Ingeniería del Software

Nombre del grupo: The Moth

Nombre del proyecto: La Mosca

Nombre del documento: Plan de Gestión de las Configuraciones

Integrantes: - Agustín Carranza
- Bruno García
- Emanuel Echazú

Fecha: 20/05/2020

Versión del documento: 1.0.1

Historial de Cambios

Version	Fecha	Resumen del Cambio	Autores
0.1.0	30/04/2020	Primera versión según el esquema de Martín M. antes de entregar.	Emanuel Echazú Bruno García Agustín Carranza
0.2.0	02/05/2020	Adaptación al esquema de Martín B. Agregado del CCB	Agustín Carranza
0.3.0	05/05/2020	Inclusión de enlaces de gestión de tareas	Bruno García Emanuel Echazú
1.0.0	06/05/2020	Inclusión de acrónimos. Modificación de redacciones. Modificación en gestión de cambios.	Agustín Carranza Bruno García
1.0.1	20/05/2020	Corrección de errores identificados por Martín M. luego de la primer entrega.	Agustín Carranza Bruno García Emanuel Echazú

Acrónimos/Glosario

Project Manager: Responsable de supervisar el proyecto general
Documentation: Responsable del repositorio de información sobre el proyecto.
Testing: Revisa los resultados finales del proyecto antes del lanzamiento
Engineering/Programming: Responsable de crear entregables
Atención al cliente: Responsable de interactuar con el consumidor
Marketing: Responsable de crear material que los consumidores verán.

INTRODUCCIÓN

Propósito y alcance del plan

Este plan tiene como propósito fundamental ayudar al equipo a la organización del desarrollo del proyecto encomendado. La Gestión de la configuración abarca la identificación, el registro y la presentación de informes de los componentes, incluidas sus versiones, componentes constituyentes y relaciones. Los servicios externos, bases de datos o APIs están fuera del alcance del presente plan.

Herramientas para la administración de configuraciones

- ***Control de versiones***

La herramienta de control de versiones elegida es Git. Se va a utilizar un repositorio remoto alojado en Github donde todos los miembros del equipo tienen acceso en calidad de colaboradores. La URL es https://github.com/agucarranza/juego-maven

- ***Gestión de tareas***

Trello para organización de tareas:
https://trello.com/b/SnpKxWpx
Slack para comunicación:
https://themothespacio.slack.com

- ***Gestión de defectos***

Utilizamos la herramienta Issues proporcionada por GitHub para hacer seguimiento de los errores y defectos. Los estados pueden ser abierto o cerrado. Esta utilidad brinda etiquetas para organizar cada *issue*. Por defecto son las siguientes:

Etiqueta	Significado
bug	Algo que no funciona.
documentation	Mejoras o adiciones a la documentación.

duplicate	El defecto o el pull-request ya existe.
enhancement	Nueva función o requerimiento.
good first issue	Bueno para los recién llegados.
help wanted	Se requiere más atención.
invalid	No parece estar correcto.
question	Se requiere más información.
wontfix	No se va a reparar.

La URL es https://github.com/agucarranza/juego-maven/issues

- ***Integración continua***

La herramienta de integración continua elegida en un principio fue Jenkins por un interés particular del equipo en conocer su funcionamiento. Luego de su estudio, se concluyó que no es conveniente en este caso porque necesita ser alojada en un servidor propio del cual no se dispone. Finalmente, se decidió utilizar Travis CI, cuya interfaz es muy amigable, y fundamentalmente, se integra con Github en la nube. La URL es https://travis-ci.com/github/agucarranza/juego-maven

- ***Construcción automática***

Para este proyecto utilizamos la IDE Intellij IDEA de Jetbrains, que provee una versión full para estudiantes. Como herramienta de construcción automática y ejecución de pruebas, utilizamos Maven por su excelente integración con Travis CI.

