

Universidad de las Ciencias Informáticas
Facultad 3



**Título: Referencia para la Gestión de Cambios en los
proyectos productivos de la Facultad 3**

Trabajo de Diploma para optar por el título de
Ingeniero en Ciencias Informáticas

Autores: Yamilka Toca Díaz

Yan Ayata Escalona

Tutor: Maikel Yelandi Leyva Vázquez

Junio de 2007

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

Yan Ayata Escalona

Yamilka Toca Díaz

Maikel Yelandi Leyva Vázquez

Firma del Autor

Firma del Autor

Firma del Tutor

OPINIÓN DEL USUARIO DEL TRABAJO DE DIPLOMA

El Trabajo de Diploma, titulado “**Referencia para la Gestión de Cambios en los proyectos productivos de la Facultad 3**”, fue realizado en la Universidad de las Ciencias Informáticas. Esta entidad considera que, en correspondencia con los objetivos trazados, el trabajo realizado le satisface

Totalmente

Parcialmente en un ____ %

Los resultados de este Trabajo de Diploma le reportan a esta entidad los beneficios siguientes (cuantificar):

Como resultado de la implantación de este trabajo se reportará un efecto económico que asciende a <valor en MN o USD del efecto económico>

Y para que así conste, se firma la presente a los ____ días del mes de _____ del año _____.

Representante de la entidad

Cargo

Firma

Cuño

OPINIÓN DEL TUTOR DEL TRABAJO DE DIPLOMA

Título: Referencia para la Gestión de Cambios en los proyectos productivos de la Facultad 3

Autor: Yamilka Toca Díaz

Yan Ayata Escalona

El tutor del presente Trabajo de Diploma considera que durante su ejecución el estudiante mostró las cualidades que a continuación se detallan.

Por todo lo anteriormente expresado considero que el estudiante está apto para ejercer como Ingeniero Informático; y propongo que se le otorgue al Trabajo de Diploma la calificación de _____.

Maikel Yelandi Leyva Vázquez

Firma

Fecha

Agradecimientos

Nuestro agradecimiento a todas las personas que de una forma u otra han hecho posible la creación de este trabajo.

A nuestros padres que siempre nos han guiado por el buen camino.

A nuestro tutor por ayudarnos y guiarnos a medida que nuestro trabajo iba avanzando.

A los sabios consejos y enseñanzas de todos los maestros y profesores que han pasado por nuestra vida estudiantil.

Sin lugar a dudas este trabajo no podría haberse realizado sin la formación que recibimos en la Universidad de las Ciencias Informáticas, es por ello que le damos nuestro más profundo agradecimiento a nuestro Comandante en Jefe Fidel Castro por su creación.

De Yamilka:

Deseo agradecer intensamente a la oportunidad que la vida me otorgó al haberme hecho germinar del vientre de una mujer tan maravillosa como lo es mi mamá, pero más aun le agradezco a mi abuelita que fue quien hizo posible que ella existiera. Agradezco mucho a la despreocupada coincidencia que hizo a mi papá haberme sembrado, y no por estar yo viva sino por ser precisamente él quien haya dado lugar a mi existencia. Papi gracias por haber llegado a mi vida, aquel día que después de haberte conocido todo se tornó maravilloso. Por otro lado tengo a mis hermanas que sin el apoyo en todo sentido de ellas, el placer cotidiano de vivir sería simple monotonía, la inocencia de mi hermanito que me hace recordar lo mucho que he crecido. Gracias a mis padres, hermanos y abuela por compartir sus vidas conmigo, por quererme tanto aunque a veces no lo merezca, por creer en mí en todo momento, por alentarme a seguir y enseñarme que siempre puede ser peor.

Deseo agradecer a los profesores de la Universidad Yoansy López, Pascual Verdecia y Carlos Yasmany Hidalgo por sus consejos y ayuda en todo momento, y por soportar mis malcriadeces. Y a mi profesora guía del PRE Nancy Richard que en esos tres años fue como una madre.

Amigos, sin ustedes a mi lado no lo hubiera logrado, les agradezco a ustedes con toda mi alma por haber llegado a mi vida, y dejar en ella un pedacito de sí por compartir momentos tan agradables, otros muy tristes pero esos momentos son los que nos hacen crecer y valorar a las personas que nos rodean, los quiero mucho, nunca los olvidaré.

Dedicatoria

De Yamilka:

Esta tesis la dedico con todo mi amor y cariño a Dios por darme la fuerza espiritual.

En especial a mi abuelita y a mi hermana menor porque en ellas no conozco la más sencilla de las maldades, en ellas solo he visto amor y comprensión.

Con mucho amor a mis padres que me dieron la vida y aunque siempre no han estado a mi lado, cuando están logran llenar los días pasados. La dedico también a esa persona que sin ser padre, ni madre, ni amigo, lo es.

Resumen

La principal meta de todo proyecto de desarrollo de software es lograr que su proceso tenga la mayor eficiencia, objetivo inalcanzable sin la aplicación de procesos de control adecuados. Gestionar los Cambios es una de las mejores prácticas para todo proceso de software recomendadas por RUP. Esta idea se ha visto frenada en la Facultad 3 de la Universidad de las Ciencias Informáticas por la poca experiencia de los integrantes de sus equipos de desarrollo. El presente trabajo, **Referencia para la Gestión de Cambios en los proyectos productivos de la Facultad 3**, surgió por la necesidad de contar con un modelo que permitiera definir los procedimientos a seguir para implementar un adecuado proceso de Gestión de Cambios en los procesos productivos de la Facultad 3. Se estudian los temas relacionados con la Gestión de Configuración de Software, y con la Gestión de Cambios específicamente. Luego se analizan las diferentes herramientas que sirven para automatizar el proceso de Gestión de Cambios, y los proyectos productivos de la Facultad 3. Se define un modelo para la Gestión de Cambios, constituido por un proceso para gestionar las Solicitudes de Cambio, técnicas para obtener un mejor provecho en las reuniones del Comité de Control de Cambios, los roles por los que debe estar integrado y un conjunto de métricas para la evaluación de la aplicación del modelo. Se propone una herramienta de Software Libre para automatizar parte del proceso de Gestión de Cambios. Finalmente toda la propuesta es validada por especialistas en el tema.

Índice

INTRODUCCIÓN	1
CAPÍTULO I. FUNDAMENTACIÓN TEÓRICA	5
1.1 INTRODUCCIÓN	5
1.2 GESTIÓN DE LA CONFIGURACIÓN DEL SOFTWARE. CONCEPTO.....	5
1.3 ORÍGENES DE LA GESTIÓN DE LA CONFIGURACIÓN DEL SOFTWARE.....	5
1.4 METODOLOGÍAS DE DESARROLLO Y LA GCS	7
1.5 CONCEPTOS Y ACTIVIDADES DE LA GCS.....	8
1.5.1 <i>Conceptos necesarios</i>	8
1.5.2 <i>Actividades que comprende la Gestión de Configuración del Software</i>	10
1.6 GESTIÓN DE CAMBIOS	13
1.6.1 <i>Proceso de Gestión de Cambios</i>	15
1.6.2 <i>Mecanismo para solicitar cambios</i>	18
1.7 HERRAMIENTAS QUE SOPORTAN EL PROCESO DE GESTIÓN DE CAMBIOS	20
1.8 CARACTERÍSTICAS DE LA PRODUCCIÓN EN LA FACULTAD 3	27
1.9 CONCLUSIONES	27
CAPÍTULO II. PROPUESTA PARA LA GESTIÓN DE CAMBIOS	28
2.1 INTRODUCCIÓN	28
2.2 CARACTERÍSTICAS DE LOS PROYECTOS PRODUCTIVOS E IMPLICACIONES PARA LA GESTIÓN DE CAMBIOS	28
2.3 NIVEL DE GESTIÓN DE CAMBIOS A APLICAR.....	28
2.4 LA SOLICITUD DE CAMBIO, EL INFORME DE CAMBIOS Y LA ORDEN DE CAMBIO DE INGENIERÍA	29
2.5 MODELO PROPUESTO	30
2.6 DESCRIPCIÓN DEL PROCESO A UTILIZAR	30
2.7 COMPOSICIÓN DEL COMITÉ DE CONTROL DE CAMBIOS. RESPONSABILIDADES.....	33
2.8 REUNIÓN PARA LA REVISIÓN DE SOLICITUDES DE CAMBIO	34
2.8.1 <i>Aspectos a considerar durante la Reunión de Revisión del CCC</i>	35
2.8.2 <i>Técnicas para el desarrollo de la reunión</i>	36
2.8.3 <i>Establecimiento de prioridades para las Solicitudes de Cambio.</i>	37

2.9 HERRAMIENTA PROPUESTA PARA SOPORTAR EL PROCESO DE GESTIÓN DE CAMBIOS.....	39
2.10 CONCLUSIONES	41
CAPÍTULO III. HERRAMIENTA PROPUESTA Y MÉTRICAS DEL PROCESO	42
3.1 INTRODUCCIÓN	42
3.2 CONFIGURACIÓN DE LA HERRAMIENTA PROPUESTA	42
3.2.1 <i>Instalación</i>	42
3.2.3 <i>Configuración de parámetros</i>	43
3.2.4 <i>Entrada de datos del proyecto</i>	43
3.2.6 <i>Usuarios</i>	45
3.2.7 <i>Adición de campos personalizados</i>	46
3.2.8 <i>Creación de Órdenes de Cambio de Ingeniería</i>	47
3.2.9 <i>Informes</i>	47
3.3 MÉTRICAS PARA EL PROCESO	48
3.4 VALIDACIÓN POR ESPECIALISTAS.....	50
3.5 CONCLUSIONES	52
CONCLUSIONES.....	53
RECOMENDACIONES.....	54
REFERENCIAS BIBLIOGRÁFICAS.....	55
BIBLIOGRAFÍA.....	57
GLOSARIO DE TÉRMINOS	58

Introducción

El mundo en que vivimos es cada vez más dependiente de las nuevas tecnologías de la Informática y las Comunicaciones, por ello es imprescindible no rezagarse en el conocimiento y uso de las mismas, Cuba no está exenta de ello. La Universidad de las Ciencias Informáticas ha sido creada con el fin de que nuestro país logre hacer competitivos en el mercado mundial sus productos de software. El plan de estudios de la Facultad 3, al igual que el de las demás, ha sido concebido para que los estudiantes se vinculen al proceso de producción de software desde la docencia. Aunque cuentan con el apoyo de profesores y especialistas de las empresas y entidades clientes, son ellos los encargados de realizar la mayor parte de las actividades. Debido a la poca experiencia acumulada en los procesos productivos de la Universidad, se requiere prestar especial atención a las técnicas de control en cada uno de los proyectos para obtener productos con calidad. Esto es apreciable a la hora de controlar los cambios que pueden ocurrir en los elementos de un sistema durante su ciclo de vida.

Para este estudio se aplicó una encuesta en 8 de los proyectos existentes en la Facultad 3, acerca de la aplicación o no en los mismos de procesos de Gestión de Cambios, se analizan a continuación los resultados de la misma:

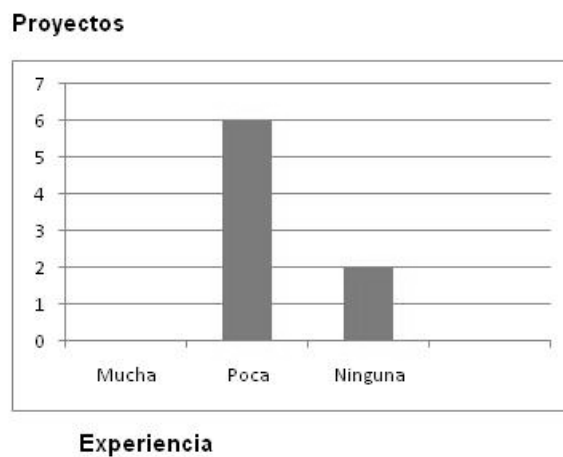


Figura 1. Experiencia del líder en la Gestión de Cambios.

Como se puede apreciar el líder del proyecto, quien es el responsable principal de los resultados de su grupo, en dos de los casos (25% de los encuestados) manifestó no tener experiencia previa en estas actividades y el 75% tiene poca experiencia.

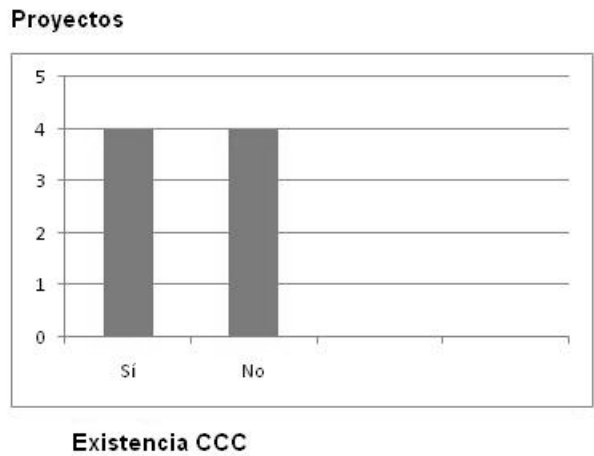


Figura 2. Existencia de un Comité de Gestión de Cambios

En la mitad de los proyectos encuestados no existe un Comité de Control de Cambios, lo que implica desorganización a la hora de realizar y registrar los cambios, y en muchas ocasiones la toma de decisiones erróneas.

