**Universidad Nacional Mayor de San Marcos**

Universidad del Perú. Decana de América  


**ÁREA DE INGENIERÍA**

**FACULTAD DE INGENIERÍA DE SOFTWARE E INFORMÁTICA**

**E.A.P: Ingeniería de software**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**PLAN DE GESTIÓN DE LA CONFIGURACIÓN DEL SOFTWARE**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**INTEGRANTES DEL GRUPO 3:**

* Ames Camayo, Daniel Vides.
* Castillo Bernal, Carlos Alberto.
* Cjumo Chumbes Jose Carlos.
* Hinostroza Quispe, Gianlucas Amed.
* Ipanaque Pazo, Jorge Paul.
* Ramírez Alvarado Piero Jaime.
* Saenz Chang Jesus Angel.

**DOCENTE:** Wong Portillo, Lenis Rossi.

**LIMA – PERÚ (2023)**

**Historial de revisiones**

| **Fecha** | **Revisión** | **Descripción** | **Autor** |
| --- | --- | --- | --- |
| 30/04/2023 | 1.0 | Primera versión del plan de gestión de la configuración del software | Consultora HomeSkill S.A. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**PLAN DE GESTIÓN DE LA CONFIGURACIÓN DEL SOFTWARE**

1. **Introducción.**
   1. **Situación de la empresa.**

* **Antecedentes de la empresa.**

La empresa es una consultora líder en el desarrollo de software innovador y de calidad, con una misión de proporcionar soluciones de software personalizadas y de alta calidad a sus clientes. Entre los clientes más importantes que atiende la consultora se encuentran empresas de renombre en el mercado local e internacional, tales como BCP, Cencosud, Clínica Internacional, entre otros. Sus proyectos han sido reconocidos por su excelencia y han permitido a los clientes mejorar su eficiencia y rentabilidad:

* **Sistema de gestión de inventarios para una empresa de retail**: Este sistema fue desarrollado utilizando la tecnología de base de datos Oracle y el lenguaje de programación Java. La solución permitió a la empresa gestionar de manera eficiente sus inventarios, desde la entrada y salida de productos hasta la realización de pedidos a proveedores. Los problemas al desarrollar el proyecto fue el control de los cambios realizados en el software y sus componentes, lo que conlleva a retrasos en el tiempo de entrega del proyecto.
* **Plataforma de e-learning para una institución educativa**: La plataforma fue desarrollada utilizando la tecnología de base de datos MySQL y el lenguaje de programación PHP. Permitió a la institución ofrecer cursos en línea a sus estudiantes, con herramientas de seguimiento de progreso y calificación automatizada. En esta ocasión, el problema fue que el código no se estaba registrando correctamente, lo que generó inconvenientes en la funcionalidad de la plataforma.
* **Sistema de gestión de ventas para una empresa de servicios**: Este sistema fue desarrollado utilizando la tecnología de base de datos SQL Server y el lenguaje de programación C#. La solución permitió a la empresa gestionar sus ventas de manera eficiente, desde la cotización hasta la facturación, con herramientas de seguimiento de ventas y estadísticas de desempeño. El problema del proyecto que tuvo fue la comunicación con el cliente, en donde los requerimientos iban aumentando y no se sentía satisfecho con lo que se le iba presentando.

Además, la empresa ha utilizado tecnologías como AngularJS, React, .NET, Python y Ruby on Rails en diferentes proyectos, según las necesidades de los clientes. También ha brindado servicios de integración de sistemas con plataformas como Salesforce y SAP.

Por último, se mencionan los diferentes problemas en los procesos que se relacionan con la gestión de la configuración como puede ser la identificación, control, análisis, seguimiento y revisión de los cambios realizados en el software y sus componentes, desde la planificación hasta la implementación.

