UNIVERSIDAD NACIONAL DE CÓRDOBA

Facultad de Ciencias Exactas, Físicas y Naturales

TRABAJO PRÁCTICO Nº 1

Plan de Configuration Management

MATERIA: Ingeniería de Software

Grupo 2cores4threads:

Di Lorenzo, Franco Martin (37619687)
Del Boca, Juan Manuel (37850419)

-2016- (1er Semestre)

Historial de Releases

Fecha	Versión	Descripción
<08/04/16>	<1.0.0>	Versión Inicial, Suma y Resta.
<09/04/16>	<2.0.0>	Se añade multiplicación y división.
<09/04/16>	<1.1.0>	A la versión <1.0.0> se le añade la multiplicación.
<09/04/16>	<1.2.0>	A la versión <1.1.0> se le añade el operador porcentaje.
<10/04/16>	<2.1.0>	A la versión <2.0.0> se le añade el operador porcentaje.

Tabla de Contenidos

1. Introducción

1.1 Propósito

El propósito de este documento es definir el plan de gestión de las configuraciones para administrar la configuración de los requerimientos, software, documentos y herramientas utilizadas en este proyecto

1.2 Alcance

El alcance de este documento es especificar los parámetros de configuración requeridos para el manejo del ambiente computacional y las herramientas de software a utilizar, especificar los responsables de cada una de las actividades halladas durante el desarrollo del presente.

1.3 Definiciones, Acrónimos, y Abreviaturas

Acronym	Description
BO	Business Operation
CCB	Change Control Board
CILOC	Configuration Items Location
CM	Configuration Management
PCM	Project Configuration Management
PSO	Professional Services Organization
SI&T	System Integration and Test
SOP	Standard Operating Procedure
SCM	Software Configuration Management
Tag	a.k.a labels. Identifiers for a Configuration Item
CI	Configuration Item

1.4 Herramientas utilizadas

Herramienta	Descripción	Link
GitHub	Repositorio remoto donde se alojará el código y se realizará la gestión de defectos.	https://github.com/juanmadelboca/ingenieria-del-software/
Travis	Servidor de integración continua integrado al repositorio en GitHub	https://travis-ci.org/juanmadelboca/ingenieria-del-software/
Gradle	Herramienta de compilación para interfaz Travis/GitHub	Integrado al GitHub
Excel	Herramienta que se utilizará para gestionar los cambios en las distintas versiones.	Local

2. Organización de la Gestión de la Configuración(CM)

2.1 Organización

Las actividades realizadas dentro del proyecto de CM van a ser coordinadas por el Gerente Global de la Gestión de la Configuración(GPCM), rol que será asignado solo a una persona.
Adicionalmente, existirá también un Vice Gerente designado dentro del equipo.
El PCM Global será responsable de las actividades como el seguimiento de las rama principal y de release, determinar cuándo se crean ramas, que tipo de actividades de desarrollo se ejecuta en cual rama, etc.
También los equipos de desarrollo, como por ejemplo equipos de scrum, pueden tener su propio CM (TPCM) para llevar a cabo las actividades de CM pertenecientes al equipo y ayudar a la GPCM con el nivel más alto de proyecto.
Actividades de gestión de configuración, procesos, procedimientos y políticas deberán ser seguidas por todos los miembros del equipo. Es la responsabilidad de cada persona para seguir y aplicar el proceso de CM adecuada, de acuerdo con sus roles / funciones asignadas.

Los responsables del GPCM se muestran en la siguiente tabla:

Rol	Nombre
Global PCM Primary	Di Lorenzo Franco
Global PCM Backup	Del Boca Juan

2.2 Responsabilidades

Los roles y responsabilidades dentro del PCM se muestran a continuación:

Rol	Responsabilidad
GPCM	Posee toda la responsabilidad por todos los ítems de configuración (CI) Responsabilidad por la creación de todas las ramas y la administración de sus reglas. Responsable de la aplicación de etiquetas en la rama principal y de release Garantizar la integridad del producto y la trazabilidad de los CI para todo el proyecto. Coordinar las actividades de CM dentro del proyecto. Garantizar la correcta ejecución del esquema de CM. Asistir en las actividades de unión(merge) de la rama principal y de release Responsabilidad en los build de la rama principal y de release. Participar en todas las auditorías. Analizar todos los hallazgos relacionados a la CM.
TPCM	Asistir en la creación de etiquetas y ramas. Asistir en las actividades de merge de nuevo código a la rama principal Responsabilidad en los build de ramas específicas del equipo. Garantizar la integridad del producto y la trazabilidad de los CI bajo responsabilidad del equipo. Participar en las auditorías. Analizar todos los hallazgos relacionados a la CM.
Team(equipo)	Ayudar a resolver conflictos durante las actividades de merge. Asegurarse de que los criterios de calidad de los entregables a la rama principal se cumplan. Seguir todos los procesos asociados, políticas y prácticas definidas por sus roles asignados.