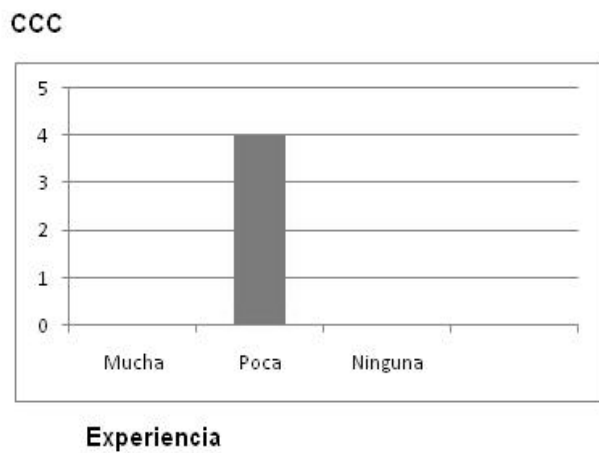


Figura 3. Experiencia del Comité de Gestión de Cambios

La totalidad de los Comités de Gestión de Cambios existentes tiene poca experiencia en sus actividades. Realizando un análisis del resultado de las encuestas y de los gráficos mostrados se pudo concluir que en la mayoría de los proyectos de esta facultad trabajan estudiantes que se encuentran en segundo o tercer año, lo que implica no haber recibido aún o encontrarse recibiendo los conocimientos relacionados con la Gestión de Configuración del Software, perteneciente al programa de la asignatura de Ingeniería y Gestión de Software, esto por supuesto que es uno de los factores que influye negativamente en la entrega a tiempo de los productos y en la calidad de los mismos. En la [Figura 1](#), se puede observar que la mayoría de los líderes de proyecto carecen de experiencia sobre el tema, o tienen muy poca, lo que es otro problema, pues su papel es imprescindible en esta actividad. Observando la [Figura 2](#) se puede ver que el 50% de los proyectos analizados no tienen un Comité de Gestión de Cambios, que se encargue de la toma de decisiones en esta actividad. La experiencia de los comités existentes es poca y en algunos ninguna. A través de la encuesta también se pudo conocer que ningún proyecto de los analizados utiliza herramientas para la Gestión de Cambios, las cuales son de gran importancia en este proceso pues automatizan y agilizan el trabajo.

Todo lo anterior permite concluir que en esta Facultad no se constituye correctamente el Comité de Control de Cambios en los proyectos, además no están definidos claramente los procesos de Gestión de Cambios, y debido a la falta de experiencia de los integrantes no se emplean herramientas para automatizar este proceso, incluso existen proyectos donde ni siquiera existe un Comité de Control de Cambios.

El **problema científico** es el siguiente: El ineficiente proceso de Gestión de Cambios es un factor que está afectando negativamente los procesos productivos en la Facultad 3.

Esta tesis se plantea como **objetivo general** desarrollar un modelo para la Gestión de Cambios adaptado a las características de la Facultad 3. Se tiene **como objeto** de estudio la Gestión de Configuración del Software en proyectos informáticos y el **campo de acción** es el proceso de Gestión de Cambios en los proyectos productivos de la Facultad 3.

Como **objetivos específicos** se tienen:

- elaborar el marco teórico
- desarrollar el modelo para la Gestión de Cambios,
- proponer qué herramienta de software libre puede ser utilizada para la Gestión de Cambios en los proyectos productivos de la Facultad 3, y

- definir una guía que ayude a adoptar los procedimientos de Gestión de Cambios que se proponen
- validar con especialistas el modelo propuesto

Hipótesis: si se desarrolla un modelo para la Gestión de Cambios adaptado a las particularidades de la Facultad 3, se influirá en la mejora de los procesos productivos.

Para lograr los objetivos trazados se realizarán las **tareas** siguientes:

1. Realizar un estudio del estado del arte de la Gestión de Cambios, para comprender este proceso y realizar un análisis crítico del mismo.
2. Estudiar las características fundamentales de los procesos de desarrollo de software en la Facultad 3 y las implicaciones que tienen en la Gestión de Cambios.
3. Proponer composición y responsabilidad del Comité de Control de Cambios, para que este se constituya en los proyectos de la Facultad 3.
4. Analizar y seleccionar técnicas que puedan ser aplicadas en las reuniones del Comité de Control de Cambios.
5. Seleccionar y definir las actividades y métricas que formarán parte del modelo.
6. Analizar y seleccionar una herramienta de software libre que pueda ser utilizada para automatizar tanto como sea posible el proceso de Gestión de Cambios.

El contenido del presente trabajo está organizado de la siguiente forma:

En el Capítulo I se elabora el marco teórico analizando los conceptos relacionados con la Gestión de Configuración del Software y la Gestión de Cambios, las herramientas que se utilizan para automatizar este proceso y se describen los polos y proyectos de la Facultad 3 de la Universidad de las Ciencias Informáticas.

En el Capítulo II se propone un modelo para implementar la Gestión de Cambios en los proyectos de la Facultad 3, se describe por quiénes debe estar constituido el Comité de Control de Cambios, qué elementos se deben discutir en la Reunión de Revisión del Comité de Control de Cambios, se sugiere una técnica para establecer las prioridades de las Solicitudes de Cambio, y por último se propone una herramienta de software libre para automatizar el proceso de Gestión de Cambios.

El Capítulo III describe la configuración de la herramienta propuesta, el sistema de métricas a utilizar para analizar los resultados de la aplicación del modelo, se valida la propuesta a través de especialistas en el tema y se valoran sus posibles impactos.

Capítulo I. Fundamentación Teórica

1.1 Introducción

El presente capítulo tiene como objetivo introducir los principales conceptos y términos utilizados en la Gestión de la Configuración del Software, su surgimiento y evolución, así como su importancia para el proceso de desarrollo de software. Será abordado con mayor profundidad, el proceso de Gestión de Cambios, tema central de la investigación, se brindan detalles de las herramientas que automatizan este proceso, para finalmente hacer una descripción de la producción en la Facultad 3.

1.2 Gestión de la Configuración del Software. Concepto.

Son varios los autores que se pueden mencionar al tratar de definir este concepto, uno de los más sencillos lo aporta Bellagio (1) planteando que es la forma en que se controla la evolución de un proyecto de software, aunque luego añade que es una disciplina de la ingeniería de software que comprende las herramientas y técnicas (procesos o metodologías) que una empresa usa para gestionar los cambios en sus recursos de software. Según Pressman (2) “es una actividad de autoprotección que se aplica durante el proceso del software”, sirve para identificar los cambios, controlarlos, garantizar que se implementen adecuadamente e informar a todos aquellos que puedan estar interesados.

Otra definición es la siguiente “...es una disciplina, cuya misión es controlar la evolución de un sistema software” (3).

Para (4) es “El arte de coordinar el desarrollo de software para minimizar la confusión...La Gestión de Configuración es el arte de identificar, organizar y controlar las modificaciones que sufre el software que construye un equipo de programación. El objetivo es maximizar la productividad minimizando los errores.”

En resumen se puede decir que es una disciplina aplicada a lo largo del todo el proceso de desarrollo de software, cuya misión es llevar a cabo el control de los cambios que puedan sufrir los artefactos que va produciendo el proceso de ingeniería de software.

1.3 Orígenes de la Gestión de la Configuración del Software

Los orígenes de la Gestión de Configuración en el mundo de la informática no son precisamente en el campo del desarrollo de software, sino que fue usada originalmente para el control de la producción y desarrollo de hardware. La mayoría de las referencias (5) que se tienen son con respecto a las

herramientas que automatizan las áreas de Gestión de Configuración del Software y las compañías que las producen. A continuación una breve cronología:

- Finales de los '60, principios de los '70: El Profesor Leon Pressor en la Universidad de California, realizó una tesis de control de cambios y control de configuración. Este concepto fue la respuesta a un contrato en el que estaba trabajando con alguien involucrado en la fabricación de motores de aeronaves para La Armada de los Estados Unidos.
- El libro Fundamentos de la Gestión de Configuración (6) en su prefacio indica: "El propósito de este libro es describir los fundamentos de la Gestión de Configuración para que así los estudiantes de ingeniería, diseñadores, ingenieros, gestores de proyectos y personal administrativo puedan tener un lenguaje común para trabajar eficientemente con sus colegas de Gestión de Configuración"
- En 1975 se publica un artículo en la IEEE: Sistema de Control de Código Fuente (SCCS), Marc Rochkind de los Laboratorios Bell.
- 197x: Pansophic's PANVALET, sistema de control de código para el mercado de mainframes.
- 1975: Con el tiempo el trabajo del Profesor Pressor se convirtió en un producto disponible comercialmente llamado Change and Configuration Control (CCC), el mismo fue vendido por la Empresa SoftTool.
- 1982: CMS (DEC). Code Management System. Soporta agrupamiento de versiones de archivos en clases.
- Revision Control System (RCS, Walter Tichy), es una implementación en software del control de versiones que automatiza el almacenamiento, la recuperación, identificación, y combinación de revisiones.
- 1984: Aide-de-Camp (ADC, actualmente TRUEchange alrededor de 1990, herramienta para el control de cambios y de versiones).
- 1986: Concurrent Version System (CVS) original. Aplicación informática que implementa un sistema de control de versiones, bajo licencia GPL.
- 1989: CVS aplicación informática que implementa un sistema de control de versiones.
- 1999: WEBDAV RFC2518 es publicado, extensiones al protocolo HTTP que permite a varios usuarios editar y administrar de forma colaborativa archivos en servidores remotos.

- Principios del 2000: se hicieron factibles sistemas distribuidos de control de versiones como BitKeeper y GNU Arch.
- 2002: Subversion. Sistema de control de versiones de código fuente abierto.

Hasta los '80, la Gestión de la Configuración del Software pudo ser solamente entendida como Gestión de la Configuración aplicada al desarrollo de software. Es solamente después de esta fecha que el uso de herramientas de software aplicadas directamente a los artefactos de software, ha permitido que la Gestión de Configuración del Software crezca como una disciplina autónoma.

1.4 Metodologías de desarrollo y la GCS

La Ingeniería de Software es “el establecimiento y uso de principios de ingeniería robustos, orientados a obtener software económico que sea fiable y funcione de manera eficiente sobre máquinas reales” (7). Al igual que en las demás ramas de la ciencia, en la producción de software se hace necesario establecer normas y principios que la guíen por caminos formales, que la hagan factible. Son varias las filosofías de desarrollo existentes, todas pretendiendo ser la que dicte el camino a seguir por los profesionales del software, y hacer de ellos los más productivos y capaces. Sin embargo el Proceso Unificado de Rational (RUP) de IBM que es el resultado de varios años de desarrollo y uso práctico en el que se han unificado técnicas de desarrollo a través del UML, ha logrado convertirse en la metodología estándar más utilizada. En el lado opuesto, están los denominados procesos ágiles, que intentan evitar los burocráticos caminos de las metodologías tradicionales enfocándose en las personas y los resultados. Intentan minimizar el riesgo, desarrollando el software en pequeñas iteraciones. Hacen énfasis en la comunicación directa entre los desarrolladores, y no a través de documentos, por lo que producen muy poca documentación, aspecto bastante criticado.

Su más alto exponente es XP o Programación Extrema, al igual que el resto de los procesos ágiles hace mayor hincapié en la adaptabilidad que en la previsibilidad. Los defensores de XP plantean que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos.

Veamos cómo es implementada la Gestión de Configuración del Software 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 Gestión de la Configuración del Software” (8).

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 Gestión de la Configuración 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 constata. (9)

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.

1.5 Conceptos y actividades de la GCS

1.5.1 Conceptos necesarios

Elemento de Configuración de Software

Para la Gestión de Configuración del Software, Configuración del Software es "toda la información o productos utilizados o producidos en un proyecto como resultado del proceso de Ingeniería de Software" (3). Configuración del Software es todo lo que sea puesto bajo control por las actividades de Gestión de Configuración del Software, como pueden ser:

- La especificación del sistema.
- El plan del proyecto.
- La especificación de requisitos.
- Un prototipo, ejecutable o en papel.
- El diseño preliminar.
- El diseño detallado.
- El código fuente.
- Programas ejecutables.

- El manual de usuario.
- El manual de operación e instalación.
- El plan de pruebas.
- Los casos de prueba ejecutados y los resultados registrados.
- Los estándares y procedimientos de ingeniería de software utilizados.
- Los informes de problemas.
- Las peticiones de mantenimiento.
- Los productos hardware y software utilizados durante el desarrollo.
- La documentación y manuales de los productos hardware y software utilizados durante el desarrollo.
- Diseños de bases de datos.
- Contenidos de bases de datos.

Para cada proyecto concreto se debe determinar qué se va a considerar como ECS. Cada uno debe constituir un elemento completo que se pueda controlar por separado.

Línea base

Una línea base es según la (10) una especificación o producto que se ha revisado formalmente y sobre la que se ha llegado a un acuerdo, y que de ahí en adelante sirve como base para un desarrollo posterior y que puede cambiarse solamente a través de procedimientos formales de control de cambios. Para (2) las líneas base nos ayudan a controlar los cambios sin impedir los cambios justificados.

“Una línea base representa una configuración estable para un grupo de archivos y directorios. Una línea base identifica actividades y una versión para cada elemento visible en uno o más componentes. Una nueva línea base incluye el trabajo que los desarrolladores han entregado a la cadena de integración desde la última línea base. Haciendo nuevas líneas base regularmente, se asegura que los desarrolladores permanecen sincronizados con los trabajos de cada uno de los otros.” (11)

Una línea base constituye “un punto de referencia en el proceso de desarrollo del software que queda marcado por la aprobación de uno o varios Elementos de Configuración del Software, mediante una revisión técnica formal.” (3)

Una línea base permite hacer formal un conjunto sucesivo de cambios que se han producido de manera informal, para su aceptación y formalización como elemento de referencia para futuros cambios.

Versiones

Para (3) 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”.

A las distintas versiones que aparecen en el tiempo, según se va avanzando en el desarrollo de un elemento, se les suele llamar también **revisiones**. La Gestión de Configuración debe permitir también especificar y gestionar distintas **variantes** de los elementos de configuración. Las variantes son versiones de un elemento de configuración que coexisten en un determinado momento y que se diferencian entre sí en ciertas características. Las variantes representan la necesidad de que un objeto satisfaga distintos requisitos al mismo tiempo. A diferencia de las revisiones, que son estrictamente secuenciales, y sólo existe una como revisión actual, las variantes se desarrollan en paralelo, y puede haber varias sobre las que se esté trabajando simultáneamente. Una variante no reemplaza a otra, como ocurre con las revisiones, sino que abre un nuevo camino de desarrollo.

