Universidad de la Ciencias Informáticas

Facultad 1

Procesos de Gestión de la Configuración de Software en el desarrollo de distribuciones GNU/Linux"

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores:

Gelsys Martínez Baeza

Yarelys Calderón Santos

Tutora:

Ing. Yusleydi Fernández del Monte

Ciudad de La Habana, 2011

"Año 52 de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y autorizamos al centro Geitel de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de junio del año 2011.

Gelsys Martínez Baeza

_____

Firma del autor

Yarelys Calderón Santos

_____

Firma del autor

Yusleydi Fernández del Monte

_____

Firma del tutor

Resumen

La Gestión de la Configuración de Software (GCS) es uno de los procesos clave para toda organización dedicada al desarrollo de sistemas informáticos, ya que posibilita una mejor organización del desarrollo y mantenimiento del producto, facilitando el resto del proceso de producción. Una correcta GCS en el desarrollo de cualquier empresa aumentaría la calidad de los productos, la satisfacción del cliente y en consecuencia la mejora de la organización, pero actualmente en el proyecto Nova no se encuentran definidos estos procesos, lo que puede traer consigo que se pierdan algunos componentes de software por no tenerlos identificados, que no se controlen los cambios no logrando un seguimiento de los mismos así como que no se garantice la evolución del proyecto. De esta problemática se deriva la necesidad elaborar un conjunto de procesos que describan la GCS en el proceso de desarrollo de distribuciones de GNU/Linux. Para la realización de este trabajo se utilizaron como métodos científicos el histórico-lógico, el analítico-sintético, la observación y la entrevista. El resultado de este trabajo describe la forma de llevar a cabo 6 procesos que garantizan una correcta GCS en proyectos que desarrollen distribuciones GNU/Linux. Cada uno de los procesos diseñados fue evaluado obteniéndose resultados satisfactorios.

PALABRAS CLAVES: control de cambios, gestión de la configuración, administración de la configuración, configuración de software.

Índice

Introducción

A pesar del bloqueo económico, comercial y financiero al que es sometido Cuba por parte del gobierno de los EE.UU, el país ha logrado desarrollarse modestamente en la industria del software y ha elaborado estrategias, políticas y formas de desarrollo que lo han impulsado a lograr posiciones competitivas en este medio. Desde la 10ma Convención y Feria Internacional Informática 2004, con la participación de alrededor de 1 600 delegados de 37 países, Cuba dio muestra de su imagen al mercado externo presentando de manera internacional la Industria Cubana del Software. A partir de este momento se hizo necesario aumentar tanto la producción nacional como la exportación de software. En función de este objetivo, se crea la Agencia de negocios para la exportación de software y la Universidad de las Ciencias Informáticas (UCI), la cual en la actualidad genera anualmente más de 150 millones de dólares por concepto de exportación de software.

Para lograr los objetivos propuestos en la UCI existe una dirección central de calidad llamada Calisoft, esta se encuentra dividida en diferentes áreas las cuales trazan los lineamientos de calidad que deben ser cumplidas por los proyectos de cada centro productivo. En cada centro productivo existe un asesor de la calidad, encargado de guiar a los administradores de la calidad de cada departamento que conforma el centro, y así velar que las políticas de calidad establecidas se cumplan en los proyectos como base del proceso. Hasta el momento todo este proceso conocido como gestión de la calidad, garantiza que cada producto que se libere cumpla con los estándares y normas requeridos, y que sean altamente competitivos en el mercado, aunque no es este el único proceso que puede alcanzar este objetivo.

Dentro de las buenas prácticas para llevar a cabo un Proceso de Desarrollo de Software (PDSW) exitoso trazadas por la metodología Rational Unified Process (RUP) se plantea que deben existir formas para controlar los inevitables cambios que sufren los software y el proceso capaz de realizar esta tarea es el Control de Cambios (CC) mediante la GCS. (EVA 2010)

El estándar ISO/IEC 12207 para procesos del ciclo de vida del software, establece el proceso de GCS como uno de los procesos de soporte del ciclo de vida, es decir, apoya a otro proceso como una parte integral pero con un propósito distinto, y contribuye al éxito y a la calidad del producto de software. Mientras que el estándar IEEE 1074-1995 lo establece como uno de los procesos integrales necesarios para completar exitosamente y con calidad las actividades del proyecto.(histaintl 2007)

Además, el modelo para la mejora y evaluación de procesos para el desarrollo, mantenimiento y operación de sistemas de software Capability Maturity Model Integration (CMMI o Integración de Modelos de Madurez de Capacidades) lo incluye dentro de sus 22 áreas de procesos.

La GCS tiene como objetivo el control de los cambios, lo que permite que los desarrolladores trabajen sobre la versión actualizada y garantiza la integridad de cada uno de los elementos de software; entiéndase como integridad que el producto satisfaga las necesidades del usuario, que cumpla con todos los requisitos de rendimiento y que se pueda trazar una línea de evolución desde que se concibe y durante todo su ciclo de vida. (Angélica de Antonio 2008)

El proyecto Nova, enmarcado en el centro productivo Geitel perteneciente a la facultad 1, se dedica a la elaboración de la distribución cubana GNU/Linux Nova para contribuir y apoyar la migración a tecnologías de Software Libre a la cual se enfrenta Cuba, como parte del proceso de Informatización de la Sociedad y la guía de migración a sistemas de software libre y de código abierto (FOSS por sus siglas en inglés), entre otras acciones para minimizar la dependencia tecnológica. (Donéstevez 2010)

Una correcta GCS en el proyecto Nova aumentaría la calidad de los productos, la satisfacción del cliente y en consecuencia la mejora de la organización, ya que maximizaría la productividad minimizando los errores. Actualmente en el proyecto Nova no se encuentra definidos los procesos que garanticen una correcta GCS, ni tan siquiera se conoce este proceso por su nombre, lo que puede afectar la calidad de los productos obtenidos en el proyecto. Aunque de manera inconsciente se lleva a cabo el control de versiones mediante la ayuda de herramientas automatizadas, las actividades que se realizan son deficientes pues no se utilizan los modelos, normas, estándares y guías basados en la experiencia de empresas y expertos en los proyectos de software, debido a que no existe la bibliografía que documente cómo se pueden adaptar los procesos de GCS existentes al desarrollo de distribuciones GNU/Linux. Esto puede traer consigo que se pierdan algunos componentes de software por no tenerlos identificados, que no se controlen los cambios no logrando un seguimiento de los mismos, que no se garantice la evolución del proyecto así como que la sustitución del personal que trabaja en la producción sea desfavorable pues no se tiene constancia de lo que se ha hecho.

Considerando la situación antes planteada se define como **problema científico** de esta investigación**:** ¿Cómo garantizar una correcta configuración de software en el proceso de desarrollo de distribuciones de GNU/Linux? Lo que enmarca el **objeto de estudio en:** los procesos de GCS en el desarrollo de software**,** centrando el **campo de acción** en los procesos de GCS en el desarrollo de distribuciones GNU/Linux**.**

Definiendo como **objetivo general** de la investigación**:** Elaborar un conjunto de procesos que describan la GCS en el proceso de desarrollo de distribuciones de GNU/Linux. Para dar cumplimiento al objetivo planteado se definen los siguientes **objetivos específicos:**

- Definir el marco teórico de la investigación.
- Diseñar los procesos de gestión de la configuración de software para el desarrollo de distribuciones GNU/Linux.
- Evaluar los procesos propuestos para verificar los resultados aplicándolos al proceso de desarrollo del proyecto Nova.

Tareas de la investigación:

1. Análisis de los procesos que componen la GCS para establecer un marco teórico.
2. Valoración de las principales tendencias de los procesos de GCS para caracterizar su estado actual.
3. Caracterización de la GCS para conocer cómo se lleva a cabo en el desarrollo de las distribuciones de GNU/Linux.
4. Elaboración de los procesos que deben ser ejecutados en el ciclo distribuciones de GNU/Linux para realizar una correcta GCS.
5. Evaluación de los procesos principales para corroborar los beneficios de los mismos, aplicándolos al proceso de desarrollo de Nova.

Idea a defender: Definiéndose correctamente los procesos de GCS en el proyecto Nova se puede garantizar la integridad de los elementos de configuración de software y mantener un mayor control sobre los cambios que se le realizan.

Durante la presente investigación se hace uso de los siguientes **métodos científicos:**

Teóricos:

Histórico–Lógico: Para la revisión bibliográfica de los colectivos de autores que llevan la vanguardia en el tema de GCS y para utilizar estos como punto de referencia y analizar los resultados alcanzados hasta la actualidad.

Analítico-Sintético: Este método permite el análisis de toda la información procesada referente a la GCS y la evaluación de las tendencias actuales en esta disciplina de control, descomponiendo el problema científico en elementos por separados, para luego sintetizarlos en los aspectos a tener en cuenta para una correcta GCS.

Empíricos:

Entrevista: Se realizaron entrevistas para tener en cuenta las experiencias que tienen algunos desarrolladores del tema y para conocer los criterios de los mismos respecto a la solución propuesta para el tema en cuestión.

Observación: Permite observar los problemas que existen en el proyecto Nova como consecuencia de la falta de un proceso que controle los cambios como lo es la GCS. La guía que se siguió para esta observación se encuentra en el **¡Error! No se encuentra el origen de la referencia.** de esta investigación.

El trabajo investigativo está estructurado en 3 capítulos:

Capítulo 1: Fundamentación Teórica de la GCS: El contenido de este capítulo está conformado por los principales aspectos y características de la GCS, así como sus principales componentes, la forma de implementarse y los factores necesarios a tener en cuenta para una correcta GCS en el PDSW.

Capítulo 2: Procesos de GCS en el desarrollo de distribuciones de GNU/Linux: Se diseñan los procesos de GCS para distribuciones GNU/Linux y la forma en la que se van a organizar dichos procesos.

Capítulo 3: Evaluación de los procesos de GCS: Se reflejan los resultados obtenidos después de ejecutar los procesos de GCS definidos, dejando claras las ventajas y desventajas de poner en práctica los mismos.

Capítulo 1

Fundamentación teórica de la GCS

En este capítulo se hace un análisis de los proceso que componen la GCS para lo cual se definen los principales conceptos que son usados en esta área de conocimiento, además se dan a conocer las principales tendencias de la GCS; se explica también cómo es visto este proceso por los diferentes modelos de calidad y todos los factores que se deben tener en cuenta para implementar una correcta GCS en cualquier empresa que se dedique a la producción de software.

1.1 Definiciones utilizadas en el área de conocimiento de la GCS

Han sido varios los ingenieros e investigadores que han dedicado su tiempo y esfuerzo a estudiar y fomentar su progreso en el proceso de la Ingeniería de Software. Son también diversos los criterios emitidos sobre la GCS, y aunque algunas definiciones sean sencilla y otras más completa ninguna discrepa de la otra, de manera general todas se centran en la organización y control del proceso de construcción del software.

Según Roger S. Pressman define la GCS como “un conjunto de actividades de seguimiento y control que se inician cuando comienza un proyecto de ingeniería del software y terminan solo cuando este se retira de operación.”

Reconoce la importancia del cumplimiento de estas actividades, cuando afirma: “Si no se controla el cambio, él toma el control y eso nunca es bueno. Es fácil que una corriente de cambios incontrolados convierta en caótico un proyecto de software bien implementado. Por esta razón, la gestión del cambio es una parte esencial de la buena gestión del proyecto y de una sólida práctica de ingeniería de software.” (Martínez 2010)

Una de las definiciones más utilizadas en el mundo es brindada por el Instituto de Ingenieros Eléctricos y Electrónicos (IEEE) en la cual establece que: “Gestión de Configuración es la disciplina que abarca todo el ciclo de vida de la producción de software y productos asociados. Específicamente, requiere de la identificación de los componentes a controlar y la estructura del producto, controla todos los cambios sobre los elementos y garantiza mecanismos para auditar todas las acciones.” (Software Engineering Standards Committee of the IEEE Computer Society 2002)

Existen muchas definiciones acerca de lo que es la GCS, pero como la definición más completa y concisa, se considera la de Ivar Jacobson, Martín Griss y Patrick Jonson en el libro "Software Reuse: Architecture, Process, and Organizartions for Busisness Success" en la que se plantea: "Gestión de Configuración: Proceso de soporte cuyo propósito es identificar, definir y almacenar en una línea base los elementos de software; controla los cambios, reporta y registra el estado de los elementos y de las solicitudes de cambio; asegura la completitud, consistencia y corrección de los elementos; controla, almacena, maneja y libera los elementos asociados al producto de software" (Martínez 2010)

El propósito de la GCS es establecer y mantener la integridad de los productos a través del ciclo de vida del proceso de desarrollo de software.

1.1.1 Conceptos generales concebidos en la configuración del software

Existen varios conceptos generales que son de gran importancia para poder entender cada uno de los procesos que conforman la GCS. Después de una **revisión** a uno o varios elementos de software puede obtenerse una nueva **versión** de un producto, una **variante** o establecerse una **línea base**, sobre la cual se mantiene un **control** y que debe estar almacenada en un **repositorio**. Todos estos son conceptos que se definen se deben tener en cuenta para una correcta administración de la configuración de software.

Elementos de Configuración de Software (ECS)

Se define como un elemento de configuración a una unidad física y/o lógica parte de un conjunto mayor de elementos, producida o adquirida, que por sus características es distinguible de las demás y cuya evolución interesa administrar. Pueden ser elementos de configuración en un proyecto de software: (histaintl 2007)

01. El plan de proyecto.
02. El plan de Gestión de Configuración.
03. El documento de definición de requerimientos.
04. Estándares de análisis, diseño, codificación, pruebas, y auditoria.
05. Documentos de análisis del sistema.
06. Documentos de diseño del sistema.
07. Prototipos.
08. Documentos de diseño de alto nivel.
09. Documentos de diseño de bajo nivel.
10. Especificaciones de prueba del sistema.

11. El plan de pruebas del sistema.
12. El Código fuente del programa.
13. Código objeto y ejecutable.
14. Especificaciones de pruebas de unidad.
15. Planes de pruebas de unidad.
16. Documentos de diseño de base de datos.
17. Datos de prueba.
18. Datos del proyecto.
19 .Manuales de usuario.

Versión: Una versión es una instancia de un elemento de configuración, en un momento dado del proceso de desarrollo, que es almacenada en un repositorio, y que puede ser recuperada en cualquier momento para su uso o modificación. El término se usa para señalar a un ECS que tiene un conjunto definido de características funcionales.(histaintl 2007)

Al grupo de versiones distintas que aparecen según transcurre el ciclo de vida del proyecto y de un producto en sí se denomina revisiones.

Revisión: Se define revisión como una versión que se construye sobre otra versión anterior. Este término generalmente se asocia a la noción de corrección de errores, esto es, hacer cambios a un programa, corrigiendo solo errores en el diseño lógico pero sin afectar las capacidades funcionales documentadas, dado que ningún requerimiento ha cambiado.(histaintl 2007)

Variantes: Se define variante como una versión que es una alternativa a otra versión. Las variantes pueden permitir a un ECS satisfacer requerimientos en conflicto. Una variante es una nueva versión de un elemento que es añadida a la configuración sin reemplazar a la versión anterior. (histaintl 2007)

Línea base (LB):

Una línea base es una especificación o producto revisado y aprobado formalmente, que sirve como base para el desarrollo posterior, y puede ser modificado solo a través de procedimientos formales de CC.

- ✓ Se definen como un punto del ciclo de vida del software en el cual se aplica el control de configuraciones a un elemento específico de la configuración.
- ✓ Es un concepto de GCS que ayuda a controlar los cambios sin impedir seriamente los cambios justificados.

- ✓ El IEEE (IEEE Std. No. 610.12-1990) define una línea base como: “Una especificación o producto que se ha revisado formalmente y se está de acuerdo con los resultados, y que a partir de ahí sirve como la base para el desarrollo posterior y que puede cambiarse sólo por medio de procedimientos formales de CC.”
- ✓ Antes que un ECS se convierta en línea base, es posible realizar el cambio rápida e informalmente. Sin embargo, una vez establecida una línea base, metafóricamente se pasa a través de una puerta giratoria de una sola dirección. Si los pasos sucesivos generan cambios en el documento después de una línea base, se requiere una revisión formal y una justificación de todas las modificaciones del documento. (histaintl 2007)

La **Figura 1** nos indica que cada documento que resulta del proceso de software, se convierte en Línea Base, una vez que dicho documento se haya revisado, corregido y aprobado.

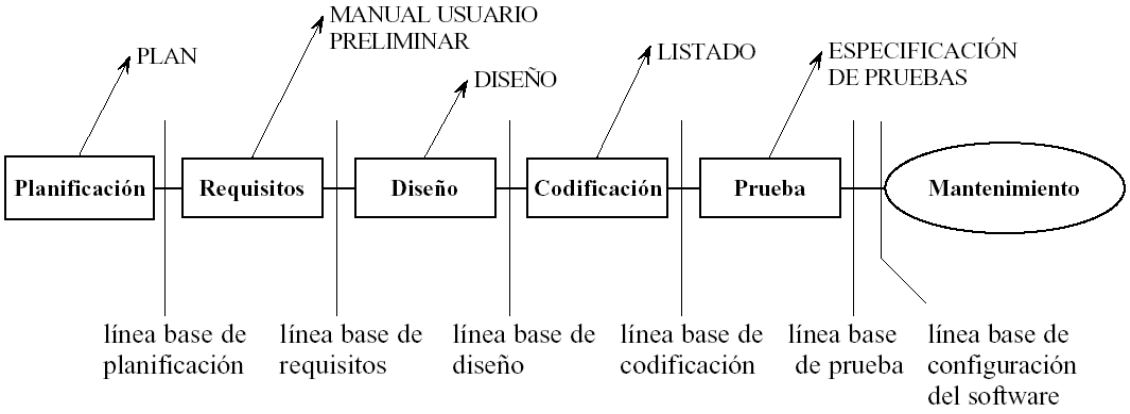


Fig 1. Definición de una línea base.

Pueden definirse diferentes líneas base como son:

- ✓ LB Funcional, que se establece al finalizar la fase de análisis y especificación de los requisitos del sistema, y comprende todos aquellos documentos en los que se define el problema a resolver, los costes del proyecto, el plan de tiempos, y los diferentes requisitos funcionales, de inter-operatividad y de interfaz del sistema.
- ✓ LB de Distribución o Asignación de funciones, que se establece al finalizar la fase de análisis y especificación de requisitos software, y comprende toda la documentación que gobernará el desarrollo de cada uno de los componentes software que se han identificado en la especificación

del sistema, y la asignación o reparto de las diferentes funciones entre los distintos componentes del sistema

- ✓ LB de Diseño Preliminar, se establece al finalizar la fase de diseño preliminar. Comprende todos aquellos documentos en los que se define la arquitectura del producto software, así como el Plan de Pruebas.
- ✓ LB de Diseño, esta se establece al finalizar la fase de diseño detallado. Comprende todos aquellos documentos que contienen el diseño detallado de la GCS y el plan de implementación, y también la descripción de los casos de prueba.
- ✓ LB de Producto, que se establece al finalizar la fase de pruebas. Comprende los programas creados y todos aquellos documentos que contienen la información relativa a las pruebas realizadas.
- ✓ LB de Operación, se establece al finalizar la fase de implantación. Comprende los manuales de usuario, guías de operación y mantenimiento, manuales de formación, etc.

(Angélica de Antonio 2008)

Control de Configuración: Una "configuración" es una combinación de versiones particulares de los componentes que forman un sistema consistente. Desde el punto de vista de evolución, es el conjunto de las versiones de los objetos componentes en un instante dado.(Dep. LSIIS - Unidad de programación 2010)

Control de versiones: Consiste en mantener un registro histórico de las diferentes versiones por las que pasan los componentes de un producto, que permita la recuperación de cualquiera de ellas.

El control de versiones combina procedimientos y herramientas para gestionar las versiones de los objetos de configuración creadas durante el proceso de ingeniería del software.(geronimo 2008)

Repositorio: El repositorio podría definirse como la base de datos fundamental para el diseño; no sólo guarda datos, sino también algoritmos de diseño y, en general, elementos de software necesarios para el trabajo de programación. Es un sitio centralizado donde se almacena y mantiene información, habitualmente bases de datos o archivos informáticos. Está preparado para distribuirse habitualmente sirviéndose de una red informática como Internet, Intranet o en un medio físico como un disco compacto. Es decir es el lugar donde comúnmente se almacenan los componentes de un sistema. El repositorio permite ahorrar espacio de almacenamiento, evitando guardar por duplicado elementos comunes a varias

versiones o configuraciones, igualmente facilita el almacenamiento de la información de la evolución del sistema. (Dep. LSIIS - Unidad de programación 2010)

Además está conformado por bibliotecas de software las cuales son una colección controlada de software y documentación asociada, estas bibliotecas constituyen el lugar donde se mantiene almacenado cada ECS para llevar a cabo el control de versiones. Estas pueden ser de diferentes tipos:

Biblioteca de trabajo: Se establece al inicio del proyecto, y comprende el área de trabajo donde los analistas y diseñadores elaboran los documentos del proyecto y donde los programadores desarrollan el software, es decir, donde se realiza la codificación y pruebas unitarias. Una vez se han completado las revisiones o pruebas y el elemento de configuración en cuestión ha sido revisado y aprobado, se inicia la transferencia del ECS a la Biblioteca de Soporte al Proyecto. En esta biblioteca el control de cambios es informal.