* **Áreas de negocio**
  + Desarrollo de software: La empresa ofrece soluciones de software dependiendo de las necesidades de sus clientes, utilizando diferentes lenguajes de programación y tecnologías.
  + Consultoría en tecnología: Ayuda a los clientes a definir la mejor estrategia de tecnología para su negocio, así como la elección y configuración de herramientas y tecnologías.
  + Integración de sistemas: Servicio que permite a sus clientes la configuración y conexión de sistemas existentes con el fin de garantizar su funcionamiento correcto y mayor eficiencia en el procesamiento de la información. Además incluye la identificación y resolución de problemas técnicos y la integración de plataformas y sistemas de terceros.
  + Mantenimiento de software: Ofrece servicios de mantenimiento de software para garantizar que los sistemas estén actualizados y funcionando correctamente, además de cumplir con los requerimientos del cliente.
* **Clientes**

La consultora HomeSkill S.A.C tiene una amplia base de clientes que se extiende a través de diferentes sectores e industrias. Algunas de sus clientes son pequeñas y medianas empresas que buscan soluciones de software personalizadas para mejorar sus procesos de negocio y aumentar su eficiencia operativa:

* Banco central del Perú (BCP).
* Cencosud.
* Clínica Internacional.
* **Estructura organizacional**

La Consultora HomeSkill S. A. C es una empresa de desarrollo de software que cuenta con los mejores analistas y programadores de la región. La estructura organizacional de la empresa se divide en cuatro áreas principales: desarrollo de software, control de calidad y pruebas, gestión de proyectos y soporte al cliente.

* **Mantenimiento del software**

El mantenimiento del software es un proceso importante en el ciclo de vida de cualquier producto de software, y nuestra consultoría HomeSkill S.A.C tiene un enfoque riguroso para garantizar que sus productos de software estén actualizados y funcionando correctamente en todo momento.

El proceso de mantenimiento de software de nuestra empresa incluye varias etapas, desde la identificación de problemas hasta la implementación de soluciones. Los problemas pueden ser reportados por los usuarios finales del software, o pueden ser detectados internamente durante las pruebas de software regulares que realiza la empresa.

* **Identificación del problema:** El problema puede ser reportado por los usuarios finales o puede ser detectado internamente durante las pruebas de software regulares.
* **Evaluación del problema:** Se evalúa la gravedad del problema y determina si se requiere una solución inmediata.
* **Desarrollo de solución:** Se desarrolla una solución para el problema, que puede incluir revisión del código fuente, identificación de errores o implementación de nuevas características o mejoras de rendimiento.
* **Prueba de la solución:** Se prueba la solución desarrollada para asegurarse de que resuelva el problema y no afecte a otras partes del software.
* **Implementación de la solución:** Se implementa la solución en el software.
* **Mantenimiento preventivo:** Se realizan actualizaciones de seguridad y actualizaciones regulares para garantizar la compatibilidad con los sistemas operativos y otras aplicaciones de software relacionadas.
* **Soporte técnico:** Se ofrece servicios de soporte técnico continuo para ayudar a sus clientes a resolver cualquier problema o incidencia que puedan tener con sus productos de software.
  1. **Propósito de Implementar la Gestión de la Configuración del Software.**

Algunos de los propósitos más importantes que tenemos en cuenta al momento de implementar el SCM en nuestra empresa son:

* **Mejorar la calidad del software:** Podemos mejorar la calidad del software al garantizar que todos los elementos de configuración estén actualizados y sean consistentes.
* **Reducir los riesgos:** Nos ayuda a reducir los riesgos asociados con la falta de control en los cambios del software así como la pérdida de información valiosa.
* **Facilitar el mantenimiento del software:** Facilitaremos el mantenimiento del software a lo largo de su ciclo de vida, lo que a su vez reduciría los costos de mantenimiento y aumentaría la eficiencia.
* **Mejorar la colaboración del grupo:** Facilitaremos la colaboración entre los miembros del equipo de desarrollo y mejoraremos la comunicación en todo el equipo.

