

# AZERTY

---

Trucardo

## Configuration Management Plan

---

**Autores:**

- Colque Ventura, Santiago Agustín
- Fernandez, Ignacio Javier
- Lopez Paviolo, Franco Marcelo

**Fecha:**

- 30 de abril de 2020

*Versión 1.0.1*  
*Grupo AZERTY*  
*Año 2020*

## ÍNDICE

	<b>HISTORIAL DE CAMBIOS .....</b>	<b>4</b>
	<b>ACRÓNIMOS/GLOSARIO .....</b>	<b>5</b>
<b>1</b>	<b>INTRODUCCIÓN .....</b>	<b>6</b>
1.1	PROPÓSITO Y ALCANCE DEL PLAN .....	6
1.2	PROPÓSITO DEL PLAN DE GESTIÓN DE CONFIGURACIONES.....	6
1.3	HERRAMIENTAS PARA LA ADMINISTRACIÓN DE LAS CONFIGURACIONES .....	6
1.4	ROLES Y RESPONSABILIDADES .....	7
<b>2</b>	<b>ESQUEMA DE DIRECTORIOS .....</b>	<b>8</b>
2.1	ESTRUCTURA DE DIRECTORIOS Y PROPÓSITO DE CADA UNO .....	8
2.2	NORMAS DE ETIQUETADO Y NOMBRAMIENTO DE ARCHIVOS .....	8
<b>3</b>	<b>GESTIÓN DE LA CONFIGURACIÓN DEL CÓDIGO .....</b>	<b>9</b>
3.1	ESQUEMA DE RAMAS .....	9
3.2	POLÍTICA DE ETIQUETADO DE LAS RAMAS .....	9
3.3	POLÍTICA DE FUSIÓN DE ARCHIVOS.....	9
<b>4</b>	<b>GESTIÓN DE CAMBIOS .....</b>	<b>11</b>
4.1	CHANGE CONTROL BOARD (CCB) .....	11
4.1.1	INTRODUCCIÓN Y OBJETIVOS .....	11
4.1.2	MIEMBROS .....	11
4.1.3	FRECUENCIA DE REUNIÓN DE TRABAJO.....	11

4.2	PROCESO DE CONTROL DE CAMBIOS .....	12
4.3	HERRAMIENTA DE CONTROL DE CAMBIOS .....	12
5	GESTIÓN DE CAMBIOS .....	12
5.1	FORMATO DE ENTREGA DE RELEASES .....	12
5.2	FORMATO DE ENTREGA DEL INSTALADOR .....	12
5.3	INSTRUCCIONES MÍNIMAS DE INSTALACIÓN .....	12

**HISTORIAL DE CAMBIOS**

<b>Versión</b>	<b>Fecha</b>	<b>Cambios</b>
<b>1.0.0</b>	<b>30 abril 2020</b>	<b>Primera versión</b>
<b>1.0.1</b>	<b>7 mayo 2020</b>	<b>Se agregó índice</b>

**ACRÓNIMOS/GLOSARIO**

<b>Acrónimo</b>	<b>Descripción</b>
<b>CCB</b>	Change Control Board
<b>CM</b>	Configuration Management
<b>SCM</b>	Software Configuration Management
<b>GPCM</b>	Global Project Configuration Manager
<b>CR</b>	Change Request

## 1 INTRODUCCIÓN

### 1.1 PROPÓSITO Y ALCANCE DEL PLAN

Este documento plantea el plan de Gestión de las Configuraciones (Configuration Management Plan) del proyecto Trucardo. El objetivo del plan de CM es el controlar la configuración de los requerimientos, documentos, software y herramientas utilizadas en este proyecto. SCM es el proceso mediante el cual se identifican las herramientas y métodos de control a lo largo del desarrollo y uso.

### 1.2 PROPÓSITO DEL PLAN DE GESTIÓN DE CONFIGURACIONES

- Mantener la integridad del proyecto
- Definir los roles del equipo dentro del proyecto
- Crear un historial del desarrollo del proyecto
- Mantener informados a los integrantes del proyecto sobre el estado actual

### 1.3 HERRAMIENTAS PARA LA ADMINISTRACIÓN DE CONFIGURACIONES

Herramienta	Descripción	Forma de acceso
Trucardo_GitHub	Control de versiones del proyecto	<a href="https://github.com/francomlopez/Trucardo">https://github.com/francomlopez/Trucardo</a>
Trucardo_Circleci	Control de Integración Continua para realizar tests	<a href="https://app.circleci.com/pipelines/github/francomlopez/Trucardo">https://app.circleci.com/pipelines/github/francomlopez/Trucardo</a>
Trucardo_Monday	Gestión de tareas del proyecto	<a href="https://azerty48.monday.com/boards/550122241">https://azerty48.monday.com/boards/550122241</a>
Trucardo_GitHub-issues	Gestión de defectos a corregir	<a href="https://github.com/ignacioFernandez1/Trucardo/issues">https://github.com/ignacioFernandez1/Trucardo/issues</a>