Roles y Responsabilidades

Miembro	Rol
Emanuel Echazú	**CM Specialist**
Bruno García	**Configuration Manager**
Agustín Carranza	**CM System Analyst**

CM System Analyst: Un integrante experimentado que realiza la entrega diaria de los servicios de CM a la organización, con instrucciones mínimas que implica centrar el trabajo en tareas priorizadas.

CM Specialist: Un integrante no tan experimentado que tiene instrucciones generales tomando las órdenes generales del CM y las órdenes específicas del CM Analyst para entregar productos de trabajo admitidas por las operaciones de gestión de la configuración en curso.
Configuration Manager: Un gerente con la capacidad y la autoridad para garantizar la entrega diaria de extremo a extremo de los servicios de CM de acuerdo con el plan de gestión de la configuración.

ESQUEMA DE DIRECTORIOS

Estructura de directorios y propósito de cada uno

```
.
├── docs
├── lib
├── src
│   ├── main
│   │   ├── java
│   │   │   ├── com
│   │   │   └── front
│   │   └── resources
│   │       └── META-INF
│   └── test
│       └── java
```

Ruta	Propósito
./docs	Documentación útil y genérica
./docs/life-cycle	Documentación referida al ciclo de vida del producto (gestión de la configuración, SRS, matriz de trazabilidad, plan de testing, etc).
./lib	Bibliotecas externas
./src/main/java/com	Código fuente del backend y otros.
./src/main/java/front	Código fuente del frontend
./src/test/java	Código de las pruebas de software
./src/main/resources/META-INF	Archivos de configuración de empaquetados.
. (root)	Archivos de configuración de las herramientas (Jenkinsfile, pom.xml, .travis.yml) y

	README.md

Normas de etiquetado y nombramiento de los archivos.

A la hora de elegir el nombre para archivos y documentos, se seguirán las siguientes pautas a los fines de estandarizar el nombramiento:

- No usar espacios en nombres de archivos ej: ~~Mi Archivo de trabajo.txt~~
- Usar - (guión medio) para separar espacios ej: web-page.htm
- Usar nombres cortos que describen el archivo ej: inventarios-enero.xls
- Usar nombres en bajas, es válido usar notación camel ej: webPage.htm
- Todos los archivos deben llevar la extensión del programa que lo genera ej: concepto.psd
- Usar sólo caracteres alfanuméricos ej: ~~Prod@A#500.xls~~ por prod-A500.xls.
- Eliminar caracteres especiales ej: ~~archivo#500.xls~~ por archivoNo500.xls.
- Usar siempre la misma estructura de archivos ej: inventarios-enero-2008.xls luego inventario-febrero-2008.xls

Para nombramiento de etiquetas se seguirá una notación numérica compuesta por tres números (y un cuarto opcional) separados por puntos con la siguiente notación:

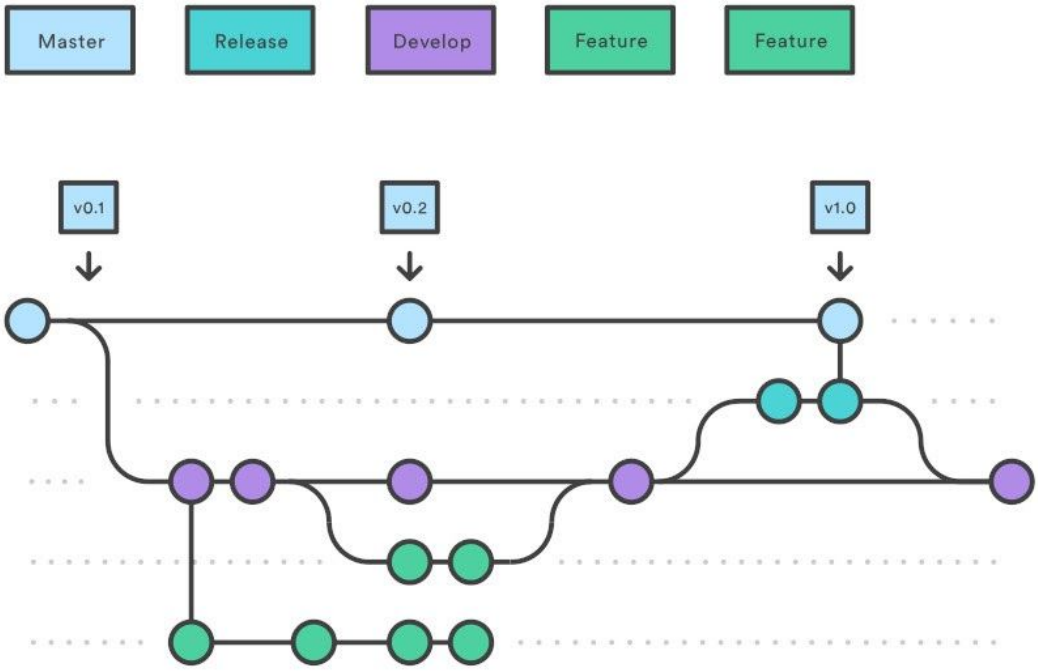
major.minor.revision[.entrega]

