Plan de Gestión de Configuración

Versión 1.5

**Integrantes:**

* Angeles Rojas, Jorge Alexander.
* Arizola Yánac, Carlos.
* Arotuma Martinez, Victor Miguel.
* Arteaga Quico, Alisson Diane.
* Barreto Trujillo, Larry Steve.
* Castillo Chávez, Luigi Jair.

**CONSULTORA:** Programsy



**Historial de Revisiones**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Descripción** | **Autor** |
| 14/09/2018 | 1.0 | Creación del documento | Alisson Arteaga |
| 14/09/2018 | 1.1 | Agregado la introducción | Equipo Programsy |
| 14/09/2018 | 1.2 | Agregado herramientas, infraestructura y entorno | Castillo |
| 14/09/2018 | 1.3 | Agregado el calendario | Arteaga |
| 14/09/2018 | 1.4 | Agregado políticas y directrices | Arizola |
| 14/09/2018 | 1.5 | Agregado roles y responsabilidades | Barreto |
| 22/09/2018 | 1.6 | Arreglado detalles del documento | Angeles |

Tabla de contenido

[Tabla de contenido 3](#_Toc525771964)

[1. Planificación de la SCM: 4](#_Toc525771965)

[1.1. Introducción 4](#_Toc525771966)

[1.2. Roles, Responsabilidades y Cantidad 5](#_Toc525771967)

[1.3. Políticas, Directrices y Procedimientos 7](#_Toc525771968)

[1.3.1. POLÍTICAS Y DIRECTRICES 8](#_Toc525771969)

[1.3.2. PROCEDIMIENTOS 8](#_Toc525771970)

[1.4. Herramientas, entorno e infraestructura 9](#_Toc525771971)

[1.4.1. HERRAMIENTAS 9](#_Toc525771972)

[1.4.2. ENTORNO 11](#_Toc525771973)

[1.4.3. INFRAESTRUCTURA 12](#_Toc525771974)

[1.5. Calendario 13](#_Toc525771975)

[2. Identificación 14](#_Toc525771976)

1. Planificación de la SCM:
   1. Introducción

Actualmente nuestra consultora cuenta con 5 proyectos de los cuales 4 en producción y uno de ellos en desarrollo que es SGVIF. Nuestra consultora cuenta con diversos productos de software los cuales durante su desarrollo surgieron inconvenientes en el manejo de las solicitudes de cambios que han generado pérdida de tiempo por parte del equipo al querer realizar cambios de versiones y dificultad el querer restaurar versiones funcionales, nuestro equipo identifica esto como una problemática originado por no contar con una guía o algún documento que permita llevar a cabo una correcta gestión de versionamiento. Debido a ello estamos realizando la siguiente propuesta para contar con un Plan de Gestión de la Configuración.

Este plan se espera que sea aplicado a todos los proyectos de nuestra consultora, sean pequeños, medianos o grandes. La gestión de la configuración del software es un proceso que incluye aplicar una serie de actividades a lo largo del ciclo de vida del software y describe un conjunto de actividades que deben ser llevados a cabo durante su desarrollo. Este conjunto de actividades que incluyen la identificación, control, estado, auditorias y revisiones de la configuración hacen que la Gestión de la Configuración sea calificada como una garantía de calidad.

La finalidad de este plan es garantizar que no se realicen cambios incontrolados y que todos los participantes en el desarrollo del software dispongan de una versión adecuada de los productos que manejan durante el ciclo de vida de desarrollo del software y de este modo facilitar el mantenimiento. Este plan pretende llevar un control, obtener informe de estado de desarrollo en el que se encuentra un software que permita aumentar la calidad del producto, logrando de esta forma una mayor satisfacción del cliente y de este modo también la mejora de nuestra consultora.