3. Change Management

3.1 Alcance

Change Management es un proceso que ocurre después de identificar y aprobar la documentación, código fuente o hardware del producto. Los cambios incluyen cambios internos en el enfoque documentado original debido a la simulación o resultados de pruebas o peticiones externas de cambios en las características o funciones.

3.2 Junta de Control de Cambio(CCB)

La CCB es un comité que asegura que cada cambio realizado es considerado adecuado por todas las partes y está autorizado antes de su aplicación. La CCB es responsable de aprobar, supervisar y controlar las solicitudes de cambio para establecer líneas de base de los elementos de configuración.

3.2.1 Integrantes

La siguiente tabla muestra los miembros del equipo que asisten a las reuniones de CCB. Los miembros opcionales se los invita si es necesario.

Rol	Miembro Obligatorio
Engineering Manager - CCB Chair	Si
Release Manager - Issue Coordinator	Si
Engineering Manager	No
Ubber Scrum Team	No
Engineering Director	No
GPCM	Si

3.2.2 Frecuencia

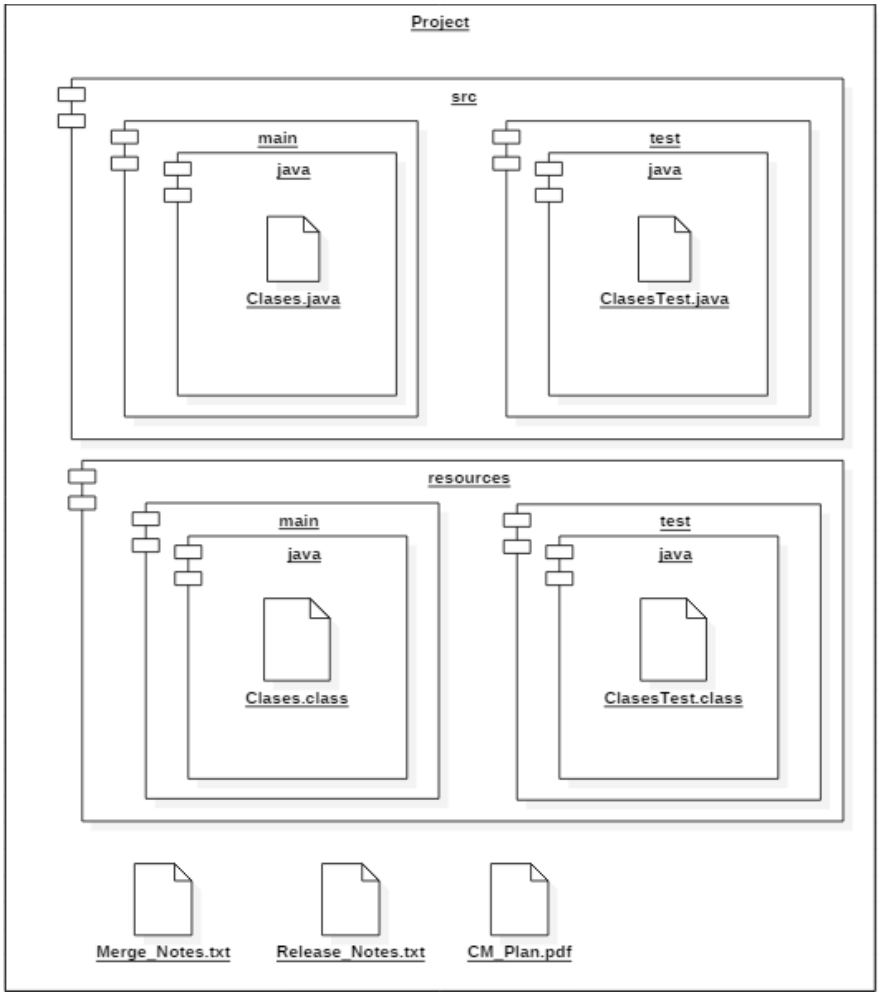
La CCB se debe juntar semanalmente o excepcionalmente la situación lo amerite.

3.2.3 Herramientas Utilizadas

Para tener un seguimiento de los cambios introducidos se utilizará como herramienta el Microsoft Excel.

4. Esquema de Directorios

El formato de directorios utilizados en el repositorio será el siguiente:



Donde en las carpetas denominadas main se guardará el código fuente principal, y en las carpetas test, los relacionados a los Unit Test.

