**Plan de Gestión de la Configuración del Software (SCMP) — Proyecto MyVet**

**Integrantes: Enzo González, María Ester Zavala, Nicolás Pardo**

**Carrera: Ingeniería en Informática**

**Asignatura: Gestión de Proyectos**

**Docente: Javier Miles**

**Fecha: 27 de jun. de 25**

Table of Contents

[**1. Introducción** 4](#_Toc201954141)

[**1.1 Propósito** 4](#_Toc201954142)

[**1.2 Alcance** 4](#_Toc201954143)

[**1.3 Glosario** 5](#_Toc201954144)

[**1.4 Referencias** 6](#_Toc201954145)

[**2. Gestión** 7](#_Toc201954146)

[**2.1 Organización** 7](#_Toc201954147)

[**2.2 Responsabilidades de SCM** 7](#_Toc201954148)

[**2.3 Gestión de Interfaces** 8](#_Toc201954149)

[**2.4 Implementación del SCMP** 8](#_Toc201954150)

[**2.5 Políticas, Directivas y Procedimientos Aplicables** 8](#_Toc201954151)

[**3. Identificación de la Configuración** 9](#_Toc201954152)

[**3.1 Convenciones** 9](#_Toc201954153)

[**3.2 Líneas Base** 9](#_Toc201954154)

[**4. Control de la Configuración** 10](#_Toc201954155)

[**4.1 Control de Código (y Documentos)** 10](#_Toc201954156)

[**4.2 Control de Medios** 10](#_Toc201954157)

[**4.3 Control de Cambios** 10](#_Toc201954158)

[**5. Contabilidad del Estado de la Configuración** 11](#_Toc201954159)

[**6. Herramientas, Técnicas y Métodos para SCM** 11](#_Toc201954160)

[**7. Control de Proveedores** 11](#_Toc201954161)

[**8. Recopilación y Retención de Registros** 11](#_Toc201954162)

[Respuesta a las preguntas El Proceso G.C.S 13](#_Toc201954163)

**Información del Servicio:**

* **Resumen:** Este documento define la organización, procedimientos, herramientas y responsabilidades para la gestión de la configuración del software (SCM) durante la fase de Diseño Detallado y Producción (DD) del proyecto MyVet. Su objetivo es asegurar la integridad, control y trazabilidad de todos los elementos de configuración generados en esta etapa.
* **Tabla de Contenidos:** (Agregar lo mismo de evaluación de proyecto o sea el indice)
* **Hoja de Estado del Documento (DSS):**
  + **Versión:** 1.0
  + **Fecha:** 27 de junio de 2025
  + **Razón del Cambio:** Versión inicial del plan para la fase DD.
* **Registro de Cambios del Documento (DCR):** No aplica, es la primera versión.

**1. Introducción**

**1.1 Propósito**

El propósito de este SCMP/DD es documentar todas las actividades de gestión de la configuración que se harán durante la fase de Diseño Detallado y Producción del proyecto MyVet. Este plan está destinado al equipo de desarrollo del proyecto (Enzo González, María Ester Zavala, Nicolás Pardo) y al profesor supervisor (Javier Miles), para garantizar que el desarrollo del software se realice de manera controlada y sistemática.

**1.2 Alcance**

Este plan aplica a todos los elementos de configuración (CIs) producidos durante la fase DD, que incluyen:

* Documento de Diseño Detallado (DDD).
* Manual de Usuario del Software (SUM).
* Código fuente de la aplicación móvil y del backend.
* Scripts de construcción y despliegue.
* Casos de prueba y datos asociados.
* Programas ejecutables (APK).

Las actividades de SCM cubiertas son: identificación, almacenamiento, control de cambios y contabilidad del estado de la configuración. El plan es aplicable a la organización del proyecto MyVet, compuesta por sus tres integrantes.

**1.3 Glosario**

**MyVet**

El nombre del proyecto, que consiste en una aplicación móvil para optimizar la gestión de citas veterinarias. Su función principal es permitir el registro de síntomas y con inteligencia artificial (IA), generar un diagnóstico preliminar para el veterinario.

**PETI (Plan Estratégico de TI)**

Es el documento que define la hoja de ruta para el uso y la administración de los recursos tecnológicos en MyVet. Su objetivo es alinear las decisiones sobre la tecnología (arquitectura, seguridad, etc.) con los objetivos estratégicos del proyecto.

**Línea Base (Baseline)**

Un conjunto de elementos de configuración (documentos, código fuente) que ha sido formalmente revisado y aprobado en una etapa específica del proyecto, como la aprobación del documento de requisitos.