Biblioteca de Integración. Es de esta biblioteca de donde se toman los ECS para su integración en ECS de nivel superior del sistema.

Biblioteca de Soporte al Proyecto: En esta biblioteca se almacenan los ECS aprobados y transferidos desde la Biblioteca de Trabajo o desde la Biblioteca de Integración. Cuando un elemento pasa a esta biblioteca, se encuentra sujeto a un control de cambios interno y semi-informal.

Biblioteca de Producción: Está compuesta por la Biblioteca de trabajo, la de integración y la Biblioteca de Soporte al Proyecto.

Biblioteca maestra: Se usa para almacenar ECS liberados para su entrega al cliente o distribución en el mercado. Los elementos en la biblioteca maestra se encuentran sujetos a un control de cambios formal y estricto. Y normalmente esta biblioteca tiene fuertes restricciones de acceso para escritura, aunque normalmente no los tiene para lectura. En esta biblioteca se almacenan los release del sistema.

Repositorio de Software: Es la entidad en la que se archivan los ECS de un proyecto tras su cierre. Se transfieren a él desde la Biblioteca Maestra. Opcionalmente se puede identificar un segmento especial en el que se almacenarán los elementos reutilizables. Todo lo que se almacena en el repositorio debe estar adecuadamente identificado y catalogado, para facilitar su recuperación en caso de necesidad. Es central y común a todos los proyectos, mientras que la biblioteca de Producción y la Maestra son individuales para cada proyecto.

Biblioteca de Backup: También debe estar adecuadamente identificada, aunque su contenido no está sujeto a GCS.

(Angélica de Antonio 2008)

1.2 Administración de la configuración de software.

La administración de la configuración debe ser documentada desde el comienzo de cada proyecto con el objetivo de definir las políticas, estándares y procedimientos que se van a utilizar para gestionar la configuración en el transcurso de dicho proyecto. Todos estos aspectos se especifican en el epígrafe 9 del Plan de desarrollo de software y debe contener según el estándar IEEE los siguientes apartados:

1. Identificación de la Configuración.

   En esta actividad se define:

- ✓ Los componentes del software que constituyen ECS.
- ✓ La descripción del ECS.
- ✓ Desarrollador responsable del ECS.
- ✓ Esquema para etiquetado y numerado de documentos y componentes software.
- ✓ Cómo identificar las versiones y los entregables

2. Líneas base del proyecto.

Se creará una LB al final de cada fase del desarrollo y las características fundamentales y elementos de cada una de estas Líneas Base serán los siguientes:

- ✓ Nombre.
- ✓ Cuándo van a ser creadas.
- ✓ El propósito.
- ✓ Cuál será su contenido (especificar los ECS que contendrá).
- ✓ Fecha de establecimiento.
- ✓ Quién las autoriza y quién las verifica.

3. Almacenamiento, manipulación y entregables del proyecto.

Describe dónde y cuándo serán almacenados los entregables del proyecto.

Además se especificará:

- ✓ Código de release o variante.
- ✓ Fecha de liberación.
- ✓ Elementos de configuración que la componen.4. Control de la configuración.

4.1 Comité de Control de Cambios (CCC).

Roles por defecto del CCC: Jefe de Proyecto, Arquitecto de Software, Administrador de la configuración,

Administrador de la calidad.

4.2 Herramientas automatizadas para el Control de Cambios**.**

Describe las herramientas utilizadas para llevar a cabo el Control de Cambios

5. Estado de la configuración

5.1 Plan de Salvas

Para realizar las salvas y recuperación de datos del proyecto se deben realizar las tareas definidas en el Estándar de Configuración para ello.

5.2 Reportes

El proyecto puede obtener los reportes de estado de la configuración de los elementos de configuración, de las Líneas Bases y del Repositorio del proyecto como se explica en el Manual de la Herramienta.

6. Auditorías a la configuración

Número de auditorías a realizar y cuándo serán llevadas a cabo.

Se realiza una Auditoría a la Configuración al final de cada fase del ciclo de vida y para cada una de ellas se debe especificar:

- ✓ Qué se va a auditar.
- ✓ Resultados de la auditoría.
- ✓ Deficiencias detectadas.
- ✓ Plan de resolución de las deficiencias encontradas.
- ✓ Recomendaciones.

1.3 Principales tendencias de la GCS

Las empresas en la actualidad se encuentran dentro de un mundo y dentro de mercados altamente competitivos, en donde el valor agregado en los productos y servicios les da una gran ventaja.

Para dar este valor agregado muchas empresas optan por dar productos o servicios de alta calidad, y para lograr esto hacen grandes inversiones. (Scribd 2011)

Una de las variantes es introducir CMMI, un modelo de evaluación de los procesos de una empresa que en la actualidad es de los más populares y todas las empresas quieren alcanzar el nivel 2 que tiene como premisa que los procesos estén planeados, ejecutados, medidos y controlados para lograr que sus servicios o productos sean de alta calidad, para lo cual incluye dentro de sus áreas de procesos la GCS que puede ser sistematizada y automatizada, lo que se denomina un Sistema de Gestión de Configuración

(SGC). Actualmente existen en el mercado diversas herramientas que permiten apoyar una o más actividades de la Gestión de Configuración.

La empresa Hista Internacional se dedica a la distribución de herramientas de Rational Software Corporation, y a brindar servicios de aseguramiento de calidad con el soporte de dichas herramientas, además reúne conocimientos específicos en la disciplina de configuración del software y experiencia implementando y aplicando estos conceptos en diferentes organizaciones. (hista internacional 2010)

Estos conocimientos se complementan y enriquecen utilizando las herramientas específicas y adaptables para el proyecto y el cliente.

La suma de estos aspectos posibilita la implementación de una solución de gestión automatizada de Configuración del Software, integrando funciones y disciplinas a herramientas y procesos automáticos.

En nuestro país existen entidades que cumplen con todas las áreas de procesos claves del nivel 2 de CMMI, pero no están acreditadas oficialmente, teniendo en cuenta parámetros como control de versiones de programas y documentos, responsabilidades, alcances, bibliotecas de trabajo entre otros aspectos con el objetivo de organizar el funcionamiento de la institución. (Arias y Parra 2007)

DESOFT es una de las empresas dedicadas a la producción de software en Cuba, esta institución tiene un procedimiento nombrado Gestión de Configuración, mediante el cual se utilizan plantillas como lista de copias autorizadas, modelo de ingreso a configuración, nota de cambio, registro de cambios del proyecto, registros de cambios por soporte y requerimiento de cambio, estas sirven de apoyo a la hora de utilizar el procedimiento para controlar la administración de configuración y cambio. (Arias y Parra 2007)

El sistema de control de configuración establecido en la empresa si está acorde con las políticas de la organización. Para llevar a cabo la administración de la configuración y cambio se entrenan a los ingenieros de software, a los desarrolladores, es decir a todo el personal involucrado en la ejecución del sistema de gestión de configuración. (Arias y Parra 2007)

En la institución existen las autoridades responsables del control de la línea base, aunque dichas líneas bases no son auditadas. Además no se controla ni el mantenimiento ni la implementación del software. La correcta ejecución de los procedimientos definidos permite conocer en cualquier momento el estado de los elementos de configuración. No siempre el jefe de proyecto controla la ejecución del sistema de control de configuración, y al no realizar este control debidamente trae como consecuencia que el grupo de control de calidad no revise de forma periódica ni audite las actividades de dicho sistema. (Arias y Parra 2007)

La empresa se rige por el estándar ISO 9003 y CMMI, pues son los más conocidos y establecidos en la industria del software a nivel internacional. Actualmente la institución no utiliza ninguna herramienta para controlar la administración de configuración y cambio, se está probando el Sistema de Control de Versiones (CVS) para ver los resultados que trae para la entidad. (Arias y Parra 2007)

SOFTEL es también otra empresa dedicada al desarrollo de software. La compañía se rige por el estándar CMMI, ya que es el más empleado en la actualidad en este tipo de empresas. Se configuró RUP a las necesidades de la empresa. Tienen definidas personas que ocupan roles específicos como el administrador de configuración, el comité de control de configuración, integrador y revisores. Se tienen establecido líneas bases, específicamente la de producción y operación. Los entregables se almacenan en carpetas, para que sean usados por los que lo requieran, según política de seguridad. (Arias y Parra 2007)

En ninguna de las instituciones antes mencionadas se encuentran formalizados los aspectos que debe ser tratados en la gestión de la configuración software, pues son procesos jóvenes que aún no están institucionalizados, por lo que la identificación, control, auditoría y los reportes de estado de la configuración no se ejecutan según el Plan de Configuración para todos los proyectos por igual. Solo en el caso de **SOFTEL** se ha elaborado un procedimiento referente a la Gestión de Configuración, utilizan plantillas como el Plan de Gestión de Configuración, Orden de Trabajo y Solicitud de Cambio. (Arias y Parra 2007)

En la UCI actualmente no existe un procedimiento general para controlar la GCS a nivel de universidad. La dirección encargada de gestionar la calidad de software en la universidad, CALISOFT, tiene un área dedicada a este proceso en específico. Hasta el momento en muchos proyectos lo único que se utilizan son herramientas para llevar a cabo el control de versiones de software.

En el informe de la segunda auditoría realizada al proyecto Nova en noviembre del 2010 por el grupo Qalit se refleja que la forma de aplicar la GCS en este proyecto es regular ya que en muchos casos ni siquiera se conoce el proceso por su nombre. Además todavía no se lleva a cabo una correcta identificación de cuáles son los verdaderos ECS y no se logra dar respuesta a las siguientes interrogantes:

- ✓ ¿Qué componentes del software nos interesa ver cómo evoluciona y controlar sus cambios?
- ✓ ¿Qué impacto trae para el desarrollo hacer el cambio X?
- ✓ ¿Qué restricciones se ven afectadas al hacer el cambio X?
- ✓ ¿En este momento del desarrollo puedo hacer el cambio X?

- ✓ Si ya estamos en fases avanzadas del desarrollo del sistema ¿Puedo hacer de manera independiente los cambios que estimo?

- ✓ ¿Qué mecanismos existen para hacer un cambio determinado?

1.4 La GCS según los modelos de calidad

1.4.1 Modelo de Calidad de Software.

En el proceso de gestión de la calidad las actividades de control y aseguramiento de la calidad son muy importantes ya que garantizan que cada producto software que es desarrollado cumpla con la calidad requerida, tan importante para las empresas en la actualidad dado que la competencia cada día es más fuerte en la industria del software. Para llevar a cabo estas actividades es muy importante la aplicación de los modelos y estándares que nos ayudan a discutir, planificar y obtener las directrices para el aseguramiento interno y externo de la calidad.

Un modelo es "un conjunto de buenas prácticas para el ciclo de vida del software, enfocado en los procesos de gestión y desarrollo de proyectos," pero debemos tener en cuenta que estos modelos nos dicen qué hacer pero no cómo hacerlo, ya que esto depende de las metodologías de desarrollo con la que se esté trabajando y de los objetivos del negocio. (Quiñones 2006)

Dentro de los modelos, normas y estándares más conocidos para la gestión de la calidad de software se encuentran Capability Maturity Model Integration (CMMI), Organización Internacional de Normalización (ISO) y el Instituto de Ingenieros Eléctricos y Electrónicos (IEEE) respectivamente.

1.4.2 CMMI (Integración de Modelos de Madurez de Capacidades o Capability Maturity Model Integration)

CMMI es el encargado de evaluar los procesos de una organización y fue desarrollado inicialmente para los procesos de desarrollo e implementación de software de la Universidad Carnegie-Mellon para el Software Engineering Institute (SEI). En septiembre de 1987 fue publicada la primera definición de un modelo de madurez para un proceso de desarrollo de software, la cual fue desarrollada por petición del Gobierno Federal de los E.U, principalmente del Gobierno de Defensa. La última versión de este trabajo fue publicada en febrero de 1993 y lo constituye CMM o SW-CMM (CMM for Software), el cual se enfoca solamente en mejorar el software. Después de este modelo fueron desarrollados otros como por ejemplo el System Engineering Capability Model (SECM) del EIA´s (Escuela de Ingeniería de Antioquia) que se enfoca únicamente a la ingeniería de sistemas; debido a que los modelos existentes hasta el momento se

concentraban solo en una parte específica de los negocios, fortaleciendo las barreras que evitaban el mejoramiento de la empresa al no lograr una integración entre el software que se utilizaba y la ingeniería en sistemas.

El uso de varios de estos modelos para lograr un objetivo y la diferencia entre estos en cuanto metodología, arquitectura y contenido devino en una barrera para el uso de estos modelos. Para dar solución a estos problemas surge en diciembre del 2001 CMMI, un modelo para la mejora y evaluación de procesos para el desarrollo, mantenimiento y operación de sistemas de software. Consistente en las mejores prácticas que están dirigidas al desarrollo y mantenimiento del producto, desde su concepción, la entrega y el mantenimiento. Existe un énfasis tanto en la ingeniería en sistemas como en la ingeniería de software y en la integración necesaria para construir y mantener el producto total

Este modelo clasifica las empresas en niveles de madurez. Para esto está dividido en 5 niveles, los cuales están conformados por 22 áreas de proceso (en la que se incluye la GCS en el nivel 2) agrupadas en 4 categorías que sirven para conocer la madurez de los procesos que se realizan para producir software. Puede ser representado de dos formas, una de forma continua, la cual define capacidades para cada nivel y otra de forma escalonada. La estructura y las diferentes representaciones de CMMI pueden ser observadas en el **¡Error! No se encuentra el origen de la referencia.** de este trabajo.

CMMI plantea como propósito de la GCS establecer y mantener la integridad de los productos de software a lo largo de su ciclo de vida y resalta además su dimensión administrativa. Plantea también que el objetivo de la gestión de la configuración es establecer y mantener la integridad de los elementos de trabajo identificando, controlando y auditando dichos elementos.

Más concretamente mediante:

- ✓ La identificación de los elementos de trabajo que componen una línea base.
- ✓ Controlando los cambios de dichos elementos.
- ✓ Proporcionando formas de construir los elementos de trabajo a partir del sistema de control de la configuración.
- ✓ Mantener la integridad de las líneas base.
- ✓ Proporcionar información precisa de los datos de la configuración a desarrolladores y clientes.(Gracia 2003)

Para lo cuál es muy importante contar con un sistema de control y gestión de versiones.

Siendo responsable además de la administración y control de los ítems que conforman el proyecto, es el proceso más largo y solo culmina cuando el software es retirado de circulación, requiere una organización impecable de los componentes en desarrollo y acompañar el software de toda la documentación necesaria para seguir su evolución, cambios, mejoras.

1.4.3 Organización Internacional de Normalización o ISO

La ISO es una organización no gubernamental, aunque se encuentra conformada por delegaciones de 163 países que pueden ser gubernamentales y no gubernamentales "encargado de promover el desarrollo de normas internacionales de fabricación, comercio y comunicación para todas las ramas industriales" y "su función principal es la de buscar la estandarización de normas de productos y seguridad para las empresas u organizaciones a nivel internacional".

Dentro de todas las normas establecidas por esta organización se encuentra la familia ISO 9000, surgida en 1987 y cuya versión actual data del 2008, la cual establece un conjunto de normas sobre calidad y gestión continua de calidad, especificando la manera en que son aplicados los estándares de calidad, tiempos de entrega y niveles de servicio en una organización. Existen más de 20 elementos en los estándares de este ISO que se relacionan con la manera en que los sistemas operan."

Como consecuencia de que las normas ISO del 2004 fueron clasificadas como burocráticas, pues estaban destinadas principalmente a organizaciones que desarrollaban procesos productivos, en este año se logró una menos pesada que podía ser usada y aplicada por todo tipo de organizaciones incluyendo empresas de servicio e incluso en la administración pública.

En lo referente a la calidad dentro de esta familia de normas la norma que más se conoce y la más establecida es la ISO 9001-2008 Sistemas de Gestión de la Calidad - Requisitos, la cual es válida para cualquier organización que busque aumentar la productividad de su empresa y la satisfacción del cliente al incrementar la calidad de sus servicios, pues no solo establece normativas para la gestión de la calidad, sino también para cualquier sistema en general.

Otra norma vinculante a la anterior es la ISO 9004:2000 la cual impone las directrices para la mejora del desempeño.

El estándar ISO/IEC 12207 para procesos del ciclo de vida del software, establece el proceso de GCS como uno de los procesos de soporte del ciclo de vida que apoya a otro proceso como una parte integral, pero con un propósito distinto, y contribuye al éxito y a la calidad del proyecto de software.

Este proceso consta de las siguientes actividades:

1. Implementación del Proceso: Se desarrolla un Plan de Gestión de Configuración que describe las actividades de Gestión de Configuración, los procedimientos y el cronograma para su realización, y los responsables de dichas actividades. Dicho plan debe ser documentado e implementado.

2. Identificación de la Configuración: Se establece un esquema de identificación de los elementos de software y sus versiones a ser controlados por el proyecto.

3. Control de la Configuración: Se identifican y registran las solicitudes de cambio, se analiza y evalúa los cambios, se aprueba o rechaza la solicitud, se implementa, verifica y distribuye el elemento de software modificado.

4. **Contabilidad de Estado de la Configuración**: Se preparan registros de gestión y reportes de estado que muestren el estado e historia de los elementos de software controlados, incluyendo líneas base.

5. **Evaluación de la Configuración**: Se determina y asegura que los elementos de software sean funcionalmente y físicamente completos.

6. **Gestión de actualización y distribución**: Se controla formalmente la actualización y distribución de los productos de software.(histaintl 2007)

La norma ISO 10007, define el objetivo principal de la GCS como: documentar y proveer visibilidad de los productos de software y del estado de progreso en la satisfacción de los requerimientos funcionales y físicos.

1.4.4 IEEE (Institute of Electrical and Electronics Engineers)

La IEEE o el Instituto de Ingenieros Eléctricos y Electrónicos, como se conoce en español es uno de los estándares más importantes en el mundo.

A la hora de hablar de los estándares para la calidad de software es prácticamente imposible dejar de mencionar el estándar IEEE 730-1998 para el Plan de Aseguramiento de la Calidad de Software. Este aborda aspectos como la administración, la documentación, el control de código, etc. Dentro de la documentación le adjudica gran importancia al Plan para la GCS reflejado en el estándar IEEE 828-1998.

Dicho plan trata todo lo referente a la asignación de las responsabilidades, la identificación de las actividades que se realizarán durante todo el proceso, la identificación de la configuración, el reconocimiento de los elementos de configuración, el control de la configuración, el acceso a las bibliotecas, la aprobación o desaprobación de un cambio y la implementación del cambio de ser aprobado.

El estándar IEEE 1074-1995 para el desarrollo de procesos del ciclo de vida del software, establece el proceso de GCS como uno de los procesos integrales. Estos son los procesos necesarios para completar

exitosamente las actividades del proyecto, y son utilizados para asegurar la finalización y calidad del mismo. Este proceso se compone por las siguientes actividades(histaintl 2007):

1. Planificar la Gestión de Configuración.
2. Desarrollar la Identificación de la Configuración.
3. Realizar el Control de la Configuración.
4. Realizar la Contabilidad de Estado.

Por otra parte la norma IEEE 828-1998 para la GCS plantea que la actividad de planificación para la GCS es una disciplina de ingeniería que proporciona los métodos y herramientas para identificar y controlar el software a través de su desarrollo y utilización; constituyendo el medio por el cual se registran la integridad y la trazabilidad del sistema de software, durante el desarrollo y mantenimiento del mismo. Apoyando además la reducción del coste total del ciclo de vida del software, proporcionando una base para la medición del producto y proyecto.

Esta norma incluye dentro de las actividades de la GCS, la identificación y el establecimiento de líneas de base, la revisión, aprobación y control de cambios, el seguimiento y presentación de informes de tales cambios, las auditorías y revisiones del producto de software en evolución, y el control de la documentación de la interfaz y el proyecto.

LA GCS es considerada además normas de buena práctica para todos los proyectos de software debido a que mejora la fiabilidad y la calidad de software ya que:

- ✓ Proporciona una estructura para la identificación y control de documentación, código, interfaces y bases de datos durante todas las fases del ciclo de vida del software.
- ✓ Apoya la metodología de desarrollo elegida ajustándola a los requisitos, normas, políticas, organización y filosofía de gestión.
- ✓ Lleva a cabo la producción de la gestión e información del producto sobre la situación de las líneas de base, control de cambios, las pruebas, nuevos productos y las auditorías.

1.5 GCS según las metodologías tradicionales y ágiles

El desarrollo de software no es una tarea fácil. Por la cantidad de actividades que trae consigo este proceso, y la dependencia de estas del tipo de producto a construir, existen varias formas de guiar la realización de un software, una de las más completas son las metodologías, estas imponen un proceso disciplinado sobre el desarrollo de software con el fin de hacerlo más predecible y eficiente. Teniendo en cuenta para esto las tareas de la planificación.

