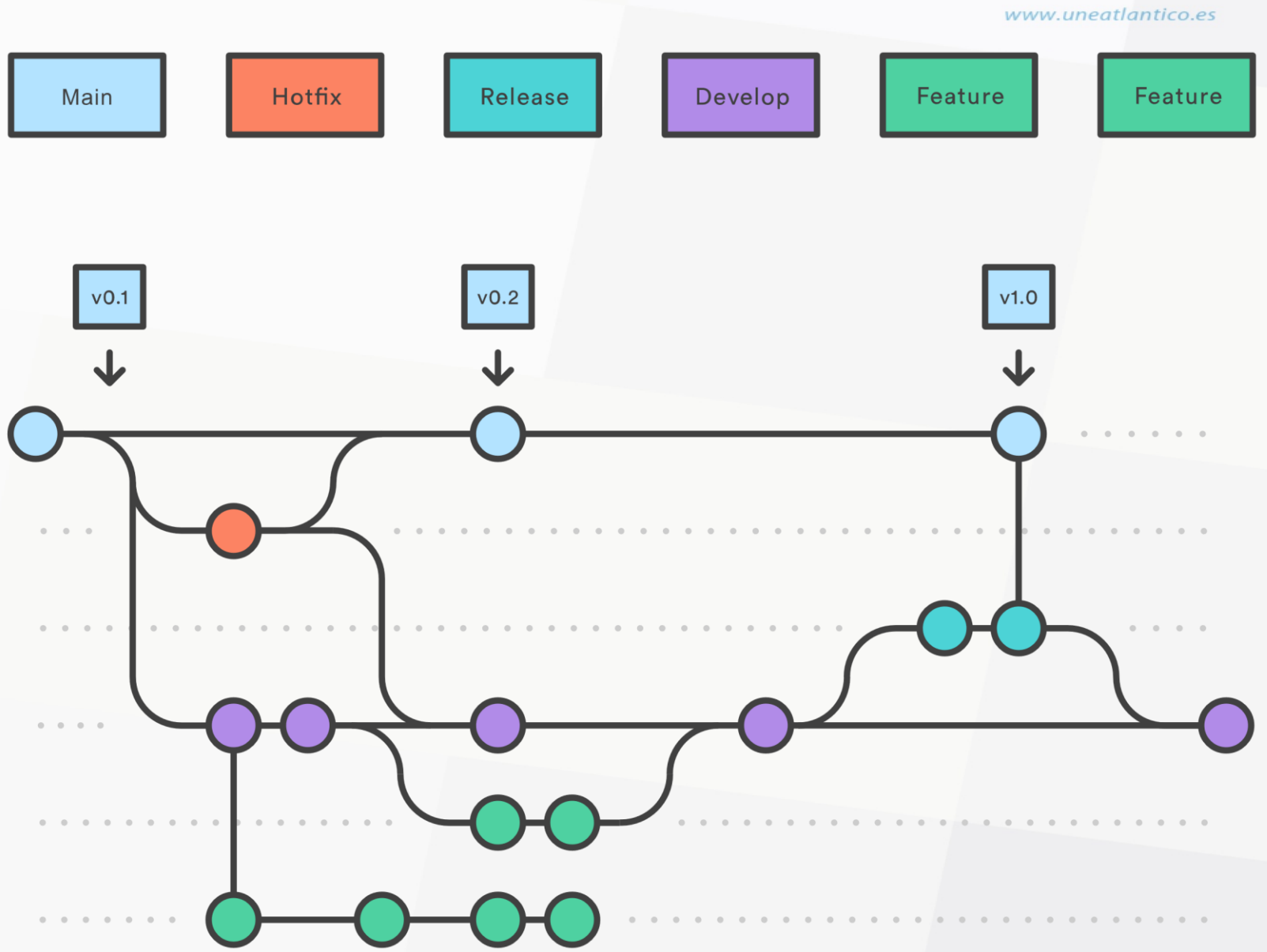
# GIT

## Flujo de trabajo



# GIT

## Flujo de trabajo




Universidad
Europea
del Atlántico

www.uneatlantico.es