**CI (Elemento de Configuración)**

Es cada uno de los productos de software generados y controlados en el proyecto. Incluye documentos como el URD, SRD, ADD, DDD, así como el código fuente, los ejecutables de la app y los datos de prueba.

**PMO (Oficina de Gestión de Proyectos)**

En el contexto de MyVet, se establece como una PMO de soporte, cuya función es asistir al equipo en la planificación, monitorear el cumplimiento de plazos y calidad, y estandarizar procesos.

**COBIT**

Un marco de gobernanza de TI que se ha aplicado a MyVet para asegurar que la tecnología se alinee con los objetivos, se gestionen los riesgos y se definan claramente los roles y procesos.

**Análisis FODA**

Una herramienta de diagnóstico utilizada en la planificación de MyVet para identificar las **F**ortalezas, **O**portunidades, **D**ebilidades y **A**menazas del proyecto y su entorno tecnológico.

**SPR (Informe de Problema de Software)**

Es el reporte formal que se utiliza para describir un problema, error o defecto encontrado en la aplicación MyVet durante las pruebas o la operación.

**SCR (Solicitud de Cambio de Software)**

Es el documento que propone y describe los cambios necesarios para solucionar un problema (reportado en un SPR) o para añadir una nueva funcionalidad. Incluye una estimación de esfuerzo y el personal responsable.

**CCC (Comité de Control de Cambios)**

Para MyVet, este rol es asumido por la **Junta de Revisión**, compuesta por los miembros del equipo y el profesor. Es la autoridad responsable de evaluar, aprobar o rechazar las solicitudes de cambio (SCRs) que afecten a una línea base establecida.

**Fase DD (Diseño Detallado y Producción)**

Es la fase del proyecto donde se lleva a cabo la codificación de cada módulo según el diseño, se realizan las pruebas unitarias y se integra el sistema. Los entregables claves de esta fase son el Documento de Diseño Detallado (DDD), el Manual de Usuario (SUM) y el código fuente validado.

**1.4 Referencias**

Los siguientes documentos son aplicables a este plan:

1. **ESA PSS-05-09:** Guide to software configuration management (Documento rector).
2. **Plan de Administración de Proyecto de Software (PAPS):** Define la organización y fases del proyecto.
3. **Plan Estratégico de TI MyVet (PETI):** Describe la infraestructura y herramientas.
4. **Ley N° 19.628:** Sobre protección de la vida privada, criterio clave para el manejo de datos.

**2. Gestión**

**2.1 Organización**

La estructura organizacional del proyecto, basada en los roles típicos definidos, se compone de:

* **Project Manager (PM):** Responsable de la planificación, supervisión general y de actuar como la autoridad para la aprobación de cambios en las líneas base. Este rol es asumido por el Director de Proyecto.
* **Programadores (Developers):** Encargados del diseño detallado, codificación, pruebas unitarias y documentación técnica. Son la autoridad de control sobre los item de configuración en sus entornos de desarrollo.
* **Configuration Manager (CM):** Responsable de la administración de las bibliotecas de software, el control de versiones y la ejecución de los procedimientos de SCM.
* **Junta de Revisión (Review Board):** Compuesta por los tres miembros del equipo y el profesor del proyecto. Es responsable de revisar y aprobar cambios formales a las líneas base establecidas

**2.2 Responsabilidades de SCM**

Las responsabilidades se distribuyen de la siguiente manera:

* **Identificación de CIs:** El equipo completo definirá los CIs al inicio de la fase.
* **Almacenamiento y Control de Bibliotecas:**
  + Los **Programadores** gestionan sus **bibliotecas de desarrollo** (ramas personales en GitHub).
  + El **PM/CM** controla el acceso y las fusiones a la **biblioteca maestra** (rama main de GitHub) y la creación de **bibliotecas de archivo** (releases/tags).
* **Control de Cambios:**
  + Los cambios informales son gestionados por cada Programador en su entorno.
  + Los cambios a la línea base deben ser solicitados formalmente (vía SPR/SCR) y aprobados por la
* **Junta de Revisión**.
* **Contabilidad del Estado:** El **PM/CM** es responsable de mantener y reportar el estado de los CIs.

**2.3 Gestión de Interfaces**

El proyecto interactúa con servicios externos como Firebase/AWS y MongoDB Atlas. La gestión de la configuración de estas interfaces es responsabilidad del **PM/CM**. No existen interfaces con otras organizaciones de desarrollo de software.