Una metodología de desarrollo de software es un conjunto de pasos, procedimientos, técnicas y herramientas que ayudan a los desarrolladores a realizar un nuevo software. Es un marco de trabajo usado para estructurar, planificar y controlar el proceso de desarrollo en sistemas de información. La finalidad de una metodología de desarrollo es garantizar la eficacia logrando con ello el cumplimiento de los requisitos iniciales y minimizar las pérdidas de tiempo en el proceso de generación de un software.

Hoy, existen numerosas propuestas metodológicas que inciden en distintas dimensiones en el proceso de desarrollo. Un ejemplo de ellas son las metodologías tradicionales que están guiadas por una fuerte planificación durante todo el proceso de desarrollo; donde se realiza una intensa etapa de análisis y diseño antes de la construcción del sistema.

Estas han demostrado ser efectivas y necesarias en un gran número de proyectos, sobre todo aquellos proyectos de gran tamaño en los que el tiempo y los recursos son factores importantes.

Entre las metodologías tradicionales o pesadas se encuentran:

- ✓ RUP (Rational Unified Procces).
- ✓ MSF (Microsoft Solution Framework).
- ✓ Win-Win Spiral Model.
- ✓ Iconix.

Sin embargo la experiencia ha demostrado que las metodologías tradicionales no ofrecen una buena solución para proyectos donde el entorno es volátil y donde los requisitos no se conocen con exactitud, porque no están pensadas para trabajar con incertidumbre.

Aplicar metodologías tradicionales nos obliga a forzar a nuestro cliente a que tome la mayoría de las decisiones al principio. Luego el coste de cambio de una decisión tomada puede llegar a ser muy elevado si aplicamos metodologías tradicionales.

Aplicando metodologías tradicionales surgieron otras metodologías que tratan de adaptarse a la realidad del desarrollo de software. Estas son las metodologías ágiles que son métodos que buscan un justo medio entre ningún proceso y demasiados procesos, proporcionando simplemente los suficientes para que el esfuerzo valga la pena. Intenta evitar enfocarse en las personas y en los resultados como lo hacen las metodologías tradicionales. Además minimiza el riesgo, desarrollando el software en pequeñas iteraciones. Hace énfasis en la comunicación directa entre los desarrolladores, y no a través de documentos, por lo que produce muy poca documentación.

Ejemplos de metodologías ágiles:

- ✓ Extreme Programming (XP).
- ✓ Scrum.
- ✓ Familia de Metodologías Crystal o Crystal Meth Family.
- ✓ Open Source Software Development.

Veamos cómo es implementada la GCS por cada una de estas metodologías. A causa de la simplicidad con que lleva a cabo el desarrollo de proyectos, XP se ocupa muy ligeramente de asuntos que RUP cubre detalladamente, como la GCS.

Al comparar los artefactos generados en este flujo de trabajo por RUP, aún en proyectos de pequeñas dimensiones, supera en completitud a los que propone XP. Este último carece, a diferencia de RUP de roles específicamente diseñados para encargarse de la GCS como el Gestor de Configuración y el Gestor de Cambios.

No obstante es señalado que aunque parezcan ausentes en XP algunos conceptos y principios de la GCS, esto no es así, es que son llamados de maneras diferentes. Algunas de las prácticas de XP parecen peligrosas e imposibles, como son la propiedad colectiva de código y la entrega frecuente de liberaciones, pero aseguran ellos que son perfectamente practicables, su experiencia así lo demuestra.

En este punto no se pone en duda que puedan tener éxito los procesos de desarrollo guiados por metodologías ágiles, pero es indiscutible, que RUP es una metodología de reconocimiento internacional, probada por el éxito de muchas grandes compañías de desarrollo de software, que cuenta con mucha más y mejor definida documentación que los procesos ágiles. Es por esta razón y por todas las buenas prácticas que brinda RUP para la GCS que en esta investigación es la metodología que se sigue y los artefactos que se usan pertenecen a su expediente de proyecto.

RUP define como responsables de realizar las actividades de la GCS los siguientes roles:

<u>Gestor de configuración:</u>

- ✓ Gestionar la planificación, identificación, control, seguimiento y auditoría de todos los elementos de configuración en la base de datos de configuración.
- ✓ Desarrollar el plan de gestión de configuración.
- ✓ Promover el uso efectivo de la base de datos de configuración dentro de la organización.
- ✓ Monitorizar y reportar los cambios no autorizados sobre los ECS.

- ✓ Asegurar la consistencia e integridad de los datos de la base de datos de configuración a través de la ejecución de procedimientos de verificación y auditoría.
- ✓ Liderar las actividades de evaluación del proceso: revisar tipos de elementos de configuración, relaciones, atributos y valores asociados, estructura de la base de datos, derechos de acceso.
- ✓ Aprobar cambios estructurales en la base de datos de configuración.

Coordinador de configuración.

- ✓ Asegurar que todos los elementos de configuración están registrados de forma adecuada en la base de datos de configuración.
- ✓ Asegurar la consistencia e integridad de los datos de la base de datos de configuración y la estructura del sistema a través de la ejecución de procedimientos de verificación y auditoría.
- ✓ Reportar cualquier discrepancia o no conformidad en los elementos de configuración al gestor de configuración.
- ✓ Participar en la mejora continua del proceso de gestión de configuración.

Responsable de elementos de configuración:

- ✓ Asegurar que los elementos de configuración de los que es responsable están registrados en la base de datos de configuración con el estado y datos de configuración apropiados.
- ✓ Verificar que los cambios sobre los elementos de configuración siguen el proceso de cambios definido.
- ✓ Asegurar la idoneidad e integridad de los elementos de configuración de los que es responsable.
- ✓ Trabajar conjuntamente con el gestor de configuración para identificar las causas de cualquier discrepancia identificada en las auditorías e implementar las acciones correctivas.

Gestor de cambio:

- ✓ Evaluar el impacto y riesgo de los cambios.
- ✓ Asegurar que los responsables de los ECS actualizan los históricos de estos elementos con los cambios implementados.

De forma resumida algunas de las actividades que realiza la gestión de la configuración se presentan en la **Tabla #1:**

Tabla #1 Actividades que realiza la GCS

Actividades	Responsable	Descripción	Entradas	Salidas

Identificación de ECS	Gestor de configuración	Identificar ECS Crear estructura del directorio de GCS	Productos del proyecto	ECS identificados Líneas base Estructura del directorio de GCS
Mantenimiento y control de la GCS	Responsable del ECS	Control de cambios sobre ECS y líneas base Obtener aprobación de solicitudes de cambio sobre productos de trabajo de línea base	Peticiones de cambio	Registro de solicitud de cambio Solicitud de cambio aprobada Líneas base
Informe de estado de la configuración	Gestor de configuración	Mantener actualizado y publicar el estado de los ECS	ECS	Informe de estado de ECS
Verificación y auditoría	Gestor de configuración	Realizar auditorías de la gestión de configuración	Registros de la GCS Líneas base Registros de cambios	Informe de auditoría de GCS
Gestión del proceso de GCS.	Gestor de configuración	Documentar el PGCS.	Necesidades del proyecto. Plan de proyecto.	PGCS aprobado.

(Díaz y Escalona 2007)

1.6 Factores de éxito para lograr una correcta GCS

En un proceso de desarrollo de software la necesidad de realizar un cambio puede surgir en cualquier momento. Si estos son realizados sin planificación y previo análisis se puede crear una confusión entre los ingenieros que puede afectar la calidad del producto. Por lo que es muy importante una correcta GCS que se convierta en una actividad de auto protección durante todo el proceso de desarrollo de software. Para esto es necesario tener en cuenta algunos factores que definen cómo llevar a cabo con éxito esta actividad entre los cuales se encuentran:

- ✓ Debe existir una planificación, un PGCS: aprobado por todos los niveles de gestión, ampliamente difundido y claramente establecido.
- ✓ Todos los elementos constitutivos del proyecto (documentos, modelos, código fuente, librerías,) deben estar almacenados en un mismo repositorio.
- ✓ Deben hacerse copias de resguardo del repositorio periódicamente.
- ✓ Cada uno de los elementos mencionados deberá encontrarse bajo control de versiones.
- ✓ Para cada cambio en cualquiera de los elementos, deberá guardarse la fecha y hora del cambio y una breve descripción del mismo.
- ✓ Se deberán trazar líneas bases para marcar un conjunto de elementos en un determinado estado del proyecto (generalmente luego de finalizado un proceso durante el desarrollo del sistema).
- ✓ Se deberá poder retroceder a cualquier línea base recuperando todos los elementos del proyecto en el estado en el que estaban al momento de trazada la línea base.
- ✓ Se deberá poder efectuar una ramificación en los elementos del proyecto, con el fin de hacer desarrollo en paralelo de distintas alternativas (sobre todo durante el proceso de construcción del sistema).
- ✓ Se deberán poder obtener listados o reportes con información sobre los cambios en los elementos (elementos que sufrieron más cambios, cambios entre dos líneas base).

1.7 Herramientas para el control de versiones en la GCS y la gestión de proyecto

1.7.1 Herramientas para el control de versiones

Uno de los principales problemas que presentan aquellos proyectos en los que no se lleva a cabo correctamente la GCS es que no se mantiene un control sobre las versiones de los diferentes elementos de la configuración de un producto software, en este caso no solo del código de fuente sino además de cada uno de los artefactos que son identificados como ECS. Aunque este proceso se puede hacer de forma manual existen muchas herramientas que nos ayudan a realizar de forma exitosa esta gestión, permitiendo que en un momento determinado se pueda saber el estado en que se encuentra un producto

en un momento de su desarrollo y llevar además un registro histórico sobre cada uno de los cambios a los que es sometido.

Existen muchas alternativas libres de estas herramientas en el mercado que además de garantizar el control que se necesita brindan otras facilidades, entre ellas encontramos Subversion (SVN), REDMINE, GIT y otros.

Subversion

Subversion es un sistema de control de versiones que maneja los archivos y las carpetas de un proyecto y sus modificaciones en el transcurso del tiempo. Fue creado en el año 2000 en CollabNet Inc, con el objetivo de ser una alternativa real y mejorada a CVS (Concurrent Versions System, un sistema de control de versiones que era la solución en la época). SVN es libre y Open Source, ya que está distribuido bajo la licencia Apache. (WANdisco 2000)

Esta herramienta trabaja sobre el modelo cliente/servidor: donde uno o más clientes se conectan a un servidor central que tiene la última copia del proyecto y la de sus versiones anteriores, aunque se trabaja con la última versión, después de hacerse los cambios las copias son subidas al repositorio creándose una copia de la versión anterior.(WANdisco 2000)

Existen varias interfaces de Subversion, ya sean programas individuales como interfaces que lo integran en entornos de desarrollo (WANdisco 2000):

- ✓ TortoiseSVN. Provee integración con el explorador de Microsoft Windows. Es la interfaz más popular en este sistema operativo.
- ✓ Subclipse. Complemento que integra Subversion al entorno Eclipse.
- ✓ Subversive. Complemento alternativo para Eclipse.
- ✓ ViewVC. Interfaz web, que también trabaja delante de CVS.
- ✓ Para Mac OS X, pueden emplearse SvnX, RapidSVN y Zigversion.
- ✓ RapidSVN también corre en Linux.
- ✓ RabbitVCS, para el administrador de archivos Nautilus del escritorio GNOME, inspirado por TortoiseSVN.
- ✓ KDESvn. Provee integración con el entorno de escritorio KDE, muy parecido en aparencia/funcionamiento/características a TortoiseSVN.
- ✓ Easyeclipse, es un paquete basado en Eclipse, con algunos complementos de código abierto.
- ✓ sventon. Interfaz web.

- ✓ Versions. Interfaz de escritorio para Mac OS X.
- ✓ AnkhSVN. Extensión para Visual Studio, para las versiones 2002, 2003, 2005, 2008 y 2010, esta última en modo experimental.

GIT.

- ✓ GIT fue diseñado por Linus Torvalds, pensando en la eficiencia y confiabilidad de mantenimiento de versiones de aplicaciones con una enorme cantidad de archivos de código fuente.
- ✓ A diferencia de SVN esta más pensado en soportar muchas modificaciones independientes bajo la premisa que los códigos subidos no necesariamente son definitivos y/o estables.
- ✓ GIT también puede ser configurado en varias instancias de subida de código, generando todo un esquema de servidores remotos intermedios hasta llegar a producción.
- ✓ Se puede unir a TRAC.

(Azcárate 2008)

A diferencia de algunas herramientas como Subversion que mantienen el control del código de forma centralizada bajo la responsabilidad de un único usuario que debe aprobar cada decisión que es tomada en la administración de la herramienta reduciendo la flexibilidad, GIT mantiene un control distribuido, donde cada usuario tiene su repositorio y se puede cambiar y mezclar revisiones sobre ellos. Debido a esta característica se diferencia de otras herramientas en aspectos como en la conectividad de la red, permite hacer cambio sin molestar a nadie, es mucho mejor para trabajar solo convirtiendo las tareas grandes en pequeñas sub-tareas, elimina los problemas de gestión relacionados con los permisos y presenta menos problemas con el renombrado de ficheros y los directorios ocultos, por lo que es considerada una alternativa muy ventajosa para el control de versiones en el proceso de GCS.

1.7.2 Herramientas de gestión de proyecto

SVN+TRAC

- ✓ Trac es un sistema web libre para la gestión de proyectos y seguimiento de errores.
- ✓ Depende de SVN y permite analizar cambios en el código, ver los responsables de los mismos y los motivos por los cuales se hicieron los cambios.
- ✓ Cuenta además con un wiki (se podría utilizar para despliegue de procedimientos o gestión de documentación de acuerdos de reuniones) y gestión de tickets.
- ✓ Tiene algunos plug-ins para diferentes trabajos, mayor seguridad y hasta métricas, existen modificaciones que lo une al gestor de proyectos dotProject.

- ✓ Tanto TRAC como SVN sirven para diferentes lenguajes de programación. (Quiñones 2008)

REDMINE

REDMINE es una herramienta de gestión de proyectos software con interfaz web, Incluye un calendario y unos diagramas de Gantt para la representación visual de la línea del tiempo de los proyectos.(Lang 2006) Después de instalada se puede dar de alta los proyectos, los desarrolladores, jefes de proyecto y administradores. Además se puede obtener una lista de tareas al definirse los hitos del proyecto, las tareas a realizar en cada hito y el desarrollador responsable. Cada desarrollador tiene en su página principal una lista con las tareas asignadas de todos los proyectos y a medida que vaya trabajando en ellas puede ir registrando el tiempo que ha trabajado en ella y el porcentaje de la tarea que ha sido realizada.(Lang 2006)

Por supuesto, además de todo esto, brinda muchas más posibilidades, como (Lang 2006):

- ✓ Wiki por proyecto.
- ✓ Foro por proyecto.
- ✓ Envío automático de e-mail a los desarrolladores cada vez que se les asigna una tarea o ante cualquier evento relacionado con el proyecto.
- ✓ Posibilidad de subir ficheros y documentos, bien al proyecto, bien como adjuntos a las tareas y errores.
- ✓ Posibilidad de definir nuevos tipos de tareas y errores, con campos personalizados, todo ello fácilmente a través de la interfaz web. Estas tareas personalizadas y campos personalizados se asignan por proyecto, por lo que unos proyectos pueden tener algunas de esas tareas y campos y otros no.
- ✓ Se puede ver a través de *REDMINE* los cambios en el repositorio. Entiende CVS, Subversion y algunos de los sistemas de control de versiones más conocidos.
- ✓ Gráficos de Gannt, consultas por filtro con posibilidad de salvar dichas consultas, proyectos con sub-proyectos.

Es muy similar a Trac, pero con una administración e interfaz web más amigable, con menos tiempo en marcha y menos pluggins disponibles. (Lang 2006)

1.8 El proceso de desarrollo de software (PDSW) en distribuciones GNU/Linux

En el PDSW de cualquier proyecto, la GCS debe implementarse durante todo el ciclo de vida del producto. Las principales actividades que tienen lugar en el desarrollo de distribuciones de GNU/Linux durante las fases inicio, desarrollo, despliegue y transición trazados por la metodología RUP se explican a continuación.

Inicio

En la primera fase se desarrollan 4 actividades fundamentales. La primera de ellas es la selección del repositorio de código fuente para lo cual se hace un estudio de varias distribuciones de GNU/Linux con el objetivo de tomar como base la que esté en función de las necesidades del cliente, luego se selecciona la distribución GNU/Linux que se va a heredar, con lo cual queda definido la paquetería a utilizar. Otra de las actividades que se realiza es la determinación de las necesidades del cliente, que en este caso aunque se tienen en cuenta las de la distribución seleccionada para heredar, también tiene en cuenta las necesidades de las empresas en las cuales el sistema ya ha sido desplegado, ya que estas constituyen el cliente fundamental para los proyectos que desarrollan distribuciones de GNU/Linux. Además como resultado de esta actividad se obtiene un listado de requisitos de los desarrolladores propios, este último listado es obtenido de la definición de desarrollos propios la cual se encarga de la elaboración de las aplicaciones que contribuyen a limar los posibles obstáculos que pueda encontrarse la migración y que recoge las especificaciones no solo de los clientes sino también de los desarrolladores propios.

Construcción

En la fase de construcción la primera actividad para este proyecto es el desarrollo de los software propios definidos en la fase anterior y se basa en el desarrollo de un prototipo de herramienta que inicialmente resuelva en parte o completamente una necesidad planteada por el cliente o por el equipo de desarrollo y a partir de aquí se comienza a perfeccionar esta solución hasta lograr la más optima y eficiente para el cliente. Después de esto se diseña la distro seleccionada partiendo de los objetivos trazados inicialmente en los que se toman ideas, proposiciones, recomendaciones de la comunidad de SWL. A partir de esto se realizan cambios en el código fuente y luego se generan los parches que vendrían siendo estas nuevas modificaciones que se han realizado. La actividad de reempaquetamiento se encarga exactamente de reempaquetar estos nuevos parches obteniéndose así un paquete modificado, luego se compila cada paquete y se almacena el paquete binario obtenido de esta actividad en el repositorio de paquetes binarios. Si es necesario seleccionar paquetes para el sistema base se crea un listado de paquetes que

van a formar parte del sistema base. Para esto se toman los paquetes de la distribución padre , además de seleccionar aquellos que cumplen con las expectativas del cliente o a petición de este, independientemente de la distribución a la que pertenecen, y los creados por el proyecto; de lo contrario se pasa directamente a la generación de metapaquetes que no son más que paquetes dentro de otros. De no existir ningún error se elabora la base del sistema operativo al construir la imagen del mismo y se crea el sistema instalable; de existir algún error se regresa al repositorio de paquetes binarios.

Transición

Se despliega el sistema después de haber sido probado y corregido los errores

1.9 Consideraciones finales

Con la investigación realizada en este capítulo se arribaron a las siguientes conclusiones:

- ✓ La GCS es una disciplina de control indispensable en el desarrollo de cualquier proyecto informático para asegurar la calidad del producto final.
- ✓ Para su puesta en práctica es necesario conocer cómo está organizado el proyecto en el que se va a aplicar el proceso y la metodología de desarrollo por la que este se organiza.
- ✓ Una correcta GCS depende de la ejecución de cada una de las actividades que esta disciplina define para su puesta en práctica, del control por parte de los responsables de llevarla a cabo y la actualización mediante las herramientas automatizadas de cada uno de los artefactos involucrados en este proceso.
- ✓ Actualmente en el proyecto Nova no se encuentra definidos los procesos de GCS ni se utilizan los modelos, normas, estándares y guías basados en la experiencia de empresas y expertos en los proyectos de software, ya que estos no se encuentran definidos para establecer las pautas necesarias que garanticen una correcta GCS en el desarrollo de distribuciones de GNU/Linux.

Capítulo 2

Procesos de GCS en el desarrollo de distribuciones de GNU/Linux

En este capítulo se realiza una especificación de los procesos necesarios para llevar a cabo la GCS en cualquier proyecto que se dedique al desarrollo de distribuciones GNU/Linux, además de las actividades y subprocesos asociados a ellos, así como las entradas y salidas necesarias, y el responsable de garantizar que cada proceso se ejecute de forma correcta; con el objetivo de proveer una guía que ayude a garantizar una correcta GCS. Para ello se explican cada uno de los procesos que se ejecutan.

2.1 La GCS dentro del PDSW de las distribuciones de GNU/Linux

Para llevar a cabo una correcta GCS en el proceso de desarrollo de distribuciones GNU/Linux la primera actividad a realizar en el inicio del desarrollo de software es la identificación de aquellos componentes que pueden constituir ECS, para ellos se evalúan los siguientes:

- ✓ Artefactos generados de la ingeniería de software.
- ✓ El sistema que se seleccionó para heredar o los paquetes que lo componen que serán los que se modificarán hasta obtener el resultado final.
- ✓ Software propios.
- ✓ Distro.
- ✓ Listado de paquetes de sistema base.
- ✓ Personalizaciones (Metapaquetes).

Ya que estos son los que evolucionarán hasta obtener el producto final.