Luego en la carpeta principal del proyecto, se almacenará el archivo de texto Merge_Notes, Release_Notes y el Plan de CM correspondiente. Adicionalmente se encontrará en el mismo lugar los archivos de gradle y travis para su correcto funcionamiento.

5. Equipos del Proyecto

El siguiente documento muestra los diferentes equipos dentro del proyecto y sus tareas relacionadas:

- Scrum Teams: están a cargo de desarrollar nuevas características y funcionalidades a través de la modificación del código fuente y su posterior merge con la rama principal. Estos mismos también se pueden dedicar a la corrección de errores o bugs.

- Release Management Team: se encargan de llevar a cabo todas las pruebas necesarias para comprobar que todos los requerimientos predeterminados para el release hayan sido cumplidos.

- Product Documentation Team: este equipo es responsable de crear y mantener la documentación del producto que se entrega al cliente. Una parte de la documentación que está integrada en los repositorios de código fuente y otra parte se entrega junto al producto. Ejemplo de archivos que gestionan: Guía de administración, Guía del usuario, etc

6. Gestión de configuración de código fuente

En esta sección se describen distintos ítems de gestion del codigo fuente. Cubre algunos aspectos sobre esquemas de ramas, etiquetas, estrategias de merge y niveles de calidad esperados para todo el producto.

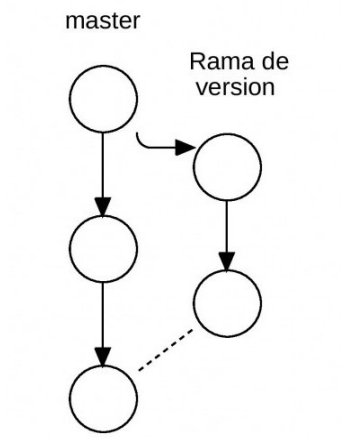
6.1 Definiciones y estrategias

6.1.1 Definición de las Ramas:

Los tipos de ramas que van a ser usadas se definen a continuación:

Master: En la rama principal se colocará todo el avance del proyecto, incluido los avances de otras ramas.

Versions: En las ramas de versiones, se colocaran solo correcciones de bugs y features pedidos por clientes que posean esas versiones. El formato de nombre de estas ramas será <numero de major version.x.x>-rel.



6.1.2 Definición de las Etiquetas

Etiquetas de Merges: Serán aplicadas cada vez que se realice un merge a la rama principal, siempre y cuando sea desde una build estable. El formato será merge-from-<Version>.
Etiquetas de Build de Release: Se aplicarán cada vez que una build de release es creada, esto conlleva que haya pasado la compilación y los unit test proveídos por travis-ci. El formato será:
- <Proyecto>-<version>--<build id>

6.1.3 Archivos Auxiliares de CM

Se utilizaran tres archivos auxiliares en este proyecto:
Merge_Notes.txt: En este archivo se registraron los merges realizados de ramas a la rama principal. Ejemplo:

> 2011-04-18 (<nombre del desarrollador>) merge-from-<etiqueta de build>.
>
> > Bug Fixes:
> > Se soluciono problema de suma con números muy grandes.
> >
> > Features:
> > Se agregó al <Proyecto> una nueva feature, la cual agrega el operador porcentaje, este devuelve el resto de una división.

Release_Notes.txt: Este archivo, contendrá todas las features del release y los tests que paso el mismo release. Ejemplo:

> Fecha: 2011/05/10
> author: <nombre-del-desarrollador>
> título: <Descripción-del-feature>
>
> Comentarios: se han corrido UnitTest para el control de las funciones de <Funciones> y han pasado exitosamente.

6.1.4 Estrategia de merge

Los merges deberán llevar una etiqueta de que indique la proveniencia del merge, y deberá ser documentada en Merge_Notes.txt, a su vez estos se realizarán solo con builds estables.

6.1.5 Estrategia de Commits

Los commits contendrán una breve descripción de los cambios agregados al proyecto, corrección de bugs o implementación de nuevos features. Ejemplo:

> “Se agrego la funcion suma y resta en clase calculadora”
> “Se implementos corrección a la función suma”

7. Build Management

Se utilizará la build de integración continua en las ramas donde se requiera chequear el código continuamente, para evitar errores introducidos durante el desarrollo. El uso de esta build en el proyecto facilitará identificar errores al poco tiempo de haber hecho un commit. La herramienta usada será Travis-ci, donde se podrá ver los Unit Tests corridos, el usuario que hizo el commit entre otros.

8. Release Management

La build de release será hecha en la misma principal (o rama de versión), luego de pasar los test de travis-ci y una revision de codigo.