**Plan de Gestión de Configuración de Software**

Contenido

[1. Introducción 3](#_Toc367492967)

[a. Propósito 3](#_Toc367492968)

[b. Aplicabilidad 3](#_Toc367492969)

[c. Gobierno 3](#_Toc367492970)

[d. Alcance 3](#_Toc367492971)

[e. Definiciones 4](#_Toc367492972)

[2. Gestión de la SCM 5](#_Toc367492973)

[a. Organización 5](#_Toc367492974)

[b. Roles y responsabilidades 6](#_Toc367492975)

[c. Políticas, directrices y procedimientos 6](#_Toc367492976)

[d. Herramientas, entorno e infraestructura 6](#_Toc367492977)

[3. Actividades de la SCM: 10](#_Toc367492989)

[a. Identificación de la configuración, Nomenclatura 10](#_Toc367492990)

# Introducción

La gestión de la configuración es uno de los componentes de éxito de los proyectos TIC. Sin una gestión de la configuración efectiva, la integridad de los ítems de configuración del proyecto y la capacidad de reportar el estado y configuración de aquellos ítems se pone en peligro.

El plan de gestión de la configuración permite establecer un método consistente para identificar y controlar formalmente los ítems de configuración del proyecto.

Los ítems de configuración del proyecto incluyen elementos de hardware y software, así como también información de gestión del proyecto como planes.

La gestión de la configuración es una función integral de la provisión de proyectos TIC porque facilita la protección de los ítems de configuración y comunica los cambios que se han hecho sobre ellos.

Una gestión de la configuración, planificada y ejecutada de manera efectiva, contribuye a la producción de productos TIC de alta calidad evitando el retrabajo. Esto aumenta el valor de los activos informáticos y ahorra costes, contribuyendo a la entrega de proyectos que satisfacen los costes, calendarios, calidad y requerimientos establecidos.

## Propósito

El objetivo de este Plan de Gestión de la Configuración (CM), es proporcionar una visión general de la organización, actividades, tareas en general, y los objetivos de Gestión de la Configuración.

Un plan de gestión de la configuración sirve como una herramienta medular de planificación que describe los esfuerzos de planificación para implementar y ejecutar la gestión de la configuración a lo largo del ciclo de vida del proyecto.

Provee visibilidad y control del producto referente a su desempeño, funcionalidad y atributos físicos.

## Aplicabilidad

El plan de gestión de la configuración debe ser desarrollado para cualquier proyecto clasificado mediano y pequeño.

## Gobierno

El Líder de proyecto es responsable de asegurar que el plan de gestión de la configuración sea desarrollado en conjunto con el plan de gestión del proyecto. El Lider de proyecto asegurará su integración en la planificación general.

## Alcance

El plan de gestión de la configuración debe involucrar a todas las fases del ciclo de vida del software. El documento permitirá mostrar los estándares de etiquetación de los productos de trabajo.

Así mismo esclarecerá el tipo de nomenclatura utilizada para el control de las versiones de los documentos que se encuentran dentro de los elementos de la gestión.

Por otra parte cualquier stakeholder podrá presentar cualquiera de los siguientes tipos de peticiones de cambio sobre el sistema, para el control de cambios:

* Petición de cambios en los requerimientos (adiciones, supresiones, modificaciones, aplazamientos) en el software actualmente en desarrollo
* Informes de los problemas en la producción corriente o sistemas de pruebas beta
* Petición de mejoras en los sistemas actuales de producción
* Petición de nuevos proyectos de desarrollo.