En la etapa de desarrollo es donde la GCS tiene que jugar un mayor papel pues se hace necesario implementar un buen proceso de control de cambios para garantizar que todas las modificaciones que se llevan a cabo sean realizadas de la forma correcta y por la persona calificada, aunque esta actividad tiene lugar en todas las etapas del desarrollo de software. Se hace necesario además mantener toda la información referente a los cambios almacenándola en los registros e informes que les permita a los desarrolladores saber qué es lo que se ha hecho y cómo se ha hecho para evitar problemas de comunicación y de desarrollo en el proyecto, de esto se encarga la generación de informes de estado que tiene lugar conjuntamente con cada una de las actividades de la GCS. Por otra parte es muy importante que antes de desplegar el sistema se realicen las revisiones y auditorías que comprueben que los

cambios realizados no afectaron en nada la integridad de los ECS y que el producto obtenido cumple con los requisitos establecidos por el cliente, por lo cual esta actividad se efectuará al final de cada fase.

La **Figura 2** muestra en qué etapas del PDSW de distribuciones de GNU/Linux tiene lugar cada una de las actividades de la GCS.

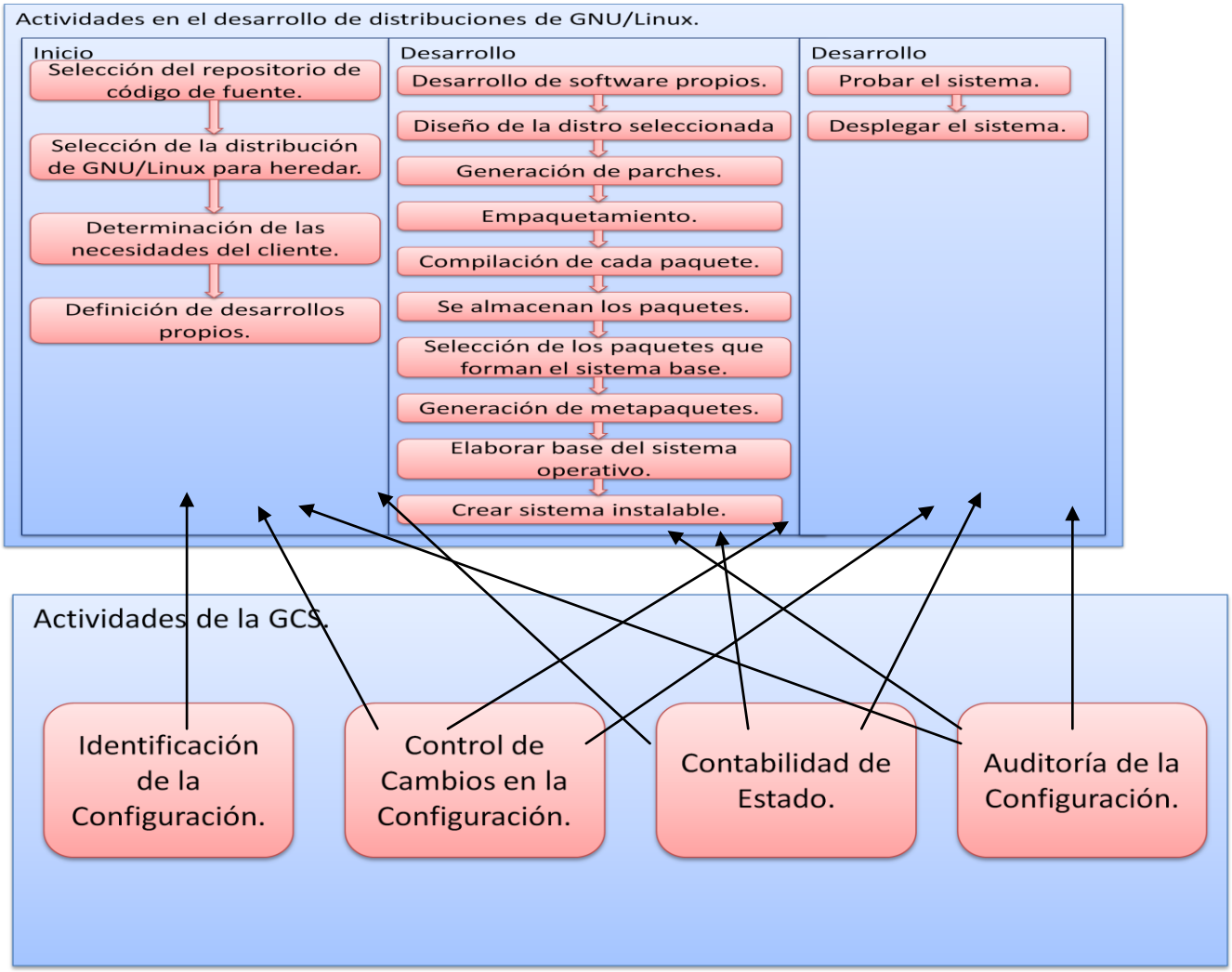


Fig 2. La GCS en el PDSW de distribuciones de GNU/Linux.

2.2 Definición de los procesos de GCS para el desarrollo de distribuciones de GNU/Linux

Definición de responsabilidades

Esta actividad se realiza con el objetivo de seleccionar a los responsables de gestionar la configuración de software en el proyecto. Para lo cual se conforma el **Grupo de Gestión de la Configuración** con los siguientes roles y responsabilidades:

Tabla #2 Grupo de GCS.

Rol	Responsabilidad
Responsable de la GCS.	Garantizar una GCS óptima.
Administrador de la configuración.	Velar por el cumplimiento de las pautas definidas en la GCS.
Gestor de Configuración.	Definir pautas para la GCS.
Administrador principal de la calidad de software.	Verificar que se definen y ejecutan correctamente los procesos de GCS.

Cada uno de los roles definidos en la tabla anterior será el encargado de llevar a cabo las actividades que se mustran en la **Tabla #3** lo especifican (El Laboratorio Nacional de calidad de Software 2008) en su "Guía Práctica de Gestión de la Configuración" y (Ramses 2011) **.**

Tabla #3 Roles y responsabilidades de la GCS.

Responsable de la GCS
✓ Asegurar que todos los elementos de configuración están registrados de forma adecuada en la base de datos de configuración. ✓ Asegurar la consistencia e integridad de los datos de la base de datos de configuración y la estructura del sistema a través de la ejecución de procedimientos de verificación y auditoría. ✓ Reportar cualquier discrepancia o no conformidad en los elementos de configuración al Gestor de configuración. ✓ Participar en la mejora continua del proceso de gestión de configuración.
Administrador de la configuración

- ✓ Planear junto al Jefe de Proyecto el proceso de Administración de la Configuración.
- ✓ Desplegar la Administración de la configuración.
- ✓ Identificar los elementos de configuración y Línea Base.
- ✓ Crear Línea Base.
- ✓ Revisar que las Solicitudes de Cambios que llegan al proyecto no se hayan emitido anteriormente por otros interesados.
- ✓ Agregar a la Solicitud de Cambio emitida anteriormente el nombre del interesado.
- ✓ Registrar las Solicitudes de Cambios.
- ✓ Actualizar el estado de las Solicitudes de Cambio a Registrada, Propuesta, Rechazada, Aprobada o Cerrada.
- ✓ Informar a los interesados acerca del estado de la solicitud de cambio.

Gestor de Configuración

- ✓ Gestionar la planificación, identificación, control, seguimiento y auditoría de todos los elementos de configuración en la base de datos de configuración.
- ✓ Promover el uso efectivo de la base de datos de configuración dentro de la organización.
- ✓ Monitorizar y reportar los cambios no autorizados sobre los elementos de configuración.
- ✓ Asegurar la consistencia e integridad de los datos de la base de datos de configuración a través de la ejecución de procedimientos de verificación y auditoría.
- ✓ Liderar las actividades de evaluación del proceso: revisar tipos de elementos de configuración, relaciones, atributos y valores asociados, estructura de la base de datos, derechos de acceso.
- ✓ Aprobar cambios estructurales en la base de datos de configuración.

Administrador principal de la calidad de software

- ✓ Revisar y aprobar ECS.
- ✓ Revisar y liberar Líneas Base.
- ✓ Revisar la correcta implementación de los cambios.
- ✓ Revisar la integridad de los ECS, de las líneas bases, y del repositorio.
- ✓ Asignar responsables de resolver las No Conformidades.
- ✓ Monitorear las No Conformidades hasta su cierre.

 **Responsable:** Líder del proyecto.

 **Salida:** Grupo de Gestión de la Configuración conformado.

 **Familiarización con la GCS**

Este proceso consiste en la capacitación de todos los desarrolladores en cuanto a la GCS, debido a que la correcta administración de este proceso depende de la colaboración de todo el equipo de desarrollo; y su conocimiento sobre el tema es fundamental para lograr establecer de manera exitosa el proceso de GCS. Para su realización cuenta con:

 **Responsable:** Grupo de Gestión de la Configuración.

Esta capacitación está enfocada a formar en cada uno de los desarrolladores las competencias necesarias para que cumplan correctamente con sus responsabilidades. Las actividades que deben poder llevar a cabo cada desarrollador para ser competente son las siguientes:

Competencia#1: "Identificar los elementos de la configuración, componentes y productos de trabajo relacionados."

Para esta competencia se deben tener las siguientes habilidades:

- ✓ Seleccionar los elementos de la configuración y los productos de trabajo que los componen, basándose en criterios documentados.
- ✓ Definir un identificador para cada elemento de configuración que facilita la localización de la información en el repositorio y evita la duplicación de información.
- ✓ Especificar las características importantes de cada elemento de configuración.

Competencia #2: "Definir un identificador para cada elemento de configuración que facilita la localización de la información en el repositorio y evita la duplicación de información."

En esta competencia se verán:

- ✓ Los mecanismos para administrar los niveles de control de la administración de la configuración.
- ✓ Los elementos de configuración elaborados durante el desarrollo del proyecto en un sistema de administración de la configuración y si están debidamente almacenados.
- ✓ Almacenar y archivar las versiones de los elementos de la configuración.
- ✓ Recuperar y almacenar los registros de la administración de la configuración y mantener los mismos actualizados para sus posteriores usos.

Competencia #3: "Crear y liberar líneas bases para uso interno y para el cliente."

En esta competencia se verá:

- ✓ Cómo está documentado el conjunto de elementos de configuración que conforman la línea base.
- ✓ Cómo saber cuando está disponible el conjunto de líneas base.

Competencia #4: "Dar seguimiento a las solicitudes de cambios y controlar los mismos en la configuración."

En esta competencia se verá:

- ✓ Registrar las solicitudes de cambios en la base de datos correspondientes.
- ✓ El análisis del posible impacto de los cambios y correcciones propuestas en las solicitudes de cambio determinando si pueden realizarse sin perjudicar el desarrollo del proyecto.
- ✓ Revisar las solicitudes de cambios para verificar si están correctamente elaboradas y si pueden ser aceptadas o no.
- ✓ Seguir al status de las solicitudes de cambio hasta que culmine.
- ✓ Controlar los cambios a los elementos de la configuración a lo largo de la vida del producto.
- ✓ Registrar las entradas y salidas de los elementos de la configuración en el sistema de administración de la configuración para la incorporación de los cambios.
- ✓ Realizar revisiones para garantizar que los cambios no han producido efectos no deseados sobre la línea base.
- ✓ Mantener registrados todos los cambios realizados a los elementos de la configuración y las razones de los mismos para posteriores trabajos durante el desarrollo del proyecto.

Competencia #5: “Establecer y mantener registros que describen lo elementos de la configuración que permitan la integridad de la línea base.”

En esta competencia se debe:

- ✓ Registrar las acciones de la administración de la configuración con el suficiente detalle para permitir que se conozca el contenido y status de cada elemento de la configuración y se puedan recuperar versiones anteriores.
- ✓ Identificar cuál es la versión de los elementos de configuración que constituyen en particular una línea base.
- ✓ Documentar las diferencias existentes entre las líneas bases.
- ✓ Revisar el estado, los cambios y otras acciones de cada elemento de configuración para mantener el control sobre cada proceso.

Competencia #6: “Ejecutar auditorías a la configuración para mantener la integridad de las líneas base de la configuración.”

En esta competencia se verá:

- ✓ Evaluar periódicamente la línea base para determinar si cumple con todos los requisitos para los que fue definida.
- ✓ Revisar la estructura y la integridad de los elementos en el sistema de gestión de configuración para su posterior confirmación.
- ✓ Confirmar que los registros de administración de la configuración identifiquen correctamente los elementos de configuración.
- ✓ Confirmar el cumplimiento a los estándares y procedimientos de la administración de la configuración.

Estas competencias serán formadas en los desarrolladores mediante un conjunto de actividades que el proyecto planifica como son:

 Conferencias.

 Talleres.

 Clases prácticas.

 Resolución de casos de estudio.

 Seminarios.

 **Identificación de la Configuración**

Este proceso es el encargado de seleccionar los artefactos que constituirán ECS, definiendo para cada uno de ellos una etiqueta que permita identificarlos de forma uniforme, además establece la forma en la que estos serán almacenados y accedidos. Para su realización cuenta con:

 **Entrada:** Productos del proyecto.

 **Responsable:** Gestor de la Configuración.

 **Salida:** ECS, Líneas base, estructura del directorio de GCS.

Para la realización de este proceso es preciso darle cumplimiento a las siguientes actividades:

 **Selección de los ECS.**

En esta actividad se determina qué se va a considerar como un ECS y se seleccionan cuales son necesarios gestionar debido a que durante el PDSW pueden verse sometidos a cambios que son necesarios controlar.

Hay que tener presente que no solo los documentos asociados al proceso de desarrollo o aquellos artefactos generados de la Ingeniería de Software son nombrados como ECS. El sistema o las partes del sistema, si se quiere dividir porque su tamaño y complejidad así lo requieren, también constituyen ECS. Además no todos los artefactos producidos durante el proceso de ingeniería de software constituyen ECS, sino solo aquellos que evolucionan y cuyos cambios se hacen necesario gestionar.

Para seleccionar los ECS hay que tener en cuenta varios criterios como son:

- ✓ La complejidad o singularidad del ECS.
- ✓ El número de personas implicadas en su mantenimiento.
- ✓ Si el elemento puede ser reutilizado o no.

- ✓ La influencia de un fallo sobre él.
- ✓ El número de elementos que lo utilizan.

Definición de un esquema de identificación

Esta actividad establece una forma de identificar uniformemente cada ECS asignándole una etiqueta que sea capaz de brindar la siguiente información:

- ✓ Número o código del ECS.
- ✓ Nombre del ECS.
- ✓ Descripción del ECS.
- ✓ Identificación del proyecto al que pertenece el ECS.
- ✓ Identificación de la línea base a la que pertenece.
- ✓ Identificación de la fase y flujo de trabajo en la que se creó.
- ✓ Tipo de elemento (documento o programa).
- ✓ Número de la versión.

Para los documentos la sintaxis de la nomenclatura es la siguiente:

[**Proyecto**]_**"Código del Subproyecto"_"Código del Software"**_**"Código del Artefacto"**_ **"Código de la Línea Base".extensión**

Para los software la sintaxis de la nomenclatura será la siguiente:

[**Proyecto**]_**"Código del Subproyecto"_"Código del Software"**_**"Código del Subsistema"**_ **"Código de la Línea Base"**

Cada uno de estos campos está definido de la siguiente manera:

Proyecto: Este campo es opcional, ya que solo es necesario especificarlo en el caso de que administre la GCS de manera global en varios proyectos.

Código del Software: Identificación del producto software que se está desarrollando.

Código del subproyecto: Identificación de la línea de desarrollo a la que pertenece el ECS.

Código del Artefacto: Código del Flujo de Trabajo + No. Consecutivo.

Código del Flujo de Trabajo: Un código que identifique el flujo de trabajo.

No. Consecutivo: Depende del orden en el que se enumeren los artefactos en este flujo de trabajo.

Código del Subsistema: Un código que identifique el subsistema en el que se está trabajando o puede identificar también a un componente dentro de ese subsistema, en este caso se especificaría después del subsistema el componente con una letra C y un número consecutivo.

Código de la Línea Base: En este campo se especifica el código de la línea base en la que el ECS fue revisado, corregido y aprobado.

Ninguno de estos códigos puede exceder los tres caracteres para permitir un mayor entendimiento del sistema de identificación.

Al mismo tiempo cada ECS lleva implícito la versión del artefacto en la que se está trabajando. Está versión está identificada con el par **X, Y**, donde **X** especifica el número de la versión del software y **Y** la versión del artefacto.

 **Definición y establecimientos de las líneas base.**

Las líneas base son establecidas al final de cada fase y en dependencia de las necesidades del proyecto y la metodología de desarrollo es definido el tipo de LB que será establecida.

Para las líneas base también son identificadas de manera uniforme, en este caso la nomenclatura será la siguiente:

LB+#_Código de la etapa.

#: El número de la línea base de acuerdo al orden en que fue establecida.

Código de la etapa: Este código es la identificación de la fase de trabajo en la que es establecida la línea base.

Además se identifica para cada línea base los ECS que están presentes en ella**.**

 **Definición y establecimiento de las bibliotecas de software**

 **Selección del tipo de biblioteca a utilizar**

En esta actividad se diseña la estructura de directorio que va a tener el repositorio de GCS teniendo en cuenta las bibliotecas por las que este va a estar conformado.

Estas bibliotecas son muy útiles para almacenar de una manera centralizada todos los elementos de un mismo sistema así como sus versiones, permitiendo siempre que se trabaje con la versión más actualizada y evitando que se dupliquen los mismos elementos o las mismas versiones o configuraciones.
La selección del tipo de biblioteca a utilizar depende únicamente de las características y necesidades del proyecto.

Selección del bibliotecario

Independientemente del tipo de biblioteca seleccionada selecciona además el desarrollador encargado de definir los mecanismos para establecer estas bibliotecas, la manera en la que se van a almacenar los elementos y los permisos para acceder a ella y los mecanismos de respaldo para salvar la información que se almacene en el repositorio. El encargado de todas estas actividades se desempeñará bajo el rol de bibliotecario.
Para la administración tanto del repositorio de código de fuente como el de la documentación se utilizada cualquiera de las herramientas descritas en el epígrafe 1.7 del capítulo anterior.
En las **Figuras 3** y **4** se muestra de forma gráfica cada una de las actividades definidas para darle cumplimiento a este proceso así como sus entradas y salida y el responsable de llevarla a cabo.

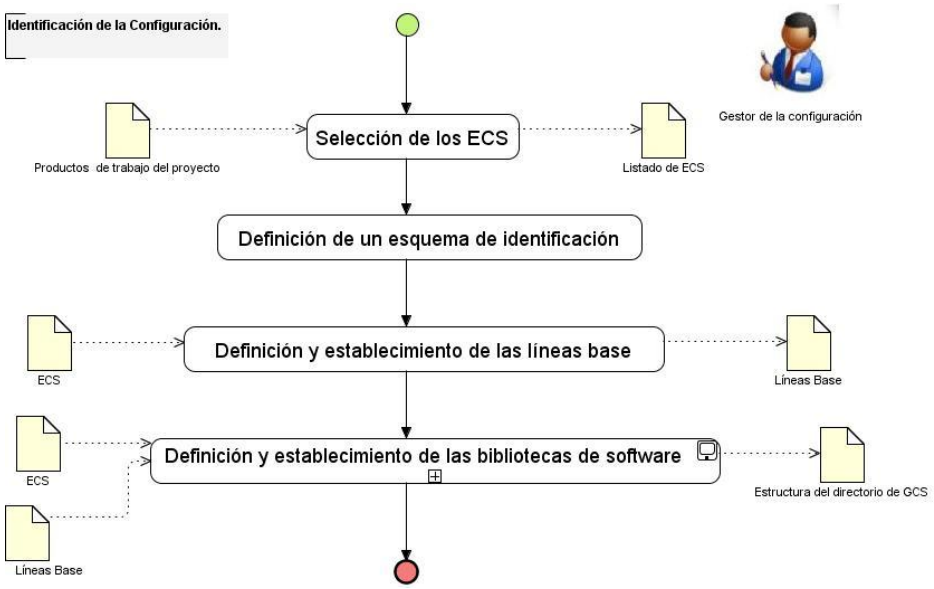


Fig 3. Modelo del proceso de Identificación de la Configuración.

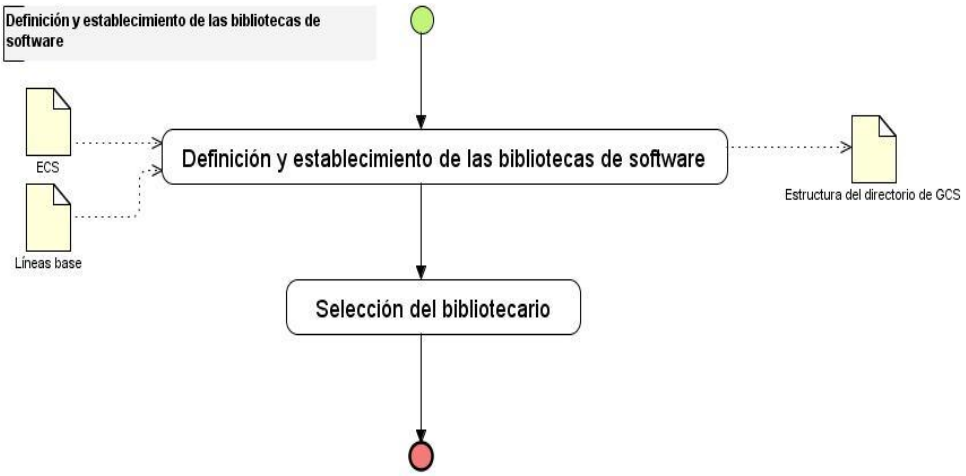


Fig 4. Modelo del Subproceso de Definición y establecimientos de las bibliotecas de software.

