|  |
| --- |
| **PROYECTO:**  SPACIA  Sistema de gestión de ambientes  **Plan de Gestión de la Configuración**  **V 1.1** |

**Contenido**

[1. Introducción 3](#_Toc513155086)

[1.1. Propósito del documento 3](#_Toc513155087)

[1.2. Alcance 4](#_Toc513155088)

[1.3. Definiciones 4](#_Toc513155089)

[1.3.1. Abreviaturas 5](#_Toc513155090)

[1.4. Referencias 5](#_Toc513155091)

[2. Gestión de la Configuración de Software 5](#_Toc513155092)

[2.1. Organización 5](#_Toc513155093)

[2.2. Roles y Responsabilidades 6](#_Toc513155094)

[2.3. Políticas, directrices y procedimientos 8](#_Toc513155095)

[2.4. Herramientas, entorno e infraestructura 13](#_Toc513155096)

[2.4.1. Herramientas de control de versiones 14](#_Toc513155097)

[2.4.2. Herramientas de entorno 15](#_Toc513155098)

[2.5. Calendario 16](#_Toc513155099)

[3. Actividades de la Gestión de la Configuración 17](#_Toc513155100)

[3.1. Lista de Items de la Configuración 17](#_Toc513155101)

# Introducción

En todo proyecto de desarrollo de software surgen cambios (tanto previstos como imprevistos) que llevan a la creación de nuevas versiones del producto a lo largo de la vida del proyecto. Es conveniente que estas versiones estén sujetas a mantenimiento y gestión para minimizar los riesgos asociados a tener varias versiones de los distintos ítems, además de la cantidad de los mismos que se maneja. La Gestión de la Configuración es entonces la disciplina que se encarga de identificar qué elementos deben estar bajo control de configuración y restringir, auditar y definir y gestionar la configuración de los mismos. Mediante la definición de distintos procedimientos y políticas, la Gestión de la Configuración busca minimizar las confusiones generadas por el simple hecho que se generan distintas versiones sobre las cuales distintas personas van a estar trabajando, lo que implica que en algún momento u otro el trabajo de las mismas se va a solapar, causando actualizaciones simultáneas o problemas en la notificación de cambios, sumados a las múltiples versiones de los elementos nombrada anteriormente.

Ante la necesidad de establecer políticas y procedimientos para una correcta gestión de la configuración del software; y de la necesidad de llevar a cabo las tareas necesarias para el mantenimiento del software desarrollado, se desarrolló este documento; el cual describe las actividades relacionadas con la gestión de la configuración y el mantenimiento del Sistema denominado SPACIA. Este último desarrollado con la finalidad de gestionar el uso de los ambientes de una empresa u organización.

## Propósito del documento

Este documento describe las actividades de gestión de configuración de software que deben ser llevadas a cabo durante el proceso de desarrollo del proyecto. Aquí se definen tanto los productos que se pondrán bajo control de configuración como los procedimientos que deben ser seguidos por los integrantes del equipo de trabajo.

Este documento está diseñado para servir de soporte al equipo encargado de la gestión de la configuración y/o mantenimiento, en el uso de métodos (y los requisitos para el cumplimiento de estos) y en la ejecución de actividades para una adecuada configuración y mantenimiento de software, enfocados en el sistema SPACIA.

## Alcance

Este documento tendrá el siguiente alcance:

* El tiempo de duración del proyecto está limitado a 9 semanas, por lo tanto se busca una rápida respuesta a los cambios, tratando que este procedimiento sea lo menos burocrático posible.
* El Modelo de Proceso se basa en un desarrollo incremental, dado por las distintas iteraciones. Resulta importante tener control sobre cada una de las iteraciones y fases, de los productos generados en estas y de los cambios surgidos, evaluados y aprobados.
* La identificación y definición de la línea base de los elementos del sistema SPACIA.
* El establecimiento de los métodos, procedimientos y herramientas para el control y registro del estado de las versiones de SPACIA.
* Procedimientos para la solicitud de cambios.
* Procedimientos a considerar para la modificación del sistema SPACIA.

