



PLAN CONFIGURACIÓN
DOSSIER DE INGENIERÍA

Código: DOC-CFG-001
Estado: VIGENTE
Clasificación: INTERNO

Ambato – Ecuador
28/01/2026

CONTROL DE DOCUMENTO

PROPIEDAD	DETALLE
Código	DOC-CFG-001
Nombre	Plan de Gestión de Configuración
Proyecto	_____
Versión	1.0
Estado	VIGENTE
Clasificación	INTERNO
Responsable	Tech Lead / DevOps

HISTORIAL DE VERSIONES

Versión	Fecha	Autor	Descripción del Cambio	Revisado por
1.0	_____	_____	Creación inicial del plan	_____

APROBACIÓN

Rol	Nombre	Firma
Tech Lead	_____	_____
DevOps	_____	_____
QA Lead	_____	_____

Contenido

PROPÓSITO	4
2. ELEMENTOS DE CONFIGURACIÓN (CI – Configuration Items).....	4
3. ESTRATEGIA DE VERSIONADO	4
4. ESTRATEGIA DE RAMAS (GITFLOW).....	4
4.1 Reglas Obligatorias.....	5
5. NOMENCLATURA DE COMMITS.....	5
6. CONTROL DE BASELINES.....	5
7. GESTIÓN DE CAMBIOS	6
8. GESTIÓN DE ARTEFACTOS	6
9. ENTORNOS	6
10. CONTROL DE ACCESOS	6
11. AUDITORÍA.....	7
12. RELACIÓN CON OTROS DOCUMENTOS.....	7

1. PROPÓSITO

Definir cómo se controlan, versionan, almacenan y protegen todos los elementos de configuración del proyecto para garantizar:

- Integridad del producto
- Reproducibilidad de builds
- Trazabilidad completa
- Prevención de cambios no autorizados

2. ELEMENTOS DE CONFIGURACIÓN (CI – Configuration Items)

Se consideran CI obligatorios:

- Código fuente
- Documentación oficial
- Scripts de infraestructura
- Archivos de configuración
- Casos de prueba
- Pipelines CI/CD

Todo CI debe vivir en repositorio controlado.

3. ESTRATEGIA DE VERSIONADO

Se adopta Versionado Semántico (SemVer):

MAJOR.MINOR.PATCH

Ejemplo: v2.3.1

Incremento	Se aplica cuando
MAJOR	Cambios incompatibles
MINOR	Nueva funcionalidad compatible
PATCH	Correcciones

4. ESTRATEGIA DE RAMAS (GITFLOW)

Ramas oficiales:

- main → Producción
- develop → Integración
- feature/* → Nuevas funcionalidades
- hotfix/* → Correcciones urgentes

4.1 REGLAS OBLIGATORIAS

- Prohibido el commit directo a las ramas main y develop.
- Todo cambio debe originarse en una rama feature/* o hotfix/*.
- Los Pull Requests (PR) requieren la aprobación del Tech Lead o un par designado antes de la integración.
- Ningún PR será aprobado si las pruebas automáticas o el análisis estático de código fallan.

5. NOMENCLATURA DE COMMITS

Formato obligatorio:

TIPO: Descripción corta

Ejemplo:

FEAT: Implementación login usuario

FIX: Corrección validación token

Tipos permitidos:

- FEAT
- FIX
- DOC
- TEST
- REFACTOR
- CHORE

6. CONTROL DE BASELINES

Se crea una baseline cuando:

- Se aprueba SRS
- Se libera versión
- Se firma aceptación

Las baselines se etiquetan en Git:

baseline-srs-v1

baseline-release-v1.0

7. GESTIÓN DE CAMBIOS

Todo cambio de alcance debe:

1. Crear DOC-CHG-001
2. Evaluar impacto
3. Aprobar cambio
4. Implementar

Sin DOC-CHG-001 aprobado → Cambio prohibido.

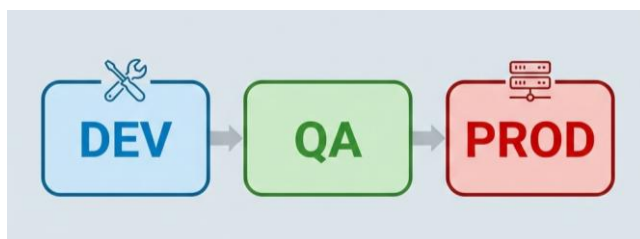
8. GESTIÓN DE ARTEFACTOS

Tipo	Herramienta	Retención
Código Fuente	GitHub	Indefinida
Documentación	SharePoint / GitHub	Historial completo
Builds / Imágenes	Docker Registry	Últimas 5 versiones
Logs	Stack Logs	30 días
Backups	Storage Seguro	90 días

9. ENTORNOS

Entorno	Propósito
DEV	Desarrollo local
QA	Validación
PROD	Producción

Flujo:



10. CONTROL DE ACCESOS

- Solo DevOps puede escribir en main
- Developers escriben en feature
- QA solo lectura en repositorio

11. AUDITORÍA

Se revisa trimestralmente:

- Historial de commits
- PRs aprobados
- Tags de versión
- Baselines

12. RELACIÓN CON OTROS DOCUMENTOS

Este plan opera en conjunto con los siguientes artefactos del dossier:

- **DOC-TRZ-001 (Matriz de Trazabilidad):** Para asegurar que cada CI esté vinculado a un requisito.
- **DOC-CHG-001 (Solicitud de Cambio):** Documento obligatorio para cualquier modificación de baselines.
- **DOC-REL-001 (Release Notes):** Registro oficial de los cambios incluidos en cada versión.
- **DOC-TEST-002 (Casos de Prueba):** Referencia para los CI relacionados con la validación de software.