Este proceso de control de cambio se aplica a los productos de línea base creados o gestionados por los miembros del sistema, incluyendo:

* El software que se ha lanzado a la producción o se encuentra en versión beta
* Requisitos de las especificaciones del sistema
* Grupo de procedimientos y procesos
* Usuarios y documentación técnica

Las siguientes clases de productos de trabajo están exentos de este proceso de control de cambios:

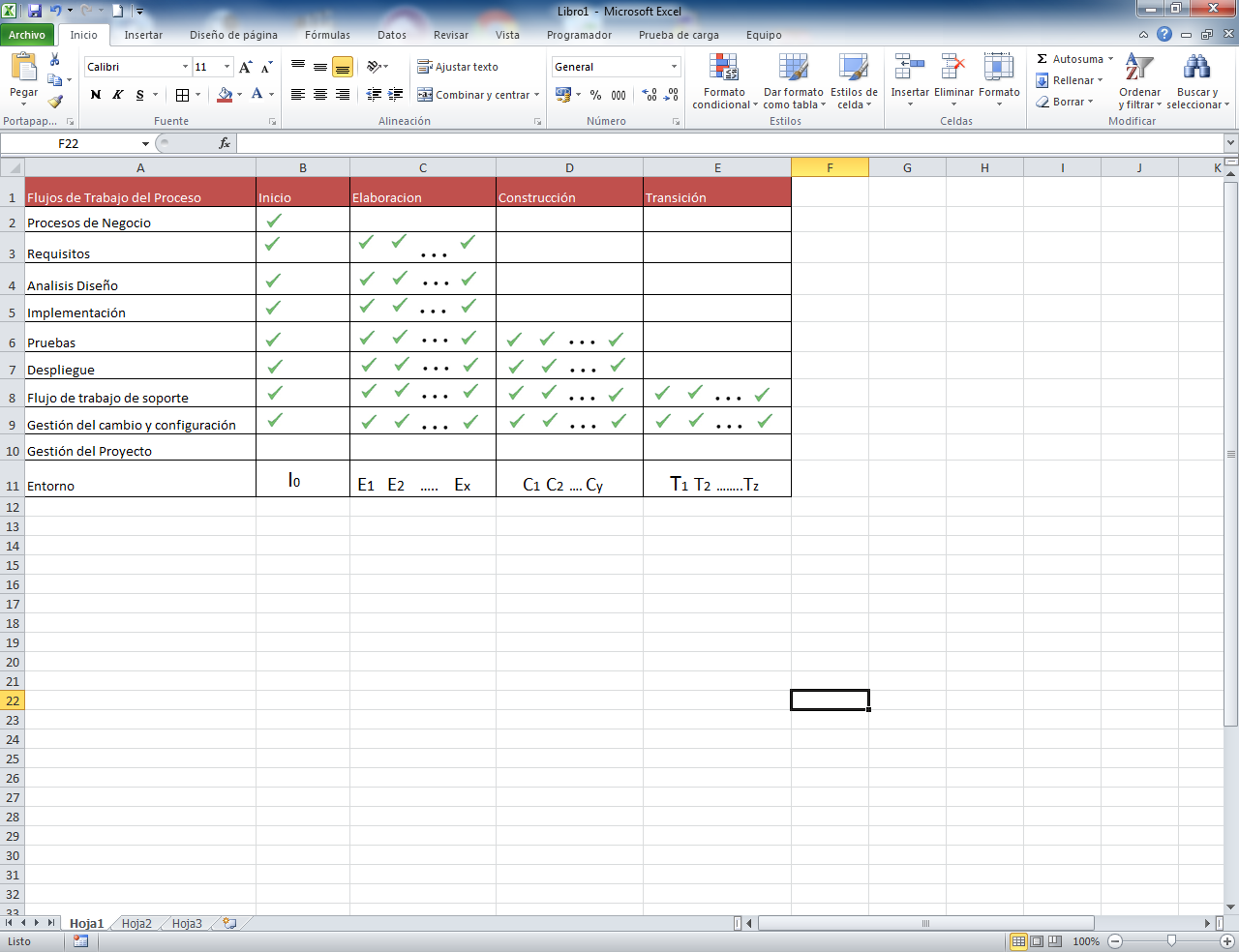
* Los productos de trabajo que están todavía en desarrollo, a excepción de cambios en los requerimientos solicitados en nuevos proyectos