El propósito de implementar la gestión de la configuración del software en nuestra empresa es asegurar la integridad y la consistencia de los artefactos de software a lo largo de su ciclo de vida. Además, nos permitirá definir y controlar las versiones y variantes de los elementos de configuración (código fuente, documentación, base de datos, etc.) que conforman el producto final. De esta manera, podremos evitar cambios incontrolados y asegurar que todos los miembros del equipo de desarrollo trabajen con la misma versión actualizada de los productos que manejan. También podremos definir las reglas de versionado y criterios de actuación para cada caso en particular, lo que nos permitirá tener un control adecuado sobre los cambios realizados en el software. Con la implementación de la gestión de la configuración, se garantiza la calidad del software entregado y se mejora la eficiencia y efectividad en la gestión de los proyectos de Consultores HomeSkill.

* 1. **Benchmarking de herramientas.**

**Comparativa entre las características de 5 sistemas de control de versiones**

**Integración / Plugs-ins IDE:** La integración y los plugins de un IDE son características importantes que pueden mejorar significativamente la eficiencia y la productividad de los desarrolladores de software.

| **Criterios** | **Git**  **(2)** | **Git Kraken**  **(2)** | **TFS**  **(3)** | **Bazaar**  **(4)** | **Mercurial**  **(5)** |
| --- | --- | --- | --- | --- | --- |
| Gratuito | Si | Si | Si | Si | Si |
| Multiplataforma | Si | Si | No | Si | Si |
| Interfaz Web | Si | Si | Si | Si | Si |
| Integración / Plugs-ins IDE | Si | Si | Si | Si | Si |
| Soporte para diferentes lenguajes | Si | Si | Si | Si | Si |
| Réplica de repositorio | Si | Si | Si | Si | Si |
| Unicode | Si | Si | Si | Si | Si |
| Mantenimiento | Si | Si | Si | No | Si |

Se colocaron algunas características importantes para poder comparar el beneficio de usar cada sistema de control de versiones, se observa que Git, Git Kraken y Mercurial poseen el 100% de las características, para escoger uno de ellos decidimos enfocarnos en la simplicidad de uso de cada uno, y con eso en mente decidimos escoger **Mercurial**, ya que tiene una estructura de datos más simple.

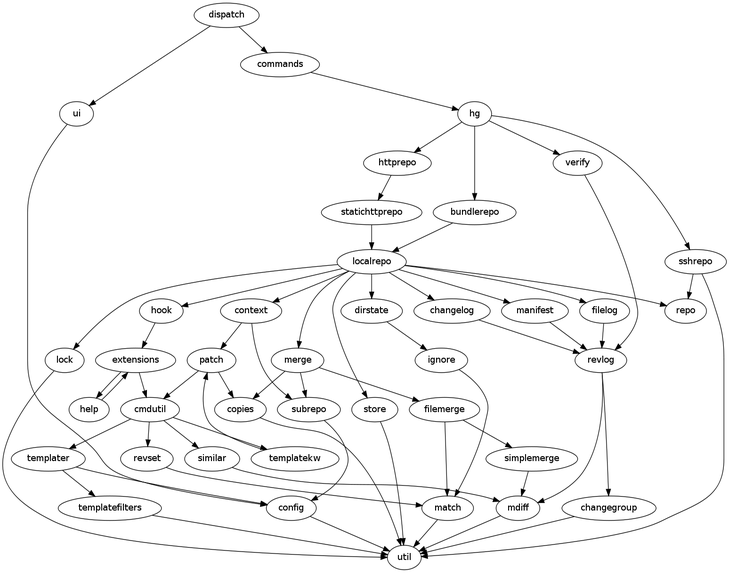
* 1. **Herramienta elegida: Mercurial.**

Mercurial es un sistema de control de versiones desarrollado en su mayoría en Python y, en un principio, fue desarrollado para sistemas GNU/Linux. Fue creado en 2005 y se ha caracterizado por su gran rendimiento y escalabilidad [1]. Entre sus características tenemos las siguientes:

* **Multiplataforma**. Mercurial es compatible con los sistemas GNU/Linux, Mac y Windows [2].
* **Distribuido.** Se dice que Mercurial tiene una arquitectura distribuida, ya que no necesita un servidor central y el equipo de desarrollo puede trabajar por separado y sin depender de una red lenta, ya que cada colaborador trabaja desde un repositorio clonado almacenado en la nube [2].
* **Rápido.** La implementación y estructuras de datos usadas por Mercurial se lo permiten. Por ello, las diferenciaciones de revisiones o el retorno a versiones anteriores del proyecto se realizan rápidamente [2].
* **Open Source.** Mercurial utiliza la licencia GNU y es considerado software libre [2].
* **Fácil de usar.** Mercurial está pensando para que la curva del aprendizaje no sea muy alta., por lo que encontramos similitudes con otros sistemas de control de versiones y el proceso de aprendizaje será más rápido [2].

A continuación, se mencionan algunas de las ventajas con las que cuenta Mercurial:

* Es escalable y se adapta al tamaño y a las exigencias del proyecto [3].
* Su instalación es rápida y liviana [3].
* Posee una interfaz web integrada y una amplia documentación [3].
* Se pueden añadir funcionalidad adicionales a Mercurial por medio de las extensiones, las cuales se pueden descargar de la Wiki de Mercurial o también se pueden escribir nuevas funcionalidades con Python, ya que Mercurial está escrito en este lenguaje [2].

**Arquitectura de Mercurial.**

**Fig. 1.** Arquitectura de Mercurial [4].

En la imagen anterior se puede apreciar un grafo que puede representar la arquitectura sobre la cual trabaja Mercurial, ya que se muestran muchos comandos conocidos para quienes ya conocemos otras herramientas de control de versiones como Git. Algunos de estos comandos familiares pueden ser *help, ignore, config, merge,* entre otros.

Los procesos que realiza mercurial comienzan cuando todos los argumentos de la línea de comandos pasan a una función en el módulo *dispatch*. Lo primero que sucede es que se crea una instancia de un objeto *ui*. La clase *ui* primero intentará encontrar archivos de configuración en varios lugares conocidos (como su directorio de inicio) y guardará las opciones de configuración en el objeto *ui*. Los archivos de configuración también pueden contener rutas a extensiones, que también deben cargarse en este punto. Además, cualquier opción global pasada en la línea de comandos también se guarda en el objeto *ui* en este punto [4].

Una vez hecho esto, tenemos que decidir si crear un objeto de repositorio. Si bien la mayoría de los comandos requieren un repositorio local (representado por la clase *localrepo* del módulo *localrepo*), algunos comandos pueden funcionar en repositorios remotos (ya sea HTTP, SSH o alguna otra forma registrada), mientras que algunos comandos pueden hacer su trabajo sin hacer referencia a ningún repositorio. La última categoría incluye el comando init, por ejemplo, que se usa para inicializar un nuevo repositorio [4].

Por lo mencionado en esta sección y en el benchmarking de herramientas, se ha optado por seleccionar el sistema de control de versiones Mercurial para los diversos proyectos que se llevan a cabo en nuestra empresa Consultora HomeSkill S.A.C.

