

Sistema de Administración de Farmacias

Manejo del Ambiente Controlado

Versión 1.3

Historia de revisiones

Fecha	Versión	Descripción	Autor
20/08/2014	1.2	Versión preliminar	Santiago Nieves
31/08/2014	1.3	Revisión y Formato	Alfonso Methol

ÍNDICE

1. Definición del Ambiente Controlado.	3
2. Descripción del uso del Ambiente Controlado.....	4
2.1 Requisitos para que hacer uso del ambiente controlado.....	4
2.2 Cuentas en bitbucket.org	4
2.3 Instalación de Git:	4
2.4 Ramas para el código fuente del proyecto.	5
2.5 Flujo normal de utilización del repositorio.	5
3. Responsabilidades.....	6

1. Definición del Ambiente Controlado.

Se utilizará un ambiente controlado para los artefactos de configuración (documentos y códigos fuentes) mediante un servicio de repositorio remoto Git que ofrece "bitbucket.org". De forma paralela se utilizará google drive para la elaboración de documentos.

Una vez los documentos alcancen una versión definida, los mismos son subidos a la línea base del proyecto en el repositorio remoto.

Url del repositorio:

<https://bitbucket.org/snieves/pis-2014.git>

El repositorio remoto se organiza de la siguiente manera:

- Los documentos se almacenan en carpetas correspondientes a cada fase
 - 1-** Fase Inicial.
 - 2-** Fase Elaboración.
 - 3-** Fase Construcción.
 - 4-** Fase Transición.

Nota: Los documentos deberán ser trasladados de google drive al repositorio, una vez que estos estén en una versión estable, dado que pasarán a formar parte de la línea base del proyecto.

- Los prototipos se organizan de forma indiscriminada en la carpeta Prototipos en la raíz del repositorio.
- Todo lo referente a código, se organiza en la carpeta Código de la raíz del repositorio.

Todavía no se definió el tema de respaldos, pero en posteriores iteraciones se definirá una manera automatizada para respaldar el repositorio.

2. Descripción del uso del Ambiente Controlado.

2.1 Requisitos para que hacer uso del ambiente controlado.

- Tener el software Git instalado.
- Conexión a internet (solo para sincronización de repositorios local y remoto, no necesario para realizar commit localmente)
- Un navegador web, para acceder a bitbucket.org/snieves/pis-2014.git y llevar un mejor control del repositorio.

2.2 Cuentas en bitbucket.org

Se proporcionará cuentas por área de trabajo, las mismas son:

User: implementador1
User: implementador2
User: testing-pis
User: user-pub
User: snieves (Administrador del Repositorio)

Las contraseñas de acceso serán brindadas a cada responsable de área, para que sean distribuidas.

2.3 Instalación de Git:

- **Debian, Ubuntu y derivados:**
 - 1- Iniciar una sesión en la Terminal.
 - 2- Digitar en la terminal, **sudo apt-get install git.**
- **Windows:**

Descargar los siguientes programas:

 - 1- <http://msysgit.github.io>
 - 2- <http://code.google.com/p/tortoisegit>

Los siguientes dos pasos son configuraciones realizadas en SO debían, Ubuntu y derivados, los manuales de uso en S.O. Windows, pueden encontrarse en la carpeta del repositorio con nombre "Tutoriales" o en google drive.

Luego de tener el Git instalado se deberán de seguir los siguientes pasos:

1- Clonar el repositorio en una carpeta local.

Se debe seleccionar una carpeta local de trabajo, una vez parados sobre ella con una terminal, se digitara el siguiente comando:

```
git clone https://nombre-de-usuario@bitbucket.org/snieves/pis-2014.git
```

En **nombre-de-usuario**, deberán poner el nombre de usuario que les fue asignado en bitbucket.org.

Con esto, se descargara el repositorio.

2- Configuración de identidad:

Se deberá abrir una terminal y ubicarse sobre la carpeta donde clonamos el repositorio, luego se deben ingresar los siguientes comandos:

```
git config --global user.name "Nombre Apellido"
```

```
git config --global user.email "correo@dominio.com"
```

Estos datos, son independientes de los brindados para acceder a bitbucket, por lo cual tendrán que poner sus nombres y correos originales.

2.4 Ramas para el código fuente del proyecto.

Git tiene como rama principal una llamada 'master', la que se utilizará como rama principal de desarrollo, esto quiere decir que la última versión del proyecto siempre estará en la rama master.

Cuando se defina la primera línea base del proyecto, el código estable será mantenido en una rama llamada 'estable'. De esta forma nuevas integraciones a la línea base del sistema se harán desde la rama master a la estable.

Es recomendable, la utilización de ramas distintas a la master para el desarrollo de nuevas funcionalidades, y una vez estas estén parcialmente o totalmente terminadas, realizar un merge de la rama con la rama master.

2.5 Flujo normal de utilización del repositorio.

Como antes se mencionó, para poder empezar a trabajar con el repositorio, el implementador deberá tener clonado el mismo en su ambiente de trabajo local (ver punto Instalación de Git).

En el momento de crear una nueva funcionalidad es de buena práctica crear una rama (distinta de "master") en su repositorio local y en

caso de ser necesario, realizar "push" de esa nueva rama al repositorio remoto para que otros implementadores trabajen sobre ella.

Esto por lo general es motivado por el hecho de que varios implementadores estarán escribiendo código al mismo tiempo y en lugares físicos distintos, lo que dificulta la comunicación.

Bajo este escenario, si ambos implementadores están desarrollando sobre la rama master, realizando "push" y "pull" cada cierto tiempo, y el código de uno de ellos deja inestable a la aplicación entera, entonces el crecimiento del programa se detiene por completo hasta solucionar dicho problema. Si cada implementador trabaja en una rama para cada nueva funcionalidad, entonces no existirá este problema porque cada uno trabajará en un ambiente separado, si el código de un implementador tranca a la aplicación entera, sólo lo estará haciendo en su rama y no afectará al resto.

Por otra parte, no hay que llevar la creación de ramas al extremo, por ejemplo cuando simplemente se debe realizar un "fix" o una modificación directa que no involucre demasiados cambios o esté presente el riesgo de dejar la versión inestable se podrá trabajar directamente sobre la rama master.

3. Responsabilidades

El responsable de la integridad del repositorio es el SCMR, mientras que es responsabilidad de los responsables de integración e implementadores el correcto merge entre las ramas creadas y la rama master.