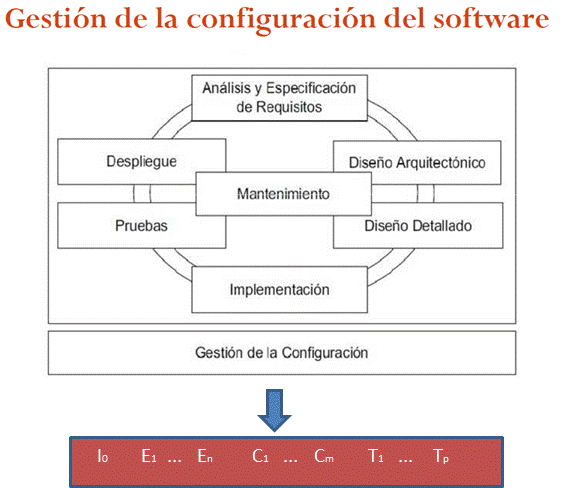
## Definiciones

* Línea Base. Un documento o producto oficial aprobado que sirve como punto de partida para futuras versiones.
* Petición de Cambio. Una solicitud que alguien ha presentado al sistema de control de cambio que describe un problema de software, una mejora solicitada, una propuesta de cambio en los requisitos de un producto en fase de desarrollo, o un nuevo proyecto que se propone.
* Stakeholder. Persona que directa o indirectamente se ve afectada por el sistema y que puede afectar el proyecto.
* Configuration Item. Los elementos que son puestos bajo el control de gestión de la configuración.
* Control de cambio. Proceso renovable de controlar todos los cambios.

# Gestión de la SCM

## Organización





## Roles y responsabilidades

El equipo de la Gestión de Configuración está conformado por los siguientes roles:

|  |  |
| --- | --- |
| **CM Rol** | **Responsabilidades** |
| Administrador de la Configuración | * Crear los repositorios de administración de configuraciones. * Configurar entornos para los proyectos * Implementar las políticas definidas por el Jefe del proyecto. |
| Administrador de Releases | * Implementar los pasos de los elementos al entorno de producción a partir de las líneas base creadas por el Sistema de Administración de Configuraciones |
| Integrador del proyecto | * Identificar posibles construcciones en el entorno de desarrollo. * Entregar los requerimientos implementados de manera formal para su integración. * Sincronizar cuando sea necesario los cambios realizados por la fábrica con el punto de entrada al Sistema de Administración de Configuraciones. |

## Políticas, directrices y procedimientos

* Cada usuario tendrá derecho a solicitar un repositorio Git para su uso personal, el cual estará restringido por una cuota y podrá distribuirla de la manera que más le acomode, pudiendo dividirla entre los repositorios que estime conveniente.
* Cada repositorio Git asignado a grupo de trabajo para los ramos impartidos por el departamento tendrá reglas diferentes a las de los personales y serán controlados por el Administrador de la configuración.
* Cada repositorio Git que sea asignado para un ramo solo durara desde el inicio del ramo hasta el final del ramo. Luego se dará un semestre para que lo respalden, luego de eso será eliminado.

## Herramientas, entorno e infraestructura

## Como herramienta se utilizara el sistema de control de versiones Git, para el entorno de desarrollo (entorno cliente) se utilizara Github.

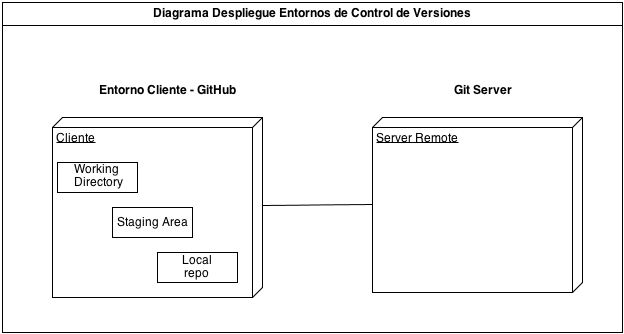


Figura 1 – Diagrama despliegue Git

|  |
| --- |
| Características mínimas Servidor Git |
| Procesador Intel® Xeon® 2.8 GHz |
| Memoria 4 Gigabytes expandibles |
| Almacenamiento Al menos un disco SAS de 300 GB |
| Comunicaciones Un puerto Ethernet 100 Mbps |
| Periféricos Lectora de DVD/CD, puerto USB |
| Sistemas Operativos Centos 6.x, Ubuntu 12.x |