Control de Cambios en la Configuración

Mediante este proceso se controlan todas las versiones y entregas del producto que se está desarrollando y los cambios a los que es sometido durante su ciclo de vida, para ello se define la manera en la que se implementará el proceso de solicitud de cambios y los responsables de llevarlo a cabo.

Para su realización cuenta con:

Entrada: Solicitudes de cambio

Responsable: Responsable del ECS a modificar.

Salida: SC, SC aprobada, Líneas base actualizadas después del cambio.

Este proceso define 2 actividades fundamentales para llevarse a cabo:

1- Crear el CCC.

2- Definir y establecer el proceso de solicitud de cambio.

Creación del CCC.

El CCC es el encargado de velar porque el proceso de cambios se lleve a cabo de manera correcta, de aprobar y controlar los cambios que se van a realizar sobre un ECS por lo que dentro de sus responsabilidades están:

- ✓ Administra las configuraciones de software.
- ✓ Recibe una solicitud de cambio.
- ✓ Evalúa el cambio y los impactos que va a causar.
- ✓ Define los elementos de configuración a extraer.
- ✓ Asigna el trabajador que lo va a ejecutar y al probador que será el encargado de probar el cambio.
- ✓ Recibe el cambio realizado y verificado y conforma la nueva configuración.

Este comité está integrado por todos los líderes del proyecto, el administrador de la calidad y el responsable de la GCS.

 **Definir y establecer el proceso de solicitud de cambio.**

Es necesario tener en cuenta que se pueden establecer 3 niveles de control de cambios:

Control de cambios informal: Son los cambios justificados que puede realizar el desarrollador de un ECS antes de que este forme parte de una línea base.

Control de cambios a nivel de proyecto o semi- informal: Son los cambios que se le realizan a los ECS cuando forman parte de una línea base y para los cuáles es necesario contar con la autorización del director del proyecto y del CCC.

Control de cambios formal: Son los cambios que se realizan sobre los ECS una vez que se haya comenzado a comercializar el producto y debe contar siempre con la aprobación del CCC.

Para el control de cambios informal no existe ningún mecanismo definido pero si hay algunos estándares como el IEEE STD 1042 Guide to Software Configuration Management que brinda algunas recomendaciones para llevar a cabo este proceso.

El control de cambios formales se puede definir de muchas formas pero las actividades típicas siempre son las siguientes:

 Iniciación del cambio

El cliente o desarrollador después de reconocer la necesidad de un cambio que pudo ser dada por una modificación en los requisitos que supone una mejora, o por la corrección de un defecto, presenta una Solicitud de Cambio (SC) al CCC y un Informe de Cambio (IC) que aunque no forma parte de los

artefactos definidos en el expediente de proyecto[1], si fue incluido en el mismo pues se considera necesario.

La SC que se presenta es un formulario que muestra información sobre quién solicita el cambio, por qué se solicita y qué es lo que se va a cambiar, una descripción del problema para poder recomendar una solución, además de especificar los ECS que se verán afectados con el cambio, así como si se aprobó el cambio (En estos momentos el estado de la SC es ***Presentada.***).

Por su parte el IC muestra el resultado del esfuerzo técnico que se debe aplicar, los posibles efectos secundarios, el impacto global sobre otras funciones del sistema y el coste estimado del cambio.

 El CCC decide.

Para decidir si se aprueba o rechaza la SC el CCC tiene en cuenta algunos criterios como son:

- ✓ Valor del cambio para el proyecto/organización
- ✓ Tamaño
- ✓ Complejidad
- ✓ impacto sobre el rendimiento del producto (uso de memoria y CPU)
- ✓ recursos disponibles para efectuar el cambio (humanos y materiales)
- ✓ relación con otros cambios ya aprobados y en progreso
- ✓ tiempo estimado para completar el cambio

El CCC evalúa si el cambio es beneficioso o no, teniendo en cuenta los datos presentados en el IC. Si el CCC decide que el cambio no procede se le informa al cliente y se modifica el estado de la SC a ***Rechazada***. Si el cambio es aceptado el IC se convierte en una Orden de Cambio (OC) que describe además el cambio que se va a realizar, las restricciones que se deben respetar y los criterios de revisión y auditoría y el estado de la SC pasa a ser ***Aprobada***.

 Asignación personalizada a los objetos de configuración.

[1] Todos los artefactos que se utilizan pertenecen al expediente de proyecto definido por el proceso de mejora en su versión 3.3

Esta OC es recibida por el líder del proyecto o el jefe del módulo al cual pertenece el ECS que será cambiado, el cual decide quién del equipo de desarrollo es el encargado de efectuar el cambio y se lo informa al Gestor de Configuración quien es el encargado de informar y documentar el control de cambios. En este momento el ECS que se va a modificar se elimina de la biblioteca.

 Se realiza el cambio y se cambia el estado de la SC a ***Cerrada***.

 Revisión del cambio.

Se audita el cambio para comprobar que este se realizó correctamente y que el problema detectado ha sido corregido y con ello se han satisfecho los requisitos modificados, además se verifica que el cambio no traiga consigo otros problemas.

 El ECS modificado se reincorpora a la biblioteca del proyecto.

 Se actualizan las líneas base integrando el cambio a la línea base del proyecto.

 Se auditan todos los ECS afectados por el cambio.

 Se reconstruye la versión adecuada del software.

 Se notifica el cambio distribuyendo la nueva versión del software.

En un proceso semi-formal, puede suprimirse la necesidad de generar la solicitud de cambio, el informe de cambio y la orden de cambio, pero sí debe realizarse la evaluación del cambio y su seguimiento.

Debido a que esta actividad necesita de otras actividades para lograr realizarse de manera íntegra es considerada un subproceso dentro del proceso de Control de Cambios en la Configuración.

En las **Figuras 5,6** y **7** se muestra de forma gráfica cada una de las actividades definidas para darle cumplimiento a este proceso, así como sus entradas y salidas y el responsable de llevarla a cabo.

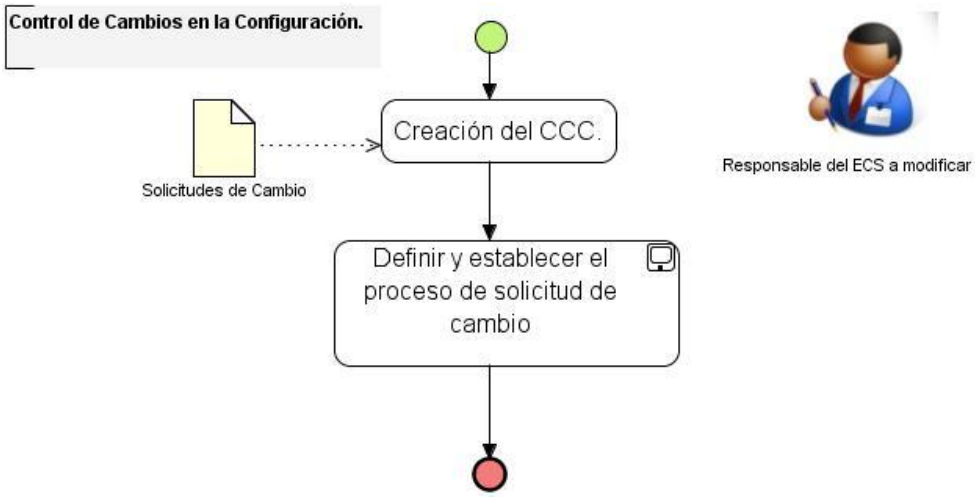


Fig 5. Modelo del proceso de Control de Cambios en la Configuración.

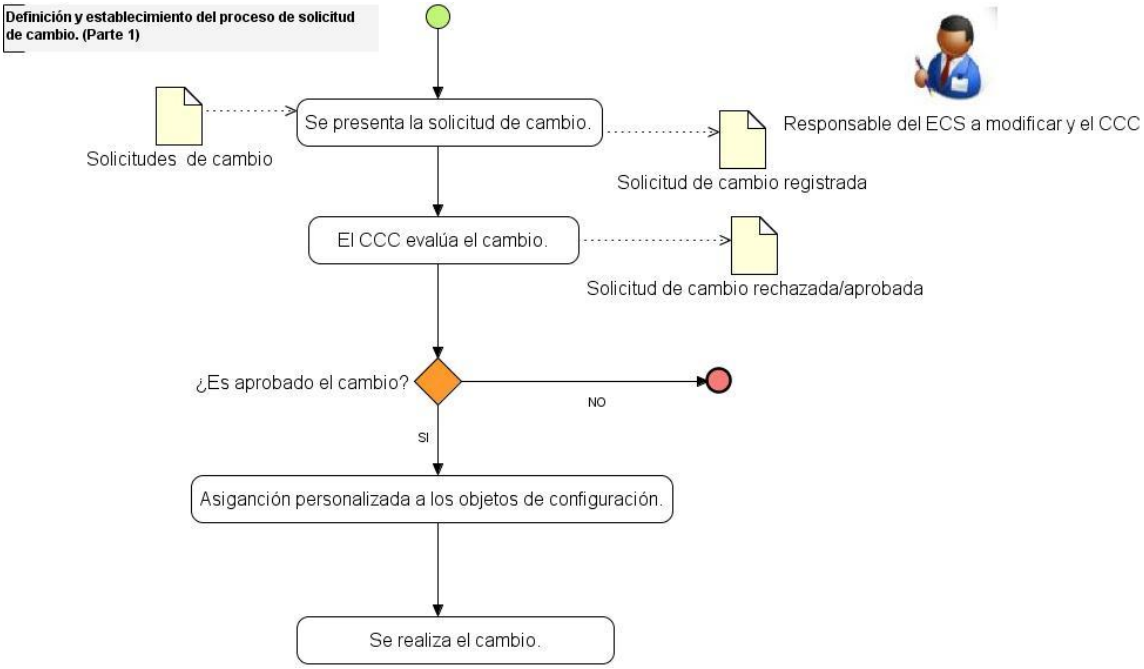


Fig 6. Modelo del subproceso de Definir y Establecer el Proceso de Solicitud de Cambios. (Parte 1)

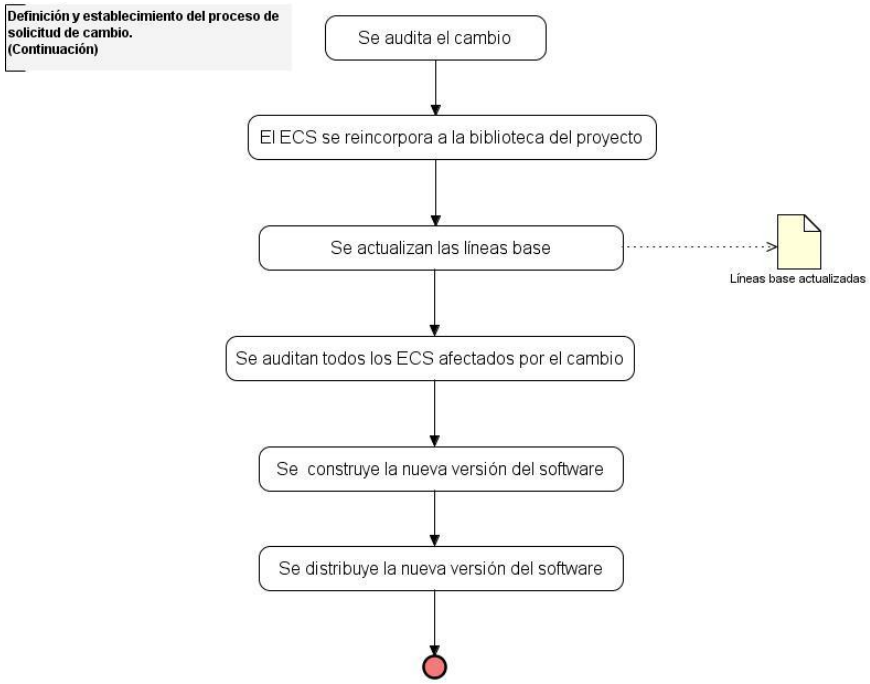


Fig 7. Modelo del subproceso de Definir y Establecer el Proceso de Solicitud de Cambios. (Parte 2)

Generación de informes de estado

Esta tarea es también conocida como Contabilidad de Estado y ayuda a minimizar los problemas de comunicación que pueden existir en el proyecto, ya que mantiene informado a los usuarios y desarrolladores sobre el estado de la configuración dando respuesta a preguntas como ¿Qué ocurrió? y ¿ Cuándo ocurrió?. Para su realización cuenta con:

Entrada: ECS.

Responsable: Gestor de Configuración.

Salida: Informe de estados de ECS, Artefactos de la GCS.

Este proceso consiste en registrar toda la información necesaria sobre lo que va ocurriendo en el proyecto para generar los registros pertinentes. Por esta razón se hace necesaria la realización de tres actividades fundamentales:

Captura de Información.

 Almacenamiento de la Información.

 Generación de Informes.

La información que se va a capturar proviene de todas las actividades que se realizan para garantizar una correcta GCS en el proyecto. Aunque la cantidad y el tipo de información a capturar dependen de las características del proyecto mientras más detallada y precisa sea mayor utilidad tiene para el equipo de desarrollo.

Debido a esto todo lo que se realice en el proyecto es recogido en diferentes registros para generar los informes necesarios.

En la **Tabla #4** se muestran algunos de los artefactos recogidos en el expediente de proyecto, definidos para documentar el proceso de GCS y otros que fueron necesarios crear con el objetivo de registrar toda la información necesaria para generar los informes que establece este proceso; debido a que se le hicieron algunos cambios (los cuales se encuentran señalados de color rojo) estos artefactos aparecen en los anexos[2] de este trabajo. Para ser consultados en su totalidad se puede acceder al expediente de proyecto.

Tabla #4 Registro de las informaciones en los artefactos del expediente de proyecto.

Artefactos	Información que registra
Plan de desarrollo de software (epígrafe Administración de la configuración) **(Ver ¡Error! No se encuentra el origen de la referencia.)**	Registra todo lo concerniente a los ECS, las líneas base y los release y variantes, demás de los resultados de la auditoría.
Plantillas de SC **(**Ver **¡Error! No se encuentra el origen de la referencia.** y **¡Error! No se encuentra el origen de la referencia.)**	Tiene todo lo que es necesario conocer acerca de las solicitudes de cambio.
Plantilla IC **(**Ver **¡Error! No se encuentra el origen de la referencia.)**	Registra toda la información sobre los cambios en general.
Registro de modificaciones del código	Contiene registro de las informaciones

[2] En el **¡Error! No se encuentra el origen de la referencia.**se puede observar la ubicación de cada uno de estos artefactos dentro del expediente de proyecto señalando en color rojo aquellos no estaban definidos pero que fueron necesarios añadir.

(Ver **¡Error! No se encuentra el origen de la referencia.)**	necesarias para mantener el control sobre los cambios a los que he sometido el código de fuente.
Registro de instalaciones **(**Ver **¡Error! No se encuentra el origen de la referencia.)**	Registra todos los lugares en los que se ha instalado un producto software.
Acta de las reuniones del CCC **(**Ver **¡Error! No se encuentra el origen de la referencia.)**	Registra todas las decisiones que son tomadas por el CCC con respecto al control de cambios en la configuración.

Los diferentes tipos de informes que se mantendrán se muestran en la **Tabla #5**

Tabla #5 Tipos de informes que se pueden generar en el proyecto.

Informes del Proyecto	
Informes	**Descripción**
Informe de estado de los cambios. **(**Ver ¡Error! No se encuentra el origen de la referencia.**)**	Es un resumen del estado en que se encuentran todas las solicitudes de cambio registradas durante un determinado período de tiempo.
Inventario de elementos de configuración. **(**Ver ¡Error! No se encuentra el origen de la referencia.**)**	Para ofrecer visibilidad sobre el contenido de las bibliotecas de proyecto.
Informe de modificaciones. **(**Ver ¡Error! No se encuentra el origen de la referencia.**)**	Es un resumen de las modificaciones que se han efectuado en el producto software durante un determinado período de tiempo.
Informe de diferencias entre versiones	Este informe no es necesario llevarlo de forma manual pues la herramienta de control de versiones lo realiza automáticamente.

En la **Figura 8** se muestra de forma gráfica cada una de las actividades definidas para darle cumplimiento a este proceso, así como sus entradas y salidas y el responsable de llevarla a cabo.

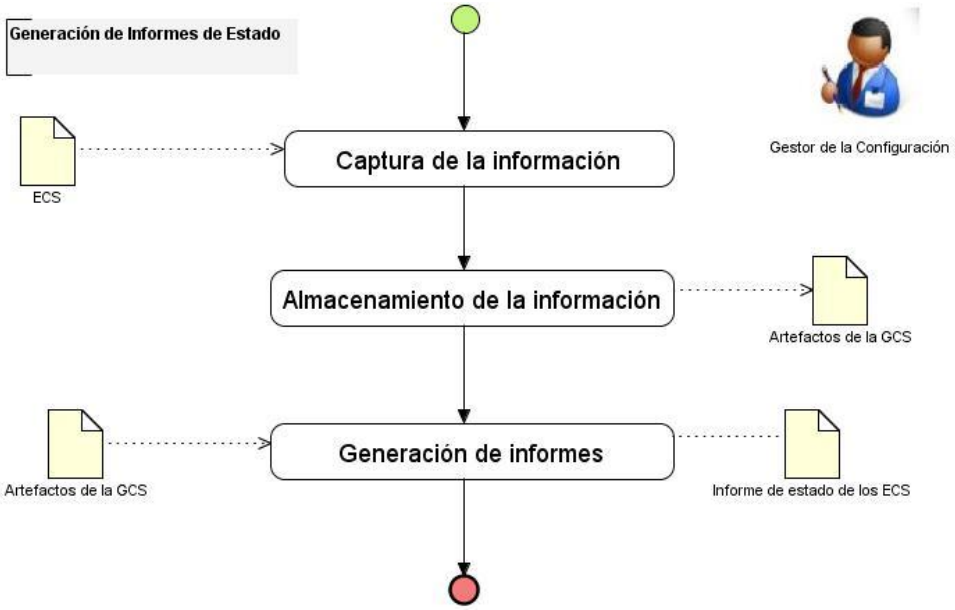


Fig 8. Modelo del proceso de Generación de Informes de Estado.

Auditoría de la configuración

Este proceso se encarga de garantizar que el cambio efectuado se haya llevado a cabo correctamente al velar por la completitud del producto y la consistencia entre sus componentes, garantizando que el producto que se está construyendo cumpla con todos los requisitos especificados.

Es considerada además una actividad para garantizar la calidad por lo que se debe efectuar por personal ajeno al equipo de desarrollo para mantener la objetividad.

Para su realización se necesita como:

Entrada: Artefactos de la GCS, Líneas base.

Responsable: Gestor de la configuración.

Salida: Informe de la auditoría de GCS[3], Documento de No Conformidades.

Para llevarse a cabo plantea dos actividades fundamentales:

[3] En este informe se recogerán los resultados de la auditoría así como los principales problemas detectados.


Revisiones Técnicas Formales (RTF):

El objetivo de esta actividad es certificar que el ECS se comporta correctamente una vez que éste se encuentra en su entorno operativo. De su realización se genera el documento de No Conformidades con todas las deficiencias encontradas.


Auditorías de la Configuración:

Esta actividad complementa las RTF y se les realiza a las líneas base para asegurar que se mantiene la calidad del producto después de los cambios realizados. Como resultado de esta actividad también se genera el documento de No Conformidades con todas las deficiencias encontradas.

Estas auditorías incluyen:

Objetivo:

El objetivo de todas las auditorías es verificar que en un momento dado la línea base se compone de una colección consistente y bien definida de productos.

Elementos de configuración bajo auditoría:

Se elegirán uno o más elementos de configuración de mayor prioridad en la línea base.

Agenda de auditorías:

Antes de la liberación o actualización.

Conducción:

Las auditoras serán dirigidas por el responsable de la administración de la configuración.

Participantes:

Responsable de la administración de la configuración y los autores de los ECS a auditar.

Documentos Requeridos:

Documentos de SC e informes del estado de la configuración generados.

Reportes de Deficiencias y Acciones Correctivas:

Determinadas por los participantes.

Criterio de Aprobación:

Lo determina el equipo de gestión de la configuración conjuntamente con el equipo de calidad.