* 1. Roles, Responsabilidades y Cantidad

La unidad de SCM de Programsy es liderada por el responsable de SCM, responsable gestionar las actividades y tareas de la unidad. En general, la unidad de SCM trabaja en equipo y la colaboración de todos los integrantes es fundamental para el correcto funcionamiento de los proyectos. A continuación, se presenta la tabla con los roles y sus responsabilidades correspondientes.

|  |  |  |  |
| --- | --- | --- | --- |
| **Roles** | **Responsabilidades** | **Cantidad** | **Niveles de autoridad** |
| **Gestor de la gestión de la configuración** | Definir una estrategia para la implantación de las prácticas de SCM  Gestionar la unidad de SCM  Planificar las actividades de SCM  Proveer el personal de SCM requerido para las actividades de desarrollo  Garantizar la capacitación del personal de SCM y los recursos necesarios para el desarrollo de sus actividades  Informar a los niveles superiores sobre el estado del proceso y las actividades de SCM en los proyectos  Promover el mejoramiento continuo del proceso de SCM  Gestionar el desarrollo de herramientas para facilitar el proceso de SCM | 1 | Es el principal responsable del proceso de gestión de configuración durante todo el ciclo de vida del software |
| **Comité de control de cambios** | Recomienda la aprobación o desaprobación del cambio  Revisar el hardware o solicitudes de cambio de clientes, en términos del cronograma del proyecto, el costo y el impacto en los clientes  Priorización de la incorporación de cambios aprobados  Informa por escrito al cliente de la decisión de cambio  Reenvía la decisión de aprobación o desaprobación del cambio al CMO | 1 | Encargado de evaluar todas las modificaciones. |
| **Equipo de desarrollo** | Aborda diversos aspectos del desarrollo, incluidos documentos, código, entrenamiento, COTS y hardware. Identifica las actualizaciones y modificaciones de CIs y CI, envía actualizaciones de línea base. Participa en el proceso de control de configuraciones | 3 | Depende de cada miembro del equipo. Esta especificado por cada Ítem de la Configuración. |
| **Bibliotecarios** | Diseñar y establecer la biblioteca del software para cada proyecto de desarrollo durante la etapa de planificación  Establecer y mantener el software y la documentación de cada proyecto de acuerdo con un proceso documentado  Proveer a los desarrolladores las copias de las líneas bases requeridas para sus diferentes tareas  Entregar la copia original para implementación de los cambios aprobados por el CCB  Mantener y distribuir un índice con el contenido de cada biblioteca  Informar a los desarrolladores sobre los cambios a los items  Apoyar la elaboración de informe sobre el estado de la configuración  Registrar y mantener copias de las antiguas versiones | 1 | Controlar el ingreso y el acceso a las baselines, garantizando el uso de los procedimientos formales definidos en el plan de SCM. |

* 1. Políticas, Directrices y Procedimientos

Con respecto a los permisos en el repositorio (en nuestro caso en GitHub), todos los integrantes tienen derechos de administrador: permisos de escritura, de lectura y de creación de documentos.

La política es la total confianza en cada uno de los integrantes del equipo.

### POLÍTICAS Y DIRECTRICES

La política en relación a la gestión de las configuraciones, se enumera en las siguientes sentencias:

* Política de seguridad (PS)
* Política de gestión de activos de información (PGAI)
* Políticas generales de la empresa (PGE)
* Política de gestión de seguridad de la información (PGSI)
* Política de privacidad (PP)
* Requisitos de gestión de configuración (RGC)
* Registro de productos (RP)

### PROCEDIMIENTOS

Los procedimientos para modificar los archivos que pertenecen al repositorio son los siguientes:

* Debe indicarse a todos los miembros que documento está a punto de ser editado, con el objetivo de evitar que un ítem se sobrescriba, o se haga el trabajo dos veces.
* Se modifica el ítem en la rama asignada al integrante
* Se suben los cambios a su rama correspondiente.
* Se hacen los merge necesarios para que la modificación del ítem se encuentre disponible para todos los integrantes.
* Informar que el cambio fue realizado.
* Si el cambio es validado, se hace un pull request a la rama principal.
  1. Herramientas, entorno e infraestructura

### HERRAMIENTAS

#### GIT

Es un sistema de control de versiones distribuido (VCS) escrito en C originalmente para albergar el código de Linux. Git permite la creación de una historia para una colección de archivos e incluye la funcionalidad para revertir la colección de archivos a otro estado.