## Definiciones

* **Elemento de configuración:** Unidad estructural fundamental de un sistema de gestión de configuración.
* **Línea base:** Punto de referencia en el ciclo de vida del desarrollo de software marcado por la finalización y aprobación formal de un conjunto de productos de trabajo predefinidos
* **Identificación de configuración:** Esta actividad se encarga de identificar los elementos, esquemas y herramientas.
* **Control de configuración:** Gestiona los cambios que se realizarán al software durante su ciclo de vida. Determina que cambio se debe hacer, si es que es correcto y la manera de su implementación.
* **Contabilidad del estado de configuración:** Proceso por el cual en el estado actual del desarrollo puede ser juzgado y la historia del ciclo de vida del desarrollo se puede rastrear.
* **Auditoría de configuración:** Identificar los elementos del sistema que satisfacen los requerimientos del cliente.

## Abreviaturas

* CCB: Tablero de Control de Configuración
* CI: Ítem de Configuración
* CM: Gestión de Configuración
* OP: Programa operacional
* SCA: Autorización de Cambios de Software
* SCI: Ítem de Configuración del Software
* SCM: Gestión de la Configuración de Software
* SCMP: Plan de Gestión de la Configuración de Software
* SMP: Gestión de Mantenimiento de Software
* SCR: Solicitud de Cambio del Sistema/Software
* CMO: Oficial de Gestión de Configuración
* SQA: Aseguramiento de calidad de software

## Referencias

[1] ANSI/IEEE Std 828-1983, IEEE Standard for Software Configuration Management Plans

[2] ANSI/IEEE Std 1042-1987, IEEE Guide to Software Configuration Management

# Gestión de la Configuración de Software

## Organización

Para obtener un trabajo alineado y que pueda cubrir las expectativas de nuestros clientes, tomando en cuenta la prevención de confusiones acerca de quién va a realizar las tareas de SCM, en la siguiente tabla (tabla 1) se trata de representar la relación que existe con las actividades de la SCM con los roles del proyecto.

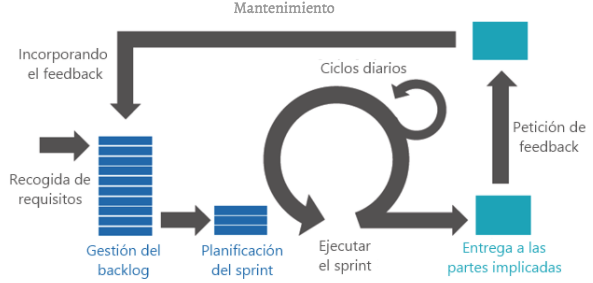


Figura 1: Relación del modelo de proceso usando, respecto a la gestión de configuración y mantenimiento

*Tabla* 1*: Relación entre las actividades generales de la SCM y los roles responsables.*

|  |  |
| --- | --- |
| Actividades | Rol Responsable |
| Planificar la configuración de SCM | CMO |
| Definición de la Línea del proyecto | CMO |
| Seguimiento de la Línea del proyecto | CMO |
| Definición del plan del contrato | CMO |
| Control de Cambios | CCB, CMO |
| Descripción de la Versión | CMO |
| Realizar informes final de SCM | CMO |

## Roles y Responsabilidades

La unidad de SCM de la NN-Consulting es liderada por el responsable de SCM (Configuration Management Officer, CMO), responsable por las tareas de gestión de la unidad. En general, la unidad actúa como un equipo en el cual sus miembros cooperan para llevar a cabo las tareas de SCM.