Para realizar estas auditarías se utilizarán como mecanismos las pruebas funcionales a la versión actual del producto para verificar que este satisface los requisitos previstos para este hito; y

además la lista de chequeo que se muestra en el **¡Error! No se encuentra el origen de la referencia.** y que está incluida dentro del expediente de proyecto. Esta lista se encarga de chequear:

- ✓ El estado de las líneas base.
- ✓ El estado de los ECS.
- ✓ El estado del repositorio.
- ✓ Informar el estado de la configuración.

Estas auditorías pueden ser antecedidas o no por una RTF al igual que las RTF pueden ser preceddas o no por una auditoría.

2.3 ¿Cómo llevar a cabo los procesos de GCS en el desarrollo de distribuciones de GNU/Linux?

De la calidad con la que se realice cada uno de los procesos definidos anteriormente depende el éxito con el que se lleve a cabo la GCS en un proyecto.

El proceso de Definición de responsabilidades es el primer proceso que se ejecutara que pretende comenzar a implementar la GCS, de esta manera son seleccionados los responsables de llevar a cabo y controlar todas las actividades que se han definido para la GCS y como primera tarea tienen la de diseñar de qué manera es puesto en práctica la Familiarización con la GCS. Este proceso garantiza que todos los desarrolladores adquieran las habilidades necesarias que facilite la realización de esta actividad en el proyecto. Después de estos 2 procesos se realiza la Identificación de la Configuración, el Control de Cambios en la Configuración y las Auditorías de la Configuración respectivamente y paralelo a cada uno de ellos se va ejecutando la Generación de Informes de Estado que documenta e informa todo lo que sucede en estos procesos, además hay que señalar que después de una auditoría de la configuración se puede establecer de nuevo el CC en la configuración.

El orden en el que se realizan los dos primeros procesos puede variar según las necesidades del proyecto. La **Figura 9** muestra una representación gráfica de lo explicado anteriormente:

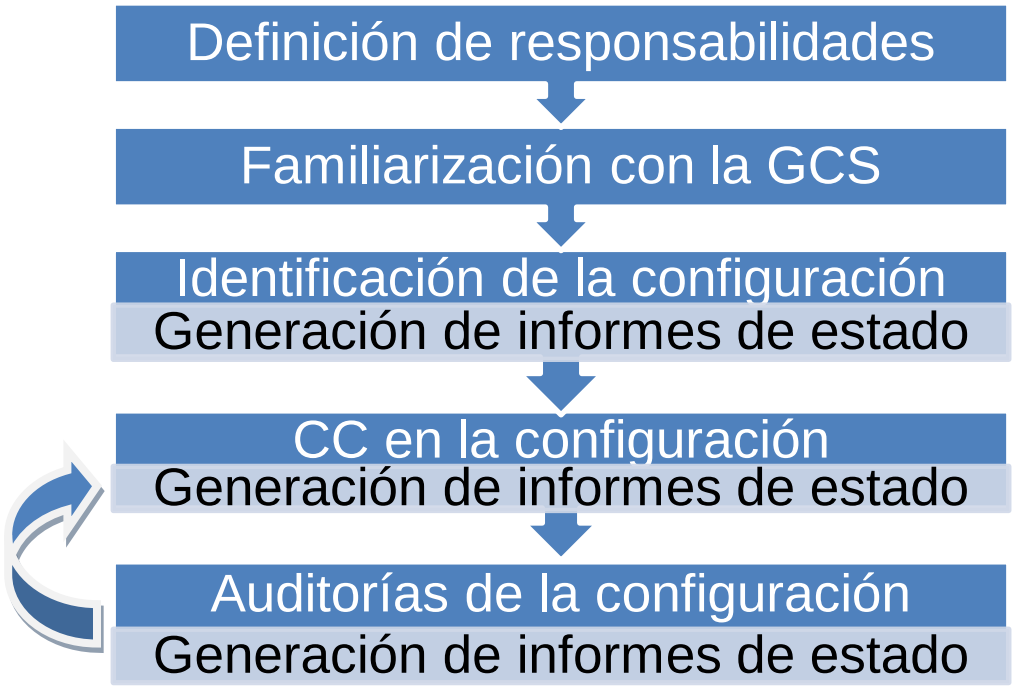


Fig 9. Orden en el que deben ser ejecutados los procesos de GCS

2.4 Consideraciones finales

- ✓ En este capítulo se diseñaron los procesos necesarios para garantizar una correcta GCS en los proyectos cuyo proceso de desarrollo esté basado en distribuciones de GNU/Linux; desde la capacitación del equipo de desarrollo en cuanto al tema como el chequeo y control de cada una de las actividades.
- ✓ Se definieron además los responsables y el orden en el que estos procesos deben ser ejecutados.

Capítulo 3

Evaluación de los procesos de GCS

Terminado el diseño de los procesos que garanticen una correcta GCS se hace necesario evaluar la integridad y valor de estos procesos, para conocer cuan aceptable puede ser la puesta en práctica de los mismos en cualquier proyecto, cuyo proceso de desarrollo esté orientado al desarrollo de distribuciones GNU/Linux. En este caso la valoración de esta solución se realiza mediante 2 métodos: el método Delphi basado en el criterio de expertos y el método experimental cuyo análisis requiere de la puesta en práctica de la solución.

3.1 Evaluación por el método Delphi.

El método de validación Delphi tiene como objetivo dar un criterio sobre lo válido que puede ser una solución brindada a una situación compleja a partir del criterio de especialistas en el tema. Para esto se realiza una selección de aquellas personas que más conocen sobre el problema planteado, los cuales a través de cuestionarios exponen su criterio sobre la propuesta planteada, que más tarde son tabuladas utilizando métodos estadísticos que nos brindan como resultado, entre otras cosas, el grado de aceptación de la solución planteada y la concordancia de criterios entre los especialistas acerca de la misma.

Sus principales características son:

- ✓ Anonimato: se expresa a través del no-conocimiento de las respuestas, puesto que los miembros del grupo contestan las preguntas sin confrontarse, incluso sin conocerse entre sí.
- ✓ Retroalimentación controlada: después de cada ronda de preguntas se tabulan las respuestas y se procesan de forma tal, que antes de la siguiente ronda los participantes pueden evaluar los resultados de la ronda anterior, así como las razones dadas para cada respuesta y su dispersión del promedio.
- ✓ Respuesta estadística del grupo: entre cada ronda de preguntas, la información obtenida se procesa por medio de técnicas estadístico-matemáticas, las que dotan al investigador de un instrumento objetivo y concreto en el cual pueden apoyarse para tomar una decisión final.

Para la realización de este método se hace necesario tener en cuenta que solo son seleccionados aquellos especialistas que estén de acuerdo en participar en la evaluación de la solución, y los que más competencias presenten sobre el tema en cuestión. Además se deben escoger los criterios de evaluación

que se tienen en cuenta para la realización de los cuestionarios y se realizan tantas iteraciones de estos como sean necesarias hasta lograr el nivel de aceptación que se quiere de la propuesta presentada.

3.1.1 Formulación del problema

Para la validación de la solución por el método Delphi lo primero es definir los criterios (C) a evaluar por los especialistas, los cuales deben ser precisos, medibles e independientes. Los atributos que se tuvieron en cuenta para evaluar la propuesta fueron los siguientes:

C1: Importancia de la propuesta.

C2: Contribución de las buenas prácticas a la solución propuesta.

C3: Contribución al aumento de la calidad en el PDSW de distribuciones de GNU/Linux.

C4: Integridad del contenido de la propuesta.

C5: Aplicación exitosa.

C6: Evaluación de la propuesta.

C7: Integridad de la propuesta.

3.1.2 Proceso de selección de especialistas.

Para la selección de los especialistas se tienen en cuenta tanto al individuo en sí como a un grupo de personas u organizaciones capaces de ofrecer valoraciones conclusivas de un problema en cuestión y hacer recomendaciones respecto a sus momentos fundamentales con un máximo de competencia.

En la investigación realizada se seleccionaron como especialistas para que valoren esta solución, a personas conocedoras tanto del campo que encierra a la GCS, como del PDSW de distribuciones de GNU/Linux, ya que es el área donde la solución es puesta en práctica. Estas personas por las competencias que poseen son capaces de emitir criterios y recomendaciones que ayudan a mejorar la solución planteada.

De una población de personas que podían considerarse especialistas para esta validación se escogió una muestra de 7 que estuvieron de acuerdo en participar en la encuesta. Ellos fueron los siguientes:

Especialista 1: Administradora de la calidad en el proyecto Identificación, Inmigración y Extranjería de la República de Cuba.

Especialista 2: Asesora del departamento central de Ingeniería y Gestión de Software.

Especialista 3: Arquitecto y desarrollador del equipo de desarrollo del instalador de Nova.

Especialista 4: Desarrollador y empaquetador en el proyecto Nova.

Especialista 5: Desarrollador, analista y arquitecto en el proyecto Nova.

Especialista 6: Jefe de la línea de desarrollo Nova Server en el proyecto Nova y administrador del repositorio.

Especialista 7: Jefe del equipo de desarrollo del instalador de Nova.

Para determinar aquellos especialistas que finalmente participan en la evaluación de la solución se calcula el coeficiente de competencia K, mediante la siguiente fórmula matemática:

$$\mathbf{K} = \frac{1}{2}(\mathrm{Kc} + \mathrm{Ka})$$

Donde:

K: Coeficiente de competencia.

Kc: Coeficiente de conocimiento.

Ka: Coeficiente de argumentación.

Para el cálculo del Kc se le realiza una encuesta a los especialistas, la cual se muestra en el **¡Error! No se encuentra el origen de la referencia.**. En esta encuesta el propio especialista valora el nivel de conocimiento que presenta sobre la GCS en una escala del 0 al 10; lo que significa que si la evaluación es "0" indica que el especialista no tiene absolutamente ningún conocimiento de la GCS, mientras que la evaluación "10" significa que el especialista tiene pleno conocimiento sobre el tema en cuestión. Luego de esto se multiplica esta evaluación por 0.1 para obtener de esta manera el coeficiente de conocimiento en un valor que oscila entre 0 y 1.

El grado de conocimiento que plantearon los especialistas tener sobre el tema se muestra en la **Tabla #6.**

Tabla #6 Grado de conocimiento de los especialistas en el tema tratado.

Grado de Conocimiento o Información de la GCS											
Especialistas	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1										X	
2										X	
3									X		
4								X			
5									X		

6								**X**			
7									**X**		

Con estos valores se calculó el Kc= criterio*0.1 obteniendo los siguientes resultados para cada especialista:

Tabla #7 Coeficiente de conocimiento de los especialistas en el tema tratado.

Especialistas	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Kc	0.9	0.9	0.8	0.7	0.8	0.7	0.8

Para el cálculo del **Ka** el especialista debe clasificar en alto, medio o bajo su grado de competencia sobre los aspectos o fuentes de argumentación sometidos a su consideración (este cuestionario también se muestra en el **¡Error! No se encuentra el origen de la referencia.** de este trabajo). Cada nivel de clasificación posee un valor y la suma de los valores marcados por cada criterio será el coeficiente de argumentación del candidato a especialista. Estos resultados se obtienen a partir de la tabla patrón que se encuentra en el **¡Error! No se encuentra el origen de la referencia..**

Por lo que el cálculo del **Ka**=∑valores **TP** donde:

valores TP: son los valores de la tabla patrón que se corresponden con lo marcado por los especialistas.

Los valores de **Ka** para cada especialista se muestran en la **Tabla #8**:

Tabla #8 Coeficiente de argumentación de los especialistas en el tema tratado.

Especialistas	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Ka	0.9	0.9	1.0	1.0	0.8	0.7	1.0

Con los valores de **Kc** y **Ka** se calculó el coeficiente de competencia **K** de los especialistas para seleccionar a los que finalmente validarán la solución. Solo serán seleccionados aquellos que obtengan un valor de **K** alto o medio, teniendo en cuenta el siguiente criterio de selección:

Si $0,8 \leq k < 1,0$ coeficiente de competencia alto.

Si $0,5 \leq k < 0,8$ coeficiente de competencia medio.

Si $k < 0,5$ coeficiente de competencia bajo.

El cálculo del coeficiente de competencia **K** de los especialistas mediante la fórmula antes planteada, teniendo en cuenta los valores de **Kc** y **Ka** antes calculados, arrojaron los siguientes resultados.

Como muestra en la **Tabla #9** todos los especialistas tienen un grado de influencia del coeficiente de competencia alto y medio, ninguno bajo, por lo que todos forman parte del panel de especialistas a los que se le aplica la encuesta para validar la solución.

Tabla #9 Coeficiente de competencia calculado para cada uno de los especialistas.

Especialistas	**Coeficiente de conocimiento. (Kc)**	**Coeficiente de argumentación. (Ka)**	**Coeficiente de competencia. (K)**	**Grado de influencia del coeficiente de competencia.**
1	0.9	0.9	0.90	Alto
2	0.9	0.9	0.90	Alto
3	0.8	1.0	0.90	Alto
4	0.7	1.0	0.85	Alto
5	0.8	0.8	0.80	Alto
6	0.7	0.7	0.70	Medio
7	0.8	1.0	0.90	Alto

3.1.3 Elaboración del cuestionario

Después de tener conformado el panel de especialistas que evalúan la solución se les hace entrega de un resumen de la propuesta para que se documenten sobre el tema y puedan responder las preguntas que se le realizan en el cuestionario.

El cuestionario realizado se encuentra disponible en el **¡Error! No se encuentra el origen de la referencia.** inicia con los datos personales del especialista y está conformado por 8 preguntas, las cuales valoran los criterios de evaluación antes presentados. De estas preguntas 2 son abiertas con el objetivo de que el especialista pueda emitir su criterio y opinión personal sobre la solución propuesta, lo que puede significar sugerencias que mejoren el resultado de la investigación; y 6 son del tipo contable, que permiten que el especialista le otorgue una evaluación al trabajo realizado. Para esto se utilizaron criterios de evaluación cualitativos (Muy alta, Alta, Media, Baja, Muy baja) y cuantitativos (estos están expresados en % y también con valores numéricos).

3.1.4 Análisis de los resultados

Para realizar el análisis de las encuestas realizadas a los especialistas se decide adoptar la escala de evaluación que se muestra en la **Tabla #10** donde las respuestas tendrán un valor de 1-5 (representando el 1 la puntuación más baja y el 5 la máxima puntuación alcanzada), para esto se realizó un balance de equivalencias entre los criterios (cualitativos y cuantitativos) de evaluación utilizados en la encuesta, con el objetivo de poder analizarlos estadísticamente logrando un resultado integrador y, mostrar gráficamente el nivel de aceptación con el que cuenta la propuesta.

Tabla #1 Criterios de evaluación para validar la propuesta.

Criterios de evaluación					
No	**Criterios Cualitativos**			**Criterios Cuantitativos (%)**	**Puntuación**
1	Contribuye muy notablemente	Excelente	Muy Alta	100	5
2	Contribuye notablemente	Muy bien	Alta	75	4
3	Contribuye medianamente	Bien	Media	50	3
4	Contribuye muy poco	Regular	Baja	25	2
5	No contribuye	Mal	Muy baja	0	1

3.1.5 Resumen de validación por especialistas.

Tabla #2 Resumen de las evaluaciones de los especialistas.

Criterios	Esp.1	Esp.2	Esp.3	Esp.4	Esp.5	Esp.6	Esp.7	Tj	PP	FA (%)
IP	5	5	5	5	4	5	5	34	4.85	97.1
CBP	5	5	4	5	4	5	5	33	4.71	94.2
CAC	4	4	4	4	4	4	4	28	4	80
CP	5	5	5	4	5	5	5	34	4.85	97.1
AE	5	5	5	4	4	5	5	33	4.71	94.2
EP	5	5	5	4	5	5	5	34	4.85	97.1
INP	4	4	5	4	4	4	4	29	4.14	82.8

Leyenda:

Esp: Especialista.

IP: Importancia de la propuesta.

CBP: Contribución de las buenas prácticas a la solución propuesta.

CAC: Contribución al aumento de la calidad en el PDSW de distribuciones de GNU/Linux.

CP: Integridad del contenido de la propuesta.

AE: Aplicación exitosa.

EP: Evaluación de la propuesta.

INP: Integridad de la propuesta.

Tj: Total de puntos obtenidos por el criterio j.

PP: Promedio de puntuación.

FA: Factor de aceptación.

3.1.6 Grado de concordancia de los expertos al conjunto de todas las preguntas

Para valorar cuantitativamente la evaluación y verificar la consistencia en el trabajo de los especialistas se determina el grado de concordancia de estos en cuanto a los criterios de evaluación, para esto se realiza el cálculo del coeficiente de Kendall (W).

Para el cálculo del W se emplea el programa estadístico Statistical Product and Service Solutions (SPSS), considerando como valores de entrada, los datos obtenidos como resultado de las encuestas realizadas a los especialistas.

Los valores del coeficiente de Kendall "W" deben oscilar entre 0 y 1 ($0 < W < 1$), si W alcanza el valor uno ($W = 1$) entonces existe una concordancia total de criterios, mientras mayor sea el valor de W, es decir, cuanto más se acerque a uno, mayor será la concordancia entre los especialistas. La concordancia se considera aceptable cuando los valores obtenidos como resultado del cálculo del W están por encima de 0.5.

Otra forma de comprobar si la concordancia entre los especialistas es significativa o no es analizar el tamaño de la muestra (N) luego de obtenido el resultado del W. En dependencia de N, se selecciona el estadígrafo que se tiene en cuenta para la prueba de hipótesis Si $N<=7$, las muestras son pequeñas y se calcula el Nivel de Significación (s), en caso de que $N>7$, se utiliza el cálculo de Chi Cuadrado ($x2$).

Si se hace uso del "s" y su valor obtenido con el empleo del SPSS es menor que 0.05, se rechaza H0 y se considera el valor de W como significativo. En caso de ser necesario el cálculo de $x2$ por presentar más de 7 criterios, debe tenerse en cuanta que si el valor de $x2$ es mayor que 0.05, entonces se rechaza la H0 y se considera significativo el valor del W.

Luego de obtener los resultados, se utiliza la Prueba de Significación de Hipótesis, planteándose la hipótesis nula (H0) y la alternativa (H1) de la siguiente forma:

H0: no existe concordancia entre los expertos. $W=0$

H1: existe concordancia entre los expertos. $W \neq 0$.

Tomando como entrada los valores obtenidos de las encuestas realizadas a los especialistas el cálculo de W mediante el programa SPSS obtuvieron los resultados que se muestran en la **Tabla #12**.

Tabla #3 Coeficiente de concordancia de Kendall

N	7
Kendall's W(a)	,555
Chi-Square	23,314
df	6
Asymp. Sig.	,001

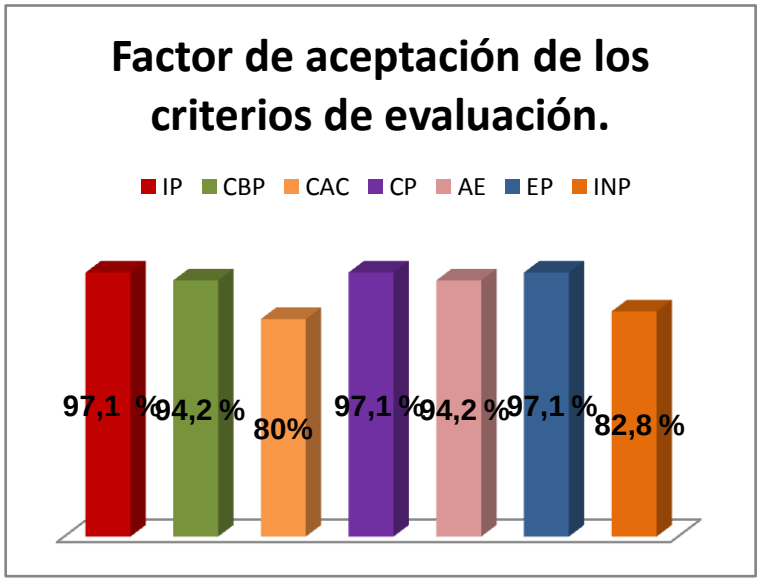
Como el valor de W es diferente de 0 se rechaza H0, pero además como W>0.5 (W=0.55) la concordancia entre los especialistas se considera aceptable.

Si tenemos en cuenta que la muestra seleccionada contiene 7 criterios de evaluación (N=7) se analiza el Nivel de Significación (s) y como s (Asymp. Sig.)=0.01 también se rechaza H0 por lo que se concluye que existe una concordancia significativa entre los criterios de los especialistas.

3.1.7 Conclusiones de la evaluación por el método Delphi.

Como se observa en la **Gráfica #1** todas las evaluaciones dadas por cada especialista a los diferentes criterios de evaluación oscilan entre 4 y 5, obteniendo un factor de aceptación para cada criterio siempre superior al 80% por lo que se puede concluir que la propuesta cumple con los objetivos trazados.

Al promediar cada un de los valores obtenidos para el factor de aceptación de cada criterio se obtuvo que el promedio de aceptación de la propuesta en general es de un 91.7 %.



Gráfica #1 Resumen de la validación de los especialistas.