**CARACTERÍSTICAS**

* El desarrollo de la aplicación será únicamente nuestro, pudiendo decidir qué parte de nuestro proyecto compartimos y con quién.
* El control de versiones se puede realizar dentro de la propia red con una mayor velocidad de acceso y escritura, así como eliminando el requisito de contar con una conexión a internet obligatoria.
* Se puede crear diferentes ramas sobre las que aplicar nuestras modificaciones en entornos aislados de la línea principal de desarrollo.
* Si en la creación de una rama del proyecto encontramos que uno de los cambios incluidos se integra sin presentar conflictos con las diferentes partes de nuestra aplicación, podremos hacer converger dicha ramificación con el desarrollo principal de forma sencilla y segura.
* Las ramificaciones nos preparan un entorno aislado de pruebas sobre el desarrollo de la línea central de nuestra aplicación.
* Se admite una multitud de configuraciones que nos permitirán dentro de su estructura organizar el trabajo tal y como deseemos.
* Se hace uso de sistemas de árbol SHA1, lo que asegura que hasta que no se realice la comprobación del cifrado o firma, los cambios no se escribirán en el servidor.
* Es un sistema de control de versiones libre de código abierto.

**LIMITES**

* No permite indexar directorios.
* Teniendo una gran curva de aprendizaje, con solo 152 comandos por aprender, la documentación de muchos de estos comandos es obsoleta.

En la figura 03 se muestra el flujo que se utilizará.



**Figura 01.** Flujo de Git

Para realizar el seguimiento de nuestro trabajo, se define los distintos estados en que pueden estar nuestros ficheros en Git.

**Sin seguimiento**: Se encuentran en ese estado todos los ficheros que han sido creados fuera de Git, y nunca los hemos incorporado al gestor de versiones.

**Sin modificar**: Se trata de la situación base en la que se encuentran todos los ficheros sujetos al gestor de versiones, y sobre los que nunca se ha hecho nada.

**Modificado**: Todos aquellos ficheros que están bajo el control de versiones, y sobre los que hemos realizado alguna modificación.

**Preparados**: Son archivos que hemos modificado, y consideramos que ya están listos para entregar, por lo que los asignamos a una entrega.

**Liberado**: Se trata de la entrega de los archivos al repositorio en el que se van almacenando las versiones.

#### GITHUB

Github es una plataforma creada para facilitar el desarrollo colaborativo de software, nos permite alojar proyectos como repositorios en la web gratuitamente, por lo general de forma pública, aunque podemos alojar los proyectos de modo privado, si pagamos una pequeña suscripción mensual.

**CARACTERÍSTICAS**

* Guardar en determinado momento los cambios efectuados a un archivo o conjunto de archivos, con la oportunidad que tener acceso a ese historial de cambios, ya sea para regresar a una de esas versiones o para hacer comparaciones entre ellas.
* Como muchos desarrolladores tienen de forma pública sus proyectos en Github es posible acceder a su código, leerlo, estudiarlo y aprender de él, e incluso podrías hacer cambios y experimentar sin afectar el código original.
* Si luego de copiar un proyecto (hacer Fork) haces ajustes que arreglan bugs o introducen una nueva funcionalidad, puedes proponerle al dueño del proyecto que integre tus cambios en su código a través de un Pull Request.
* Permite trabajar conjuntamente en una idea con un amigo o colega, además se puede invitar a otros usuarios como colaboradores, permitiendo así que estos lean y escriban directamente sobre el repositorio.
* Posee un estupendo visor de código mediante el cual, a través del navegador, podremos consultar en cualquier instante el contenido de un archivo determinado.
* Tiene un sistema de notificaciones con el cual se puede estar al tanto de las actividades alrededor de un repositorio en el cual estas participando o uno en el cual estés interesado.
* Cada proyecto creado en Github incluye un sistema de seguimiento de problemas, del estilo sistema de tickets.
* Es una plataforma web, por tanto, es independiente del sistema operativo que utilices, y además Git que es la herramienta que si requiere instalación es compatible con todos los sistemas; Linux, OSX y Windows.
* Cuenta con una herramienta de revisión de código, donde se pueden añadir anotaciones en cualquier punto de un fichero.
* Es completamente gratis e ilimitado para proyectos públicos.