Repositorio

Es un término utilizado en el dominio de las herramientas CASE. 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 o en un medio físico como un disco compacto.

1.5.2 Actividades que comprende la Gestión de Configuración del Software

La Gestión de Configuración del Software forma parte de las disciplinas de control en el desarrollo del software, específicamente se encarga del control de los cambios, según el Modelo de Madurez de Capacidades del Instituto de Ingeniería de Software (SEI CMM) (12), “La Gestión de Configuración y Solicitudes de Cambio”, controla el cambio y mantiene la integridad de los artefactos de un proyecto.

Plantea (2) que estos cambios pueden estar referidos a:

- nuevos negocios o condiciones comerciales que dictan los cambios en los requisitos del producto o en las normas comerciales
- nuevas necesidades del cliente que demandan la modificación de los datos producidos por sistemas de información, funcionalidades entregadas por productos o servicios entregados por un sistema basado en computadora
- reorganización o crecimiento/reducción del negocio que provoca cambios en las prioridades del proyecto o en la estructura del equipo de ingeniería del software
- restricciones presupuestarias o de planificación que provocan una redefinición del sistema o producto

RUP (8) considera que estos cambios pueden ser para incluir nuevas funcionalidades, debido a solicitudes de mejoras, defectos o cambios en los requerimientos.

Ahora, veamos qué importancia tiene la Gestión de Configuración del Software. Sus objetivos son al parecer de (3):

- establecer y mantener la integridad de los productos generados durante un proceso de desarrollo de software y a lo largo de todo el ciclo de vida del producto
- evaluar y controlar los cambios sobre ellos, es decir, controlar la evolución del sistema de software
- facilitar la visibilidad sobre el producto

Un sistema de Gestión de Configuración es esencial para controlar los numerosos artefactos producidos por muchas personas que trabajan en un proyecto común. El control ayuda a evitar la costosa confusión, y asegura que los artefactos resultantes no estén en conflicto debido a algunos de los siguientes tipos de problemas:

- Actualización simultánea
- Notificación limitada
- Versiones múltiples (8)

Para lograr cumplir con sus actividades la Gestión de Configuración del Software se divide en las siguientes tareas:

- *Identificación de la Configuración:* Consiste en identificar la estructura del producto, sus componentes y el tipo de estos, y en hacerlos únicos y accesibles de alguna forma.
- *Gestión de Cambios en la Configuración:* Consiste en controlar las versiones y entregas de un producto y los cambios que se producen en él a lo largo del ciclo de vida.
- *Generación de Informes de Estado:* Consiste en informar acerca del estado de los componentes de un producto y de las solicitudes de cambio, recogiendo estadísticas acerca de la evolución del producto.
- *Auditoría de la Configuración:* Consiste en validar la completitud de un producto y la consistencia entre sus componentes, asegurando que el producto es lo que el usuario quiere. (3)

En la figura mostramos las actividades que propone RUP (8) en el Flujo de trabajo que define como Gestión de Configuración y Cambios, así como los trabajadores que responden por ellas:



Figura 1.1. Actividades del Flujo de Trabajo: Gestión de Configuración y Cambios.

Se pueden apreciar cuáles son los roles involucrados en este proceso, así como las actividades que realizan, en el siguiente epígrafe profundizaremos en el proceso de Gestión de Cambios, centro de nuestro trabajo.

1.6 Gestión de Cambios

En su trabajo (2) cita a James Bach, al decir que la Gestión de Cambios en el contexto de la Ingeniería de Software moderna "...es vital. Pero las fuerzas que la hacen necesaria también la hacen molesta..." El problema es que hay que saber equilibrar la forma en que se implementa, porque un excesivo uso puede llegar a convertirlo en una pesada carga que en vez de resolver, cree problemas. Pero ante todo es la vía a seguir para evitar que se propague la confusión en los equipos de desarrollo.

Este primer acercamiento a esta disciplina puede hacer parecer que, lejos de traer beneficios, podría llegar a entorpecer el proceso de desarrollo, por la carga adicional que supone el tener que disponer de personal que se encargue de estas actividades, así como la necesidad de adquirir nuevos conocimientos.

Debido a que "no importa en qué momento en el ciclo de vida se encuentra el sistema, este va a cambiar y el deseo de cambio persistirá a lo largo de todo el ciclo de vida del mismo" (13), la Gestión de Cambios constituye la actividad de Gestión de Configuración más importante y su objetivo es "proporcionar un mecanismo riguroso para controlar los cambios" (3). Es decir que es preciso estar preparado para enfrentar el cambio de manera adecuada, en el momento que sea preciso, pues es inevitable. Aunque muchas veces las solicitudes de cambio se producen durante la fase de mantenimiento del producto, esta puede aparecer en cualquier momento durante el ciclo de vida del software.

Angélica de Antonio (3) considera dos tipos fundamentales de cambios:

- Corrección de un defecto: Es la forma en que es visto generalmente por los clientes.
- Mejora del sistema: Así lo ven los desarrolladores.

Sobre esto es preciso señalar otra causa que pudiera generar la necesidad de un cambio: un cambio en el negocio que provoque la necesidad de cambiar los requisitos del sistema en cuestión.

Un cambio se puede realizar a diferentes niveles, (3) menciona los siguientes:

- Informal: Antes de que un Elemento de Configuración del Software pase a formar parte de una línea base, aquel que haya desarrollado el Elemento de Configuración del Software podrá realizar cualquier cambio justificado por el proyecto (siempre que no impacte en otros requisitos del sistema más amplios) sobre él.

- Al nivel del proyecto o semi-formal: Una vez que un Elemento de Configuración del Software pasa la revisión técnica formal y ha sido aprobado y por tanto se ha convertido en una línea base, para que el encargado del desarrollo pueda realizar un cambio debe recibir la aprobación de:
 - El jefe del proyecto, si es un cambio local
 - El Comité de Control de Cambios (o Autoridad de Control de Cambios), si el cambio tiene algún impacto sobre otros Elementos de Configuración del Software
- Formal: Se adopta cuando se ha distribuido el producto al cliente, o sea cuando se empieza a comercializar. Todo cambio deberá ser aprobado por el Comité de Control de Cambios.

Como se puede apreciar cuando el cambio es al nivel semi-formal o formal se encuentra mediándolo el Comité de Control de Cambios. El Comité de Control de Cambios es una persona o grupo de personas en dependencia del tamaño del proyecto, que se encarga de tomar las decisiones finales acerca del estado y la prioridad de las solicitudes de cambio. Su papel es el de tener una visión general, evaluar el impacto que pueda tener el cambio fuera del Elemento de Configuración de Software en que se produce, en aspectos como el hardware, el rendimiento, la percepción del cliente sobre el producto, y la calidad y fiabilidad del mismo. (2)

Además del Comité de Control de Cambios, tienen responsabilidad sobre el proceso de Gestión de Cambios todos los miembros del proyecto, pues pueden ser quienes soliciten los cambios o quienes los realicen, y el jefe del proyecto que es quien debe asegurar que los procedimientos de Gestión de Cambios, se llevan a cabo de forma adecuada, y debe evaluar aspectos tales como el impacto sobre el proyecto y el costo de proceder con el cambio. Tanto los miembros del proyecto que estén realizando los cambios, como el Jefe del Proyecto son responsables por mantener informados de los cambios al resto de los miembros.

El nivel de burocracia del proceso de Gestión de Cambios, puede causar incomodidad a los que no estén acostumbrados, más aún cuando se describa en detalle la forma de llevarlo a un nivel formal. Pero para (2), esto debe ser normal, ya que “sin la protección adecuada, la Gestión de Cambios puede demorar el progreso y crear un papeleo innecesario”.

Es necesario establecer de forma precisa, al comienzo de cada proyecto, cuál será el proceso de Gestión de Cambios que se va a utilizar, para ello se deben definir:

- Políticas a nivel organizativo que promuevan las actividades de Gestión de Cambios.
- Los estándares que se van a adoptar y a los que será necesario ajustarse.

- Los procedimientos que se van a utilizar para poner en práctica las políticas. (3)

1.6.1 Proceso de Gestión de Cambios

El proceso de Gestión de Cambios en un determinado proyecto, define los pasos a seguir una vez dada algunas de las condiciones que generan la necesidad de hacer una solicitud de cambio.

Los pasos a seguir para llevar a cabo la Gestión de Cambios formal, sobre una línea base existente los describe (2) de la siguiente manera:

Se hace una **Solicitud de Cambio** y se evalúa para calcular el esfuerzo técnico, los posibles efectos secundarios, el impacto global sobre otras funciones del sistema y sobre otros objetos de la configuración. Los resultados de la evaluación se presentan como un **Informe de Cambios** al Comité de Control de Cambios una persona o grupo de personas que toman la decisión final del estado y la prioridad del cambio. Para cada cambio aprobado se genera una **Orden de Cambio de Ingeniería (OCI)**. La OCI describe el cambio a realizar, las restricciones que se deben respetar y los criterios de revisión y de auditoría. El objeto a cambiar es “dato de baja” de la base de datos del proyecto y se realiza el cambio. Luego, el objeto es “dato de alta” en la base de datos y se usan los mecanismos de control de versiones apropiados para crear la siguiente versión del software.

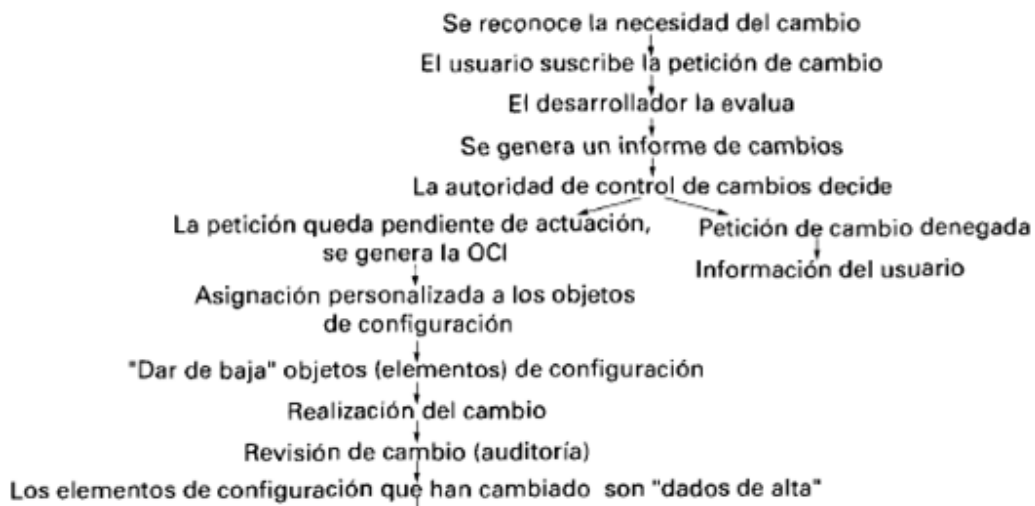


Figura 1.2. El proceso de Gestión de Cambios (2)

La visión que tiene (3) sobre cómo se debe llevar a cabo la Gestión de Cambios formal, sobre una línea base existente es la siguiente, nótese que incluye un paso final ausente en el descrito por Pressman (2), la notificación al originador del cambio luego de la realización del cambio (véase la [Figura 1.3](#)):

1. Iniciación del Cambio: se presenta una Solicitud de Cambio, que puede venir provocada por un problema que se ha detectado o por un cambio en los requisitos.
2. Clasificación y registro de la Solicitud de Cambio.
3. Aprobación o rechazo inicial de la Solicitud de Cambio. De ello suele ser responsable el *Comité de Control de Cambios*.
4. Evaluación de la Solicitud de Cambio, si ha sido aprobada, para calcular el esfuerzo técnico, los posibles efectos secundarios, el impacto global sobre otras funciones del sistema y el coste estimado del cambio. Como resultado se obtiene un *Informe de Cambio*.
5. Se presenta el Informe de Cambio al Comité de Control de Cambios. Si se considera que el cambio es beneficioso se genera una *Orden de Cambio* (también llamada Orden de Cambio de Ingeniería), que describe el cambio a realizar, las restricciones que se deben respetar y los criterios de revisión y de auditoría. Esta Orden de Cambio es asignada a alguno de los ingenieros de software para que se encargue de llevarlo a cabo. En este momento, el objeto a cambiar se da de baja.
6. Se realiza el cambio, entrando en un proceso de seguimiento y control.
7. Una vez finalizado el cambio, se certifica, mediante una revisión, que se ha efectuado correctamente el cambio y con ello se ha corregido el problema detectado o bien se han satisfecho los requisitos modificados. El objeto se da de alta.
8. Se notifica el resultado al originador del cambio.

En esta descripción de los pasos a seguir en un proceso formal o semi-formal, (3) pasa por alto evaluar el impacto que pueda tener el cambio propuesto, sobre otros Elementos de Configuración de Software, que incluso pueden ser de línea base. Además hace responsable al Comité de Control de Cambios de la revisión inicial de todas las Solicitudes de Cambio recibidas, algo que puede resultar muy engorroso.

Continúa (3) refiriéndose a un proceso de Gestión de Cambios semi-formal diciendo que “puede suprimirse la necesidad de generar la Solicitud de Cambio, el Informe de Cambios y la Orden de Cambio, pero sí debe realizarse la evaluación del cambio y su seguimiento.”

Considera necesario:

- Definir los mecanismos para solicitar cambios sobre los Elementos de Configuración.
- Definir los mecanismos para analizar y evaluar el impacto de las Solicitudes de Cambio.
- Definir los mecanismos para aprobar o rechazar las Solicitudes de Cambio.
- Definir los mecanismos para controlar la realización de los cambios aprobados.