**2.4 Implementación del SCMP**

Los eventos clave para la implementación de este plan son:

* **Establecimiento de herramientas SCM:** Uso de GitHub, Planner y Google Drive desde el inicio de la fase DD.
* **Establecimiento de la Línea Base DD:** Al aprobarse el Documento de Diseño Detallado (DDD) en la revisión formal (DD/R).
* **Junta de Revisión de Cambios (CCC):** Se activará formalmente después de establecer la línea base DD para gestionar todos los cambios posteriores.

**2.5 Políticas, Directivas y Procedimientos Aplicables**

Este SCMP/DD es el documento principal que rige las actividades de SCM. Se utilizan herramientas de acceso libre y recursos personales.

**3. Identificación de la Configuración**

**3.1 Convenciones**

Cada elemento de configuración (CI) se identificará de forma única.

* **Nomenclatura de CIs:** Se utilizará el siguiente formato: MYVET-[TIPO]-[NOMBRE]-[VERSION]
  + **[TIPO]:** DOC (Documento), SRC (Código Fuente), TEST (Prueba), EXEC (Ejecutable).
  + **[NOMBRE]:** Nombre descriptivo del CI (ej. DDD, UserAuth, LoginTests).
  + **[VERSION]:** Issue.Revision para documentos (ej. 1.0) y Major.Minor.Patch para código (ej. 1.0.0).
* **Ejemplos:**
  + MYVET-DOC-DDD-1.0
  + MYVET-SRC-SymptomAnalysis-1.2.0.kt
  + MYVET-EXEC-App-1.0.0.apk
* **Cabecera de Módulos de Código Fuente:** Todos los archivos de código fuente deben incluir una cabecera estándar con:
  + Identificador del CI (nombre, tipo, versión).
  + Autor original.
  + Fecha de creación.
  + Historial de cambios (versión, fecha, autor, descripción del cambio, SPR/SCR de referencia).

**3.2 Líneas Base**

Para la fase DD, se establecerá una única línea base principal:

* **Identificador de la Línea Base:** BL-DD-1.0
* **Contenido:**
  + **Software:** MYVET-DOC-DDD-1.0, MYVET-DOC-SUM-1.0, todo el código fuente validado y el ejecutable inicial.
  + **Herramientas:** Versiones específicas de Android Studio, JDK, Gradle, y librerías de IA (ej. TensorFlow Lite).
  + **Software de Prueba:** Scripts y datos de prueba para pruebas unitarias y de integración.
* **Evento de Aprobación:** La Revisión de Diseño Detallado (DD/R).
* **Criterios de Aceptación:** Aprobación del DDD y demostración de que los componentes de software codificados son consistentes con el diseño.

**4. Control de la Configuración**

**4.1 Control de Código (y Documentos)**

Se utilizará una estructura de tres bibliotecas, implementada con GitHub y Google Drive.

* **Bibliotecas de Desarrollo (Dinámicas):** Corresponden a las ramas de características (Feature) en GitHub creadas por cada Programador. El control es individual.
* **Biblioteca Maestra (Controlada):** Es la rama main del repositorio central en GitHub. Los cambios se integran únicamente mediante.

*Pull Requests* revisados y aprobados por el PM/CM.

* **Biblioteca de Archivo (Estática):** Se implementa usando "Releases" y "Tags" en GitHub para cada línea base y entrega final. Estas son inmutables.

**4.2 Control de Medios**

* **Medios Electrónicos:** El control se centra en la gestión de los repositorios en la nube (GitHub, Google Drive) y la instancia de MongoDB Atlas.
* **Etiquetado:** Los repositorios, carpetas y "releases" seguirán la nomenclatura del proyecto y de la línea base correspondiente.
* **Respaldo:** Se confía en los mecanismos de respaldo de los proveedores de servicios en la nube. Se realizarán respaldos locales del ejecutable y la documentación final antes de cada entrega.

**4.3 Control de Cambios**

* **4.3.1 Niveles de Autoridad:**
  1. **Programador:** Autoridad sobre su biblioteca de desarrollo.
  2. **PM/CM:** Autoridad para aprobar la integración de cambios a la biblioteca maestra.
  3. **Junta de Revisión:** Única autoridad para aprobar cambios a una línea base ya establecida.
* **4.3.2 Procedimientos de Cambio:**
  1. **Documentación:** Los cambios a documentos en una línea base seguirán el proceso RID/DCR/DSS, gestionado a través de *Issues* de GitHub.
  2. **Código:** Se utilizará el proceso SPR/SCR/SMR. Los problemas se reportan con un
  3. **Software Problem Report (SPR)** (un *issue* en GitHub). La solución se propone en un
  4. **Software Change Request (SCR)** (otro *issue* o propuesta en *Pull Request*). La implementación se documenta en un
  5. **Software Modification Report (SMR)** (descripción del *Pull Request* y mensajes de *commit*).
* **4.3.3 Junta de Revisión:**
  1. **Miembros:** El equipo de desarrollo y el profesor supervisor.
  2. **Funcionamiento:** Se reunirá según sea necesario para evaluar los SCRs que afecten a la línea base. Las decisiones se registrarán.