* Configuration Management Officer

El CMO es responsable de:

* Gestionar la unidad de SCM.
* Definir una estrategia para la implantación de las prácticas de SCM.
* Proveer el personal de SCM requerido para las actividades de desarrollo.
* Planificar las actividades de SCM.
* Informar a los niveles superiores sobre el estado del proceso y las actividades de SCM en los proyectos.
* Garantizar la capacitación del personal de SCM y los recursos necesarios para el desarrollo de sus actividades.
* Promover el mejoramiento continuo del proceso de SCM.
* Gestionar el desarrollo de herramientas para facilitar el proceso de SCM.
* Control Configuration Board (CCB)

Un grupo de stakeholders del proyecto responsables de:

* Revisar el hardware o solicitudes de cambio de clientes, en términos del cronograma del proyecto, el costo y el impacto en los clientes.
* Recomienda la aprobación o desaprobación del cambio.
* Priorización de la incorporación de cambios aprobados.
* Reenvía la decisión de aprobación o desaprobación del cambio al CMO
* Informa por escrito al cliente de la decisión de cambio.
* Ingenieros

Los ingenieros miembros de la unidad de SCM, que se constituyen como sus representantes en los diferentes proyectos de desarrollo, deben:

* Desarrollar y coordinar las actividades de SCM planificadas para el proyecto.
* Establecer y participar de la identificación, control, auditoría e informe sobre el estado de la configuración.
* Participar en el establecimiento de la biblioteca del software.
* Monitorear el correcto funcionamiento de la biblioteca del software.
* Interactuar con los desarrolladores y la unidad de SQA.
* Bibliotecarios

Los bibliotecarios son responsables de:

* Diseñar y establecer la biblioteca del software para cada proyecto de desarrollo durante la etapa de planificación.
* Establecer y mantener el software y la documentación de cada proyecto de acuerdo con un proceso documentado
* Proveer a los desarrolladores las copias de las líneas bases requeridas para sus diferentes tareas.
* Mantener y distribuir un índice con el contenido de cada biblioteca.
* Entregar la copia original para la implementación de los cambios aprobados por el CCB.
* Informar a los desarrolladores sobre los cambios a los ítems
* Registrar y mantener copias de las antiguas versiones.
* Apoyar la elaboración de informe sobre el estado de la configuración.
* Personal de apoyo

Las responsabilidades del personal de apoyo incluyen:

* Desarrollar tareas administrativas para los representantes de SCM.
* Completar los informes de SCM.
* Interactuar con las diferentes herramientas utilizadas para SCM durante un proyecto.
* Interactuar con el personal de SCM según sea necesario.

## Políticas, directrices y procedimientos

Respecto a los documentos en el repositorio de GitHub, todos tienen derechos de administrador, es decir, permiso de lectura, escritura y creación de nuevos documentos. Se tiene como política la total confianza en todos los integrantes del equipo.

**Políticas**

* Seguir los estándares de nomenclatura plasmados en el presente documento.
* Cada rol debe cumplir las actividades designadas en el documento.
* Cada miembro debe trabajar únicamente y exclusivamente sobre su branch.
* La filtración de información a agentes externos se encuentra estrictamente prohibida.
* Correcta comunicación entre los miembros encargados de la gestión de la configuración.
* Toda dependencia en los proyectos deberá ser estrictamente documentada.

**Directrices**

* Se encuentra estrictamente prohibido el versionamiento en el nombre del ítem. Ej.: No se permite el nombre “SPC\_AC\_v1.1.docx”
* Para los ítems de gestión de la configuración:
  + Los ítems deberán llevar el siguiente estándar de nombre:

<INICIALES DEL DOCUMENTO>.\*

Ej.: PGC.docx (Plan de gestión de la configuración)

* En caso dos ítems de gestión de la configuración llegasen a tener la misma nomenclatura, el ítem más reciente tendrá la nomenclatura:
  + <INICIALES DEL DOCUMENTO>\_<TAG>.\*

Donde <TAG> es un identificador para dar soporte a la identificación del documento más reciente.

