


# Trabajo en Clase: SCM

 Fecha de Entrega	@21 de mayo de 2025 16:00
--	---------------------------

**Escuela Politécnica Nacional**  
**Facultad de Ingeniería de Sistemas**

**Construcción y Evolución de Software**  
**(ISWD633)**

Valencia David y Vásconez Gabriel

**Prof.:** Vicente Eguez

**Paralelo:** GR1SW

## Índice:

[Recursos](#)

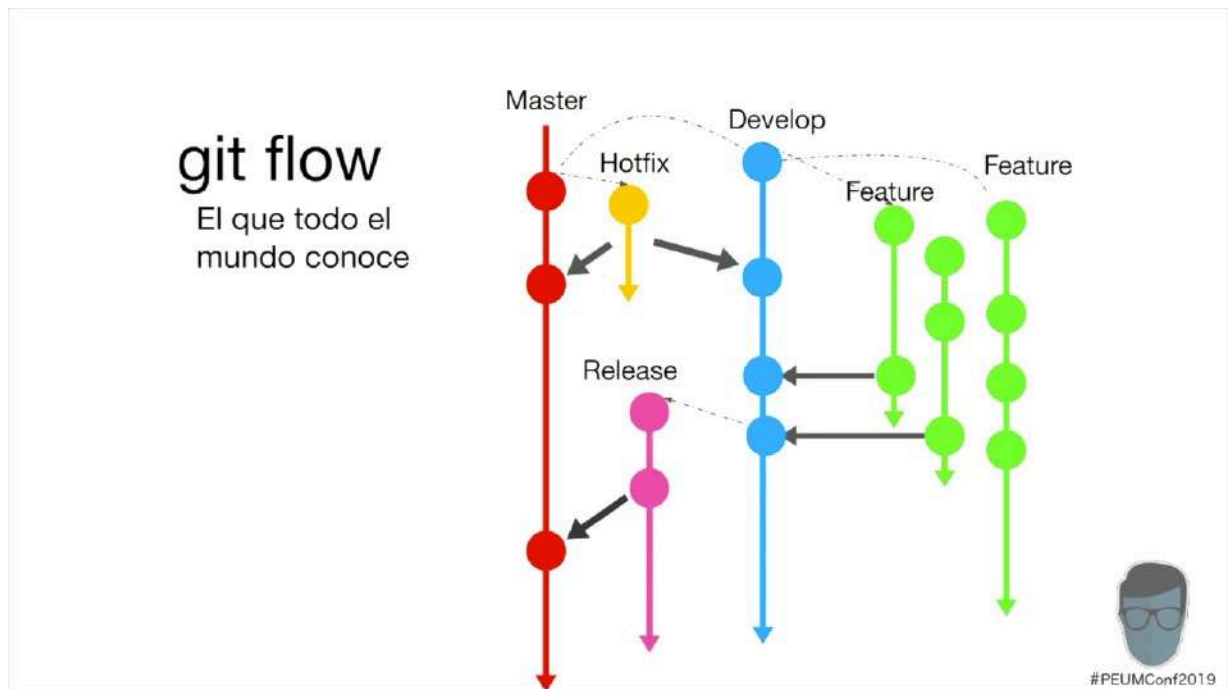
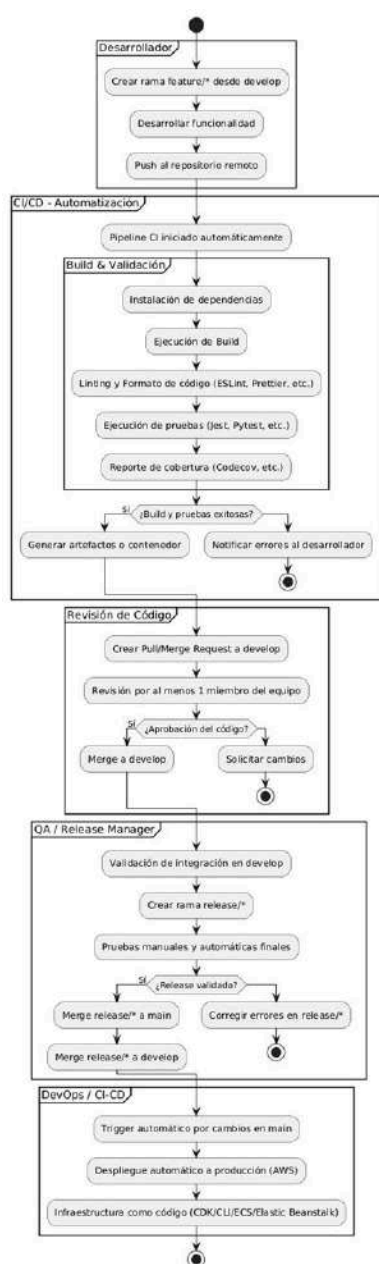
### Proceso y herramientas

- [1. Gestión con Git](#)
- [2. Herramientas para CI/CD y repositorio](#)
- [3. Análisis de código](#)
- [4. Pautas de revisión de código](#)
- [5. Flujo de despliegue del aplicativo](#)
- [Diagrama en PlantUML](#)

## Recursos

(Aquí se encuentran diversos materiales: archivos, enlaces, repositorios, etc.)