3.2 Evaluación por el método experimental.

A diferencia del método Delphi, que solo brinda una predicción de cuán aceptable puede ser la propuesta de solución, el método de validación experimental demuestra realmente que tan factible es la propuesta, ya que sus resultados se obtienen de la puesta en práctica de la solución para comprobar sus beneficios. Para llevarlo a cabo se escogen una serie de variables que constituyen las partes de la propuesta de

solución que se va a probar y asociado a cada uno de ellas un conjunto de atributos que representan los criterios de evaluación para la solución. Después de esto se hace una valoración de la situación ***Antes*** y ***Después*** de haber puesto en práctica la solución para determinar si la misma es factible o no.

Para la evaluación de los resultados de esta investigación se decide valorar cada uno de los atributos en la situación ***Antes*** y ***Después*** para así definir cuánto mejora la situación problemática planteada en esta investigación después de haber puesto en práctica la solución, para ello se evaluará cada uno de los atributos utilizando una escala del 1 al 5 en una escala del 1 al 5 (5 es excelente, 4 es muy bien, 3 es bien, 2 es regular y 1 es mal).

El proyecto Nova organiza su PDSW en diferentes líneas de desarrollo, cada una de ellas trabaja en variantes diferentes de la distribución Nova con el objetivo de lograr un producto final cuyos componentes son: la imagen liberada (ISO) y el repositorio de aplicaciones, cada uno de estos construidos sobre la base de diferentes paquetes los cuales constituyen el principal componente de software a controlar en el proyecto. Debido a que no está establecido el proceso de GCS ninguno de estos componentes es identificado como ECS y tampoco son sometidos a un control y seguimiento de sus cambios. Por esta razón 2 de los procesos que forman parte de esta solución son probados en el PDSW del proyecto Nova.

Las variables seleccionadas lo constituyeron 2 de los procesos que forman parte de la solución y los atributos los aspectos fundamentales a tener en cuenta para evaluar el comportamiento de estas variables. La **Tabla #13** muestra la relación de estas variables y sus atributos.

Tabla #4 Variables y atributos seleccionados para la validación por el método experimental.

Variables	Atributos
Identificación de la Configuración.	Selección de los ECS.
	Definición de un esquema de identificación.
	Establecimiento de líneas base.
	Establecimiento de las bibliotecas de software.
Control de cambios en la Configuración.	Creación del Comité de Control de Cambios.
	Establecimiento del proceso de solicitud

	de cambios.

3.2.1 Identificación de la Configuración

Situación ***Antes***:

Selección de los ECS:

Están identificados algunos componentes de software en este caso paquetes, a los cuales es necesario controlarles los cambios pero no se reconocen como ECS, además no son solo los paquetes los que pueden constituir ECS dentro del proyecto Nova.

Selección de un esquema de identificación:

Por lo planteado anteriormente no existe ningún esquema que identifique de forma uniforme todos los ECS, pues estos no están definidos.

Definición y establecimiento de las líneas base:

No se definen ni establecen líneas base.

Definición y establecimiento de las bibliotecas de software:

Existe un repositorio para el código de fuente administrado mediante las herramientas reprepro y pbuilder con la ayuda de los script en bash y otra para la documentación administrada con el SVN de REDMINE en el que están almacenados los paquetes y los expedientes de proyecto de cada una de las líneas que se desarrollan, debido a que son considerados de importancia para el proyecto , pero ninguno de estos

dos repositorios están organizados por las bibliotecas de software que deben existir dentro del mismo, además algunos de los cambios que se le realizan en el REDMINE a estos componentes no son registrados.

Situación ***Después***:

Con la puesta en práctica del proceso Identificación de la Configuración mucho de los problemas que existían anteriormente son eliminados:

Selección de los ECS:

A partir de estos momentos se reconocen como ECS en el proyecto Nova todos los artefactos generados de la ingeniería de software, el conjunto de las imágenes liberadas hasta el momento y todos los paquetes y metapaquetes cuyos cambios son necesarios mantener controlados. La **Tabla #14** muestra todos estos elementos y la etapa del desarrollo en la que se crean o modifican.

Tabla #5 Artefactos de la ingeniería de software que constituyen ECS

Disciplina	Artefactos (ECS)	Fase (Creación/Modificación)			
		I	E	C	T
Requerimientos	Especificación de requisitos de software	x	x		
	Especificación de casos de uso	x	x		
	Reporte de trazabilidad	x	x	x	x
	Salidas del sistema	x	x	x	x
	Criterios para validar requisitos del producto	x	x	x	x
	Evaluación de requisitos	x			
	Registro de revisiones de inconsistencias	x	x	x	x
	Lista de chequeo para detectar Inconsistencias en REQM	x	x	x	x
	Evaluación de Casos de Uso	x	x		

Análisis y Diseño	Modelo de diseño		x		
	Arquitectura de Software Guía Base	x	x		
	Arquitectura de Integración	x	x		
	Arquitectura Vista de Proceso	x	x		
	Arquitectura Vista de Seguridad	x	x	x	x
	Arquitectura Vista Sistema	x	x	x	x
	Arquitectura Vista de Infraestructura	x	x	x	x
	Arquitectura Vista de Presentación	x	x	x	x
	Arquitectura Vista de Despliegue	x	x	x	
	Diseño de Casos de Pruebas basado en Requisitos	x	x	x	
	Planilla de Solicitud de Pruebas de Liberación			x	
	Planilla de Artefactos para Pruebas de Liberación			x	
	Manual de usuario				x
Despliegue	Modelo de despliegue			x	
Gestión de Proyecto	Plan de Pruebas	x			
	Plan de Desarrollo de Software	x			
	Proyecto Técnico	x			
	Registro de Revisiones para el Compromiso al Plan	x			
	Plantilla Problemas, Desviaciones y Acciones	x	x	x	x
	Plantilla Planes y Registro de Monitoreo	x	x	x	x

	Método de Estimación Calisoft	x			
	Método_de_Estimación_Post-Arquitectura_UCI	x			
	Acta de aceptación	x			
	Acta de Inicio	x			
	Selección de Proveedores	x			
	Lista de adquisiciones	x			
	Solicitud de Oferta de Productos y Servicios	x			
	Acuerdo con el Proveedor	x			
	Evaluación del Proveedor	x			
	Plantilla de Reporte de Estado del Proyecto	x	x	x	x
	Plantilla Reporte de Hitos para Alta Gerencia	x	x	x	x
	Reporte de Estado del Proveedor	x	x	x	x
	Minuta de reunión	x	x	x	x
	Reporte de Investigación	x			
	Diseño de Investigación	x			
	Estado del Arte del producto a desarrollar	x			
Soporte	Plantilla de Solicitud de Mejora de Proceso o Producto	x	x	x	x
	Plantilla de Registro de Evaluaciones del proyecto	x	x	x	x
	Glosario de Términos	x	x	x	x

	Solicitud de Cambios para Mejora	x	x	x	x
	Solicitud de Cambios para Error	x	x	x	x
	Lista de Verificación para Auditorías a la Configuración	x	x	x	x
	Plantilla de Recolección Manual de Medidas	x	x	x	x

Versiones liberadas:

nova-2010-acm-i386

nova-2010-desktop-i386

nova-server-alpha2

nova-desktop-2011-amd64

nova-desktop-2011-amd64.iso.md5

nova-desktop-2011-i386

nova-desktop-2011-i386.iso.md5

nova-desktop-2011-r1-i386

nova-desktop-2011-r1-i386.iso.md5

nova-desktop-2011-r2-i386

nova-desktop-2011-r2-i386.iso.md5

nova-desktop-2011-r3-i386

nova-desktop-2011-r3-i386.iso.md5

nova-light-2011-i386

nova-light-2011-i386.iso.md5

ADRIANE-KNOPPIX V6.2.1CD-2010-01-31-DE

KNOPPIX V6.2.1DVD-2010-01-31-EN

linuxmint-10-gnome-cd-i386

Linuxmint-debian-201101-gnome-dvd-i386

openSUSE-11.4-DVD-i586

openSUSE-11.4-NET-i586

openSUSE KDE 4 desktop.i686-0.0.4-personalizado
Puppy-4.2-k2.6.25.16-seamonkey
kubuntu-10.10-desktop-i386
kubuntu-10.10-dvd-amd64
lubuntu-10.10
lubuntu-10.10.iso.md5
ubuntu-10.10-alternate-i386
Ubuntu-10.10-desktop-amd64
xubuntu-10.10-desktop-amd64
Natty-desktop-i386. dailyBuild-january4-
Zentyal-2.02-i386

Paquetes a controlar

A continuación se listan los paquetes y metapaquetes que son identificados como ECS ya que son considerados críticos en el PDSW de cada una de las variantes de Nova que desarrolla el proyecto por lo que se hace necesario controlarle sus cambios, también se muestran los que son comunes para todas las líneas de desarrollo.

Genéricos:

2011|main|source:apt0.8.3ubuntu7nova1
2011|main|source:apt-extract0.12
2011|main|source:apt-setup1:0.42ubuntu3nova1
2011|main|source:atk1.01.32.0-0ubuntu1nova1
2011|main|source:audacity1.3.12-7nova1
2011|main|source:base-files5.0.0ubuntu20.10.04.3nova2~2011
2011|main|source:base-installer1.103ubuntu7nova1
2011|main|source:bash4.1-2ubuntu3nova1
2011|main|source:brltty4.1-2ubuntu6nova1
2011|main|source:busybox1:1.13.3-1ubuntu11nova1~2011
2011|main|source:cairo1.10.0-1ubuntu3nova1
2011|main|source:casper1.248nova11
2011|main|source:cdebconf-keystep0.10nova1

2011|main|source:checkbox0.9.1nova1
2011|main|source:chromium-browser10.0.648.133~r77742-0ubuntu0.10.04.1nova1
2011|main|source:cryptsetup2:1.1.0~rc2-1ubuntu13nova1
2011|main|source:debootstrap1.0.25nova6~2011
2011|main|source:dhcp33.1.3-2ubuntu3nova1
2011|main|source:directfb1.2.8-5ubuntu2nova1
2011|main|source:doublecmd0.4.6-4000Mnova1~2011
2011|main|source:eglibc2.11.1-0ubuntu7.8nova1
2011|main|source:ejecter0.4.1-0nova4
2011|main|source:expat2.0.1-7ubuntu1nova1
2011|main|source:faenza-cupertino0.1
2011|main|source:faenza-icon-theme0.8nova7
2011|main|source:firefox3.6.16+build1+nobinonly-0ubuntu0.10.10.1nova1
2011|main|source:fontconfig2.8.0-2ubuntu1nova1
2011|main|source:fuse2.8.1-1.1ubuntu3.1nova1
2011|main|source:gdk-pixbuf2.22.0-0ubuntu1nova1
2011|main|source:glib-networking2.26.0-1
2011|main|source:gmultimms0.1-2nova1
2011|main|source:gnome-codec-install0.4.7ubuntu2nova1
2011|main|source:gnome-control-center1:2.30.1-0ubuntu1nova1
2011|main|source:gnome-desktop1:2.30.2-0ubuntu1nova1
2011|main|source:gnome-icon-theme2.31.0-0ubuntu1nova1
2011|main|source:gnome-main-menu0.9.13-5nova5
2011|main|source:gnome-panel1:2.30.2-0ubuntu0.2nova3
2011|main|source:gnome-screensaver2.30.0-0ubuntu2nova1
2011|main|source:gnome-session2.32.0-0ubuntu1nova1
2011|main|source:gnome-shell2.31.5-2ubuntu2nova1
2011|main|source:gnome-themes-nova0.2
2011|main|source:gnupg1.4.10-2ubuntu1nova1
2011|main|source:grub21.99really1.98.20100804-14nova4

2011|main|source:gtk+2.02.22.0-0ubuntu1nova1
2011|main|source:gtk2-engines1:2.20.1-1ubuntu1nova1
2011|main|source:gtkdialog2:0.7.20-4nova1
2011|main|source:guano-default-settings0.9.5
2011|main|source:guano-session0.9.10
2011|main|source:indicator-me0.2.10-0ubuntu1nova1
2011|main|source:inkscape0.48.0-1ubuntu1nova1
2011|main|source:iso-scan1.28ubuntu2nova1
2011|main|source:jfsutils1.1.12-2.1nova1
2011|main|source:kdenlive0.7.9+svn20110304.r5471-0ubuntu0nova1
2011|main|source:kiwix0.9-alpha7.1nova2
2011|main|source:klibc1.5.17-4ubuntu1nova1
2011|main|source:lazarus0.9.30.1-30490M~lucidnova1
2011|main|source:libaio0.3.107-3ubuntu2nova1
2011|main|source:libapt-pkg-perl0.1.24build1nova1
2011|main|source:libdebian-installer0.68ubuntu3nova1
2011|main|source:libfm0.1.15+git20110210
2011|main|source:libgcrypt111.4.4-5ubuntu2nova1
2011|main|source:libgpg-error1.6-1ubuntu2nova1
2011|main|source:libqt4pas2.1Qt4.5.3-1
2011|main|source:libx112:1.3.3-3ubuntu1nova1
2011|main|source:libxau1:1.0.6-1nova1
2011|main|source:libxcb1.6-1nova1
2011|main|source:libxcursor1:1.1.10-2nova1
2011|main|source:libxdmcp1:1.0.3-2nova1
2011|main|source:libxext2:1.1.2-1nova1
2011|main|source:libxfixes1:4.0.5-1nova1
2011|main|source:libxi2:1.3-4nova1
2011|main|source:libxinerama2:1.1-3nova1
2011|main|source:libxrender1:0.9.6-1nova1

2011|main|source:likewise-open5.4.0.42111-2ubuntu1.2nova1
2011|main|source:lintian2.4.3ubuntu1nova2
2011|main|source:linux2.6.32-31.61.2
2011|main|source:linux-meta2.6.32.31.37.1
2011|main|source:linux-ntfs2.0.0-1ubuntu4nova1
2011|main|source:loop-aes-utils2.15.1~rc1-2ubuntu1nova1
2011|main|source:ltsp5.2.4-0ubuntu6nova3
2011|main|source:lvm22.02.54-1ubuntu4.1nova1
2011|main|source:lxdm0.2.0~svn2307+git20100408-0ubuntu1nova4
2011|main|source:lxpanel0.5.5-0ubuntu2nova3
2011|main|source:main-menu1.30ubuntu1nova1
2011|main|source:mdadm2.6.7.1-1ubuntu15nova1
2011|main|source:menu-cache0.3.2-2
2011|main|source:mplayer2:1.0~rc4~try1.dsfg1-1ubuntu1nova1
2011|main|source:multimms1.0-2nova1
2011|main|source:multipath-tools0.4.8-14ubuntu4nova1
2011|main|source:nautilus1:2.32.2-0ubuntu4nova1~ppa171
2011|main|source:ne-terminal-config1.1-0nova1~ppa0
2011|main|source:net-retriever1.24ubuntu2nova1
2011|main|source:netbook-launcher-efl0.2.6-0ubuntu2nova1
2011|main|source:netcfg1.51ubuntu2nova1
2011|main|source:nova-artwork3.0~rc6
2011|main|source:nova-docencia0.13nova1
2011|main|source:nova-docs1.0~rc11
2011|main|source:nova-keyring2010.09.21
2011|main|source:nova-meta3.0~rc20
2011|main|source:nova-sounds0.2
2011|main|source:nova-wallpapers0.4
2011|main|source:ntfs-3g1:2010.3.6-1ubuntu1nova1
2011|main|source:open-iscsi2.0.871-0ubuntu4nova1

2011|main|source:openoffice.org1:3.2.0-7ubuntu4.2nova5
2011|main|source:openssl0.9.8o-1ubuntu4.4nova1
2011|main|source:packagekit0.6.8-0ubuntu3.2nova1
2011|main|source:packagekit-gnome2.32.0-0ubuntu1nova1
2011|main|source:pango1.01.28.2-0ubuntu1.1nova1
2011|main|source:partman-auto-lvm33ubuntu4nova1
2011|main|source:pcmanfm0.9.11+git20110210
2011|main|source:pcre37.8-3build1nova1
2011|main|source:pidgin-webkit0.1-1nova2
2011|main|source:pixman0.18.4-1nova1
2011|main|source:plymouth0.8.2-2ubuntu22nova3~2011
2011|main|source:popt1.15-1nova1
2011|main|source:python-apt0.7.96.1ubuntu11.1nova2
2011|main|source:qemu-kvm0.12.3+noroms-0ubuntu9.4nova1
2011|main|source:reiserfsprogs1:3.6.21-1build1nova1
2011|main|source:rootskel-gtk1.17ubuntu1nova1
2011|main|source:rtmpdump2.3-2nova1
2011|main|source:seabios0.6.1.2-2nova1
2011|main|source:serere2.2.5
2011|main|source:slang22.2.2-2ubuntu1nova4
2011|main|source:software-properties0.75.10.1nova2
2011|main|source:synapsis0.1-0nova10
2011|main|source:synaptic0.63.1ubuntu14nova5
2011|main|source:tasksel2.73ubuntu26nova1
2011|main|source:telepathy-gabble0.11.0-1nova1
2011|main|source:telepathy-glib0.13.4-1nova1
2011|main|source:ttf-dejavu2.30-2nova1
2011|main|source:ubufox0.9~rc2-0ubuntu2.1nova3
2011|main|source:ubuntu-keyring2010.11.09nova1
2011|main|source:ubuntu-meta1.197nova1

2011|main|source:udev151-12.3nova2~2011

2011|main|source:udisks1.0.1-1ubuntu1nova1

2011|main|source:usb-creator0.2.25nova1

2011|main|source:user-setup1.28ubuntu7nova1

2011|main|source:util-linux2.17.2-0ubuntu1.10.04.2nova2~2011

2011|main|source:vlan1.9-3ubuntu3nova1

2011|main|source:vte1:0.26.0-0ubuntu2nova1

2011|main|source:wireless-tools30~pre9-3ubuntu4nova1

2011|main|source:xdg-utils1.0.2-6.1ubuntu3nova1

2011|main|source:xfsprogs3.1.0ubuntu1nova1

2011|main|source:xft2.1.14-2ubuntu1nova1

2011|restricted|source:adobe-flashplugin10.1.102.65-2lucid1

2011|restricted|source:nvidia-graphics-drivers195.36.24-0ubuntu1nova1~2011

2011|universe|source:alarife3.0-0nova1

2011|universe|source:multimms0.9.3-1nova1

2011|universe|source: uck2.0.12-0ubuntu2nova

Los paquetes específicos por cada línea son los siguientes:

Línea de desarrollo: Nova Server

Metapaquetes: nova-server, nova-server-ui, nova-server-session, nova-server-default-settings.

Paquetes: db-installer, Apache, linux-server, ebox, dhcp3-server, bind9, vsftpd, PostgreSQL, squid, openldap, postfix, openssh, mysql-server, phpmyadmin, phpldapadmin, phppgadmin

Línea de desarrollo: Nova Base

Metapaquetes: nova-minimal

Línea de desarrollo: Software propios

Desarrollo: Serere

Paquetes: python-parted, python-augeas, python-pexpect, Python-dbus, os-prober, Hal, squashfs-tools, reiserfsprogs.

Desarrollo: OSPlugger

Metapaquetes: dbus-c++, gnucommon, libconfig, pqxx, glibmn, poco, libxmlrpc-c3, libstd-c++, dmsetup, coreutilus, bash.

Línea de desarrollo: Clientes Ligeros

Metapaquetes: guano.

Paquetes: debian-installer.

Línea de desarrollo: Nova Desktop

Metapaquetes: chromium-browser, debootstrap, ejecter, faenza-icon-theme, firefox, gnome-codec-install, gnome-session, gnome-shell, gnome-themes-nova, grub2, lintian, linux, linux-meta, nautilus, ne-terminal, nova-artwork, nova-docs, nova-meta, open-office, packagekit, packagekit-gnome, plymouth, sinapsis, synaptic, usb-creator

Existen otras líneas de desarrollo como AGN y NBS de las cuales no se especifican los paquetes a controlar porque no se tiene acceso a su información debido a que no se están desarrollando en el proyecto.

Establecimiento de un esquema de identificación:

Cada uno de los ECS definidos es identificados de manera uniforme como se explica a continuación:

La nomenclatura para los artefactos es: **Código de la línea de desarrollo_código del flujo de trabajo_ código de la LB_nombre del artefacto.extensión.**

Código de la línea de desarrollo: nombre de la línea de desarrollo a la que pertenece.

El código para el flujo de trabajo se muestra en la **Tabla #15**:

Tabla #6 Código de flujos de trabajo.

Código de Flujos de Trabajo	
Flujo de Trabajo	**Sintaxis**
Requisitos	REQ
Diseño	AD
Implementación y Prueba	IMP
Despliegue	DE
Gestión del Proyecto	GP
Soporte	SOP

Código de la LB: Identificación de la LB en la que el ECS fue revisado, corregido y aprobado.

Las versiones se definen de la siguiente manera **X, Y**, **X** es la versión del producto y **Y** la versión del artefacto.

La nomenclatura para los artefactos será: **[Código de la línea de desarrollo]_Código del Subsistema_[Código del metapaquete]_nombre del paquete**

Código de la línea de desarrollo: este campo es opcional, solo se especifica cuando el paquete no es utilizado por todas las líneas de desarrollo.