## En el entorno de trabajo cliente se manejan las siguientes estaciones de trabajo:

* Working Directory: Es la carpeta donde están todos los archivos del proyecto. Respecto a git hay dos tipos de archivos: los que figuran en el "Local repo" y se registra un seguimiento de cambios (tracked) y los que no (untracked) que no se registran.
* Staging Area: Se refiere a la lista de archivos cuyos cambios se registrarán en "Local repo" durante el próximo commit (independientemente de si previamente figuraban o no en "Local repo") por ejemplo el comando git add archivo.txt, agrega el archivo.txt a la "Staging Area" es decir lo deja 'preparado' y con el comando git commit se registra en el "Local repo" ya sea su registro inicial o actualizando sus cambios.
* Local repo: Lugar donde se almacena el registro de los cambios en archivos, sólo se registrarán las modificaciones que ocurrieron entre comandos commit, lo cual tiene sus detalles, git almacena todo este registro en una carpeta llamada .git.

## Herramientas: Git es un software de control de versiones, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de código fuente. Entre las características más relevantes se encuentran:

## Fuerte apoyo al desarrollo no-lineal, por ende rapidez en la gestión de ramas y mezclado de diferentes versiones. Git incluye herramientas específicas para navegar y visualizar un historial de desarrollo no-lineal. Una presunción fundamental en Git es que un cambio será fusionado mucho más frecuentemente de lo que se escribe originalmente, conforme se pasa entre varios programadores que lo revisan.

## Gestión distribuida. Git le da a cada programador una copia local del historial del desarrollo entero, y los cambios se propagan entre los repositorios locales. Los cambios se importan como ramas adicionales y pueden ser fusionados en la misma manera que se hace con la rama local.

## Los almacenes de información pueden publicarse por HTTP, FTP, rsync o mediante un protocolo nativo, ya sea a través de una conexión TCP/IP simple o a través de cifrado SSH. Git también puede emular servidores CVS.

## Gestión eficiente de proyectos grandes, dada la rapidez de gestión de diferencias entre archivos, entre otras mejoras de optimización de velocidad de ejecución.

## Todas las versiones previas a un cambio determinado, implican la notificación de un cambio posterior en cualquiera de ellas a ese cambio.

## Resulta algo más caro trabajar con ficheros concretos frente a proyectos, eso diferencia el trabajo frente a CVS, que trabaja con base en cambios de fichero, pero mejora el trabajo con afectaciones de código que concurren en operaciones similares en varios archivos.

## Los renombrados se trabajan basándose en similitudes entre ficheros, aparte de nombres de ficheros, pero no se hacen marcas explícitas de cambios de nombre con base en supuestos nombres únicos de nodos de sistema de ficheros, lo que evita posibles, y posiblemente desastrosas, coincidencias de ficheros diferentes en un único nombre.



**Entorno:** GitHub es un software web para alojar proyectos utilizando el sistema de control de versiones Git. Utiliza el framework Ruby on Rails por GitHub

Entre sus principales características se puede mencionar:

* Wiki para cada proyecto.
* Página web para cada proyecto.
* Gráfico para ver cómo los desarrolladores trabajan en sus repositorios y bifurcaciones del proyecto.
* Funcionalidades como si se tratase de una red social

**Infraestructura:**

Requisitos del sistema Cliente

* Sistema operativo
  + Microsoft® Windows XP, Windows Vista o Windows 7 (versiones de 32 y 64 bits).
* Especificaciones técnicas

|  |  |
| --- | --- |
| Procesador | Intel Core 2 Duo de 2 GHz (o equivalente) |
| Memoria | 2 GB RAM y 1 GB de espacio en el disco duro (sistemas operativos de 32 bits) 4 GB RAM y 1 GB de espacio en el disco duro (sistemas operativos de 64 bits) |
| Tarjeta gráfica | NVIDIA GeForce 5 o una versión más reciente |
| Otros | Unidad de DVD para la instalación del software |