**LIMITES**

* Github tiene dos modalidades: una gratuita y otra de pago. En la versión gratuita, se podrán crear “n” repositorios de acceso público; pero para el uso de repositorios privados tendrá un coste.
* Limitación de versionamiento para elementos de texto plano.

### ENTORNO

El ambiente de trabajo estará compuesto por:

**Desarrolladores:** Las personas involucradas en el proyecto, tienen acceso para poder modificar los documentos del repositorio.

**Administrador:** Se encarga de verificar los cambios de los documentos, y revisar que se trabaja en el branch establecido. Así mismo debe dar los permisos necesarios a los desarrolladores para realizar el desarrollo.

**Repositorio Remoto de producción:** Github.

**Repositorio Remoto de pruebas:** Github.

### INFRAESTRUCTURA

Se manejará 2 tipos de ramas o branch, que nos va servir para controlar mejor los commit, se va desarrollar en la rama de desarrollo (branch development) que corresponderá 1 branch para cada miembro del equipo y la rama maestra (branch master) la cual alojará las versiones cuyos cambios hayan sido aprobados y estén listos para su respectivo release.

* **Branch master**: Esta rama será la principal, donde se pondrá los cambios aprobados por el administrador.
* **Branch Development**: Esta rama será para los desarrolladores, donde podrán hacer sus cambios previa aprobación, sin alterar la rama principal.



* 1. Calendario

|  |  |  |
| --- | --- | --- |
| ACTIVIDAD | TIEMPO(días) | ROL |
| 1.Planificación de la SCM | 12 días | Gestor de la configuración |
| Identificar la problemática de la empresa. | 2 días | Gestor de la configuración |
| Definir el propósito y finalidad del plan. | 1 día | Gestor de la configuración |
| Identificar roles y responsabilidades. | 2 días | Gestor de la configuración |
| Identificar políticas y responsabilidades. | 2 días | Gestor de la configuración |
| Analizar herramientas. | 3 días | Gestor de la configuración |
| Elaborar el calendario de PGC | 2 días | Gestor de la configuración |
| 2.Identificación de la SCM | 8 días | Gestor de la configuración |
| Identificar ítems de configuración. | 3 días | Gestor de la configuración |
| Clasificar ítems de configuración | 2 días | Gestor de la configuración |
| Definir la nomenclatura de los elementos. | 2 días | Gestor de la configuración |
| Listar ítems con la nomenclatura. | 1 día | Gestor de la configuración |

1. Identificación

**2.1 Identificación de los ítems de configuración (CI)**

**En la siguiente tabla se listan los ítems de configuración identificados con su extensión.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TIPO  (E=Evolución  F=Fuente  S=Soporte) | Nombre del Ítem (CI) | Fuente  (E=Empresa  P=Proyecto  C=Cliente  V=Proveedor) | Extensión | Proyecto |
| Evolución | **Plan de gestión de la configuración** | **Empresa** | **.docx** | **-** |
| Evolución | **Documento del Negocio** | **Proyecto** | **.docx** | **SGIVF** |
| Evolución | **Plan del proyecto** | **Proyecto** | **.docx** | **SGIVF** |
| Evolución | **Listado de requerimientos** | **Proyecto** | **.docx** | **SGIVF** |
| Evolución | **Documento de análisis** | **Proyecto** | **.docx** | **SGIVF** |
| Evolución | **Documento de diseño** | **Proyecto** | **.docx** | **SGIVF** |
| Evolución | **Manual de usuario** | **Proyecto** | **.docx** | **SGIVF** |
| Evolución | **Documento de casos de prueba** | **Proyecto** | **-docx** | **SGIVF** |
| Evolución | **Documento de especificación de Casos de uso** | **Proyecto** | **.docx** | **SGIVF** |
| Evolución | **Documento de diseño de la base de datos** | **Proyecto** | **.docx** | **SGIVF** |
| Evolución | **Documento de aceptación del cliente** | **Proyecto** | **.docx** | **SGIVF** |
| Evolución | **Documento de arquitectura** | **Proyecto** | **.docx** | **SGIVF** |