Cada uno de estos números tienen el siguiente significado:

- **major:** indica la versión principal del software, consistiendo en un conjunto de funcionalidades concretas que son recogidas y cubiertas en dicha versión.
- **minor:** indican funcionalidad menor cubierta en la versión de software entregada.
- **revisión**: se modifican cuando hay revisiones de código ante fallos de la aplicación.
- **entrega:** este dígito tiene el objetivo de llevar la cuenta del número de veces que una entrega se rechaza, por incumplimiento de algún requisitos de la gestión de entregas o del proyecto.

GESTIÓN DE LA CONFIGURACIÓN DEL CÓDIGO

Esquema de ramas



El equipo va a seguir el esquema de GitFlow. Es un flujo de trabajo basado en Git que brinda un mayor control y organización en el proceso de integración continua. Este esquema aumenta la velocidad de entrega de código terminado al equipo de pruebas, disminuye los errores humanos en la mezcla de las ramas. y elimina la dependencia de funcionalidades al momento de entregar código para ser puesto en producción.

Deben existir dos ramas principales para que el flujo de trabajo funcione correctamente: master y develop.

Política de etiquetado de las ramas

GitFlow creará por defecto los siguiente prefijos para las ramas auxiliares, los cuales ayudan a identificar y tener control en el repositorio: feature, release, hotfix, bugfix, support. Nuestro equipo planea utilizar sólo las dos primeras. También agregamos el prefijo «v» a las etiquetas.

Política de fusión de archivos

Todos los miembros del equipo pueden hacer merge en las ramas que no sean *master* sin limitación. Para esta última, está creada una regla de protección que corre con Travis CI las pruebas automatizadas antes de mergear.

Para fusionar nuevas ramas que pudieran surgir con el código principal, el administrador realizará la fusión haciendo un rebase del código junto a los desarrolladores. De esta forma rápidamente se podrán identificar errores, los desarrolladores podrán trabajar en resolverlos y el administrador estará al tanto de la situación para coordinar otro eventual cambio en el código.

Los criterios para modificar (incrementar) cada uno de los contadores de la etiqueta de versión son los siguientes:

- **major:** nuevas funcionalidades claves de la aplicación respecto a la versión anterior debido a la inclusión de nuevos requerimientos para el sistema, como la inclusión de nuevos módulos o una revisión completa de los existentes.
- **minor**: cambios significativos en la forma en la que se ofrece la funcionalidad existente, corrección de grandes fallos del sistema o nuevas versiones evolutivas que modifican significativamente la funcionalidad ofrecida.
- **revision:** se modifica por cada entrega de software que se realice.
- **entrega:** al rechazarse una entrega se incrementa este contador en la siguiente. Cuando la entrega se aceptase se crearía un tag público que sólo conservaría los tres primeros dígitos (major, minor, revision).

