iSoft Empresa

PROYECTO DE PÁGINA WEB DE VENTA DE INDUMENTARIA DEPORTIVA

Plan de gestión de las configuraciones

Autores: Cruzado, Patricia
Rebola, Cristian
Robles, Karen Yessica
Vazquez, Franco

Versión del documento: 1.0.0

Historial de revisiónes y modificaciones

Version	Fecha	Cambio	Autor
1.0.0	8-May-2017	Version base	iSoft

Indice de contenidos

INTRODUCCION

Este documento cubre el plan de Configuation Managment que se utilizara para el proyecto PAGINA ONLINE PARA VENTA DE INDUMENTARIA DEPORTIVA. Abarca el seguimiento del control de versiones, integración continua, gestión de defectos, esquema de directorios, normas de etiquetado de archivos, esquema de ramas a usar, configuración de la CCB, entre otras cuestiones.

Objetivos del plan:

- Garantizar que no se realicen cambios de forma inapropiada.
- Garantizar integridad del producto a lo largo de su ciclo de vida.
- Evitar problemas por falta de estandarización de las distintas actividades.

GLOSARIO

- CCB: Change Control Board
- CM: Configuration Managment
- SCM: Software Configuration Manager

CONTROL DE VERSIONES

Como su nombre lo indica, le concierne el control de versiones de un componente, de un producto o programa. Se trata con conjuntos de items elementales que están bajo el control de configuracion, con el fin de producir un componente o producto coherente. En consecuencia esta actividad está basada en una generación controlada de un número de versiones para items del proyecto.

En nuestro caso utilizaremos Git y GitHub como la herramientas para el control de versiones, que no es necesario confiar todo en un repositorio central lo cual nos permite que se pueda trabajar sobre copias de trabajo o hacer nuevos modelos de trabajo a partir de dicho repositorio como si fuera el proyecto general.

Se accedera al repositorio de trabajo creando una cuenta de usuario en GitHub y luego se obtendran los permisos de edicion correspondientes para poder editar el proyecto a traves de la siguiente direccion URL:

https://github.com/francovqz/TP1-iSoft

INTEGRACION CONTINUA

La integracion continua constituye una parte importante al Sofware Configuration Management ya que se asegura que cada cambio realizado sea integrado al proyecto principal lo antes posible lo cual proporciona una informacion viable sobre la salud del proyecto.

La idea basica es integrar nuevas partes de codigo tan rapido como sea posible y testear las modificaciones en el contexto del proyecto en su totalidad. Para alcanzar este objetivo un servidor de integración continua recupera la fuente del código del proyecto despues de cada cambio producido, lo compila y corre los tests correspondientes para verificar la funcionalidad.

Se utilizara Travis CI ya que es bastante convenido con GitHub. Tiene varias ventajas, entre ellas; no es necesario tener proporcionado un server de IC dedicado ademas de que el servicio es gratuito para proyectos de open source.

Para usar Travis , el usuario debe entrar en su cuenta de GitHub, luego la pagina de perfil de Travis permite al usuario activar IC para cualquiera de sus proyectos. Despues de la activacion, un build sera ejecutado cada vez que se produzcan cambios (push) al repositorio.

Para decirle a Travis como construir el proyecto, un archivo config llamado .travis.yml debe ser agregado al directorio del repositorio. La configuración contiene el lenguaje del programa que es usado en el proyecto asi como tambien todas las versiones con las que el sofware sera testeado.

La direccion del servidor de Integracion continua de Travis es la siguiente:

https://travis-ci.org/francovqz/TP1-iSoft

Junto con Travis se utilizará la herramienta Gradle para la realización de los builds y su comprobación. El link de descarga de Gradle es el siguiente:

http://gradle.org/gradle-download/

Se deberá incluir en el repositorio los archivos necesarios para que gradle realice los builds. Debido a que en el proyecto no utilizamos Test Unitarios por lo cual gradle será configurado unicamente para realizar los builds de java.