Código del Subsistema: Un código que identifique el subsistema en el que se está trabajando o puede identificar también a un componente dentro de ese subsistema.

Código del metapaquete: Es especificado solo en caso de que el paquete forme parte de un metapaquete.

Las versiones se definen de la siguiente manera **X, Y**, **X** es la versión del producto y **Y** la versión del paquete.

La nomenclatura para los ISO será: **Nombre de la distribución-Año-Línea de desarrollo-versión-arquitectura para la cual se desarrolló el ISO.**

Versión: X, Y

X: Lanzamiento del producto.

Y: Número de la versión de ese lanzamiento.

Definición y establecimiento de las líneas base:

Las líneas base se establecen al final de cada fase de desarrollo y se identifican de la siguiente forma: LB+#_Código de la etapa.

#: El número de la línea base de acuerdo al orden en que fue establecida.

El código de la etapa se establece como se muestra en la **Tabla #16.**

Tabla #7 Código de las etapas

Código de Etapas	
Etapa	**Código**
Inicio	I
Elaboración	E
Construcción	C
Transición	T

LB por etapas y ECS que las conforman:

En todas las líneas base van a estar presente los siguientes ECS:

- ✓ Reporte de trazabilidad.

- ✓ Salidas del sistema.
- ✓ Criterios para validar requisitos del producto.
- ✓ Registro de revisiones de inconsistencias.
- ✓ Lista de chequeo para detectar Inconsistencias en REQM.
- ✓ Arquitectura Vista de Seguridad.
- ✓ Arquitectura Vista Sistema.
- ✓ Arquitectura Vista de Infraestructura.
- ✓ Arquitectura Vista de Presentación.
- ✓ Plantilla Problemas, Desviaciones y Acciones.
- ✓ Plantilla Planes y Registro de Monitoreo.
- ✓ Plantilla de Reporte de Estado del Proyecto
- ✓ Plantilla Reporte de Hitos para Alta Gerencia.
- ✓ Reporte de Estado del Proveedor.
- ✓ Minuta de reunión.
- ✓ Plantilla de Solicitud de Mejora de Proceso o Producto.
- ✓ Plantilla de Registro de Evaluaciones del proyecto.
- ✓ Glosario de Términos.
- ✓ Solicitud de Cambios para Mejora.
- ✓ Solicitud de Cambios para Error.
- ✓ Lista de Verificación para Auditorías a la Configuración.
- ✓ Plantilla de Recolección Manual de Medidas.

LB1_I (Línea base de la fase de inicio):

ECS que la conforman:

- ✓ Especificación de requisitos de software.
- ✓ Especificación de casos de uso.
- ✓ Evaluación de requisitos.
- ✓ Evaluación de Casos de Uso.
- ✓ Modelo de diseño.

- ✓ Arquitectura de Software Guía Base.
- ✓ Arquitectura de Integración.
- ✓ Arquitectura Vista de Proceso.
- ✓ Arquitectura Vista de Despliegue.
- ✓ Diseño de Casos de Pruebas basado en Requisitos.
- ✓ Plan de Pruebas.
- ✓ Plan de Desarrollo de Software.
- ✓ Proyecto Técnico.
- ✓ Registro de Revisiones para el Compromiso al Plan.
- ✓ Plantilla Problemas, Desviaciones y Acciones.
- ✓ Plantilla Planes y Registro de Monitoreo
- ✓ Método de Estimación Calisoft.
- ✓ Método_de_Estimación_Post-Arquitectura_UCI.
- ✓ Acta de aceptación.
- ✓ Acta de Inicio.
- ✓ Selección de Proveedores.
- ✓ Lista de adquisiciones.
- ✓ Solicitud de Oferta de Productos y Servicios.
- ✓ Acuerdo con el Proveedor.
- ✓ Evaluación del Proveedor.
- ✓ Plantilla de Reporte de Estado del Proyecto.
- ✓ Plantilla Reporte de Hitos para Alta Gerencia.
- ✓ Reporte de Estado del Proveedor.
- ✓ Minuta de reunión.
- ✓ Reporte de Investigación.
- ✓ Diseño de Investigación.
- ✓ Estado del Arte del producto a desarrollar.
- ✓ Plantilla de Solicitud de Mejora de Proceso o Producto.
- ✓ Plantilla de Registro de Evaluaciones del proyecto.
- ✓ Glosario de Términos.

- ✓ Solicitud de Cambios para Mejora.
- ✓ Solicitud de Cambios para Error.
- ✓ Lista de Verificación para Auditorías a la Configuración.
- ✓ Plantilla de Recolección Manual de Medidas.

LB2_E (Línea base de la etapa de Elaboración)

ECS que la conforman:

- ✓ Especificación de requisitos de software.
- ✓ Especificación de casos de uso.
- ✓ Reporte de trazabilidad.
- ✓ Salidas del sistema.
- ✓ Criterios para validar requisitos del producto.
- ✓ Registro de revisiones de inconsistencias.
- ✓ Lista de chequeo para detectar Inconsistencias en REQM.
- ✓ Evaluación de Casos de Uso.
- ✓ Modelo de diseño.
- ✓ Arquitectura de Software Guía Base.
- ✓ Arquitectura de Integración.
- ✓ Arquitectura Vista de Proceso.
- ✓ Arquitectura Vista de Seguridad.
- ✓ Arquitectura Vista Sistema.
- ✓ Arquitectura Vista de Infraestructura.
- ✓ Arquitectura Vista de Presentación.
- ✓ Arquitectura Vista de Despliegue.
- ✓ Diseño de Casos de Pruebas basado en Requisitos.

LB3_D (Línea base de la etapa de Desarrollo)

ECS que la conforman:

- ✓ Arquitectura Vista de Despliegue.
- ✓ Diseño de Casos de Pruebas basado en Requisitos.
- ✓ Planilla de Solicitud de Pruebas de Liberación.

- ✓ Planilla de Artefactos para Pruebas de Liberación.
- ✓ Modelo de despliegue.

LB4_T (Línea base de la etapa de Transición)

ECS que la conforman:

- ✓ Manual de usuario.

Definición y establecimiento de las bibliotecas de software:

El repositorio donde se encuentran los ECS producto del trabajo de la Ingeniería de Software queda organizado con las siguientes bibliotecas de software:

Biblioteca de trabajo: En esta biblioteca se encuentran almacenados todos los artefactos y paquetes definidos como ECS será el lugar de trabajo de cada desarrollador donde podrán modificar los ECS pues los cambios se realizan sin control alguno.

Biblioteca de soporte: Se almacenan las líneas base conformada por todos los ECS que fueron revisados y aprobados. Los cambios en esta biblioteca solo se realizaran por el personal autorizado por el líder del proyecto.

Biblioteca Maestra: En esta biblioteca están almacenados todos los release y un cambio en estos solo puede ser realizado con autorización del líder de proyecto.

El repositorio de código de fuente se mantiene con la estructura organizativa que tenía hasta el momento debido a que Dariem Pérez Herrera quien ocupa el rol de administrador del repositorio considera que esta es mejor manera de gestionar los cambios que pueden darse en los diferentes paquetes.

Para definir la forma en la que se van a almacenar los ECS y los permisos para acceder a ellos en las bibliotecas se nombra como bibliotecario a Yusleydi Fernández del Monte y en el repositorio de paquetes se mantiene como bibliotecario su administrador.

3.2.2 Control de Cambios en la Configuración

Situación ***Antes***:

Creación del CCC:

No está creado el CCC.

Definición y establecimiento del proceso de solicitud de cambios:

No está definido ningún proceso de solicitud de cambios.

Situación ***Después***:

Creación del CCC:

Este comité está conformado por: el jefe de departamento, el jefe de proyecto, jefes de cada línea de desarrollo, la administradora de la calidad de software, responsables de la GCS y son los encargados de velar porque el proceso de cambios se lleve a cabo de manera correcta, además de aprobar y controlar los cambios que se van a realizar sobre un ECS. El CCC del proyecto Nova queda conformado de la siguiente forma:

Tabla #8 CCC del proyecto Nova

Nombre y Apellidos	Rol
Allan Pierra Fuentes	Jefe del departamento
Abel A. Fírvida Donéstevez	Líder del proyecto
Dariem Pérez Herrera	Líder de Nova Server
Jorge Luis Machín Castillo	Líder Nova Base
Daniel Hernández Bahr	Líder de Nova Escritorio
Yunier Soler Franco	Líder de la línea de desarrollos propios
Yusleydi Fernández del Monte	Administradora de la calidad de software
Yarelys Calderón Santos	Responsable de la GCS

Definición y establecimiento del proceso de solicitud de cambios:

Para llevar a cabo el proceso de solicitud de cambio se siguen los siguientes pasos:

1. El cliente[4] o cualquier miembro del equipo del proyecto presenta la solicitud del cambio que quiere efectuar al CCC acompañada de un informe de cambio que debe incluir el resultado del esfuerzo técnico que se debe aplicar, los posibles efectos secundarios, el impacto global sobre otras funciones del sistema y el coste estimado del cambio.

[4] Las necesidades del cliente llegan al proyecto por mediación del proyecto SYMAYS.

2. El CCC planifica una reunión para evaluar la SC y decidir si el cambio es beneficioso o no teniendo en cuenta el IC presentado. Si el CCC rechaza esta SC, se le informa al cliente, de lo contrario el IC se convierte en una OC que especifica el cambio que se va a realizar, las restricciones que se deben respetar y los criterios de revisión y auditoría.

3. El líder de la línea de desarrollo le asigna el cambio a un integrante de su equipo de desarrollo.

4. Se efectúa el cambio.
5. Se evalúa el cambio para comprobar que este se realizó correctamente
6. El ECS modificado se reincorpora a la biblioteca a la que pertenece.
7. Se actualizan las líneas base integrando el cambio realizado.
8. Se auditan todos los ECS afectados por el cambio.
9. Se construye la versión adecuada del software.
10. Se notifica el cambio.

En las tablas **18** y **19** se muestra una evaluación de cada uno de los criterios tenidos en cuenta de los procesos Identificación de la Configuración y Control de Cambios en la Configuración, los cuales constituyeron las variables utilizadas en la evaluación de la propuesta por el método experimental; con el objetivo de realizar un análisis que nos demuestre si la ejecución del mismo es beneficiosa o no para el desarrollo del proyecto Nova.

Tabla #9 Evaluación de la variable Identificación de la Configuración.

Identificación de la Configuración				
	Selección de los ECS	**Definición de un esquema de identificación**	**Establecimiento de líneas base**	**Establecimiento de las bibliotecas de software**
Antes	**3**	**1**	**1**	**1**
Después	**5**	**5**	**4**	**3**

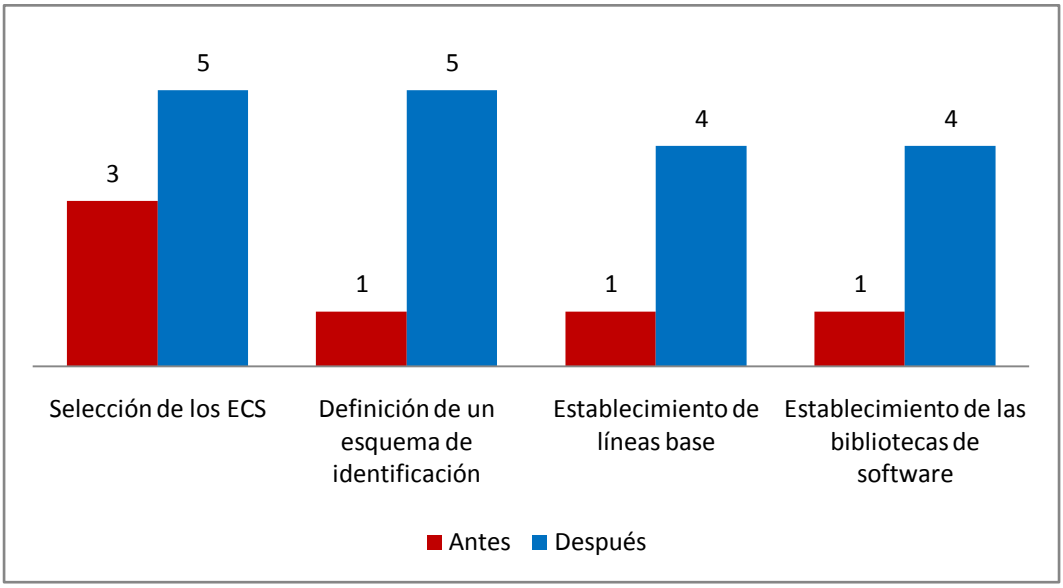
Tabla #10 Evaluación de la variable Control de Cambios en la Configuración.

Control de Cambios en la Configuración		
	Creación del Comité de Control de Cambios	**Establecimiento del proceso de solicitud de cambios**
Antes	**1**	**1**
Después	**5**	**4**

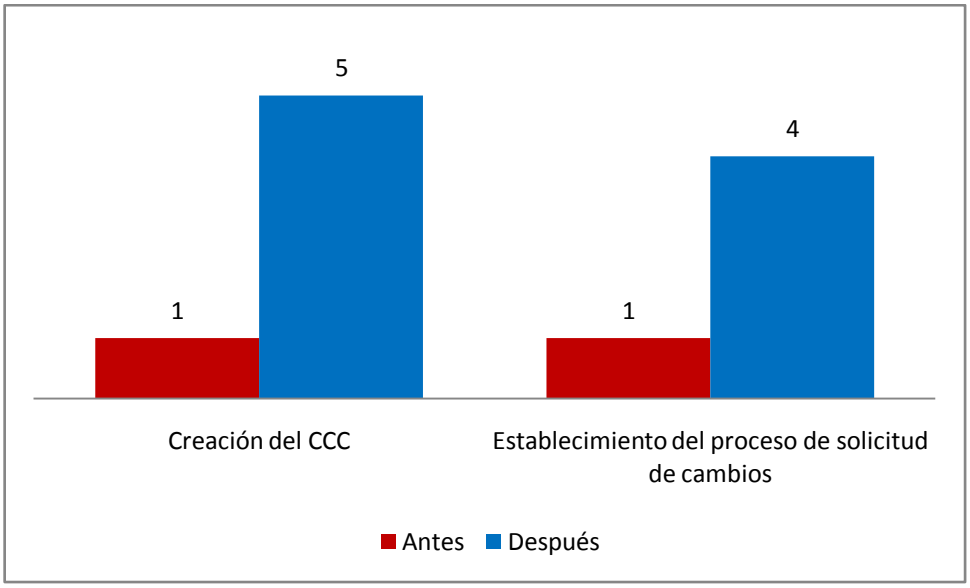
3.2.3

Conclusiones de la evaluación por el método experimental

En las **gráficas 1 y 2** obtenidas a partir de las tablas **18** y **19,** se puede observar que la situación después de haber puesto en práctica estos procesos mejoró notablemente, debido a que se establecieron las bases para llevar a cabo una correcta GCS, por lo que se puede considerar que la solución que brinda la propuesta: "Procesos de GCS en el desarrollo de distribuciones GNU/Linux" es aceptable.



Gráfica #1 Representación de la evaluación del proceso Identificación de la Configuración.



Gráfica #2 Representación de la evaluación del proceso Control de cambios en la configuración.

3.3 Consideraciones finales

Con el desarrollo de este capítulo se demuestra que la GCS es un proceso que se debe tener en cuenta en cualquier proyecto para mantener un mayor control sobre los diferentes componentes de software y los cambios a los que estos son estaban definidos los procesos que garanticen una correcta GCS, por lo que se diseñó la solución, la cual al ser evaluada tanto por el método Delphi como por el método experimental arrojó resultados favorables demostrando que puede ser una solución factible para resolver el problema científico planteado en esta investigación.

Conclusiones generales

Como resultado del presente trabajo investigativo se arribaron a las siguientes conclusiones:

- ✓ La GCS es una disciplina de control indispensable en el desarrollo de cualquier proyecto informático para asegurar la calidad del producto final.
- ✓ Una correcta GCS depende de la ejecución de cada una de las actividades que esta disciplina define, del control por parte de los responsables de llevarla a cabo y la actualización mediante las herramientas automatizadas de cada uno de los artefactos involucrados en este proceso.
- ✓ Se diseñaron un conjunto de procesos que van a guiar la forma de llevar a cabo la GCS en cualquier proyecto de desarrollo de distribuciones GNU/Linux.
- ✓ Actualmente en el proyecto Nova se encuentran establecidas las pautas necesarias para llevar a cabo una correcta GCS teniendo en cuenta los modelos, normas, estándares y guías basados en la experiencia de empresas y expertos en los proyectos de software.
- ✓ Los procesos de GCS propuestos fueron evaluados satisfactoriamente por un grupo de especialistas.

Recomendaciones

- ✓ Impartir con mayor profundidad el contenido de GCS en la asignatura de Ingeniería de Software.
- ✓ Implementar el proceso de manera íntegra en proyectos que desarrollen distribuciones GNU/Linux para evaluar los resultados obtenidos y mejorar así la aplicación de los mismos.
- ✓ Investigar cómo se llevan a cabo los procesos de GCS en desarrollos colaborativos para hacer la propuesta extensible a otros entornos de desarrollo.

Referencias Bibliográficas:

EVA. 2010. Introducción a la Gestión de Configuración de Software (GCS).Control de versiones y filosofía del [Consultado el: 13 de octubre del 2010] Disponible en: http://eva.uci.cu/mod/resource/view.php?id=21006

Antonio, Angélica de. 2008. Gestión de Configuración. [Consultado el: 12 de noviembre del 2010] Disponible en: http://is.ls.fi.upm.es/udis/docencia/plani/G_Configuracion.pdf

Donéstevez, Ing. Abel Alfonso Fírvida. 2010. NOVA 3.0, AVANCES Y ESPECTATIVAS DE LA DISTRIBUCIÓN CUBANA DE GNU/LINUX. Taller: Uso y desarrollo de aplicaciones de código abierto. [Consultado el: 9 de octubre del 2010] Disponible en:

LSIIS, Unidad de programación-Dep. 2010. Gestión de configuración. [Consultado el: 2 de diciembre del 2010] Disponible en: http://lml.ls.fi.upm.es/ep/versiones.html#toc6

geronimo. 2008. Gestión de la configuración del software [Consultado el: 2 de diciembre del 2010] Disponible en: http://www.geronet.com.ar/?p=90

Scribd. 2006. cmmi. [Consultado el: 2 de noviembre del 2010] Disponible en: http://www.scribd.com/doc/41340693/cmmi

histaintl, hista internacional. 2010. Home | Hista. [Consultado el: 10 de octubre del 2010] Disponible en: http://www.histaintl.com/

Diana Ibett Arias Herrera, Yulien Gallego Parra. 2007. Procedimiento de control y aseguramiento de la calidad en la Administración de la Configuración y Cambio. [Consultado el: 7 de octubre del 2010] Disponible en:

Quiñones, Ernesto. 2006. modelos_de_calidad_y_software_libre. [Consultado el: 2 de enero del 2011] Disponible en: http://www.eqsoft.net/presentas/modelos_de_calidad_y_software_libre.pdf

Gracia, Joaquín. 2003. CMM - CMMI Nivel 2. Calidad. Ingenieria del Software. [Consultado el: 11 de enero del 2011] Disponible en: http://www.ingenierosoftware.com/calidad/cmm-cmmi-nivel-2.php

Yamilka Días Toca, Yan Ayata Escalona. 2007. Referencia para la Gestión de Cambios en los proyectos productivos de la Facultad 3. [Consultado el: 2 de enero del 2011] Disponible en:

Anaya, Miguel Martínez. 2010. Propuesta de Plan de Gestión de Configuración de Software y Control de Cambios en el Departamento de Producción de Herramientas Educativas. [Consultado el: 3 de octubre del 2010] Disponible en:

WANdisco. 2000. Comunidad, y desarrolladores de CollabNet, Elego, VisualSVN, Subversion. [Consultado el: 13 de febrero del 2011] Disponible en: http://es.wikipedia.org/wiki/Subversion

Quiñones, Ernesto Azcárate. 2008. Herramientas de apoyo al desarrollo de software. [Consultado el: 13 de febrero del 2011] Disponible en: http://www.eqsoft.net/presentas/herramientas_libres_para_el_apoyo_al_desarrollo_de_software.pdf

Lang, Jean-Philippe. 2006. Redmine - Wikipedia, la enciclopedia libre. [Consultado el: 13 de febrero del 2011] Disponible en: http://es.wikipedia.org/wiki/Redmine

LNCS., Laboratorio Nacional de Calidad de Software. 2009. Guía práctica de Gestión de Configuración. [Consultado el: 16 de mayo del 2011] Disponible en: http://eva.uci.cu/file.php/102/Curso_20102011/Clases/Semana_03/Conferencia_5/Materiales_complementarios/Guia_practica_de_Gestion_de_Configuracion.pdf>

Ramses. 2011. 0515_Roles asociados a las áreas de proceso del nivel 2 de CMMI. [Consultado el: 14 de marzo del 2011] Disponible en: http://calisoft.uci.cu/index.php/proceso-de-mejora/Procesos