



# Gestion de Configuracion del Software

## ▼ Concepto General

La administración de configuración de software busca algo fundamental: **proteger la integridad del software**. Esto significa que todo el conjunto de algoritmos que conforman el sistema se mantenga consistente y confiable a lo largo del tiempo.

Se la suele llamar **disciplina paraguas** porque no se limita a una sola etapa, sino que atraviesa todo el ciclo de vida del proyecto. Es decir, está presente desde el inicio hasta el final, actuando como soporte constante.

## ▼ Calidad en el Proceso y en el Producto

Existe una premisa clave: **si el proceso es de calidad, el producto resultante también lo será**. En teoría, lo lógico sería que las mismas entradas generen siempre las mismas salidas. Sin embargo, en el software esto no es tan simple: lo que sí se puede garantizar es el mismo nivel de calidad, aunque el producto en sí pueda variar.

Esto se debe a que el software no es un producto estático, sino que depende de factores externos y de cómo se desarrollan los procesos.

## ▼ Cambios en los Requerimientos

Uno de los grandes desafíos es que los **requerimientos cambian con frecuencia**. Cuando esto sucede, las fuentes de información anteriores pueden quedar inválidas y se producen problemas de versiones.

Estos cambios pueden surgir por distintos motivos:

- modificaciones en el **ambiente** donde corre el software,
- cambios en las **leyes** o regulaciones,

- **parches del sistema operativo,**
- o transformaciones en el propio **negocio.**

En cualquiera de estos escenarios, la integridad del software queda en riesgo, y aquí es donde SCM se vuelve indispensable para mantener todo bajo control.

## ▼ Configuración vs. Aseguramiento de Calidad 🛡️

La administración de configuración de software suele verse como una **revisión de calidad tardía**, porque muchas veces se valida cuando el producto ya está terminado.

Pero la esencia de esta disciplina es distinta: busca asegurar la calidad **desde el principio hasta el final**, justamente porque acompaña a lo largo de todo el ciclo de vida. Por eso se la considera un pilar fundamental del aseguramiento de calidad.

## ▼ Procesos y Personas 👤

En el desarrollo de software siempre hablamos de las **4 P**: procesos, personas, proyecto y producto.

- En un **proceso empírico** no existen planillas ni un esquema definido.
- En un **proceso definido**, en cambio, sí se documenta, se registran datos y se siguen protocolos claros.

El producto final siempre será el **software**, que nace de la interacción de personas que utilizan herramientas para crearlo, y luego será ejecutado por una computadora en su estado más desarrollado.

Algo interesante: en este ámbito se dice que **"primero muere el proyecto, pero el producto sigue vigente"**. Es decir, aunque el equipo se desarme, el software puede seguir funcionando en producción.

## ▼ Control Sistémico de Cambios 🛠️

Para que los cambios no generen caos, es necesario que exista un **proceso definido de control de cambios**. Se habla de control sistémico porque no es algo improvisado, sino una práctica organizada.

Dentro de este control entran varios ítems de configuración que deben estar bajo supervisión:


- El **código fuente**, que es el más crítico y debe estar bien nombrado, versionado y cumplir estándares.
- Las **pruebas unitarias**, que aseguran que las partes funcionen correctamente.
- La **última versión de los requerimientos validados**.
- Incluso algunos **mails de validación de requerimientos**, que son ítems importantes a conservar.

Lo interesante es que los ítems de configuración **evolucionan como si fueran fotos**: cada versión es un registro en el tiempo. Y además, esas versiones pueden tener variantes, por ejemplo, si el software se adapta a distintos sistemas operativos.