**1.4 ROLES Y RESPONSABILIDADES**

<b>Rol</b>	<b>Descripción</b>	<b>Responsable</b>
Gestor global de configuración (GPCM)	<ul style="list-style-type: none"><li>● Responsable de asegurar que se cumpla con lo establecido en el plan de gestión de configuración.</li><li>● Promover el uso efectivo de las herramientas elegidas para la realización del proyecto</li><li>● Monitorear y reportar los cambios en las branches del proyecto</li><li>● Monitorear y controlar el correcto uso de labels en las distintas versiones</li><li>● Aprobar cambios estructurales en el plan de gestión de configuración.</li></ul>	Ignacio Fernandez
Coordinador de configuración	<ul style="list-style-type: none"><li>● Asegurar que todos los elementos de configuración están registrados de forma adecuada en el plan de configuración.</li><li>● Asegurar la consistencia e integridad de los datos del plan de configuración y la estructura del sistema a través de la ejecución de procedimientos de verificación y auditoría.</li><li>● Reportar cualquier discrepancia o no conformidad en los elementos de configuración al gestor de configuración.</li><li>● Participar en la mejora continua del proceso de gestión de configuración.</li></ul>	Franco Lopez

Responsable de elementos de configuración	<ul style="list-style-type: none"><li>● Asegurar que los elementos de configuración de los que es responsable están registrados en el plan de configuración con el estado y datos de configuración apropiados.</li><li>● Verificar que los cambios sobre los elementos de configuración siguen el proceso de cambios definido.</li><li>● Trabajar conjuntamente con el gestor de configuración para identificar las causas de cualquier discrepancia identificada en las auditorías e implementar las acciones correctivas.</li></ul>	Santiago Colque
---	---	-----------------

## 2 ESQUEMA DE DIRECTORIOS

### 2.1 ESTRUCTURA DE DIRECTORIOS Y PROPÓSITO DE CADA UNO

Nuestro proyecto se almacenará dentro de un directorio base llamada Trucardo. Dentro del mismo podremos encontrar:

- Trucardo\Documentacion : Planes generales y archivos relevantes que aplican para todos los releases.
- Trucardo\Releases: dentro de este directorio se encontrarán todos los releases realizados con su documentación correspondiente.
- Trucardo\src: se encuentra el código en Java de nuestro proyecto y los test correspondientes.

### 2.2 NORMAS DE ETIQUETADO Y NOMBRAMIENTO DE ARCHIVOS

Para las diferentes versiones de los documentos seguiremos un estándar del tipo <NombreDoc>V1.x.x.

Para las distintas releases que se vayan realizando utilizaremos el siguiente formato Rel<ID>.



### 3 GESTIÓN DE LA CONFIGURACIÓN DEL CÓDIGO

#### 3.1 ESQUEMA DE RAMAS

Se utilizará un esquema con cuatro tipos de rama. Estos son: *master*, *stable*, *feature*, *bug* y *hotfixes*.

Las ramas *master* y *stable* serán las principales, y perpetuarán indefinidamente.

La rama *stable* se utilizará para versiones estables y releases, mientras que la rama *master* será la rama principal. Los desarrolladores realizarán merges y branches tomando como punto de partida a esta rama.

Las ramas *feature* tendrán lugar a la hora de desarrollar una nueva función o realizar una mejora que potencialmente acabarán incorporándose al proyecto. Una vez integrada la nueva funcionalidad a la rama *master*, esta terminará su vida útil.

Las ramas *bug* se crearán al descubrir un bug en la rama *master*, y al igual que en *feature*, llegarán a su fin una vez realizado el merge correspondiente hacia *master*.

Una rama *hotfix* surge de la necesidad de actuar inmediatamente al tomar conocimiento de un comportamiento indeseado en un release, parte desde la rama *stable* de manera tal que posibilita trabajar en el fix y en *master* al mismo tiempo.

#### 3.2 POLÍTICA DE ETIQUETADO DE RAMAS

Si la rama no existe todavía, se creará una nueva rama con un número de id de la forma <tipo\_de\_branch-id>. Por ejemplo *feature-1* o *bug-2*.

#### 3.3 POLÍTICA DE FUSIÓN DE ARCHIVOS

La política de fusión de cada rama se detalla a continuación:

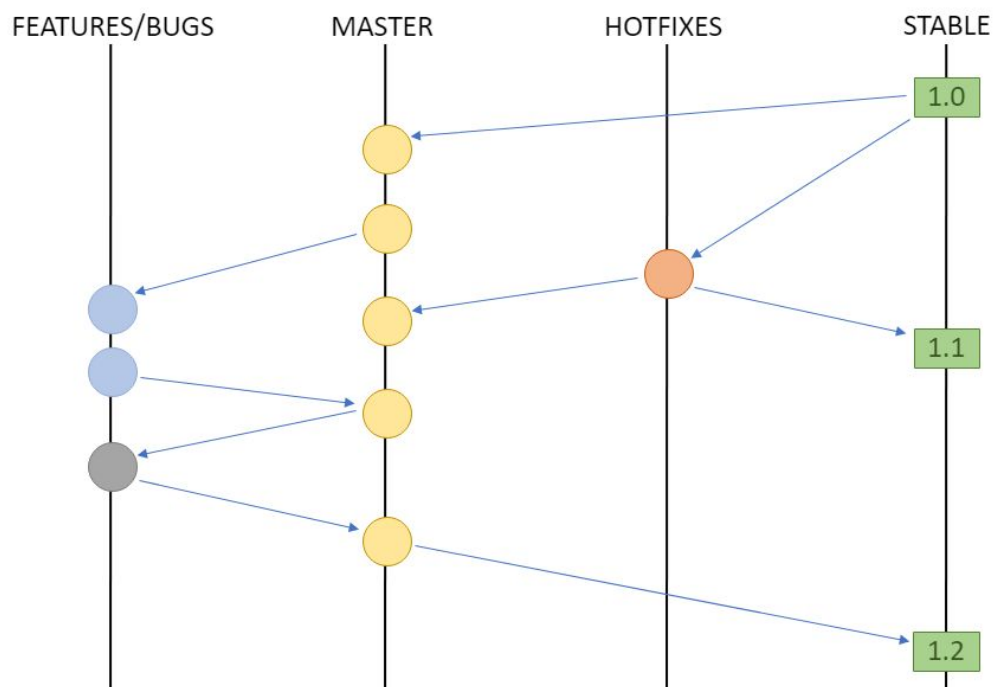
- *feature*:
  - Debe salir de la rama *master*
  - Debe mergear a la rama *master*
  - Cualquier cambio en *master* debe ser reflejado en *feature* antes de volver a mergear a *master*

- *bug*:
  - Debe salir de la rama *master*
  - Debe mergear a la rama *master*
  - Cualquier cambio en *master* debe ser reflejado en *bug* antes de volver a mergear a *master*
- *hotfix*:
  - Debe salir de la rama *stable*
  - Debe mergear a la rama *master* y a la rama *stable*

Además, cuando el código de la rama *master* es estable y apto para release, este debe ser mergeado hacia *stable*.

No se podrá mergear a *master* si no se pasan los test que son corridos en Circleci.

Todo esto se puede ver de una manera más gráfica, en el ejemplo expuesto en el diagrama siguiente:



## 4 GESTIÓN DE CAMBIOS

### 4.1 CHANGE CONTROL BOARD (CCB)

#### 4.1.1 INTRODUCCIÓN Y OBJETIVOS

En este comité se revisarán los cambios a introducir en el proyecto ya sea en la documentación o en el código fuente del producto. El objetivo de este comité es aplicar los cambios necesarios en el proyecto con la aprobación de todos sus miembros.

#### 4.1.2 MIEMBROS

Rol	Responsabilidad	Responsable
CCB Líder	Responsable de manejar la herramienta de gestión de defectos (GitHub Issues), documentando específicamente sobre cada problema que se trabaja.  Dar la aprobación final al cambio a realizar en el proyecto y documentando la finalización del problema.	Franco Lopez Paviolo
Gestor de problemas	Encargado de traer los problemas a las reuniones del comité con todas las especificaciones necesarias.	Santiago Colque
Gestor global de configuración	Asegurar que el procedimiento de corrección de errores cumpla con lo dispuesto en el plan de gestión de configuración.	Ignacio Fernandez

#### **4.1.3 FRECUENCIA DE REUNIÓN DE TRABAJO**

Las reuniones se realizarán 2 o más veces a la semana y a medida que el proyecto avance, la frecuencia podría llegar a aumentar.

#### **4.2 PROCESO DE CONTROL DE CAMBIOS**

Una vez que se recibe una CR, la CCB evalúa si es válida la solicitud. Si no es válida, se rechaza y la CR se cierra. En caso contrario se busca una solución que luego debe ser aceptada por la CCB. Si es aceptada se cierra la CR, pero si no lo es, se busca una nueva solución.

#### **4.3 HERRAMIENTA DE GESTIÓN DE CAMBIOS**

Como herramienta de gestión de cambios se utilizara GitHub-Issues.

### **5 GESTIÓN DE ENTREGAS**

#### **5.1 FORMATO DE ENTREGA DE RELEASES**

Para el formato de entrega de releases se incluirá en la entrega lo siguiente:

- Programa de instalación del software
- Archivo README.txt el cual incluye información sobre versión de release, ayuda con la instalación del software y bugs conocidos.

Con el avance del proyecto este formato podría ser modificado.

#### **5.2 FORMATO DE ENTREGA DEL INSTALADOR**

El formato de entrega del instalador será a través de un archivo comprimido .jar

.

#### **5.3 INSTRUCCIONES MÍNIMAS DE INSTALADOR**

Aún no se ha determinado cómo será el proceso de instalación. Las instrucciones serán incluidas en un documento de texto adjunto a la primera entrega.