## Calendario

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nombre de tarea | Duración | Comienzo | Fin | Predecesoras |
| **Plan de la gestión de la configuración** | **10.5 días** | **sáb 31/08/13** | **lun 16/09/13** |  |
| Inicio | 0 días | sáb 31/08/13 | sáb 31/08/13 |  |
| **Definición de la estrategia de la configuración** | **1.5 días** | **lun 02/09/13** | **mar 03/09/13** |  |
| **Definición de la gestión de la configuración** | **3 días** | **mar 03/09/13** | **vie 06/09/13** |  |
| **Identificación de la configuración** | 1 día | lun 09/09/13 | lun 09/09/13 | 7 |
| **Establecer control de la configuración** | 3 días | mar 10/09/13 | jue 12/09/13 | 13 |
| **Obtener el estado de la configuración** | 1 día | vie 13/09/13 | vie 13/09/13 | 14 |
| **Obtener la auditoría de la configuración** | 1 día | lun 16/09/13 | lun 16/09/13 | 15 |
| **Gestión y entrega de releases del software** | 1 día | mar 17/09/13 | mar 17/09/13 | 16 |

# 

# 3. Actividades de la SCM:

# Identificación de la configuración, Nomenclatura

**i. Ítems de Configuración.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tipo** (E= Evolución F= Fuente S= Soporte) | **Nombre del Ítem (CI)** | **Fuente** (E= Empresa P= Proyecto C= Cliente V= Proveedor) | **Extensión** | **Proyecto** |
| E | Plan de la Gestión de la Configuración | E | DOCX | - |
| E | Cronograma | E | MPP | - |
| - | Manuel de Instalación GitHub | E | DOCX | - |
| E | Plan de Proyecto | [P](https://github.com/jorgedison/Proyecto-STD/blob/master/METODOLOGIA%20RUP/PROYECTO%20STD/00%20GESTION%20DE%20PROYECTO/STD_PROJ_CHAR.docx) | MPP | STD |
| E | Enunciado del alcance | [P](https://github.com/jorgedison/Proyecto-STD/blob/master/METODOLOGIA%20RUP/PROYECTO%20STD/00%20GESTION%20DE%20PROYECTO/STD_ENUN_ALC.docx) | DOCX | STD |
| E | EDT | [P](https://github.com/jorgedison/Proyecto-STD/blob/master/METODOLOGIA%20RUP/PROYECTO%20STD/00%20GESTION%20DE%20PROYECTO/STD_EDT_PRO.wbs) | WBS | STD |
| E | Cronograma del Proyecto STD | P | MPP | STD |
| E | Documento de Procesos de Negocio | P | DOCX | STD |
| E | Lista de Requisitos Funcionales y no Funcionales | P | DOCX | STD |
| E | Trazabilidad Casos de Usos y Requisitos | P | XLSX | STD |
| E | Especificaciones de Caso de Usos | P | DOCX | STD |
| E | Trazabilidad Casos de Usos y Clases de Análisis | P | XLSX | STD |
| E | Documento de Análisis | P | DOCX | STD |
| E | Documento de Diseño | P | DOCX | STD |
| E | Documento de Microdiseño | P | DOCX | STD |
| E | Fuentes | P | JAVA | STD |
| E | Documento de arquitectura | P | DOCX | STD |
| E | Ejecutables | P | JAR | STD |
| E | Casos de Prueba unitarias x Escenarios | P | XLSX | STD |
| E | Despliegue | P | DOCX | STD |
| E | Plan de Despliegue | P | MPP | STD |
| E | Diagrama de Despliegue | P | DOCX | STD |