## ▼ Configuración y Línea Base

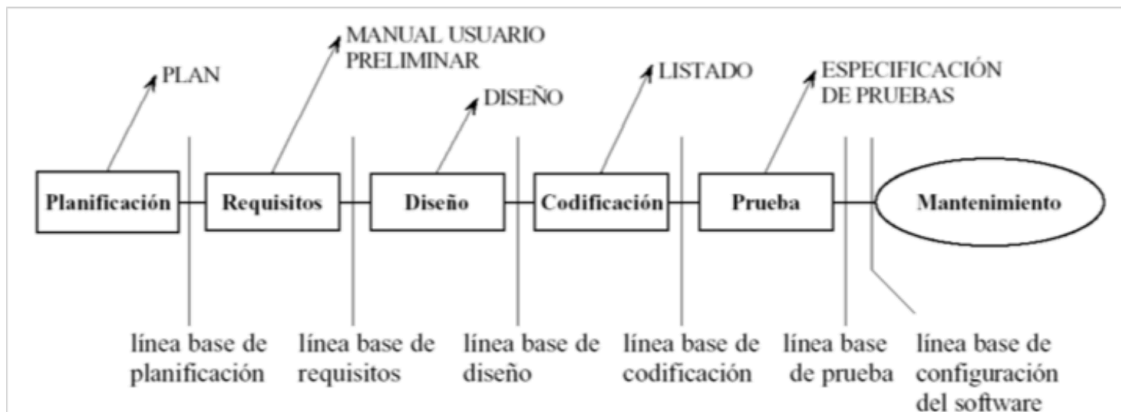
La **configuración de software** es como una foto de un momento que marca una versión específica.

En cambio, la **línea base** es una versión considerada íntegra y estable, un verdadero "punto seguro".

Para entenderlo mejor, pensemos en el juego de Mario Bros : cuando llegás a una banderita, marcás un checkpoint. Si avanzás y algo sale mal, volvés a ese punto y no desde el inicio. En software pasa lo mismo: la línea base es ese lugar seguro al que podés regresar si algo falla.

Eso sí, modificar una línea base no es algo trivial. Se necesita un **procedimiento formal** aprobado o rechazado por un comité conformado por representantes de todas las áreas involucradas.

Además, es recomendable que al final de cada etapa importante del proyecto se marque una nueva línea base.



## ▼ Repositorios y Reglas

Los **repositorios** son los lugares donde se almacenan los ítems de configuración junto con todo su historial. Esto no solo permite guardar, sino también analizar y evaluar los cambios que se proponen.

Para que el trabajo en repositorios sea efectivo, se deben definir:

- reglas claras de **nomencatura**,
- la **estructura** y la ubicación,
- y un orden preacordado que asegure prolijidad y organización.

Dentro de los repositorios existen dos acciones fundamentales:

- **Check-out** : cuando me traigo la última versión o alguna anterior.
- **Check-in** : cuando subo mi nueva versión y la devuelvo al repositorio.

Mientras más estructurado y controlado esté este proceso, mejor será la calidad del producto final, porque habla directamente de la calidad del proceso que se siguió.

## ▼ Auditorías

Un proceso de auditoría finaliza con un **informe de hallazgos**, que deja registro de lo que se analizó.

Aquí se distinguen dos conceptos clave:

- **Validar** : comprobar que se construyó el software correcto.
- **Verificar** : comprobar que se construyó el software correctamente.

Las auditorías suelen ser realizadas por alguien externo a la organización o al proyecto, lo que asegura imparcialidad. Estos informes resultan esenciales para rastrear y justificar los cambios que se han realizado.

## ▼ SCM en Metodologías Ágiles

Cuando llevamos estas prácticas a metodologías ágiles, es necesario relacionarlas con los principios del **Manifiesto Ágil**.

### 1. Individuos e interacciones sobre procesos y herramientas

- SCM debe ajustarse a la forma de trabajo del equipo y no imponer rigidez innecesaria.

### 2. Software funcionando sobre documentación exhaustiva

- En vez de depender de documentos, SCM puede automatizar procesos con scripts que garanticen la calidad.