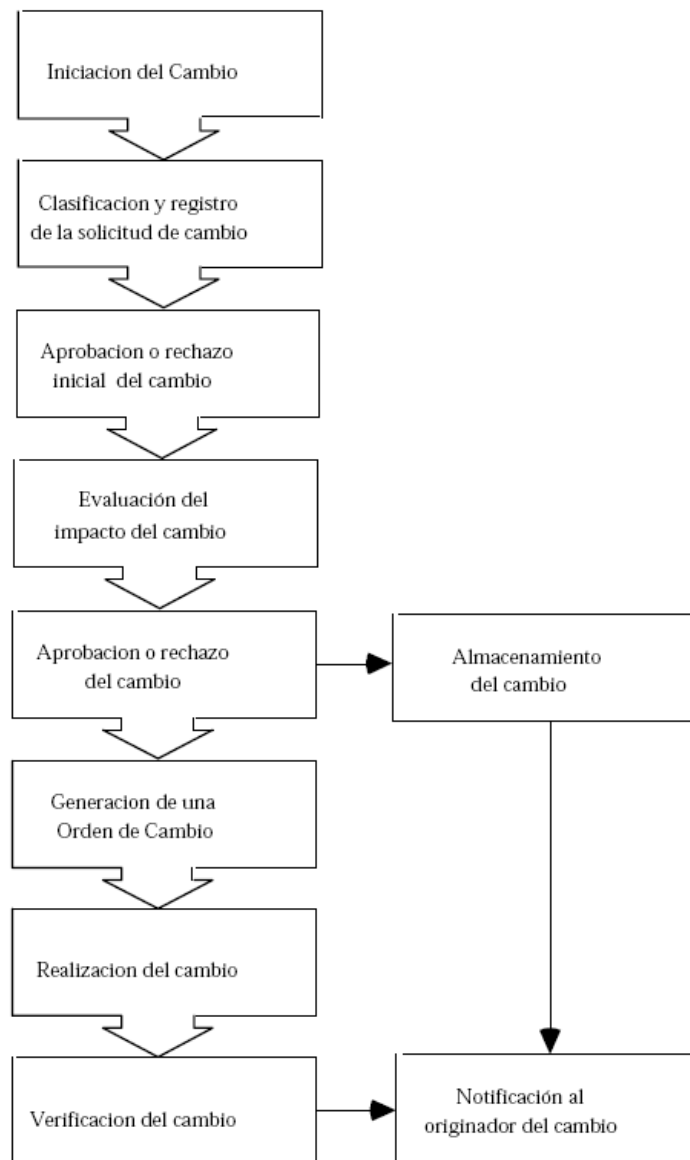


Figura 1.3. El proceso de Gestión de Cambios (3)

1.6.2 Mecanismo para solicitar cambios

Como ya fue dicho hay que dejar bien definido cuál va a ser el mecanismo para solicitar cambios. Se debe establecer por tanto qué contendrá la planilla o formulario a utilizar para las Solicitudes de Cambio. Entre los aspectos que no deben faltarle están:

- Razón
- Elementos de Configuración de Software a cambiar
- Solicitud a nombre de...
- Descripción detallada del problema para que se le pueda recomendar solución
- Descripción de qué se puede ver afectado por el cambio

En cualquier caso debe ser simple, y aceptado por las personas que lo tendrán que manejar. El formulario más común es el de Solicitud de Cambio. Sin embargo, se pueden utilizar formularios diferentes para solicitar una mejora y para informar de un problema o deficiencia detectada durante una auditoría, una prueba o el uso del sistema, que posiblemente requerirá un cambio.

RUP (8), en el Flujo de Trabajo que denomina Gestión de Cambios y Configuración propone construir el artefacto Solicitud de Cambio, el cual considera que debe registrar las decisiones tomadas en cuanto a los cambios para que en el proyecto se comprendan los impactos que pueda tener. El trabajador responsable es el Gestor de Cambios, el que debe asegurarse mediante la generación y control de las Solicitudes de Cambio, que los cambios al sistema se realicen de manera controlada, de forma tal que se pueda predecir su efecto en el sistema.

A los formularios que se usan para informar de problemas se les suele llamar Informes de Incidencias o Informes de Problemas. Recogen información adicional sobre el incidente que ha que ha expuesto la existencia del problema:

- fecha y hora en que ocurrió
- descripción del incidente
- efectos que ha producido
- referencia al tipo de prueba que se estaba efectuando

Este Informe de Incidencia es analizado por los desarrolladores, y estos pueden recomendar alguna de las siguientes acciones:

- No requiere acción: Cuando lo que se describe en el informe de incidencia no es realmente una deficiencia. Esta situación suele ser debida a malentendidos acerca de la forma de funcionamiento

del sistema. También se puede dar esta situación cuando ya se ha informado previamente de una incidencia similar, y se están tomando las acciones correctivas necesarias.

- **Solicitud de Cambio:** La implementación se corresponde con el diseño del sistema, pero una mejora en el diseño del sistema solucionaría el problema. Se genera entonces una Solicitud de Cambio, que se tratará por los cauces normales.
- **Notificación de Cambio:** Cuando la deficiencia que se describe en el informe de incidencia se debe a una mala implementación que debe ser corregida. Se informa entonces al Comité de Control de Cambios y se pasa a corregir la deficiencia.

Una Notificación de Cambio es otro formulario que, puede dar respuesta a varios Informes de Incidencias, cuando todos ellos corresponden al mismo problema.

Para solicitar cambios en la documentación se pueden utilizar los formularios anteriores o un nuevo formulario al que se suele llamar Formulario de Seguimiento de la Documentación.

Mecanismo para aprobar o rechazar las Solicitudes de Cambio

Algunos criterios que se pueden tener en cuenta para tomar la decisión de aprobar o rechazar las Solicitudes de Cambio son los siguientes:

- valor del cambio para el proyecto u organización
- retorno de la inversión
- tamaño
- complejidad
- impacto sobre el rendimiento del producto
- recursos disponibles para efectuar el cambio
- relación con otros cambios ya aprobados y en progreso
- tiempo estimado para completar el cambio
- existencia de alternativas

Seguimiento de los cambios. Gestión de Problemas

Una vez aprobado un cambio debido a un problema se debe realizar un seguimiento del mismo. A este proceso se le llama Gestión de Problemas.

La Gestión de Problemas se considera una actividad complementaria a la de Control de Cambios, que consiste en gestionar la evolución de los problemas detectados sobre el software desarrollado, tanto aquellos que se detectan en la fase de pruebas como los informes de problemas que llegan del usuario.

Algunos de los datos que puede resultar interesante almacenar acerca de los problemas son:

- descripción
- severidad
- urgencia o prioridad
- causa del problema (omisiones en el análisis, error en la documentación de entrada, falta de experiencia)
- solución al problema
- módulos afectados
- persona que lo notificó
- persona responsable
- fechas de notificación, resolución...
- fase o etapa en la que se originó el problema
- fase o etapa en la que se detectó el problema

1.7 Herramientas que soportan el proceso de Gestión de Cambios

Son numerosas las herramientas que se pueden encontrar en el mercado, a continuación se describen las características de algunas de ellas, las facilidades que ofrecen. Puede ser encontrada más información al respecto en (14).

20s Change Coordinator

20s Change Coordinator, es un producto de 20smackers, permite:

- Gestionar el proceso de solicitudes de cambios de software
- Realizar seguimiento de defectos
- Construir un proceso de gestión de Solicitudes de Cambio de software específico para el usuario
- Pre-establecer valores aceptables de entrada asociados con las Solicitudes de Cambio
- Archivar registros de Solicitudes de Cambio según se necesite
- Mantener un hilo de comunicación por Solicitud de Cambio

- Generar informes impresos por cada Solicitud de Cambio

20s Change Coordinator simplifica el proceso de control de Solicitudes de Cambio. El enfoque que brinda permite al usuario configurar la herramienta, para adecuar los detalles del proceso de solicitudes de cambio usado al entorno del usuario.

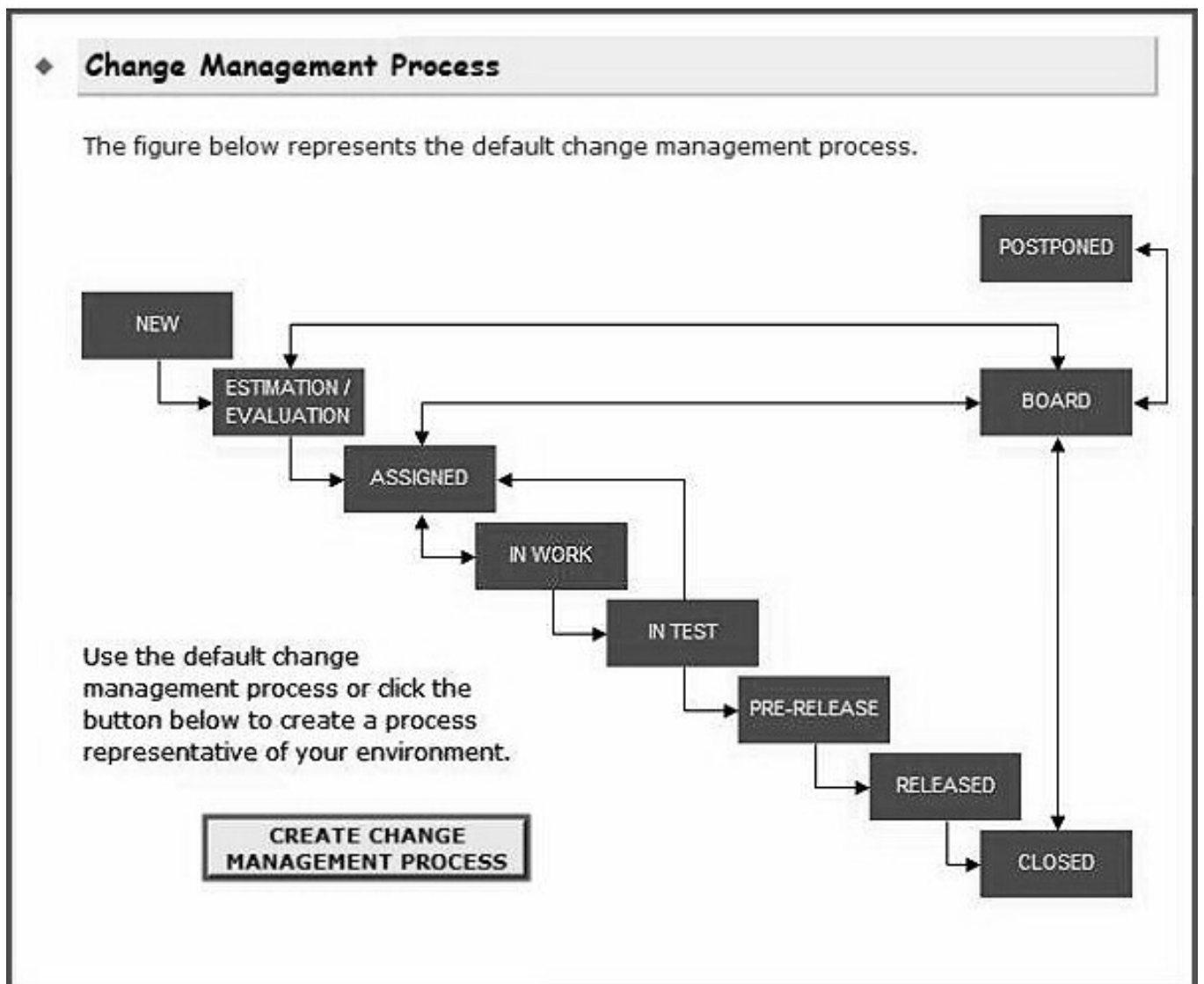


Figura 1.4. Proceso de Gestión de Cambios en 20s Change Coordinator

Los registros de Solicitudes de Cambio se representan con un ID único, y una vez ingresados pueden ser editados utilizando la opción “Edit” o haciendo doble clic sobre la celda a modificar.

CHANGE_REQUEST						ADD RECORD	ACTIONS...	LINKS...	?	www.20smackers.com
7 <= TOTAL CHANGE REQUESTS										
* ID	NUMBER OF COMMENTS	CHANGE TYPE	* STATE	ASSIGNED TO	* TITLE	REPORTED BY	PRIORITY	* DATE SUBMITTED		
01-CR	0	BUG	CLOSED	TOM	Change Request #1	ELLEN	HIGH	11/18/03		
02-CR	3	FUNCTIONALITY	IN WORK	SUE	Change Request #2	ROB	LOW	11/18/03		
03-CR	0	LOOK & FEEL	IN TEST	SAM	Change Request #3	SAM	MEDIUM	11/18/03		
04-CR	0	USER INTERFACE	ASSIGNED	ROB	Change Request #4	SUE	HIGH	11/18/03		
05-CR	2	LOOK & FEEL	IN TEST	ELLEN	Change Request #5	TOM	LOW	11/18/03		
06-CR	1	FUNCTIONALITY	IN TEST	SUE	Change Request #6	SAM	HIGH	11/18/03		
07-CR	0	USER INTERFACE	IN TEST	SAM	Change Request #7	ROB	MEDIUM	11/18/03		

Figura 1.5. Área de Gestión de Cambios en 20s Change Coordinator

El hilo de comunicación asociado con cada registro de solicitud de cambio, es almacenado por separado y se accede a través de un vínculo desde el registro de la solicitud de cambio padre.

El responsable de esta actividad puede desear saber el número de Solicitudes de Cambio asignadas por cada individuo, o el porcentaje de Solicitudes de Cambio con prioridad “alta”, activas en un momento determinado. 20s Change Coordinator satisface esta necesidad a través de una matriz bidimensional. Ver [Figura 1.5](#).

BMC Remedy Change Management

Según BMC, la compañía a cargo del desarrollo de Remedy Change Management, esta herramienta proporciona normas, gestión de procesos y capacidad de planificación para incrementar la velocidad y coherencia en la implantación de los cambios y, al mismo tiempo, minimizar los riesgos para el negocio (15). Desde la solicitud del cambio hasta su verificación, pasando por su planificación y aplicación, BMC Remedy Change Management ayuda a evaluar el impacto, los riesgos y los requisitos de recursos asociados a los cambios.