# ii. Convención de Nomenclatura de Ítems de Configuración

|  |  |  |
| --- | --- | --- |
| N° | Entregables | Nomenclatura |
| 1 | Plan de Proyecto | [STD\_PROJ\_CHAR](https://github.com/jorgedison/Proyecto-STD/blob/master/METODOLOGIA%20RUP/PROYECTO%20STD/00%20GESTION%20DE%20PROYECTO/STD_PROJ_CHAR.docx) |
| 2 | Enunciado del alcance | [STD\_ENUN\_ALC](https://github.com/jorgedison/Proyecto-STD/blob/master/METODOLOGIA%20RUP/PROYECTO%20STD/00%20GESTION%20DE%20PROYECTO/STD_ENUN_ALC.docx) |
| 3 | EDT | [STD\_EDT\_PRO](https://github.com/jorgedison/Proyecto-STD/blob/master/METODOLOGIA%20RUP/PROYECTO%20STD/00%20GESTION%20DE%20PROYECTO/STD_EDT_PRO.wbs) |
| 4 | Cronograma del Proyecto XYZ | STD\_CRON\_PROY |
| 5 | Documento de Procesos de Negocio | STD\_PRO\_NEG |
| 6 | Lista de Requisitos Funcionales y no Funcionales | STD-REQS |
| 7 | Trazabilidad Casos de Usos y Requisitos | STD-TCUR |
| 8 | Especificaciones de Caso de Usos | STD-ECUS |
| 9 | Trazabilidad Casos de Usos y Clases de Análisis | STD-TCUA |
| 10 | Documento de Análisis | STD-DANA |
| 11 | Documento de Diseño | STD-DDIS |
| 12 | Documento de Microdiseño | STD-MDIS |
| 13 | Fuentes | STD-FUEN |
| 14 | Documento de arquitectura | STD-DAS |
| 15 | Ejecutables | STD-EJEC |
| 16 | Casos de Prueba unitarias x Escenarios | STD-CRON |
| 17 | Despliegue | STD-DESP |
| 18 | Plan de Despliegue | STD-PDES |
| 19 | Diagrama de Despliegue | STD-DDES |

# Documentos

# El identificador consiste de una a tres partes separadas por guión en el formato: PROYECTO-ACRÓNIMO, o PROYECTO-ACRÓNIMO-COMPONENTE.

# Los ítems en evolución se utiliza un identificador de dos partes: ACRÓNIMO\_PROYECTO y ACRÓNIMO derivado del tipo de artefacto. Como ejemplo, para identificar la especificación de Casos de Usos del proyecto STD, tendríamos: STD-ECUS.

# Los ítems en evolución que son específicos de un proyecto y están asociados con un componente específico, utilizan un identificador de tres partes: ACRÓNIMO\_PROYECTO, ACRÓNIMO\_COMPONENTE, y, ACRÓNIMO derivado del tipo de artefacto.

# El nivel de versión se mantiene como un identificador numérico con dos componentes: Versión.Revisión. Ejemplo: 1.1; (Versión 1, Revisión 1).

# Número de versión

# El número de versión cambia únicamente cuando la arquitectura principal del ítem ha cambiado, o cuando el ítem es completamente reconstruido, con cambios internos sustanciales. En este caso la versión 1.1 se convertirá en versión 2.0.

# Número de revisión

# El número de revisión cambia cuando el contenido ha cambiado, pero la estructura principal y el flujo del ítem se mantiene igual. La secuencia normal de las revisiones es: 1.0, 1.1, 1.2, 1.3, etc.

# Archivos ejecutables y de soporte

# El esquema de numeración de versiones consiste de tres componentes: Versión.RevisiónActualización. Ejm: 1.1a.

# Identificación de ítems fuente

# Esto se maneja en base a la herramienta de gestión de la configuración GitHub.

# Identificación de ítems de soporte

# Son identificados por su nombre y el número de versión necesario para soportar el entorno de producción o desarrollo. Como ejemplo, si un editor se actualiza de la versión 2.1 a 2.2a, el rango de versión del ítem de