1. **Identificación.**
   1. **Clasificación de Ítem**

| **Tipo** | **Nombre del Item(CI)** | **Nomenclatura** | **Extensión** | **Proyecto** |
| --- | --- | --- | --- | --- |
| E | Plan de la Gestión de la configuración del Software | PGCS | DOCX | - |
| E | Cronograma del proyecto | GS-CP | XLSX | GS |
| E | Plan de proyecto | GS-PY | DOCX | GS |
| E | Documento arquitectura del software | GS-AS | DOCX | GS |
| E | Documento de especificación de UI | GS-UI | DOCX | GS |
| E | Documento de guía de estilos | GS-GE | DOCX | GS |
| E | Documento de especificación de la BD | GS-DEBD | DOCX | GS |
| E | Plan de Fase | GS-PF | DOCX | GS |
| E | Plan de iteración 1 | GS-PI-01 | DOCX | GS |
| E | Documento de especificación de requisitos 1 | GS-DER-01 | DOCX | GS |
| E | Documento de especificación de requisitos 2 | GS-DER-02 | DOCX | GS |
| E | Documento de especificación de requisitos 3 | GS-DER-03 | DOCX | GS |
| E | Documento de especificación de requisitos 4 | GS-DER-04 | DOCX | GS |
| E | Documento de especificación de requisitos 5 | GS-DER-05 | DOCX | GS |
| E | Documento de especificación de requisitos 6 | GS-DER-06 | DOCX | GS |
| E | Documento de especificación de requisitos 7 | GS-DER-07 | DOCX | GS |
| E | Documento de casos de uso 1 | GS-CU-01 | DOCX | GS |
| F | Código del requisito 1 | Desarrollo\_GoShop/GoShop  Desarrollo\_GoShop/Ventas\_Retail  Desarrollo\_GoShop/db.sqlite3  Desarrollo\_GoShop/manage.py | PY  HTML  CSS  JS | GS |
| F | Código del requisito 2 | Desarrollo\_GoShop/GoShop  Desarrollo\_GoShop/Ventas\_Retail  Desarrollo\_GoShop/media/shop  Desarrollo\_GoShop/db.sqlite3  Desarrollo\_GoShop/manage.py | PY  HTML  CSS  JS | GS |
| F | Código del requisito 3 | Desarrollo\_GoShop/GoShop  Desarrollo\_GoShop/Ventas\_Retail  Desarrollo\_GoShop/media/shop  Desarrollo\_GoShop/db.sqlite3  Desarrollo\_GoShop/manage.py | PY  HTML  CSS  JS | GS |
| F | Código de prueba unitaria del requisito 1 | tests\_01 | PY | GS |
| F | Código de prueba unitaria del requisito 2 | tests\_02 | PY | GS |
| F | Código de prueba unitaria del requisito 3 | tests\_03 | PY | GS |
| E | Documento de informe de pruebas 1 | GS-IP-01 | DOCX | GS |
| E | Reporte del desarrollo de software 1 | GS-RDS-01 | DOCX | GS |
| E | Plan de iteración 2 | GS-PI-02 | DOCX | GS |
| E | Documento de casos de uso 2 | GS-CU-02 | DOCX | GS |
| F | Código del requisito 4 | Desarrollo\_GoShop/GoShop  Desarrollo\_GoShop/Ventas\_Retail  Desarrollo\_GoShop/db.sqlite3  Desarrollo\_GoShop/manage.py | PY  HTML  CSS  JS | GS |
| F | Código del requisito 5 | Desarrollo\_GoShop/GoShop  Desarrollo\_GoShop/Ventas\_Retail  Desarrollo\_GoShop/db.sqlite3  Desarrollo\_GoShop/manage.py | PY  HTML  CSS  JS | GS |
| E | Reporte del desarrollo de software 2 | GS-RDS-02 | DOCX | GS |
| E | Plan de iteración 3 | GS-PI-03 | DOCX | GS |
| E | Documento de casos de uso 3 | GS-CU-03 | DOCX | GS |
| F | Código del requisito 6 | Desarrollo\_GoShop/GoShop  Desarrollo\_GoShop/Ventas\_Retail  Desarrollo\_GoShop/db.sqlite3  Desarrollo\_GoShop/manage.py | PY  HTML  CSS  JS | GS |
| F | Código del requisito 7 | Desarrollo\_GoShop/GoShop  Desarrollo\_GoShop/Ventas\_Retail  Desarrollo\_GoShop/media/shop  Desarrollo\_GoShop/db.sqlite3  Desarrollo\_GoShop/manage.py | PY  HTML  CSS  JS | GS |
| E | Reporte del desarrollo de software 1 | GS-RDS-01 | DOCX | GS |
| E | Manual de Usuario | GS-MU | DOCX | GS |
| E | Acta de cierre del proyecto | GS-ACP | DOCX | GS |