**5. Contabilidad del Estado de la Configuración**

La contabilidad del estado (CSA) se realizará para rastrear el progreso y el historial de cada CI.

* **Recopilación de Datos:** La información se recolectará de forma continua a partir de:
  + Logs de git para el historial de cambios en el código.
  + El tablero de *Issues* de GitHub para el estado de SPRs, SCRs y RIDs.
  + La sección de "Releases" en GitHub para el contenido de las líneas base archivadas.
  + La Hoja de Estado del Documento (DSS) en cada documento.
* **Informes:** Se generará un **Informe de Estado de la Configuración** semanalmente para las reuniones de equipo, resumiendo el estado de los cambios y la evolución de los CIs.

**6. Herramientas, Técnicas y Métodos para SCM**

El proyecto utilizará un conjunto de herramientas de nivel básico-avanzado.

* **GitHub:** Para el almacenamiento de CIs de código, control de versiones, control de cambios y bibliotecas.
* **Google Drive:** Para el almacenamiento de CIs de documentación y entregables finales.
* **Trello:** Para la gestión visual de tareas y seguimiento de alto nivel del flujo de trabajo de cambios.
* **Android Studio:** Entorno de desarrollo con integración a git.

**7. Control de Proveedores**

Los proveedores del proyecto son las plataformas en la nube utilizadas (GitHub, Google, MongoDB). No se subcontratará ninguna parte del desarrollo. El control se limita a la gestión de la configuración de dichos servicios, aceptando sus términos y prácticas de gestión.

**8. Recopilación y Retención de Registros**

* **Registros a Retener:** Se conservará toda la información relacionada con la SCM, incluyendo el historial de CIs, registros de cambios y decisiones de la Junta de Revisión.
* **Método de Retención:** Todos los registros se mantendrán en formato electrónico en las plataformas designadas (GitHub, Google Drive).
* **Período de Retención:** Los registros se conservarán indefinidamente dentro de las plataformas y, como mínimo, durante todo el ciclo académico del proyecto y por dos años adicionales como archivo de consulta.

**Respuesta a las preguntas El Proceso G.C.S**

* **¿Cómo identifica y gestiona una organización las muchas versiones existentes de un programa (y su documentación) de forma que se puedan introducir cambios eficientemente?** La Identificación de la configuración: Esta tarea establece un esquema y estándares para identificar todos los elementos que componen el software. Se asignan identificadores únicos a cada programa, documento. lo que permite gestionar las distintas versiones y facilita el control de cambios y actualizaciones.
* **¿Cómo controla la organización los cambios antes y después de que el software sea distribuido al cliente?** La respuesta es el **Control de cambios**. Este proceso consiste en la evaluación, registro y aprobación de todas las modificaciones realizadas a los elementos de la configuración del software a lo largo de su ciclo de vida. Se aplican diferentes niveles de control, como el individual (informal), el de gestión (durante el desarrollo) y el formal (durante el mantenimiento), para gestionar los cambios de manera ordenada.
* **¿Quién tiene la responsabilidad de aprobar y de asignar prioridades a los cambios?**
* La responsabilidad recae en el **Comité de Control de Cambios (CCC):** Este comité está compuesto por miembros de las organizaciones de usuarios y desarrolladores. Entre sus tareas se encuentran analizar el impacto de los cambios, así como categorizar y asignar prioridades a las modificaciones solicitadas y aprobadas.
* **¿Cómo podemos asegurar que los cambios se han llevado a cabo adecuadamente?** la Auditoria de Configuraciones, Junto con las revisiones técnicas formales, las auditorías sirven para garantizar que un cambio se ha implementado de manera correcta. Verifican si el cambio se realizó según lo especificado en la orden de cambio, si se siguieron los estándares de ingeniería y si todos los elementos de configuración relacionados fueron actualizados adecuadamente.
* **¿Qué mecanismos se usan para avisar a otros de los cambios realizados?** El mecanismo principal es la **Generación de informes**. Estos informes documentan y comunican los cambios realizados, sirviendo como el mecanismo para notificar a las partes interesadas.