GESTIÓN DE CAMBIOS

Change Control Board (CCB)

Introducción y objetivos

La Junta de Control de Cambios (CCB) es un grupo de personas dentro de la empresa que revisa y prioriza las solicitudes de cambio relacionadas con un proyecto. Este grupo puede ser tan pequeño como el gerente del proyecto y la persona que ha patrocinado el proyecto, o tan grande como un comité de representantes que representan diversas funciones dentro de la

organización. Algunos miembros pueden verse directamente afectados por el proyecto, pero no es un requisito ser miembro del comité.

Roles

Miembro	Rol
Bruno García	**Documentation / Testing**
Emanuel Echazú	**Project Management / Programming**
Agustín Carranza	**Engineering / Testing**

Frecuencia de reunión de trabajo

Considerando las restricciones vigentes a nivel nacional, las reuniones presenciales están absolutamente descartadas. Se programó una reunión semanal con el equipo mediante la utilidad Meet de Google, los días viernes a las 18 hs. La URL es https://meet.google.com/aur-tgew-kik

Proceso de control de cambios

1. Análisis de implementación: El miembro encargado recibirá la solicitudes pedidas por el usuario. Las priorizará de acuerdo a una escala de urgencia (baja, media, crítica). Realizará una reunión con el equipo de trabajo para discutir si la implementación solicitada es coherente y viable. En caso afirmativo se avanza al siguiente punto; en caso negativo se descarta el pedido justificando la elección realizada junto con el equipo de trabajo.
2. Análisis de costo e impacto: Una vez aprobada la viabilidad, se procede a analizar tanto las consecuencias de no realizar el cambio, como los beneficios, costos, usuarios afectados e impacto en los planes de implementarlo. Para esto el CCB tendrá en cuenta la naturaleza del mismo. Por ejemplo, si el cambio se debe a una falla crítica el costo se considera alto. Distinto si fuera una nueva funcionalidad deseada.
3. Planificación: Se comenzará viendo qué parte específica del proyecto se realiza el cambio, la designación del responsable de las modificaciones, el cálculo de las horas hombre que llevará el trabajo de acuerdo a la disponibilidad considerando días no laborables entre semana, las gráficas estimativas de los periodos de trabajo para comenzar con el análisis profundo, desarrollo y testing de la implementación.
4. Implementación del cambio: Una vez asignado el desarrollador y realizado el análisis previo de las modificaciones a realizar, se comienza con la implementación en una nueva rama del proyecto a fin de no entorpecer el trabajo del resto del equipo y generar una vía segura en caso de que el desarrollo pueda impactar de forma negativa de alguna forma pasada por alto.

5. Verificación: El CCB se reunirá con una representación del área de testing donde se establecerá si la modificación satisface la solicitud de cambio. El área testing puede concluir que esto no ha ocurrido o que se han encontrado errores, ante lo cual se vuelve al área de desarrollo.
6. Aceptación del cambio: Si todos los requisitos del proceso se consideran cumplidos, el CCB da por cerrada la solicitud para ser documentada y archivada.

Herramienta de gestión de cambios

La sección "issues" del repositorio Github. La URL es https://github.com/agucarranza/juego-maven/issues

GESTIÓN DE ENTREGAS

Formato de entrega de releases

Cuando el equipo decida publicar una de las entregas, serán publicadas en la sección *releases* del repositorio de Github. Estas consistirán en un empaquetado tipo .zip o .tar donde estará contenido un archivo .jar y un script de bash para ejecutarlo en Linux, junto con un script equivalente .bat para Windows. Será un software portable. La URL es https://github.com/agucarranza/juego-maven/releases

Formato de entrega del instalador

No se prevé la entrega de un instalador considerando el tamaño de proyecto.

Instrucciones mínimas de instalación

Se entregará un archivo README.txt dentro del empaquetado con las instrucciones para ejecutar.