¿Qué es SCM?

Disciplina DE SOPORTE que **identifica, organiza, controla y audita** todos los elementos que componen un producto de software a lo largo de su ciclo de vida. Su PROPOSITO es que cada cambio sea **trazable, reversible, verificable y reproducible**.

¿Para qué sirve?

- **Controlar cambios** de forma ordenada.
- **Asegurar integridad** del producto y sus artefactos en cada versión/release.
- **Trazabilidad** completa.
- **Reducir riesgos** en entregas, hotfixes y mantenimiento.

¿Por qué garantiza integridad?

- **Línea base**: arrancar desde un "momento" aprobado previamente.
- **Control de versiones** con historial, etiquetas (tags) y firmas.
- **Políticas de branching + revisiones (PR/code review)** + CI que valida.
- **Auditorías de configuración**: comprobar que lo entregado coincide con lo aprobado.
- **Gestión de dependencias y artefactos** (hash, checksums, locks) para builds repetibles.

¿Qué problemas resuelve?

- Cambios **caóticos**.
- **Regresiones** difíciles de rastrear.
- **Divergencias** entre equipos/entornos.
- **Dependencias** rotas o no declaradas.
- **Falta de trazabilidad**.
- **Releases irrepetibles**.

Conceptos generales (rápidos)

- **Ítems de Configuración (CI/IC):** unidades gestionadas: código, scripts, infra-as-code, docs, binarios, plantillas, parámetros, etc. Se **identifican**, **versionan** y **auditan**.
- **Repositorio:** es como una base de datos de ICs. Puede ser de **código fuente** (Git) y/o de **artefactos** (paquetes, imágenes Docker).
- **Versión:** estado concreto de un IC o conjunto en un momento de tiempo.
- **Rama (branch):** línea de trabajo paralela. Al terminar de trabajar o se descarta o se hace merge con la main.
- **Control de versiones:** mecanismos/herramientas para registrar y coordinar cambios. Permiten **historial**, **diff**, **merge**, **revert**.

- **Variantes:** ediciones del mismo producto para **segmentos distintos**, activadas por **feature flags**, perfiles de build o ramas de larga vida.
- **Configuración de SW:** parámetros que **cambian comportamiento sin recompilar**. Deben ser **versionados** (o gestionados en vaults) y segregados por ambiente.

¿Qué es una línea base?

Un **conjunto aprobado** de ítems/versiones que sirve de **punto de referencia**. Es un momento estable del proyecto. A partir de ella:

- Se **trabaja** (se derivan ramas, se planifican cambios).
- Se **audita** (que el incremento respete lo aprobado).
- Se **reproduce** (compilar/instalar exactamente esa versión).

Tipos comunes:

- **Funcional** (requisitos/alcance aprobado).
- **De diseño** (arquitectura/interfaces congeladas).
- **De producto/release** (lo que se entrega a QA/Producción).

Si se desea cambiar algo de la línea base, existe un protocolo formal de control de cambios, dirigido por el Comité de Control de Cambios, formado por los principales actores de cada área.

Actividades típicas de la Gestión de la Configuración

1. **Planificación de SCM**
   Definir qué se controla, políticas de versionado, naming, roles, herramientas.
2. **Identificación de configuración**
   Catalogar ICs, su granularidad y metadatos (dueño, versión, ubicación).
3. **Control de cambios**
   Proceso formal para proponer, evaluar, aprobar y aplicar cambios. Suele haber un **CCB** (Change Control Board) para ítems críticos.
4. **Contabilidad de estado**
   Registro y reportes: qué versión está dónde, qué cambió, qué se aprobó.
5. **Auditorías de configuración**
   - **FCA (Functional Configuration Audit):** lo implementado cumple lo requerido.
   - **PCA (Physical Configuration Audit):** lo construido coincide con lo documentado (números de versión, contenidos del paquete, checksums).
6. **Gestión de builds y releases**
   Pipelines, artefactos firmados, etiquetado, notas de versión, rollback.

7. **Gestión de librería/repositorio**
   Permisos, backups, retención, limpieza, mirrors.
8. **Gestión de entornos**
   Infra como código, paridad Dev/QA/Prod, secretos, parámetros por ambiente.

SCM en entornos ágiles

En las metodologías ágiles la gestión de configuración **cambia el enfoque**, ya que la misma es de utilidad para los miembros del equipo de desarrollo y no viceversa. En orden con la caracterización de los equipos ágiles, decimos que la gestión de configuración posibilita el **seguimiento y la coordinación del desarrollo** en lugar de controlar a los desarrolladores. Los equipos ágiles **son auto organizados**, por lo que las Auditorías de configuración no son una actividad propia de la gestión de configuración en los ambientes ágiles. Todos los procesos definidos establecidos en la gestión de configuración del enfoque tradicional son relativizados en agile. Por ejemplo, no existe un comité para el proceso formal de control de cambios.