CHANGE REQUEST TOTALS																
											LINKS...	7	www.20smakers.com			
CLEAR	* ID	NUMBER OF COMMENTS	CHANGE TYPE	* STATE	ASSIGNED TO	* TITLE	REPORTED BY	PRIORITY	* DATE SUBMITTED	VERSION FOUND	CHANGE OWNER	* DAYS OPEN	VERSION FIXED	TARGET RELEASE		
CLEAR																
* ID			PERCENT	43%	29%	29%										
NUMBER OF COMMENTS			TOTALS	3	2	2										
CHANGE TYPE	PERCENT	TOTALS	7	HIGH	LOW	MEDIUM										
* STATE	14%	1	TOM	1	0	0										
ASSIGNED TO	29%	2	SUE	1	1	0										
* TITLE	29%	2	SAM	0	0	2										
REPORTED BY	14%	1	ROG	1	0	0										
PRIORITY	14%	1	ELLEN	0	1	0										
* DATE SUBMITTED																
VERSION FOUND																
CHANGE OWNER																
* DAYS OPEN																
VERSION FIXED																
TARGET RELEASE																

Figura 1.6. Estadísticas en 20s Change Coordinator

Software para la Gestión de Cambios de la Empresa Élite

Proporciona un flujo de trabajo flexible, automatizando la aprobación de cambios y la documentación del proceso, el sistema provee acceso a toda la información relevante, actual e histórica, asociada con una Solicitud de Cambio.

Es ideal para organizaciones que buscan establecer o automatizar los procedimientos y procesos de Gestión de Cambios. Alivia el trabajo propio de algunos procesos que son muy intensos manualmente (documentos Word, hojas de cálculo Excel, etc.), reduce los defectos y las horas de trabajo, gestionando las Solicitudes de Cambio.

Controla todas las actividades de cambio, reduciendo los casos en que se dan pérdidas de calidad, normalmente asociados con cambios no coordinados.

Provee la capacidad de supervisar y aprobar las Solicitudes de Cambio desde cualquier ubicación, de calcular el riesgo asociado a una Solicitud de Cambio, brinda acceso a información clave del proceso acerca de cambios planeados e implementados y sus ramificaciones, permite personalizar los formularios de Solicitudes de Cambio para satisfacer los procesos y la terminología existentes de Gestión de Cambios. Se pueden documentar los planes paso a paso, para si se necesitara restaurar el estado original al fallar un cambio. Provee la capacidad de crear relaciones Padre/Hijo entre varias Solicitudes de Cambio.

Hansky Butterfly

Butterfly, ofrece una solución para gestionar los procesos de desarrollo de software. Centrándose en el ciclo de vida de desarrollo del proyecto, es capaz de seguir las aéreas asignadas y supervisar el progreso, gestionar los defectos y cualquier cambio que pueda ocurrir durante el desarrollo del software, así como mantener el control de la distribución de recursos y costos en el proyecto.

ManageWare CM-Synergy

Es un sistema de Gestión de Configuración usado por Nice, Interwise, Nokia, Philips y otras grandes compañías.

CM-Synergy provee una interfaz de usuario intuitiva y fácil de usar, brinda al desarrollador:

- Un área de trabajo gestionada, constantemente monitoreada por el servidor. Permite al usuario obtener su entorno de desarrollo desde cualquier máquina.
- Visibilidad sobre el área de trabajo de cualquier miembro del equipo, sobre los archivos dados de baja y las tareas, con el objetivo de incrementar la colaboración.
- Tareas dirigidas al desarrollo y la configuración, permiten al usuario añadir y eliminar cambios completos, revisar el contenido de una configuración basado en las tareas incluidas.
- Ver reportes de errores y solicitudes de cambio relacionadas.
- Combinar archivos, directorios, ramas y proyectos completos.
- Definir procesos de desarrollo como cooperativos (todos los miembros del equipo obtienen los cambios inmediatamente que son dados de alta), o aislados (los miembros del equipo obtienen los

cambios sólo después de ser aprobados) o compartidos (todos los miembros del equipo trabajan en la misma área del proyecto).

Herramienta de Gestión de Cambios MATCHPOINT, de Heinrich Informatik AG

Esta herramienta garantiza una segura y eficiente gestión de todos los cambios del software. Apoya completamente los procesos de desarrollo y despliegue, desde la creación de una Solicitud de Cambio hasta el despliegue automatizado en el entorno de producción.

MATCHPOINT permite conocer el estado y las actividades para cada Solicitud de Cambio. Las Solicitudes de Cambio registran datos como el estado, responsabilidad, fecha tope, y prioridad. Es también posible adjuntar documentos u otros objetos a una Solicitud de Cambio, lo que facilita la documentación. Las Solicitudes de Cambio pueden ser filtradas y organizadas, dando a cada miembro del proyecto la vista que necesita de los datos.

Para las Solicitudes de Cambio que provocan un cambio en la aplicación, los objetos modificados o nuevos pueden ser adjuntados a la Solicitud de Cambio. Los objetos adjuntos pueden ser desplegados automáticamente al entorno requerido. Ver [Figura 1.7](#).

Algunas de las funcionalidades que ofrece son: *notificación por correo electrónico* (los usuarios involucrados pueden ser notificados de nuevas Solicitudes de Cambio, o del cambio de estado de alguna de ellas); *dependencia entre Solicitudes de Cambio* (si una Solicitud de Cambio depende de otra, esta dependencia puede ser ingresada); *documentos adjuntos* (para facilitar la documentación, pueden ser adjuntados a la Solicitud de Cambio, conceptos, capturas de pantalla, etc.); *registro completo de todas las acciones* (cada cambio, manual o automático es registrado en un archivo, lo que permite tener trazas de todas las acciones durante el proceso de Gestión de Cambios).

Rational ClearQuest

Rational ClearQuest proporciona un seguimiento flexible de defectos y cambios. Es un producto de IBM, fue ganador del premio de la Visual Studio Magazine en el 2006, como la mejor herramienta de seguimiento de errores. Posee las siguientes funcionalidades:

- Seguimiento basado en actividad de cambios y defectos.
- Soporte robusto y flexible para flujos de trabajo, que incluye notificaciones por correo electrónico y opciones de envío.

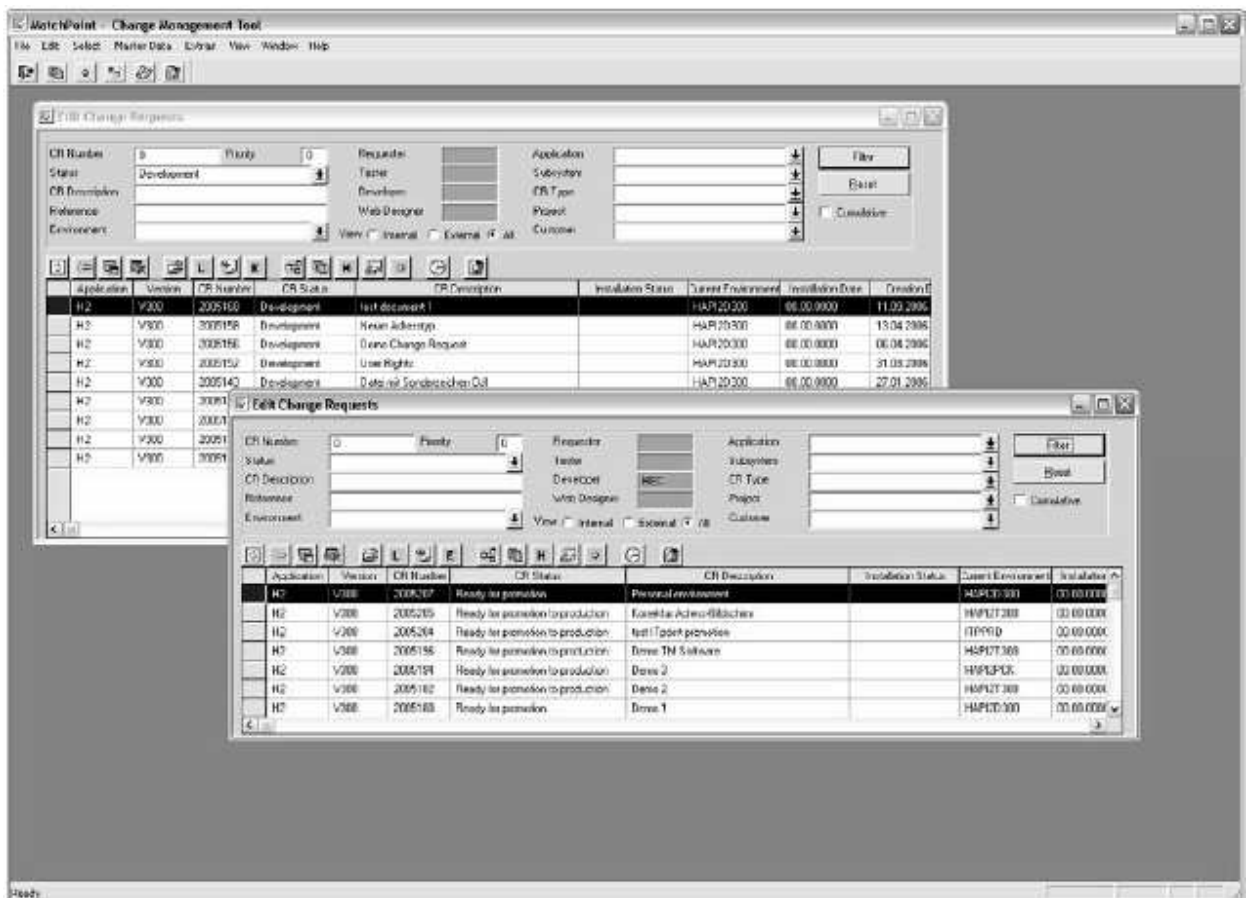


Figura 4.7. Solicitudes de Cambio en Matchpoint

- Fácil personalización.
- Soporte completo para consultas con generación de multitud de informes y gráficos.
- Interfaz web para acceder fácilmente desde cualquier navegador web estándar.
- Integración transparente con Rational ClearCase para conseguir una solución de Gestión de Configuración del Software completa.
- Integrado con los Entornos de Desarrollo Integrados (IDE por sus siglas en inglés) líderes en el sector, como WebSphere Studio, Eclipse y Microsoft .NET

Luego de hacer un análisis de algunas de las herramientas para la Gestión de Cambios, actualmente en el mercado, se pudo constatar que existen muchas posibilidades, todas de excelente calidad. No obstante también se pudo apreciar que la mayoría es software propietario, lo que presenta la gran desventaja de sus elevados precios, que no están al alcance de todos, y los problemas relacionados con las licencias.

1.8 Características de la producción en la Facultad 3

La Facultad 3 brinda servicios informáticos relacionados con el turismo y la gestión de documentos jurídicos, gestión de proyectos y la aplicación de técnicas de calidad. También trabaja en el área de la Inteligencia Artificial aplicada y el reconocimiento de patrones.

La producción se divide en tres Polos Productivos: Gestión de Recursos, Sistemas Legales y Gestión de Proyectos. Las plataformas más utilizadas son J2EE y .NET.

1.9 Conclusiones

La Gestión de Cambios contribuye en gran medida a evitar problemas en los procesos productivos.

La mayoría de las herramientas existentes para la Gestión de Cambios, son software propietario, lo que trae consigo una serie de limitaciones conocidas por todos.

Se analizaron las características de la producción en la Facultad 3, por tanto este capítulo constituye la base de búsqueda de nuevas alternativas para la creación de un proceso de Gestión de Cambios aplicable a dichos proyectos

Capítulo II. Propuesta para la Gestión de Cambios

2.1 Introducción

Aquí se proponen las soluciones a los objetivos planteados para este trabajo, se propondrá qué nivel de Gestión de Cambios adoptar en cada caso, por quiénes debe estar compuesto el Comité de Control de Cambios y qué debe ser discutido en sus reuniones. Se describe cuál va a ser el proceso a seguir para implementar la Gestión de Cambios, los aspectos que deberán contemplar los artefactos que se generan una vez que se da la necesidad de solicitar un cambio, y finalmente la herramienta que dará soporte al mismo. El modelo propuesto es descrito con más detalles en la Guía Práctica.

2.2 Características de los proyectos productivos e implicaciones para la Gestión de Cambios

Para la elaboración del modelo se identificaron características en los proyectos productivos de la facultad, como las que a continuación se enuncian, que de estar presentes inciden en una mayor formalidad en la aplicación del modelo.

Algunos de los proyectos se encuentran distribuidos geográficamente, por lo que es vital mantener la coordinación entre todos sus integrantes acerca de todos los cambios que ocurran. Existen proyectos con un gran número de miembros, lo que hace más difícil las tareas de gestión de no aplicarse rigurosamente. Como consecuencia la metodología utilizada en todos es RUP, la que exige que se mantengan numerosos artefactos y se garantice su trazabilidad entre todos los Flujos de Trabajo, además recomienda como una buena práctica que se gestionen los cambios.

También se puede señalar la existencia de proyectos con constantes cambios en los requisitos debido a negocios variables, ej.: el proyecto de Registro y Notarías tiene constantes cambios debido a la transformación de las leyes venezolanas.

2.3 Nivel de Gestión de Cambios a aplicar

El centro de atención se colocará sobre los cambios a nivel semi-formal y a nivel formal. A nivel informal no deberá ser necesario aplicar restricciones, pues los cambios contemplados aquí son los que se realizan sobre Elementos de Configuración de Software que no pertenecen a la línea base, los desarrolladores podrán realizar aquellos cambios que consideren pertinentes, siempre y cuando sean con

el objetivo de alcanzar la meta que le haya sido asignada, y no afecten objetos de configuración de la línea base.