*Nota.* E = Evolución, F = Fuente, S = Soporte

* 1. **Definición de Nomenclaturas**

Incluiremos una lista de términos y acrónimos utilizados en el contexto de nuestros proyectos de software.

| Caso 1 | Cuando es un ítem único que pertenece a un proyecto único | ACRÓNIMO DEL PROYECTO + ‘-’+ACRÓNIMO DEL ELEMENTO | GS-CP |
| --- | --- | --- | --- |
| Caso 2 | Cuando un ítem no pertenece a un proyecto, es decir, pertenece a la empresa en general | ACRÓNIMO DEL ELEMENTO | PGCS |
| Caso 3 | Cuando es un ítem que hace referencia a un tipo de elemento y son más de uno, se agrega la numeración | ACRÓNIMO DEL PROYECTO + ‘-’+ ACRÓNIMO DEL ELEMENTO + ‘-’ + CORRELATIVO | GS-PI-01  GS-PI-02  GS-PI-03 |
| Caso 4 | Cuando el acrónimo de un ítem se repite con otro, se agregan las dos primeras letras del elemento para diferenciarlos. | ACRÓNIMO DEL PROYECTO + ‘-’+ ACRÓNIMO DEL ELEMENTO (DOS PRIMERAS LETRAS DE LA PALABRA CON LETRA INICIAL REPETIDA) | GS-DAR  GS-DAN |

* 1. **Diseño de la estructura del repositorio**

***2.3.1 Librería Documentos***

* **Responsable**

Jefe de proyecto: Jorge Paul Ipanaque Pazo

* **Actividades**

Almacenar y gestionar documentos, manuales, guías de usuario, especificaciones y otros recursos importantes relacionados con el desarrollo y gestión del software

* **Contenido**

Se almacenan en las carpetas todos los documentos relacionados a la empresa.

* Estándares
* Manuales
* Plantillas
* Políticas
* **Accesos**

| **Rol** | **Tipo de acceso** |
| --- | --- |
| Jefe de proyecto | * Leer * Escribir * Ejecutar * Eliminar |
| Desarrolladores y Testers | * Leer |

***2.3.2 Librería Desarrollo***

* **Responsable**

Arquitecto de software: Jorge Paul Ipanaque Pazo

* **Actividades**

Permite la gestión y el control de versiones del código fuente del software que se está desarrollando junto con las dependencias y herramientas necesarias para construir y ejecutarlo

* **Contenido**

Se almacenan en carpetas las aplicaciones desarrolladas para cada cliente, en donde se pueden encontrar manuales sobre el uso del programa y la explicación de cada funcionalidad de la misma.

* + AirGo
  + Emsamlud
  + GoShop
* **Accesos**

| **Rol** | **Tipo de acceso** |
| --- | --- |
| Arquitecto de Software | * Leer * Escribir * Ejecutar * Eliminar |
| Desarrolladores | * Leer * Escribir * Ejecutar |
| Testers | * Leer * Ejecutar |

***2.3.3 Librería Línea Base***

* **Responsable**

Arquitecto de Software: Jorge Paul Ipanaque Pazo

* **Actividades**

Permite la gestión de versiones estables donde se puede trabajar y realizar pruebas en él sin problemas para el cliente y el equipo de trabajo.

* **Contenido**

Se almacenan en carpetas copias de versiones de la aplicación funcional con los ítems correspondientes por el cronograma

* AG
* EMS
* GS
* **Accesos**

| **Rol** | **Tipo de acceso** |
| --- | --- |
| Arquitecto de Software | * Leer * Escribir * Ejecutar * Eliminar |
| Desarrolladores | * Leer * Escribir * Ejecutar |
| Testers | * Leer * Ejecutar |

***2.3.4 Librería Cliente***

* **Responsable**

Jefe de proyecto: Jorge Paul Ipanaque Pazo

* **Actividades**

Se realiza la gestión de los requerimientos y cualidades de los clientes con los que trabajaremos como pueden ser sus necesidades, documentación de los acuerdos y contratos establecidos entre el cliente y la empresa.