* Para los ítems de gestión de proyectos:
  + Los ítems deberán llevar el siguiente estándar de nombre:

<INICIALES DEL PROYECTO>\_<INICIALES DEL DOCUMENTO>.\*

Ej.: SPC\_AC.docx

* Los nombres de los branches deberán ser hecho según la inicial del nombre del responsable y del primer apellido completo. Ej.: Eduardo Angulo -> branch “EAngulo”
* Para los ítems de desarrollo de proyectos Frontend:
  + Todo folder o ítem que cuente con más de 2 palabras, deberá ser nombrado en CamelCase. Ej.: Dashboard home-> DashboardHome
  + Los componentes:
    - Los ítems que expongan los componentes deberán ser nombrados de la siguiente manera:

<NOMBRE DEL COMPONENTE O FEATURE>component.js

* + Las vistas o templates:
    - Deberán ir en el folder de su respectivo componente o feature (característica de la aplicación). En este caso se utilizará el Views como carpeta de las vistas.
    - Dentro de la carpeta Views tendremos las carpetas Home y Shared. Shared nos permitirá tener las vistas compartidas como el \_Layout y la vista de carga de datos en otra vista así como también la vista de Errores.
    - Deberán ser nombradas de la siguiente manera:

<NOMBRE DEL COMPONENT O FEATURE>Index.cshtml

* + Los controladores
    - Deberán ir en el folder de su respectivo componente o feature (característica de la aplicación).
    - Deberán ser nombradas de la siguiente manera:

<NOMBRE DE COMPONENT O FEATURE>HomeController.cshtml

* + Los services(servicios)
    - Deberán ir en el folder src/app/core del proyecto.
    - Dependiendo de la utilidad del servicio, este deberá ir en la siguiente carpetas:
      * Utils.- servicios que utilizan componentes.
      * Helpers.- servicios para evitar la repetición de código o snippets.
      * Services.- servicios con lógica de procesos.
    - Los ítems que expongan los servicios, deberán ser nombrados de la siguiente manera:

<NOMBRE DEL SERVICIO>.service.js

* + Los resources(recursos de datos)
    - Deberán ir en el folder src/app/core/resources del proyecto.
    - Dependiendo de su fuente, deberán ir en las carpetas:
      * Local.- Extracción de datos almacenados en el navegador.
      * Test.- Para mock.
      * Remote.- Extracción de datos de fuentes remotas(APIs).
  + Los estilos de la aplicación
    - Deberán ir en src/assets.
    - Se deberá seguir el patrón de diseño 7-1. Ref: [https://sass-guidelin.es/es/#el-patron-7-1](https://sass-guidelin.es/es/" \l "el-patron-7-1)
  + Imágenes y documentos
    - Deberán ir en las carpetas src/images y src/docs respectivamente.
* Para los items de proyectos BackEnd:

Todo ítem con más de dos palabras deberá ser nombrado en Camel case. Ejm: controlador de usuario -> UserController

* + Será usado como estructura del proyecto el generado por el Framework Laravel:
  + El directorio raíz
    - El directorio app
    - El directorio bootstrap
    - El directorio config
    - El directorio database
    - El directorio public
    - El directorio resources
    - El directorio routes
    - El directorio storage
    - El directorio tests
    - El directorio vendor
  + El directorio app
    - El directorio Console
    - El directorio Events
    - El directorio Exceptions
    - El directorio Http
    - El directorio Jobs
    - El directorio Listeners
    - El directorio Mail
    - El directorio Notifications
    - El directorio Policies
    - El directorio Providers
    - El directorio Rules
* Para los ítems de proyectos móvil:
  + Todo folder el cual contenga más de 2 palabras como por ejemplo: android Test, será nombrado del siguiente modo: androidTest. En resumen del siguiente modo: <nombreDeLaCarpeta>
    - Solo en el caso de tratarse de las carpetas de los componentes gráficos tendrán esta estructura:

En resumen del siguiente modo: <drawable-hdpi>

* + Los componentes serán nombrados del siguiente modo:

Las clases hechas en JAVA las son expresadas del siguiente modo:

<NombreDeLaClase>.java

* + Los layouts y componentes gráficos o vistas se organizarán del siguiente modo:

Los archivos xml están con la siguiente nomenclatura: <activity\_main>.xml

Los archivos como fotos o imágenes tienen la siguiente nomenclatura: <ic\_add\_without\_circle>.png

* + Todo archivo que contenga configuración del mismo proyecto Android como el graddle o los services de Google API terndrán la siguiente nomenclatura:

<nombre-archivo>.<extension>

Ejm:

google-services.json

proguard-rules.pro

* + - * + Todas las imágenes se encuentran distribuidas en la carpeta <nombre- app>/<app>/<src>/<main>/<res>, y dependiendo su dimensión se pueden ubicar en drawable, drawable-nodpi, etc.