Esto evita la excesiva burocracia que a veces se le atribuye a la Gestión de Cambios, dando a los desarrolladores una mayor libertad, con lo que se agiliza la realización de cambios. Siempre se debe tener en cuenta que sólo será aplicable a cambios que no afecten otros Elementos de Configuración que no estén bajo su responsabilidad.

Todo lo que se tratará en el resto del trabajo, estará referido a los cambios formales o a nivel del proyecto.

2.4 La Solicitud de Cambio, el Informe de Cambios y la Orden de Cambio de Ingeniería

La Solicitud de Cambio, el Informe de Cambios y la Orden de Cambio de Ingeniería son los artefactos que deberán ser manejados en el proceso de Gestión de Cambios.

El primer artefacto que es generado en el proceso de Gestión de Cambios, es la Solicitud de Cambio. Cualquier interesado en que se realice algún cambio deberá completar el formulario correspondiente, estas son utilizadas para documentar y seguir los defectos, las solicitudes de alguna mejora o cualquier otra solicitud de cambio sobre el producto. Permiten mantener registradas todas las decisiones tomadas y asegurar que los impactos de los cambios son entendidos a través del proyecto.

Siempre que vaya a ser realizada una Solicitud de Cambio, a nivel semi-formal o formal, no deben faltar los aspectos que se muestran tanto en el anexo como en la guía (ver [Anexo 1](#)).

Una vez que ha sido emitida una Solicitud de Cambio, y esta ha sido evaluada por el personal de Ingeniería de Software, se toma una decisión, que queda plasmada en el Informe de Cambio que va dirigido al Comité de Control de Cambios (ver [Anexo 2](#)).

El tercer y último artefacto que se propone sea utilizado es la Orden de Cambio de Ingeniería. Este es el artefacto que se emite cuando ha sido aprobada una Solicitud de Cambio para un Elemento de Configuración de Software de línea base, esta indica los requisitos para realizar el cambio, entre otros aspectos ([ver Anexo 3](#)).

2.5 Modelo propuesto

El modelo propuesto está compuesto principalmente por los siguientes aspectos:

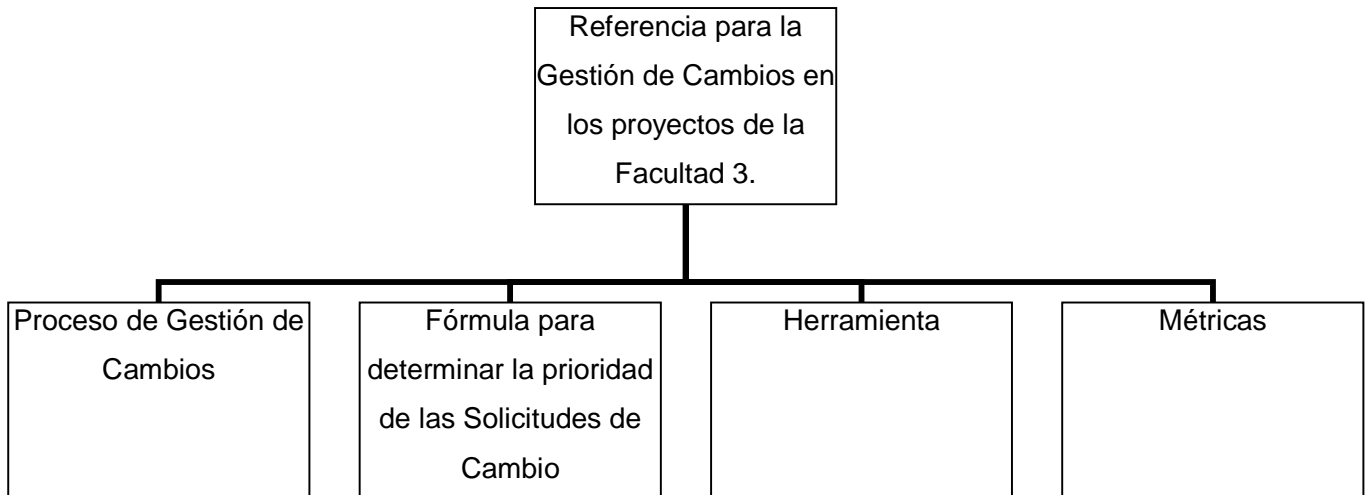


Figura 2.1. Partes integrantes del modelo propuesto.

2.6 Descripción del proceso a utilizar

Los procedimientos descritos a continuación sólo serán aplicables a para los niveles de Gestión de Cambios semi-formal y formal. Este proceso es nuestra propuesta de la secuencia de actividades a seguir, que realizarán los solicitantes de los cambios (clientes, desarrolladores o personal de calidad), el Comité de Control de Cambios, los encargados de implementarlos y el personal de pruebas. Para su elaboración se analizaron diferentes fuentes (2) (3) (8).

Teniendo un proceso de Gestión de Cambios estándar y bien documentado, se garantiza que los cambios se realizan de forma consistente. Permite que los interesados se mantengan informados del estado del producto, los cambios sobre el mismo, para poder evaluar el impacto de estos cambios sobre el costo y el cronograma.

Para formalizar el proceso propuesto se definieron: **Objetivo, Roles, Entradas, Definiciones, Actividades del procedimiento, Salidas y Gráficos.**

Nombre del proceso:

Proceso de Gestión de Cambios.

Objetivo:

Establecer una guía para la aplicación del Proceso de Gestión de Cambios en los proyectos productivos de la Facultad 3.

Establecimiento de Roles:

Cualquier interesado en hacer una Solicitud de Cambio (usuario, cliente o miembro del equipo de desarrollo), Gestor de Cambios, Probador, Desarrollador, Evaluador

Entradas:

Solicitud de Cambio.

Definiciones:

- Solicitante: Persona que emite una Solicitud de Cambio.
- Evaluador: Rol que se propone crear para que sea el encargado de hacer la revisión inicial de las Solicitudes de Cambio, es quien emite un Informe de Cambio en caso de considerarla válida. Se recomienda sea desempeñado por el Gestor de Cambios.

Actividades en el procedimiento:

Actividad	Descripción	Responsable
Enviar Solicitud de Cambio	Cualquier persona que guarde algún tipo de relación con el proyecto (cliente, usuario) puede enviar una Solicitud de Cambio. La Solicitud de Cambio es registrada en el Sistema de Seguimiento de Solicitudes de Cambio que se esté utilizando y es puesta en la cola de revisiones del CCC, el estado de la Solicitud de Cambio pasa a ser "Enviada".	Solicitante
Evaluar Solicitud de Cambio	La función de esta actividad es hacer una revisión inicial de las Solicitudes de Cambio "Enviadas", para descartar las no válidas.	Evaluador
Generar Informe de Cambio	De considerarse válida la Solicitud de Cambio se emite un Informe de Cambio.	Evaluador
Evaluar Informe de Cambios	Se evalúa el Informe de Cambios para decidir si se va a proceder con el cambio o no.	Comité de Control de Cambios

Generar Orden de Cambio de Ingeniería	Una vez que ha sido “Aceptada” una Solicitud de Cambio, el Jefe del Proyecto asignará entonces el trabajo al miembro del equipo apropiado, dependiendo del tipo de solicitud (solicitud de mejora, defecto, cambio en la documentación, defecto de prueba, etc.), y hacer cualquier actualización necesaria al cronograma del proyecto	Comité de Control de Cambios
Realizar los cambios	El miembro del equipo asignado realiza las actividades definidas dentro de la sección adecuada del proyecto (requisitos, análisis y diseño, implementación, etc.) para ejecutar los cambios solicitados. Estas actividades incluirán toda la revisión normal y actividades de prueba de unidad como están descritas en el proceso normal de desarrollo. La Solicitud de Cambio pasará a “Resuelta”.	Desarrollador del Equipo Asignado
Verificar cambios	Después de que los cambios han sido realizados por el miembro del equipo asignado, los cambios son ubicados en una cola de pruebas, para ser asignados a un probador y verificados.	Probador
Informar acerca de los resultados de las pruebas	Se le informa al Comité de Control de Cambios, si los cambios realizados pasaron exitosamente las pruebas realizadas	Probador
Informar al Solicitante si los resultados de los cambios	El Comité de Control de Cambios le informa a quien hizo la Solicitud de Cambio si se realizaron los mismos.	Comité de Control de Cambios

Tabla 2.1. Descripción de las actividades del proceso propuesto.

Salidas:

Solicitud de Cambio -aceptada, rechazada o aplazada- sobre un Elemento de Configuración de Software de la línea base, Orden de Cambio de Ingeniería, Elemento de Configuración de Software cambiado.

Gráfico del proceso:

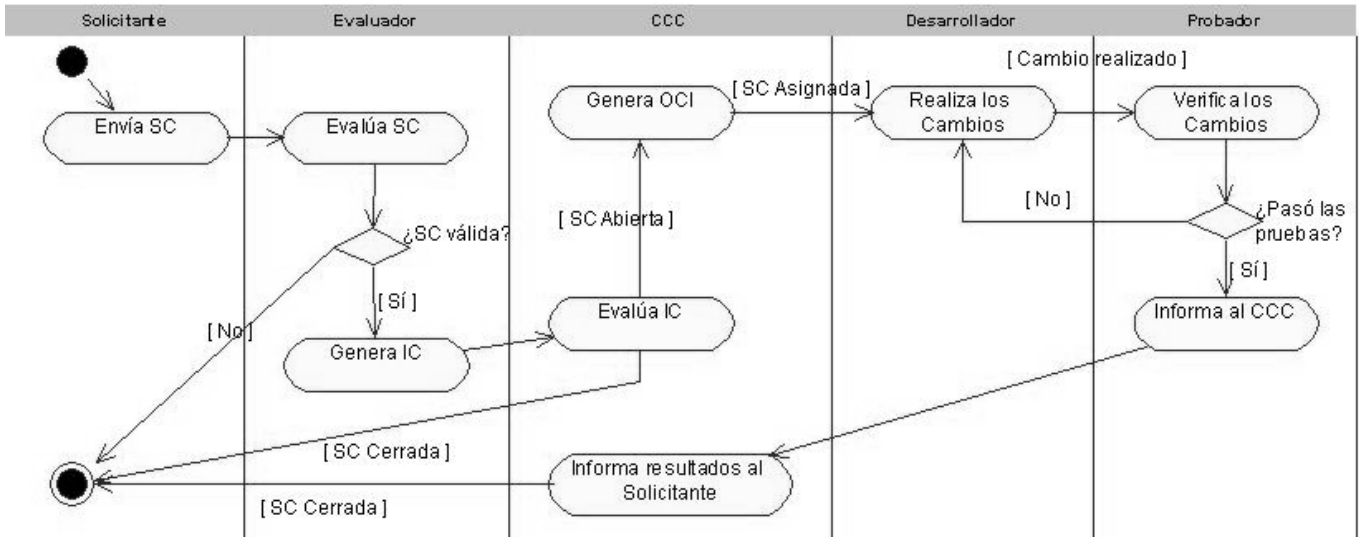


Figura 2.1. Gráfico del proceso de Gestión de Cambios.

Es aconsejable almacenar las Solicitudes de Cambio en una base de datos, de forma que puedan ser gestionadas más fácilmente (por ejemplo, ordenándolas por prioridad, etc.).

Las Solicitudes de Cambio podrán ser elevadas en cualquier momento del proyecto debido a que los defectos pueden aparecer en cualquier punto del ciclo de vida del desarrollo de software.

Cualquier persona que forme parte del personal del proyecto debe poder elevar una Solicitud de Cambio. Sin embargo, éstas necesitan ser revisadas y aprobadas de manera apropiada. La decisión final sobre cada Solicitud de Cambio debe ser hecha por el Comité de Control de Cambios.

2.7 Composición del Comité de Control de Cambios. Responsabilidades.

Como ya quedó definido en el Capítulo I, es el comité encargado de la parte fundamental del proceso de Gestión de Cambios, de tomar la decisión de proceder o no con el cambio. El Comité de Control de Cambios es una persona o grupo de personas en dependencia del tamaño del proyecto, con la responsabilidad de tomar las decisiones concernientes con las Solicitudes de Cambio, debe realizar el análisis, revisión y aceptación o rechazo de las mismas. Las Solicitudes de Cambio se acumulan y mantienen de modo que sean revisadas en la Reunión de Revisión de Solicitudes de Cambio, para analizarlas y ver si son aprobadas o rechazadas.

Se recomienda para los proyectos pequeños que esté constituido siempre por más de una persona, preferiblemente tres, para que las decisiones tomadas no dependan de la voluntad de uno solo, y en caso de existir diversidad de criterios, estos se logren unificar a través de la opinión de la mayoría. Estaría conformado siempre por el Líder del Proyecto, el Arquitecto de Software y el Gestor de Cambios, en caso de ser personas diferentes, de lo contrario, se completaría con algún otro miembro.

En los demás casos debe estar integrado por personas pertenecientes a todas las partes interesadas en el desarrollo del proyecto (clientes, usuarios y desarrolladores), se propone que esté compuesto por quienes se encuentren desempeñando los siguientes roles:

- Jefe de Proyecto que es la máxima autoridad en las decisiones tomadas en el proyecto
- Arquitecto de Software, el responsable de la arquitectura del software y pieza clave en todas las decisiones técnicas que se puedan tomar sobre el diseño y la implementación del proyecto.
- Gestor de Cambios que es quien define y supervisa el proceso de Gestión de Cambios.
- Analista de Pruebas, es el encargado de confirmar que ha sido realizado el cambio adecuadamente.
- Integrador de Sistemas, quien responde por la integración de cada construcción cuando sus partes han sido implementadas.
- Desarrollador, quien implementa los componentes.

Para los proyectos con miembros distantes geográficamente, se propone que en cada módulo exista un Comité de Control de Cambios, su constitución dependerá también de su tamaño, atendiendo a la cantidad de personas que posean.