GESTION DE DEFECTOS

Para nuestro proyecto utilizaremos la herramienta Issues proporcionada por GitHub para el reporte de defectos y su seguimiento. Para acceder a la misma es necesario entrar al apartado de Issues de GitHub con las cuentas de en las cuales tienen acceso como colaboradores al proyecto principal.

Los permisos de acceso a esta herramienta están incluidos al pertenecer al grupo de colaboradores al repositorio de GitHub.

Es importante revisar el apartado en el cual se publican todos los defectos para no tener problemas con los mismos.

https://github.com/francovqz/TP1-iSoft/issues

MODO DE REPORTE DE DEFECTOS

En caso de observar algún defecto en el proyecto es importante dirigirse inmediatamente a la herramienta Issues y crear un nuevo reporte explicando detalladamente el defecto que se encontró, el lugar en el cual se encuentra el defecto y una posible solución del mismo. Es importante garantizar que realmente el defecto existe antes de realizar un reporte.

El Issue Coordinator determinará quién se hará cargo de resolver el defecto y realizará un seguimiento del mismo hasta garantizar que el defecto está solucionado. Una vez solucionado se cerrará el Issue en la herramienta dejando activos únicamente los defectos que todavìa no fueron resueltos.

La prioridad de resoluciòn del defecto va a ser determinada por el Issue Coordinator, teniendo en cuenta los problemas que puedan causar el postergar la resoluciòn del defecto.

ESQUEMA DE DIRECTORIOS

El esquema de directorios que se va a usar va a ser el típico de un proyecto java con Maven. Esto seria: un directorio src/main/java en la cual se almacena el código, otro src/main/resourses para los recursos que se usan en el código java y otro src/main/test para las pruebas a realizar . (véase la imagen).

NORMAS DE ETIQUETADO Y NOMBRAMIENTO DE ARCHIVOS

Clases: Solo se permitirán clases para implementar las operaciones de movimiento que realizara la pagina. Estas deberán llamarse con la notación CamelCase. Como siempre en la clase Main se iniciara la interacción entre los objetos de las demás clases, dando inicio al programa.

Recursos: Se requerirán archivos de texto, interfaces graficas para interaccion e imágenes. Los archivos de texto debera ser en formato .txt y deben llamarse de forma que con solo leer su denominación ya se pueda dar una idea clara de qué rol tienen dentro del programa. Las interfaces se implementaran adecuadamente para permitir la interaccion entre el usuario y la aplicación, las interfaces se proveerán de imágenes para hacer mas calido el nexo entre el usuario y la pagina de venta.

PLAN DEL ESQUEMA DE RAMAS A USAR

En el proyecto utilizaremos únicamente 3 ramas: el branch master y dos ramas auxiliares para poder realizar los releases 1.1.0, 1.2.0 y 1.3.0 mientras que en el master se trabajara para realizar los releases 1.0.0, 2.0.0 y 2.1.0.

El nombre de la rama auxiliar tiene que tener la primera letra en mayúscula y seguido por minúscula, en cambio el nombre de la rama master tendrá el nombre asignado por defecto. Al terminar el release 1.3.0 hay que realizar un merge con la versión 2.0.0 para volver a la rama master y asi poder terminar el release 2.1.0.

POLITICA DE FUSION DE ARCHIVOS Y ETIQUETADO

La fusión de archivos se realizará a base de los tipos de cambios que se iran realizando en los CI correspondientes.

Luego de las consultas necesarias con los miembros de trabajo y a partir de las necesidades requeridas por los clientes o por la exigencia de reparación de algun inconveniente o actualización del proyecto, se trabajarán en distintas funcionalidades que se desarrollaran en ramas o branchs del thread master inicial para luego acoplarse a éste segun ya esten completos los cambios y las necesidades que se requerian en un principio y que llevaron a cabo su desarrollo.

Los etiquetados, tags o label se realizaran cada vez que el proyecto se encuentre en un estado que cumpla con las exigencias o requerimientos por las que fueron ejecutadas su desarrollo en base a la necesidad originaria, describiendo adecuadamente las caracteristicas de las nuevas versiones producidas, como asi tambien los datos de miembros de trabajo involucrados, respetando un orden cronologico en los nombres correspondientes y teniendo en cuenta de respetar siempre la integridad del proyecto en su conjunto y totalidad.