**Procedimientos**

* Respecto a la modificación de archivos pertenecientes a la gestión de la configuración:
  + - Indicar a demás miembros encargados de los documentos qué documento está siendo manipulado, con el fin de evitar modificaciones en el mismo por parte de otros miembros, lo cual repercutiría en un sobreescribimiento del ítem.
    - Modificar el ítem apropiadamente.
    - Subir los cambios.
    - Hacer los merge necesarios para que el ítem modificado se encuentre disponible para todos.
    - Informar que el cambio fue realizado.

## Herramientas, entorno e infraestructura

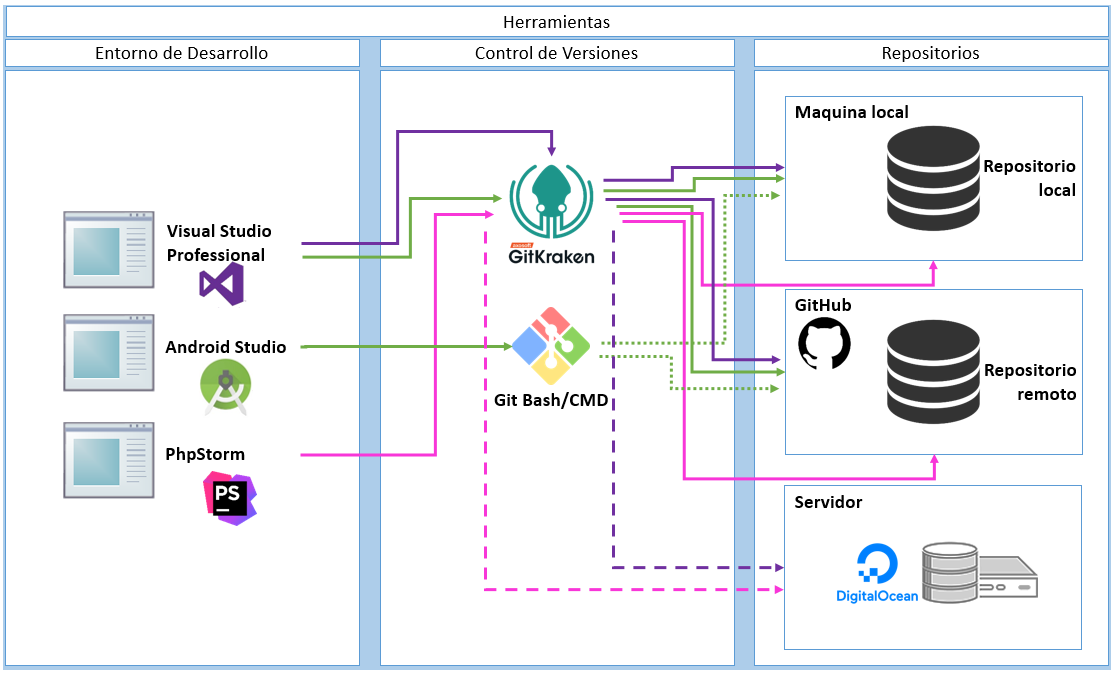


Figura 2: Mapeo del uso de las herramientas de control de versiones y entorno

## Herramientas de control de versiones

* + - Github

Es una plataforma de desarrollo colaborativo online, utiliza control de versiones git para el mantenimiento y versionado del código fuente, añadiendo una serie de servicios extras para la gestión del proyecto y el código fuente. Permite alojar nuestro código en repositorios públicos, la versión Premium permite tener repositorios privados, pero para el desarrollo de este proyecto se usará la versión gratuita. En la actualidad ofrecen varias herramientas útiles para el trabajo en equipo, entre ellos cabe destacar