- Diagrama y flujo de trabajo en Git:**



Redacten un documento que describa el proceso completo de integración y entrega continua (CI/CD) que utilizará su equipo para desarrollar y desplegar el aplicativo. Deben incluir:

#### 1. Uso de Git:

- ¿Cómo se gestionarán las ramas ( `feature/*` , `develop` , `release/*` , `main` )?
- ¿Cuál es la convención de nombres y el flujo de trabajo con Git?

#### 2. Herramientas de CI/CD y repositorio:

- ¿Qué herramientas utilizarán para configurar los pipelines (por ejemplo, GitHub Actions, GitLab CI, Jenkins, etc.)?
- ¿Qué plataforma se usará para alojar el repositorio (GitHub, GitLab, Bitbucket, etc.)?

#### 3. Análisis de código automatizado:

- ¿Qué herramientas se usarán para análisis estático, formateo y pruebas automatizadas? (por ejemplo, ESLint, Prettier, Jest, Pytest, Codecov, etc.)
- ¿En qué parte del pipeline se ejecutan y cómo se manejan los errores?

#### 4. Pautas para revisión de código:

- ¿Cómo se solicita una revisión?
- ¿Cuántos revisores son necesarios?
- ¿Qué se evalúa antes de aprobar una Pull/Merge Request?

#### 5. Flujo de despliegue (CD):

- ¿Qué eventos (por ejemplo, merge a `main` ) desencadenan el despliegue?
- ¿Cómo se gestiona la validación de versiones ( `release/*` )?
- ¿Cómo y dónde se realiza el despliegue (por ejemplo, AWS, Docker, Elastic Beanstalk, etc.)?

## Proceso y herramientas

# 1. Gestión con Git

- **Repositorio:** GitHub
- **Convención de ramas:**
  - `develop` : rama base para desarrollo colaborativo.
  - `feature/*` : ramas para nuevas funcionalidades, basadas en `develop` .
  - `release/*` : ramas de estabilización antes de producción.
  - `main` : rama estable para producción.
- **Flujo:**
  1. Crear `feature/*` desde `develop` .
  2. Hacer cambios y pruebas.
  3. Pull Request hacia `develop` (con revisión).
  4. QA valida `develop` y crea `release/*` .
  5. Si la release es válida, se hace merge a `main` y despliegue.

# 2. Herramientas para CI/CD y repositorio

- **Repositorio:** GitHub
- **Pipeline:** GitHub Actions
- **Ambientes:**
  - `develop` : entorno de pruebas internas.
  - `main` : entorno de producción (AWS).
- **Automatización:** GitHub Actions ejecuta pipelines al hacer push o merge.

# 3. Análisis de código

- **Formato y estilo:**
  - `ESLint` para JavaScript/TypeScript.
  - `Prettier` para formato de código.
- **Pruebas:**
  - `Jest` o `Pytest` según el lenguaje.
- **Cobertura:** `Codecov`
- **Orden en el pipeline:**
  1. Instalar dependencias.
  2. Ejecutar build.
  3. Ejecutar linters y formateadores.
  4. Ejecutar pruebas.
  5. Reportar cobertura.
  6. Si hay errores, se notifica al desarrollador.

# 4. Pautas de revisión de código

- **Pull/Merge Request:** se crea hacia `develop` .
- **Revisión obligatoria:** mínimo 1 miembro del equipo.
- **Criterios:**
  - Estilo y formato correctos.
  - Tests pasados.
  - Legibilidad y claridad del código.

- Validación funcional básica.
- **Acciones:**
  - Si es aprobado, se hace merge.
  - Si requiere ajustes, se solicita cambios.

## 5. Flujo de despliegue del aplicativo

- **Validación en rama `release/*`:**
  - QA realiza pruebas manuales y automáticas.
- **Merge a `main`:**
  - Se dispara automáticamente el pipeline de despliegue.
  - Se crea imagen o artefacto, se sube a AWS (ECS o Elastic Beanstalk).
- **Herramientas utilizadas:**
  - GitHub Actions para CI/CD.
  - AWS como plataforma de despliegue.
  - Infraestructura como código: CDK o CloudFormation.

## Diagrama en PlantUML