2.8 Reunión para la revisión de Solicitudes de cambio

Es la reunión que debe efectuar el Comité Control de Cambios para revisar y evaluar las Solicitudes de Cambio emitidas. En la primera revisión se determina si la Solicitud de Cambio es válida o inválida. En caso de ser válida se determina si su resolución se debe llevar a cabo en la liberación actual, teniendo en cuenta la prioridad, gravedad, esfuerzo, recursos libres y riesgo de la Solicitud de Cambio. En esta asamblea pueden participar representantes del cliente, si el caso lo requiere. Esta reunión debe tener lugar cada vez que sea necesario evaluar algún Informe de Cambios, la frecuencia estará según las

necesidades y volumen de las Solicitudes de Cambio. El Comité de Control de Cambio debe asistir de forma íntegra a cada una de las reuniones, de ser necesario deben ser invitados otros miembros del equipo de trabajo o clientes implicados con la Solicitud de Cambio.

Se propone que el encargado de evaluar las Solicitudes de Cambio y emitir los Informes de Cambio asignen prioridades usando una escala de cinco valores, desde P1 hasta P5, atendiendo a la prioridad asignada inicialmente por quien solicita el cambio en la Solicitud de Cambio, y a los análisis hechos durante su evaluación, estas prioridades serán rectificadas posteriormente al aplicarse el método propuesto en el epígrafe 2.7.3.

2.8.1 Aspectos a considerar durante la Reunión de Revisión del CCC

El jefe de proyecto reúne la información, no debe solo pensar en la información necesaria para poder identificar las modificaciones, sino también en todo el entorno en el que opera el proyecto.

Deben cuestionarse los siguientes aspectos:

	Pregunta	Proceso relacionado
1	¿Que?	Alcance
2	¿Por qué?	Estratégico / Interdependencias
3	¿Cuánto?	Coste
4	¿Cuándo?	Tiempo
5	¿Quién?	Recursos / personal
6	¿Cómo?	Comunicaciones / Compras
7	¿Que pasa si?	Riesgos

Tabla 2.2. Procesos relacionados con el cambio o modificación

En la [Tabla 2.2](#) se aprecian cuáles son los procesos van asociados a un cambio, por lo que consideramos necesario que los siguientes aspectos sean tenidos en cuenta por los miembros del Comité de Control de Cambios, antes de tomar cualquier decisión sobre las Solicitudes de Cambio: ¿cuánto trabajo existente

tendrá que cambiar? ¿Cuánto trabajo nuevo se necesitará agregar? ¿Hay alguna alternativa? ¿Es fácil de hacer el cambio propuesto? ¿Qué posibles consecuencias puede traer hacer este cambio? ¿Qué impacto traería no implementar esta solicitud? ¿Hay involucrada pérdida de trabajo o de datos? ¿Es una solicitud de “mejora”? ¿Para cuándo se requiere el cambio? ¿Es factible? ¿Qué consecuencias traerá hacer el cambio? ¿Hay que hacer algún tipo de pruebas especiales para verificar que el cambio ha tenido éxito?

2.8.2 Técnicas para el desarrollo de la reunión

Las técnicas de trabajo en grupo permiten sacar un mejor provecho de las reuniones y aumentar el aporte que de los participantes. Algunas de sus ventajas son:

- Enseñan a pensar activamente y a escuchar de modo comprensivo.
- Vencen temores, inhibiciones y crean seguridad.
- Refuerzan el diálogo interpersonal y consiguen que los acuerdos grupales sean más ricos.
- Ponen en evidencia las actitudes y papeles representados efectivamente por cada participante.
- Capacitan para la cooperación, el intercambio, la autonomía y la creación.
- Ayudan al análisis y reflexión, tanto del contenido de la sesión como del proceso de la misma.

Se propone que sean utilizadas las siguientes técnicas:

Que se comience la reunión con la técnica llamada “Estudio de casos”, el Comité de Control de Cambios realiza el estudio exhaustivo de cada una de las situaciones concretas, de cada una de las Solicitudes de Cambio que se planificó discutir, con el fin de obtener conclusiones. Se persigue con esto los siguientes objetivos: provocar la toma de conciencia y la búsqueda de soluciones realistas y concretas, además de fomentar en el grupo la capacidad crítica de sus integrantes, consiguiendo su participación de forma respetuosa, exponiendo sus ideas y escuchando las de los demás. Esta técnica tiene como ventajas que estimula la creatividad y la participación activa de cada integrante, al tener que dar soluciones, refuerza el conocimiento y la experiencia, favorece la comprensión del problema y ayuda a extraer conclusiones realistas y aplicables, facilita el estudio de problemas que el grupo tienda fácilmente a abstraer y generalizar, sin planteárselos de forma concreta.

Luego, en caso de hacerse difícil ponerse de acuerdo ante la toma de alguna decisión se puede aplicar la “Tormenta de ideas”, los participantes en la reunión aportan, durante un tiempo previamente establecido, por ejemplo, diez minutos, el mayor número de ideas posibles sobre una Solicitud de Cambio a la que no se ha logrado llegar a un consenso sobre su aprobación. Interesa, en primer lugar, la cantidad de ideas;

conviene que las aportaciones sean breves, que nadie juzgue ninguna, que se elimine cualquier crítica o autocrítica y que no se produzcan discusiones ni explicaciones. Las ventajas de esta técnica son que se desarrolla en un ambiente totalmente informal que permite producir ideas y expresarlas con absoluta libertad y sin inhibiciones; favorece al máximo la creatividad de todos los miembros del grupo y facilita que afloren todas las alternativas posibles al problema; posteriormente se podrán estudiar con mayor reflexión y análisis.

2.8.3 Establecimiento de prioridades para las Solicitudes de Cambio.

Se propone que se aplique, de forma experimental, el siguiente método para tener una medida de cómo los cambios pueden afectar el desarrollo de software, y utilizarla para determinar de forma numérica las prioridades con que deben ser realizados. Este método constituye una adaptación del Método de Hanlon (16) y depejde de 4 componentes:

- Componente A = Magnitud del problema
- Componente B = Gravedad del problema
- Componente C = Eficacia estimada de la solución
- Componente D = Factibilidad de la solución, dada por la factibilidad económica, la disponibilidad de recursos y la conveniencia

Estos componentes se relacionan a través de una fórmula que proporciona una puntuación numérica, de manera que una mayor puntuación se traduce en una prioridad más alta para las Solicitudes de Cambio.

La máxima puntuación que se puede obtener es de 100 puntos. El componente A puede tener desde 0 hasta 10, el B desde 0 hasta 20, por lo que tiene un mayor peso, el C desde 0 hasta 10 y el D puede tomar como valor 0 ó 1. La fórmula es la siguiente:

$$[(A+B) C/3] \times D$$

Componente A: Magnitud del problema

Mientras mayor sea la cantidad de Elementos de Configuración de Software afectados por un problema o mejora, mayor será la prioridad que tendrá una Solicitud de Cambio. Este componente puede aportar hasta 10 puntos.

	0-10%	10-20%	20-30%	30-40%	40-50%	50-60%	60-70%	70-80%	80-90%	90-100%
Puntos	1	2	3	4	5	6	7	8	9	10

Tabla 2.3. Por ciento de ECS afectados.

Componente B: Gravedad

La gravedad de un problema o solicitud de mejora dependerá de en qué medida se encuentren afectados los factores de este componente, el cual tendrá una mayor influencia que el resto (puede aportar hasta 20 puntos) debido a que tiene en cuenta aspectos tan importantes para el proyecto como son la arquitectura del sistema y la pérdida económica que se prevé que haya de no hacerse el cambio solicitado.

	Cuánto afecta la Arquitectura	Pérdida económica de no hacerse el cambio
Puntos	0-10	0-10

Tabla 2.4. Factores del Componente B.

	0	1-4	5-8	9-10
Puntos	Ninguno	Poco	Considerable	Mucho

Tabla 2.5. Escala para la puntuación de los factores del Componente B.

Componente C: Eficacia de la solución

Este es el componente más subjetivo, pues debe dar una medida de la valoración que el Comité de Control de Cambios da acerca de la eficacia de la solución que se pretende dar. Su aporte puede ser hasta diez puntos.

Componente D: Factibilidad de la solución

Es el resultado del producto de los siguientes factores, que pueden tomar los valores de “0” ó “1” por lo que si alguno de ellos no toma valor de cero, anula completamente la puntuación de la Solicitud de Cambio en cuestión, o sea le asignaría prioridad cero, lo que significaría que no se procediera con la realización del cambio.

Factibilidad económica

Disponibilidad de recursos

Conveniencia

2.9 Herramienta propuesta para soportar el Proceso de Gestión de Cambios

Cuba está organizando la migración progresiva de las computadoras instaladas en los organismos de la Administración Central del Estado hacia el software libre, sobre la base del sistema operativo Linux, eliminando así la presencia casi exclusiva del Windows en las máquinas (9). Esto demuestra la política de nuestro país con respecto al software libre en general, el impulso que está dispuesto a brindarle por la gran cantidad de ventajas que ofrece.

Es por ello que se trató de encontrar una herramienta que fuera libre, pues las usadas tradicionalmente, entre las que se destacan el Rational ClearQuest, no permiten su uso a menos que los costosos precios de su licencia sean cubiertos, además de que los requerimientos de software y de hardware son bastante elevados.

La propuesta que se ofrece como herramienta a utilizar para que soporte la parte de la gestión de las Órdenes de Cambio de Ingeniería en el proceso de Gestión de Cambios en los Proyectos de la Facultad 3 es el Bugzilla 3.0.

Bugzilla es una herramienta basada en Web de seguimiento de errores (en inglés: bugs), originalmente desarrollada y usada por el proyecto Mozilla. Lanzado como software de código abierto por Netscape Communications en 1998, Bugzilla ha sido adoptado por una variedad de organizaciones con el fin de utilizarlo usarlo para seguimiento de defectos (errores) entre ellas existe una gran lista de más de 400 compañías, de gran prestigio, entre ellas se destacan Yahoo! Inc., AT&T, AMD. (10)

Su propósito original brinda la posibilidad de categorizar los defectos de software de acuerdo a su prioridad y severidad, así como asignarles versiones para su solución. También permite anexar comentarios, propuestas de solución, responsables a quienes asignar la solución y el tipo de resolución que tuvo el defecto, todo ello llevando un seguimiento de fechas en las cuales sucede cada evento y, si se configura adecuadamente, enviando mensajes de correo a los interesados en el defecto. Los errores pueden ser enviados por cualquiera, y pueden ser asignados a un desarrollador en particular. Cada error puede tener diferente prioridad y estar en diferentes estados, así como ir acompañado de notas del usuario o ejemplos de código que ayuden a corregir el error.

Bugzilla utiliza un servidor HTTP (como puede ser Apache) y una base de datos (normalmente MySQL) para llevar a cabo su trabajo, además de alguna versión de Perl 5. Puede ser utilizado tanto en Windows como en Linux, este último con la ventaja de que incluye el Perl dentro de su instalación y, tanto el Apache como el MySQL son fácilmente instalados desde un repositorio.

Un bug es un error, defecto, fallo en un programa de computadora que impide que se comporte de la forma prevista. La mayoría surge de errores cometidos por las personas en el código fuente o en el diseño del programa. Los informes que detallan los bugs en un programa son comúnmente conocidos como informes de bugs, informes de defectos, informes de fallos, informes de problemas, y hasta incluso, solicitudes de cambio.

La noción de "error" en Bugzilla es muy general, por ejemplo, Mozilla.org (17) lo utiliza también para registrar las solicitudes de nuevas funcionalidades, funciona como una base de datos con registros de bugs y solicitudes de mejoras para sus productos (Firefox, Thunderbird, SeaMonkey, Camino). Esta última idea es la que pretendemos utilizar.

Otro de los elementos que nos ayudan a sustentar la idea de que el Bugzilla puede ser utilizado con el propósito de gestionar los cambios en un proyecto de desarrollo de software, es que el Eclipse Process Framework (EPF) (18), un proyecto open source, de la Eclipse Foundation, utiliza el Bugzilla para registrar las Solicitudes de Cambio. Además posee varias funcionalidades muy útiles, el envío automático de correo electrónico y la generación de informes, en tablas o gráficos.

El próximo capítulo detalla la configuración que se hará para gestionar las Órdenes de Cambio de Ingeniería.

2.10 Conclusiones

Se ha desarrollado la propuesta del proceso a utilizar para gestionar los cambios en los proyectos productivos de la Facultad 3.

Se logró definir un Comité de Control de Cambios, sus integrantes y métodos a aplicar para mejores resultados, así como una fórmula que permite establecer las prioridades para las Solicitudes de Cambio.

Se propuso una herramienta para la Gestión de Cambios en la que se profundizará en el próximo capítulo.

Capítulo III. Herramienta propuesta y métricas del proceso

3.1 Introducción

Se describirán los principales cambios que será necesario hacer en la configuración del Bugzilla 3.0 con el objetivo de adaptarlo para gestionar las Órdenes de Cambio de Ingeniería, estos aspectos serán cubiertos con mayor detalle en la Guía Práctica. Se propone un conjunto de métricas para la evaluación de la aplicación del modelo propuesto. Finalmente se hace una validación del modelo propuesto con especialistas y se valoran los posibles beneficios que podría traer la aplicación del modelo.

3.2 Configuración de la herramienta propuesta

Para poder adaptar el Bugzilla para gestionar las Órdenes de Cambio de Ingeniería, es necesario seguir las indicaciones que en este epígrafe se detallan.

3.2.1 Instalación

El proceso de instalación del Bugzilla puede ser encontrado en el sitio oficial del proyecto (19), se requiere:

- Instalar Perl (5.8.1 o superior para Windows, 5.8.0 o superior para el resto)
- Instalar un servidor de base de datos (se recomienda MySQL 4.1.2 o superior)
- Instalar un servidor web (se recomienda Apache 2.x o superior)
- Instalar un Agente de Transporte de Correos.

Para más detalles sobre dónde obtener lo anterior, ver la Guía Práctica.