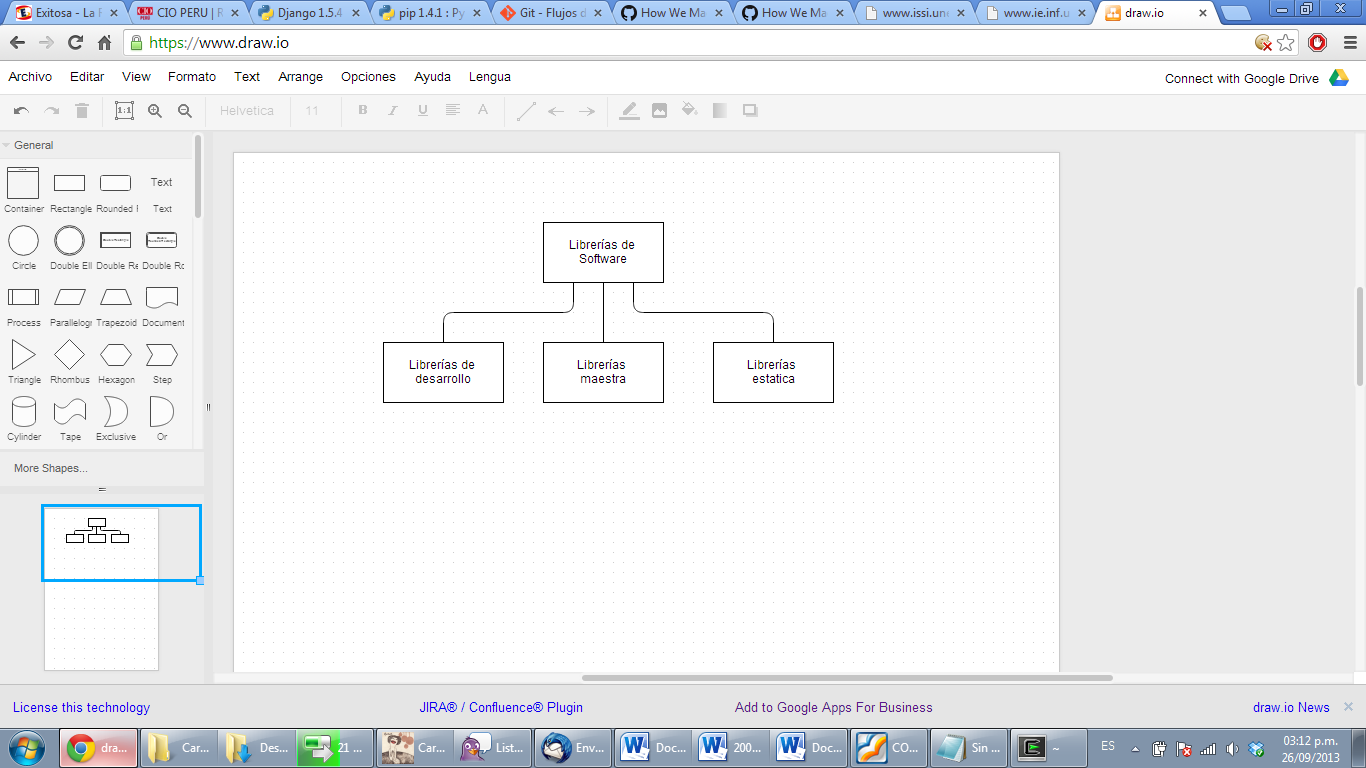
# configuración será 2.1 – 2.2a

# b .Control de la configuración

i. Línea Base

|  |  |  |
| --- | --- | --- |
| **Línea Base** | **Evento/Hito** | **Ítems de Configuración** |
| **Línea Base Funcional** | | |
| Línea base de planificación | Fin de Fase de Planificación, iteración preliminar. |   Casos de uso del negocio |
|   Plan de gestión del proyecto |
|   Especificación preliminar de requisitos |
| **Líneas Base de Concepción** | | |
| Línea base de especificación de requerimientos | Fin primera iteración de concepción. |   Especificación de Requerimientos; |
| **Configuración durante el desarrollo** | | |
| Línea base de elaboración | Última iteración de elaboración |   Diseño preliminar; |
|   Diseño detallado; |
|   Plan de pruebas: unitario, integración, aceptación y sistema; |
| Línea base de construcción | Al final de cada iteración |   Especificación de casos de prueba; |
|   Especificación de procedimientos de prueba; |
|   Código fuente; |
|   Documentación del código; |
|   Resultados de pruebas unitarias; |
| Línea base de integración y pruebas | Al final de la última iteración de construcción |   Resultados de las pruebas de integración y sistema; |
| **Línea Base de Producto** | | |
| Línea base de aceptación y entrega | Al final de la fase de ejecución, última iteración de transición |   Software desplegado |
|   Documentación del software; manuales |
|   Descripción de la versión del software |

ii. Librerías controladas



**Librería Desarrollo**

• Librerías de diseñadores y programadores

• Repositorio de trabajo propia de los programadores, gestionada por ellos

• Contiene el código fuente de SW en desarrollo.