* Una wiki: para el mantenimiento de distintas versiones de las páginas.
* Un sistema de seguimiento de problemas, que permite a un miembro de tu equipo hacer mejoras, sugerencias y optimizaciones.
* Herramienta de versión de código, donde puedes añadir anotaciones en cualquier punto del proyecto.
* Un visor de ramas o branch: donde se puede conocer el progreso que llevamos en nuestro proyecto, mostrando las actualizaciones o commits realizados a partir de nuestro master.
  + - Git

Es una herramienta de tipo consola que posee múltiples comandos que ayudan a tener un buen control de versiones, permite manipular y gestionar todo el proceso a realizarse en el proyecto. Posee la misma funcionalidad tanto en Windows, Linux o Mac, permite un desarrollo no lineal lo que permite una gran rapidez en la gestión de ramas. La herramienta será usada tanto en el desarrollo Front End, Back End, Android y documentación. Git Bash es la herramienta para consola más utilizada por los desarrolladores de NN-Consulting.

* + - Git Kraken

Proporciona una interfaz gráfica multiplataforma para git. Es bastante amigable y nos permite visualizar y gestionar de una forma más sencilla y hacer un seguimiento a nuestro historial en el repositorio y sus distintas ramas al igual que git esta herramienta será usada tanto en el desarrollo FrontEnd, BackEnd, Android y documentación.

## Herramientas de entorno

* + - Visual Studio Professional (Frontend-Web)

Visual Studio Professional es un entorno de desarrollo integrado (IDE, por sus siglas en inglés) para sistemas operativos Windows y **MacOS** (Actualmente-2018 Sistema Operativo más utilizado por los desarrolladores en NN-Consulting). Soporta múltiples lenguajes de programación, tales como C++, **C#** (Actualmente-2018 el lenguaje más utilizado por los desarrolladores en NN-Consulting), Visual Basic .NET, F#, Java, Python, Ruby y PHP, al igual que entornos de desarrollo web, como **ASP.NET MVC** (Actualmente-2018 el más utilizado por los desarrolladores en NN-Consulting), Django, etc. Permite administrar el código en repositorios GIT que hospede cualquier proveedor, como GitHub y Visual Studio Team Services.

* + - Android Studio (Móvil)

Es el entorno de desarrollo integrado oficial para la plataforma Android. Fue anunciado el 16 de mayo de 2013 en la conferencia Google I/O, y reemplazó a Eclipse como el IDE oficial para el desarrollo de aplicaciones para Android. La primera versión estable fue publicada en diciembre de 2014. Está basado en el software IntelliJ IDEA de JetBrains, y es publicado de forma gratuita a través de la Licencia Apache 2.0. Está disponible para las plataformas Microsoft Windows, Mac OS X y GNU/Linux.

* + - PhpStorm (Back-End)

PhpStorm es un IDE de programación desarrollado por JetBrains. Es uno de los entornos de programación más completos de la actualidad, permite editar código no sólo del lenguaje de programación php como lo indica su nombre. PhpStorm permite trabajar con Symfony, Drupal, WordPress, Zend Framework, **Laravel** (Actualmente-2018 framework más utilizado por el equipo de desarrollo en NN-Consulting), Magento, Joomla!, CakePHP, Yii, y otros frameworks.

* + - Servidor-Digital Ocean

DigitalOcean es un proveedor estadounidense de servidores virtuales privados, basado en la ciudad de Nueva York. La compañía alquila instalaciones de centros de cómputo existentes, incluyendo sitios como Nueva York, Toronto, Bangalore, Ámsterdam, San Francisco, Londres y Singapur. El servidor con el que cuenta NN-Consulting tiene las siguientes características: VPS Linux (Ubuntu 16.04) 1 CPU, 1GB RAM, 25 GB SSD.