### 3. Colaboración con el cliente sobre negociación contractual

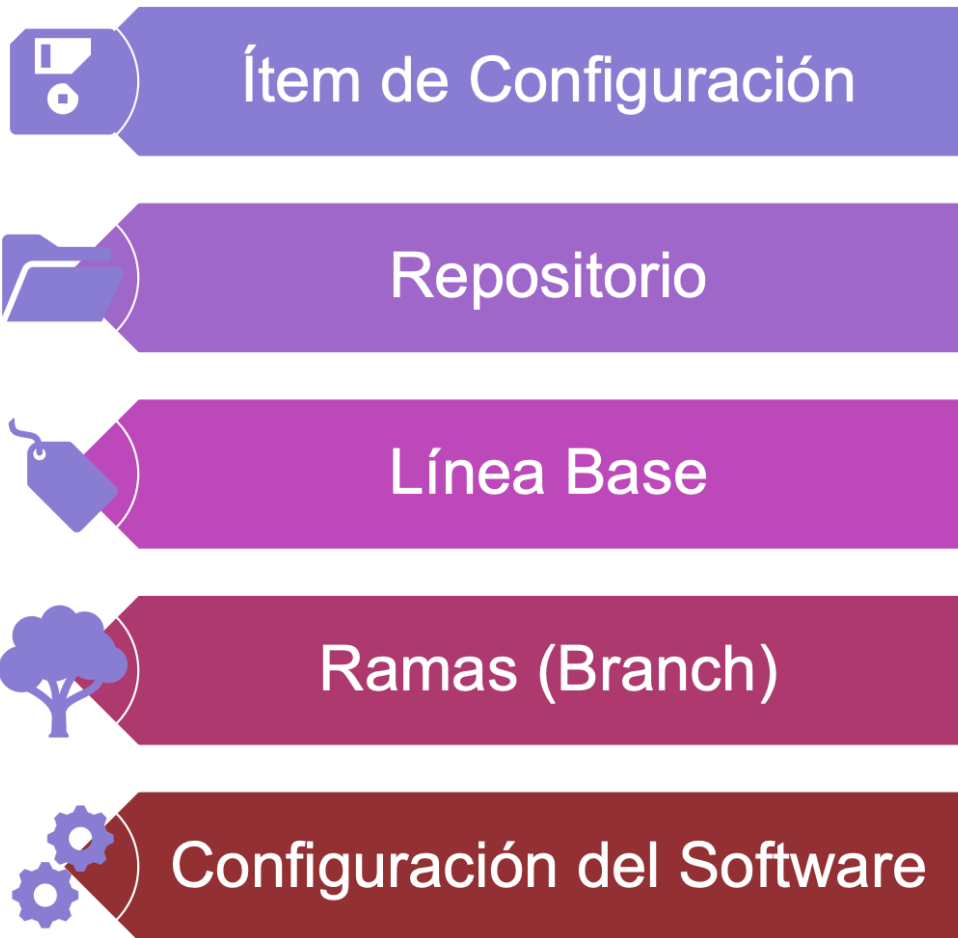
- SCM ayuda a la comunicación, da visibilidad al cliente y le permite involucrarse activamente.

### 4. Responder al cambio sobre seguir un plan

- El corazón de SCM es facilitar el cambio, nunca bloquearlo. La flexibilidad es esencial en entornos ágiles.

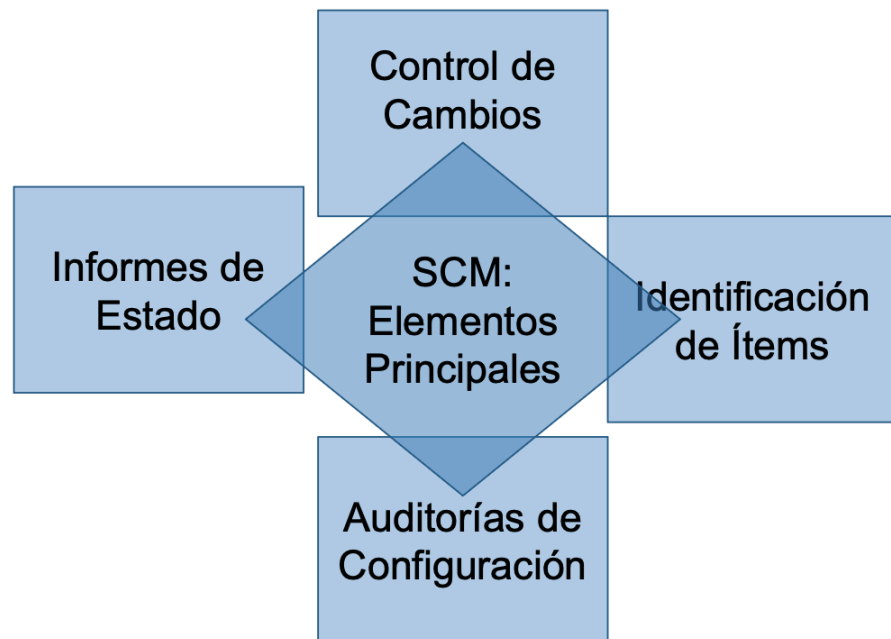


## ▼ Conceptos Clave



## ▼ Actividades de SCM

# Actividades Fundamentales de la Gestión de Configuración de Software



## ▼ Idea Final ✨

En conclusión, las tareas de integridad son **intrínsecas** a esta disciplina. Sin SCM, sería muy difícil garantizar que el software se mantenga confiable, estable y seguro a lo largo del tiempo.