GESTION DE LANZAMIENTO DE RELEASES(cambios o agregados al proyecto)

Este proceso es el responsable de planear, distribuir, y controlar la construccion, el testeo y los lanzamientos de "releases" asi como tambien del envio de nuevas funcionalidades al proyecto que sean requeridas por los clientes que usaran el producto siempre teniendo en cuenta de proteger y mantener la integridad de los servicios y features existentes.

Para asegurar que sean elegidos los CI adecuados para ser "release" y evitar fallas o errores durante el proceso de implementacion, la gestion de release debe verificar la correcta funcionalidad y su uso antes de la entrega mediante pruebas y tests.

El plan para la realizacion de los release debe definir:

- Alcance y contenido del release
- Evaluacion de riesgo y perfil de riesgo para el release
- Usuarios/clientes afectados.
- Miembros de la CAB (Change Advisory Board) que aprobaron la recomendacion para el release
- Equipo responsable del release
- Estrategia de lanzamiento
- Recursos disponibles

A su vez las actividades del proceso de entrega y desarrollo incluyen:

1. Construir y probar antes de realizar el despliegue de produccion
2. Crear pilotos de prueba
3. Logistica y plan de envio
4. Como y donde seran enviadas las unidades de release
5. Como son los plazos de entrega
6. Como llevar un rastreo del envio y obtener informacion de confirmacion
7. Metricas para el seguimiento y la determinacion del exito de los esfuerzos de implementacion del release.
8. Desarrollo y testeo de los releases
9. Documentacion del release y su construccion
10. Lanzamiento del packaging del release

CHANGE CONTROL BOARD

La CCB es el grupo encargado de verificar y autorizar cada cambio que se quiera realizar sobre el plan, el producto, el código o la documentación del mismo. Ningún cambio que tenga consecuencias o que produzca conflictos con el cliente o con algún miembro del personal que esté trabajando en el proyecto puede ser realizado sin la autorización de la CCB.

Las decisiones deben ser tomadas teniendo en cuenta las posibles consecuencias políticas o económicas que los cambios puedan ocasionar tanto con el cliente como con los miembros del proyecto.

La CCB evaluará cada pedido de cambio que se realice a cualquier miembro de la CCB, este pedido tendrá que ser realizado por escrito detallando los cambios que se quieren implementar.

Miembros de la CCB

Engineering Manager – CCB Chair: Su presencia es indispensable para realizar un control de cambios. Es el que dirige la CCB y es el que se encarga de coordinar las distintas áreas involucradas en el proyecto por lo tanto conoce las incidencias que un cambio puede causar en todas las áreas. En caso de no poder asistir tiene que autorizar a otro miembro de la CCB a ocupar su lugar.

Release Manager – Issue Coordinator: Su presencia es indispensable para realizar un control de cambios. Es el encargado de coordinar la forma en la cual se hacen y se entregan los realeases. Además se encargar de coordinar y solucionar todos los problemas que puedan surgir una vez realizados los cambios.

Uber Scrum Team: Su presencia es indispensable para realizar un control de cambios. Es el encargado de coordinar a todos los Scrum Masters de la empresa, y por lo tanto tiene un conocimiento de cómo afectarán los cambio con otros proyectos.

Director Grafico**:** Su presencia es opcional en caso que se lo requiera. En caso de necesitar realizar cambios que involucren la parte gráfica del proyecto es necesaria su participación.

Configuration Manager: Su presencia es opcional en el caso que se lo requiera. En el caso que los cambios a realizar involucren alguna parte del CM Plan es necesario su participación ya que es la persona que lo diseño.

Periodicidad de la reuniones de la CCB

Las reuniones de la CCB se van a realizar 2 veces por semana durante el desarrollo del proyecto, con la posibilidad de llamar a reuniones especiales en caso de ser necesario avisando por email o por teléfono con 24hs de antelación.