## Calendario

A continuación, se presenta la tabla de desarrollo de actividades (hitos) incluidas dentro del Proceso de Gestión de la Configuración de Software para NN-Consulting (Ver Tabla 2).

Tabla 2: Calendario del Plan de Gestión de la Configuración

|  |  |  |  |
| --- | --- | --- | --- |
| Actividad | Rol | Dependencia | Comienzo |
| Realizar el plan de gestión de la configuración | CMO, Ingenieros | Ninguna | 1 semana |
| Realizar la actividad de identificación de SCM | CMO, Ingenieros | Plan de gestión de la configuración | 1 semana |
| Realizar la definición de la línea base y estructura de librerías | CMO, Bibliotecarios, Ingenieros | Plan de gestión de la configuración | 1 semana y media |
| Realizar el plan de gestión de cambios | CMO, CCB, Ingenieros | Plan de gestión de la configuración | 1 semana y media |
| Realizar los reportes de estado | CMO, CCB, Bibliotecarios, Personal de apoyo | Plan de gestión de la configuración y plan de gestión de cambios | 1 semana |
| Realizar los reportes de auditorias | CMO, CCB, Ingenieros, Personal de apoyo | Plan de gestión de la configuración y plan de gestión de cambios | 1 semana |
| Realizar la gestión de release | CMO, CCB, Personal de apoyo | Plan de gestión de la configuración | 1 semana |

# Actividades de la Gestión de la Configuración

## Lista de Items de la Configuración

|  |  |  |  |
| --- | --- | --- | --- |
| **Tipo** | **Nombre de Ítem** | **Fuente** | **Extensión** |
| E | Plan de la gestión de la configuración | E | DOC |
| E | Documento de especificación de requisitos | P | DOC |
| E | Documento de diseño y arquitectura de software | P | DOC |
| E | Documento de Pruebas | P | DOC |
| E | Manual de usuario | P | DOC |
| E | Aplicación móvil | P | APK |
| F | Código fuente de la aplicación móvil | P | ZIP |
| F | Script de la base de datos | P | ZIP |
| F | Scripts de front-end | P | ZIP |
| F | Vistas del front-end | P | ZIP |
| F | Compilados del código fuente front-end | P | ZIP |
| E | Documentación del Api | P | DOC |
| F | Código fuente back-end | P | ZIP |
| F | Manual de instalación | P | DOC |

## Definición de la Nomenclatura de los Ítems de Configuración

### Ítems en evolución

Los ítems en evolución son de dos tipos: documentos, y archivos ejecutables o de soporte.

* **Documentos**

Los ítems en evolución tipo documento se asignan identificadores únicos que permiten identificar el proyecto y componente (si es aplicable) con el cual está asociado, junto con el nivel de revisión actual. El identificador consiste en una a tres partes separadas por guión bajo en el formato: PROYECTO\_ACRÓNIMO, ó, PROYECTO\_ACRÓNIMO\_COMPONENTE.

Los ítems en evolución que son ítems no específicos a un proyecto único, tales como políticas, descripciones de procesos y guías, son identificados únicamente por su acrónimo, por ejemplo: PGC (Plan de Gestión de Configuración)

Los ítems en evolución que son específicos de un proyecto, pero no asociados con un componente del proyecto, utilizan un identificador de dos partes: ACRÓNIMO\_PROYECTO y ACRÓNIMO derivado del tipo de artefacto. Por ejemplo, para identificar el plan de proyecto para el proyecto SPACIA, tenemos: SPACIA\_PP.

Los ítems en evolución que son específicos de un proyecto y están asociados con un componente específico, utilizan un identificador de tres partes: ACRÓNIMO\_PROYECTO, ACRÓNIMO\_COMPONENTE, y, ACRÓNIMO derivado del tipo de artefacto. Por ejemplo, para identificar el documento de especificación de requerimientos, del componente B2B (Business to Business) del proyecto ACME, tenemos: ACME-B2B-SRD.