Actividades:

Mantener actualizado el repositorio durante el ciclo de vida del proyecto.

Responsable:

Integrador del Proyecto.

Contenido

* Línea base de diseño
* Línea base de construcción
* Línea base de integración y pruebas

Accesos:

|  |  |
| --- | --- |
| Rol | Tipo de acceso |
| Gestor de Configuración | Leer  Escritura  Ejecutar  Eliminar |
| Gerente del Proyecto | Leer  Ejecutar |
| Desarrollo | Leer  Ejecutar |

Dirección de repositorio: <https://github.com/alexgino147/lib-dev>

**Librería maestra**

• Repositorio de trabajo propio de la Organización de Gestión de Configuración.

• Contiene SW de soporte de control, SW liberado para su integración.

Actividades:

Repositorio donde se encuentra el código fuente integrado listo para ser pasado a producción, el cual ya paso por una verificación.

Responsable:

Arquitecto de Software.

Contenido

* Línea base de planificación.
* Línea base de especificación de requerimientos
* Línea base de diseño
* Línea base de construcción
* Línea base de integración y pruebas
* Línea base de aceptación y entrega

Accesos:

|  |  |
| --- | --- |
| Rol | Tipo de acceso |
| Gerente del Proyecto | Leer  Escritura  Ejecutar  Eliminar |
| Arquitectura | Leer  Ejecutar |
| Analista de Calidad | Leer |

Dirección de repositorio: <https://github.com/alexgino147/lib-mst>

**Librería estática**

• Almacén SW o archivo de seguridad.

• Repositorio de almacenamiento propio de la Organización de Gestión de Configuración

• Contiene el SW que haya contenido la librería maestra a lo largo del tiempo.

• No permite el cambio de elementos contenidos, sí puede ampliarse y reducirse

Responsable:

Gestor de la Configuración.

Contenido:

* Línea base de integración y pruebas
* Línea base de aceptación y entrega
* Código Fuente, Frameworks, plugins

Accesos:

|  |  |
| --- | --- |
| Rol | Tipo de acceso |
| Gestor de Configuración | Leer  Escritura  Ejecutar  Eliminar |
| Gerente de Proyecto | Leer  Ejecutar |

Dirección de repositorio: <https://github.com/alexgino147/lib-stc>

iii. Control de cambios

El proceso de control de cambios gestiona la solicitud, evaluación, aprobación y ejecución de cambios. Cuando la solicitud de cambio afecta a un ítem bajo línea base requerirá aprobación del Comité de Cambios (CCC); caso contrario serán gestionadas por el gerente del proyecto y el arquitecto de software.

## Comité de Control de Cambios (CCC)

El CCC estará conformado por los siguientes miembros:

* Líder ejecutivo del proyecto;
* Gerente de proyecto;
* Arquitecto de Software;
* Usuario afectados por la solicitud de cambio;

## Solicitudes de Cambio

El control de configuración debe cubrir lo siguiente:

* Identificación y documentación de la necesidad de cambio;
* Análisis y evaluación del pedido de cambio;
* Aprobación o desaprobación del pedido;
* Verificación, implementación y liberación de cambios

Plantilla Base para Solicitud de cambios

|  |  |
| --- | --- |
| ID | <número 2 dígitos – Nombre de la petición del cambio> |
| Proyecto | <Nombre del proyecto para el que se solicita el cambio> |
| Fecha | <Fecha de la petición del cambio Formato dd-mm-aaaa> |
| Fuentes | <Persona que ha identificado la necesidad del cambio> |
| Descripción | <Descripción del cambio, incluye el motivo, el propósito, los ítems de configuración, los recursos necesarios para su implementación> |
| Justificación | <Justificación del cambio solicitado> |
| Observaciones | <Descripción del proceso de reversión |
| Estado | <Estado de la solicitud del cambio> |
| Responsables | <Lista de responsables de la implementación del cambio> |  |