Gestionar los requerimientos y cualidades de los clientes con los que trabajaremos, jerarquizando prioridades para el avance de las líneas base.

* **Contenidos**

Se almacena en carpetas a los clientes en donde contienen la documentación necesaria donde se mencionan los requerimientos que están solicitando para el desarrollo de su aplicación.

* + SkiMall
  + MeditCare
  + ProSupply
* **Accesos**

| **Rol** | **Tipo de acceso** |
| --- | --- |
| Jefe de Proyecto | * Leer * Escribir * Ejecutar * Eliminar |
| Desarrolladores y testers | * Leer * Ejecutar |

* 1. **Definición de Líneas Base (GS)**

| **Línea Base** | **Hito** | **Items** |
| --- | --- | --- |
| LB1 | 1 | - GS-PP.docx  - GS-CP.xlsx  - GS-PF.docx  - GS-PI-01.docx  - GS-DER-01.docx  - GS-DER-02.docx  - GS-DER-03.docx  - GS-DER-04.docx  - GS-DER-05.docx  - GS-DER-06.docx  - GS-DER-07.docx  - Desarrollo\_GoShop/GoShop  - Desarrollo\_GoShop/Ventas\_Retail  - Desarrollo\_GoShop/media/shop  - Desarrollo\_GoShop/db.sqlite3  - Desarrollo\_GoShop/manage.py  - tests\_01.py  - tests\_02.py  - tests\_03.py  - GS-GE.docx  - GS-UI.docx  - GS-CU-01.docx  - GS-IP-01.docx  - GS-RDS-01.docx |
| LB2 | 2 | - GS-AS.docx  - GS-UI.docx  - GS-GE.docx  - GS-DEBD.docx  - GS-PF.docx  - GS-PI-02.docx  - GS-CU-02.docx  - Desarrollo\_GoShop/GoShop  - Desarrollo\_GoShop/Ventas\_Retail  - Desarrollo\_GoShop/media/shop  - Desarrollo\_GoShop/db.sqlite3  - Desarrollo\_GoShop/manage.py  - tests\_04.py  - tests\_05.py  - GS-IP-02.docx  - GS-RDS-02.docx |
| LB3 | 3 | - GS-PI-03.docx  - GS-CU-03.docx  - Desarrollo\_GoShop/GoShop  - Desarrollo\_GoShop/Ventas\_Retail  - Desarrollo\_GoShop/media/shop  - Desarrollo\_GoShop/db.sqlite3  - Desarrollo\_GoShop/manage.py  - tests\_06.py  - tests\_07.py  - GS-IP-03.docx  - GS-MU.docx  - GS-PF.docx  - GS-AS.docx  - GS-UI.docx  - GS-GE.docx  - GS-DEBD.docx  - GS-RDS-03.docx  - GS-ACP.docx |

1. **Control.**
2. **Estado.**
3. **Entrega y gestión de release.**
4. **Referencias.**

[1] EcuRed (s.f.). Mercurial (sistema de control de versiones) [Online]. Available: <https://www.ecured.cu/Mercurial_(sistema_de_control_de_versiones)>

[2] Mercurial (s.f.). Mercurial source control management [Online]. Available: <https://www.mercurial-scm.org/about>

[3] G. B. Chiriboga (Octubre, 2013). Sistemas de Control de Versiones [Online]. Available: <https://portfoliogabrielfcr.wordpress.com/2013/10/25/62/>

[4] D. Ochtman (s. f.).The Architecture of Open Source Applications (Vol 1) Mercurial [Online]. Available: <https://aosabook.org/en/v1/mercurial.html>

[5] IBM Rational Build Forge (Versión 7.1.3) [Software de computadora]. (2021). IBM. <https://www.ibm.com/docs/es/rational-build-forge/7.1.3?topic=welcome-to-build-forge-713-documentation>