El nivel de versión de cada ítem se mantiene como un identificador separado. Esto permite que el identificador principal sea utilizado como parte del nombre de archivo o URL para acceder a la versión más actualizada sin necesidad de requerir cambios a todos los ítems referenciados. El nivel de versión se mantiene como un identificador numérico con dos componentes: Versión.Revisión. Ejemplo: 1.1; Versión 1, Revisión 1.

**Número de versión**

El número de versión cambia únicamente cuando la arquitectura principal del ítem ha cambiado, o cuando el ítem es completamente reconstruido, con cambios internos sustanciales. En este caso la versión 1.1 se convertirá en versión 2.0.

**Número de revisión**

El número de revisión cambia cuando el contenido ha cambiado, pero la estructura principal y el flujo del ítem se mantiene igual. La secuencia normal de las revisiones es: 1.0, 1.1, 1.2, etc.

**Archivos ejecutables y de soporte**

Los ejecutables del software y los archivos de soporte son identificados generalmente por el nombre y el número de versión, tales como “Main DB v1.1a"

El esquema de numeración de versiones consiste en tres componentes: Versión.RevisiónActualización. Ej: 1.1a.

**Número de versión**

El número de versión cambia únicamente cuando la arquitectura principal del ítem de software cambia, cuando migramos de un nivel de herramienta de desarrollo a otro, cuando una aplicación es totalmente reconstruida, o cuando se producen cambios sustanciales en la GUI. En este caso, la versión 1.1a se convierte en la versión 2.0

**Número de revisión**

El número de revisión es actualizado cuando se añade nuevas características, funcionalidad y otro contenido, o estas han cambiado significativamente. Normalmente la arquitectura principal o la GUI ha sido extendida o limitada de alguna manera. La razón más común de cambiar el número de revisión es cuando añadimos un nuevo módulo u otra funcionalidad al ítem de software. La secuencia normal de revisión es 1.0, 1.1, 1.2, etc.

**Carácter de actualización**

El carácter de actualización se incrementa cuando el único cambio al ítem de software es corregir uno o más defectos, sin añadir ninguna nueva funcionalidad. Las actualizaciones evolucionan 1.1a, 1.1b, etc. Esta actualización se sobrescribe cuando una revisión combinada, que incluye arreglar defectos y añadir nuevas características, se lleva a cabo. En tal caso, se incrementa el número de revisión y se descarta el carácter, es decir: 1.1b a 1.2.

**Identificación de ítems fuente**

Esto se maneja en base a la herramienta de gestión de la configuración utilizada, o de las utilidades del entorno de desarrollo para versionamiento.

**Identificación de ítems de soporte**

Son identificados por su nombre y el número de versión necesario para soportar el entorno de producción o desarrollo. Por ejemplo, si un editor se actualiza de la versión 2.1 a 2.2a, el rango de versión del ítem de configuración será 2.1 – 2.2a.

* 1. **Listado de Items con la Nomenclatura**

En esta sección, a cada uno de los ítems que ya fueron identificados

anteriormente en una tabla, se les asigna una nomenclatura en base a las reglas que fueron definidas.

Tabla 3: Nomenclatura de los ítems durante el ciclo del desarrollo del software

|  |  |  |
| --- | --- | --- |
| **Nomenclatura** | **Descripción del Item(CI)** | **Fuente** |
| SPC\_PGC | Plan de la gestión de la configuración | E |
| SPC\_DER | Documento de especificación de requisitos | P |
| SPC\_DDAS | Documento de diseño y arquitectura de software | P |
| SPC\_DP | Documento de Pruebas | P |
| SPC\_MU | Manual de usuario | P |
| SPC\_AM | Aplicación móvil | P |
| SPC\_AMCF | Código fuente de la aplicación móvil | P |
| SPC\_SBD | Script de la base de datos | P |
| SPC\_SFE | Scripts de front-end | P |
| SPC\_VFE | Vistas del front-end | P |
| SPC\_CFE | Compilados del código fuente front-end | P |
| SPC\_DAPI | Documentación del Api | P |
| SPC\_CFBE | Código fuente back-end | P |
| SPC\_MI | Manual de instalación | P |