Es preciso instalar una serie de módulos Perl, los mismos también son especificados en (19), y configurar ciertos aspectos en el archivo *localconfig* del Bugzilla y el *httpd.conf* del Apache, del resto se encarga la ejecución del script *checksetup.pl*.

3.2.3 Configuración de parámetros

Luego de haber entrado por primera vez al Bugzilla se le solicita al administrador del mismo que se configuren los parámetros que definirán su comportamiento.

El primer parámetro que se solicita es la dirección de correo electrónico de la persona responsable de la instalación y mantenimiento del Bugzilla y a quién deberá notificarse cualquier problema que esté ocurriendo.

La seguridad del sistema puede configurarse para evitar que los usuarios sin tener una cuenta vean alguno de los datos del proyecto.

También brinda la opción de configurar el proceso de creación de nuevas cuentas, si será por parte de los propios usuarios o si el administrador será quien las cree.

Una de las principales funcionalidades que se debe ser aprovechada es el envío automático de correo electrónico, se debe establecer por medio de qué método se hará (Sendmail, SMTP, etc.). Por medio del correo electrónico serán avisados los desarrolladores cuando se la haya asignado la realización de un cambio, y el Gestor de Cambios cuando alguno de los desarrolladores haya realizado un cambio. Los usuarios del sistema podrán actualizar sus direcciones de correo electrónico en los casos que sea necesario.

Entre las ventajas que ofrece desde el punto de vista administrativo está la posibilidad de añadir y eliminar usuarios.

Puede establecerse seguridad a través de la creación de grupos, se recomienda crear un grupo por cada módulo encargado de un producto específico, así al asignarse un Orden de Cambio de Ingeniería, esta sólo podrá ser vista por los miembros del grupo al que corresponde el desarrollador asignado.

3.2.4 Entrada de datos del proyecto

Se verá cómo adicionar los productos del proyecto en cuestión, sobre los que pudieran realizarse cambios. Para esto se debe seguir el vínculo **Products**. Una vez aquí, se puede ver que existe un

producto de prueba a modo de ejemplo, este se puede eliminar. Para adicionar un nuevo producto se hace clic en el vínculo **Add a product**.

Edit product...	Description	Open For New Bugs	Votes Per User	Maximum Votes Per Bug	Votes To Confirm	Action
TestProduct	This is a test product. This ought to be blown away and replaced with real stuff in a finished installation of bugzilla.	Yes	0	10000	0	Delete

[Redisplay table with bug counts \(slower\)](#)

[Add a product.](#)

Figura 3.1. Insertando un nuevo producto.

Luego se deben llenar los campos que lo describen y que se observan en la siguiente pantalla:

The screenshot shows the 'Add a product' form in Bugzilla. It contains the following fields and options:

- Product:** A text input field.
- Description:** A large text area with scrollbars.
- Closed for bug entry:** A checkbox that is currently unchecked.
- Voting Settings (shaded grey area):**
 - A note: "The 'usevotes' parameter is currently 'off'. These voting settings will take effect when the parameter is next enabled."
 - Maximum votes per person:** Input field with value 0.
 - Maximum votes a person can put on a single bug:** Input field with value 10000.
 - Confirmation threshold:** Input field with value 0. Text below: "Enter the number of votes a bug in this product needs to automatically get out of the UNCONFIRMED state."
- Version:** Input field with value 'unspecified'.
- Add:** A button at the bottom left.

Figura 3.2. Campos requeridos para un producto.

La región sombreada en gris debe obviarse pues se refiere a la utilización de votos en los proyectos de software libre para los que Bugzilla es destinado originalmente.

Product: Nombre del producto

Description: Descripción del producto

Closed for bug entry: No debe seleccionarse esta opción, para poder entrar Órdenes de Cambio de Ingeniería para ese producto

Version: *La versión del producto*

Finalmente se hace clic en el botón **Add** lo que conduce a una pantalla donde se pide especificar al menos un componente para el producto seleccionado.

Se sigue el vínculo y se completan los campos solicitados:

Component: *Nombre del componente*

Description: *Descripción*

Default Assignee: *Dirección de correo electrónico de la persona a la que se asigna por defecto al ser creado el componente en el Bugzilla*

Se completa la adición haciendo clic en el botón **Add**.



The image shows a web form with three input fields. The first field is labeled 'Component:' and is a single-line text box. The second field is labeled 'Description:' and is a multi-line text area with a vertical scrollbar on the right side. The third field is labeled 'Default Assignee:' and is a single-line text box. Below these fields, there is a horizontal line and a small button labeled 'Add'.

Figura 3.3. Campos requeridos para un componente.

3.2.6 Usuarios

Existe una sección **Users**, donde el administrador podrá adicionar nuevos usuarios al sistema, o sea adicionar desarrolladores en cada uno de los grupos existentes, también es posible hacer búsquedas de usuarios. Para adicionar nuevos usuarios deberán especificarse los siguientes campos:

Login name: *una dirección de correo electrónico*

Real name: *el nombre de la persona*

Password: *la contraseña*

Bugmail Disabled: *de marcarse esta opción se deshabilitaría el envío de correos electrónicos para este usuario*

Disable text: *de poseer contenido, se deshabilitaría esta cuenta de usuario, el contenido debe ser la explicación del motivo*

3.2.7 Adición de campos personalizados

Gracias a esta funcionalidad es que es posible que se puedan gestionar las Órdenes de Cambio de Ingeniería a través del Bugzilla. La sección **Custom Fields**, es la que posibilitará la adición de los campos requeridos para adaptar la entrada de un “Bug” a una Orden de Cambio de Ingeniería. Los campos que se adicionarán que se corresponden con los aspectos que debe contemplar una Orden de Cambio de Ingeniería según la plantilla que se propone utilizar (ver [Anexo 2](#)):

Evaluador

Datos de contacto del Evaluador

Datos de contacto del Solicitante

Otros ECS afectados

Usuarios interesados

Trabajo requerido

Herramientas o recursos

Impacto en otros sistemas

Riesgos

Modificaciones en el Modelo del Análisis

Modificaciones en el Modelo del Diseño

Revisiones que se harán

Pruebas que se harán

En la página de creación de campos personalizados, se dan las indicaciones correspondientes, deberá comenzar su nombre con **cf_**, se deberá suministrar una descripción, que es el nombre con el que aparecerá en la página de creación de Órdenes de Cambio de Ingeniería, el tipo a escoger será **Free Text**, pues la otra opción es **Drop Down**, y se marcará la casilla de verificación **Can be set on bug creation**, para poder especificarlos en la creación de cada nueva Orden de Cambio de Ingeniería.

- Custom field names must begin with "cf_" to distinguish them from standard fields. If you omit "cf_" from the beginning of the name, it will be added for you.
- Descriptions are a very short string describing the field and will be used as the label for this field in the user interface.

Name: **Can be set on bug creation:**

Description: **Displayed in bugmail for new bugs:**

Type: **Is obsolete:**

Sortkey:

Figura 3.4. Creando un campo personalizado.

3.2.8 Creación de Órdenes de Cambio de Ingeniería

Para crear una nueva Orden de Cambio de Ingeniería debe usarse el vínculo **New**, originalmente utilizado en el Bugzilla para entrar un nuevo "Bug". Una vez hecho esto se muestran los productos de con los que está trabajando el proyecto em cuestión, se debe seleccionar el producto al que se le va a hacer el cambio, se completan los campos requeridos y finalmente **Commit**, tras lo cual es enviado un correo electrónico al desarrollador que se haya especificado como asignado para realizar el cambio.

Desde la perspectiva de los desarrolladores, al autenticarse en su sesión, deben visitar el vínculo **My Bugs**, para poder ver las Órdenes de Cambio de Ingeniería que le han sido asignadas, además de la correspondiente notificación por correo electrónico.

3.2.9 Informes

Finalmente se verá una de las más valiosas funcionalidades que brinda el Bugzilla 3.0, su capacidad de generar informes, que permite ver y dar seguimiento a la base de datos de Órdenes de Cambio de Ingeniería. Aquí hay dos opciones, los informes o los gráficos. A esta opción se puede acceder a través del vínculo **Reports**. Se tiene la posibilidad de escoger qué campos se pondrán en cada eje del gráfico (horizontal y vertical), y de seleccionar qué tipo de informe se desea: en forma de tabla (**Tabular Report**), o gráfico (**Graphical reports**); los informes gráficos pueden ser en forma de línea, de pastel o de barras. A través de estos informes podrán ser vistos los gráficos de algunas de las métricas propuestas en el siguiente epígrafe, específicamente las relacionadas con las Órdenes de Cambio de Ingeniería.

3.3 Métricas para el proceso

Asegura (2) que la medición es fundamental para cualquier disciplina de ingeniería, pues nos permite tener una visión más profunda, proporcionando un mecanismo para la evaluación objetiva. Se aplican con el objetivo de mejorar los resultados en una amplia gama de aspectos durante un proceso de desarrollo de software.

Las métricas son también utilizadas para señalar tareas con problemas de manera que se puedan desarrollar los remedios y mejorar el proceso del software. Para obtener los mejores resultados se recomienda aplicar un plan de medición sencillo pero consistente, que no evalúe, premie o castigue el rendimiento individual.

Los indicadores de proceso permiten a una organización de ingeniería del software tener una visión profunda de la eficacia de un proceso ya existente. También permiten que los gestores evalúen lo que funciona y lo que no. Las métricas del proceso se recopilan de todos los proyectos y durante un largo período de tiempo. Su intento es proporcionar indicadores que lleven a mejoras de los procesos de software a largo plazo.

Los principios que se pueden asociar con la formulación de las métricas técnicas son los siguientes:

- Los objetivos de la medición deberían establecerse antes de empezar la recogida de datos.
- Todas las técnicas sobre métricas deberían definirse sin ambigüedades.
- Las métricas deberían obtenerse basándose en una teoría válida para el dominio de aplicación.
- Hay que hacer las métricas a medida para acomodar mejor los productos y procesos específicos.

Atributos que deberían acompañar a las métricas:

- Simples y fáciles de calcular
- Empírica e intuitivamente persuasivas
- Consistentes y objetivas
- Consistentes en el empleo de unidades y tamaños
- Un eficaz mecanismo para la realimentación de calidad

Utilizando las plantillas para las métricas que sugiere (8), estas son las propuestas:

Nombre	SC x P.
Definición	Número de Solicitudes de Cambio por producto.
Metas	Conocer qué producto está recibiendo más Solicitudes de Cambio.

Tabla 3.1. Solicitudes de Cambio por producto.

Nombre	SC x M.
Definición	Número de Solicitudes de Cambio por módulo de un producto.
Metas	Conocer qué módulos en cada producto está recibiendo más Solicitudes de Cambio.

Tabla 3.2. Solicitudes de Cambio por módulo.

Nombre	SCA
Definición	Solicitudes de Cambio abiertas.
Metas	La cantidad de Solicitudes de Cambio que se encuentran abiertas en el proyecto, permite conocer el volumen de cambios que el proyecto deberá realizar.

Tabla 3.3. Estado de las Solicitudes de Cambio.

Nombre	TSCA
Definición	Tasa de Solicitudes de Cambio abiertas = Número de Solicitudes de Cambio abiertas / Número total de Solicitud de Cambio.
Metas	Permite conocer qué parte de las Solicitudes de Cambio que se están recibiendo, se les está dando solución.

Tabla 3.4. Tasa de Solicitudes de Cambio abiertas.

Nombre	OCI x D
Definición	Número de Órdenes de Cambio de Ingeniería por desarrollador
Metas	Permite saber el peso que está recibiendo cada desarrollador, en la implementación de los cambios.

Tabla 3.5. Órdenes de Cambio de Ingeniería por desarrollador.

Nombre	SCA x Solicitante
Definición	Número de Solicitudes de Cambio abiertas para un determinado solicitante, este es igual al número de Órdenes de Cambio de Ingeniería.
Metas	Permite determinar quién está detectando más errores o fallos, en caso de cambios solicitados por clientes, quién tiene mayor grado de inconformidad.

Tabla 3.6. Solicitudes de Cambio abiertas por solicitante.

La obtención de los resultados de las métricas propuestas se encuentra soportada por la herramienta propuesta, debido a su capacidad de generar informes.

3.4 Validación por especialistas

Como no fue posible la aplicación del modelo propuesto para realizar un análisis que permitiera comparar los resultados, fue necesario obtener información de los especialistas.

Para ello fueron seleccionados siete especialistas, número que coincide con valoraciones que indican que no deben ser menos de cinco especialistas ya que se pierde la idea de la valoración colectiva, y que la muestra no sea tan grande que aumente innecesariamente el gasto para la elaboración de los criterios.

La calidad de los especialistas influye decisivamente en la exactitud y fiabilidad de los resultados y en ello interviene la calificación técnica, los conocimientos específicos sobre el objeto a evaluar y la posibilidad de decisión entre otros. Existen varios procedimientos para la **selección de especialistas**, uno de ellos es el basado en la auto-evaluación de los mismos. Este método consiste en calcular el coeficiente de competencia **K**. Para calcularlo se tiene en cuenta la autovaloración del experto acerca de su competencia o poder de decisión en el tema, **Kc** (Coeficiente de conocimiento), y las fuentes que le permiten argumentar sus criterios **Ka** (Coeficiente de argumentación).

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

Donde:

Kc: escala de 0 a 10 multiplicado por 0.1, "0" indica no tener ningún conocimiento

Ka: Evaluación del experto en *alto, medio, bajo*.

Se definió que los especialistas debían conocer sobre los procesos de desarrollo de software, el control de cambios en un proyecto y sobre métricas. Los especialistas seleccionados son Ingenieros en Informática y líderes de proyectos en la Facultad 3.

Se realizó una primera encuesta para calcular el coeficiente de información de los especialistas y según los resultados seleccionar a los que se les realizaría la segunda encuesta para determinar la aplicabilidad y cumplimiento de los objetivos del modelo. Después de este análisis se decidió tener en cuenta el criterio de seis especialistas debido a que el valor del coeficiente de competencia de los mismos permitía la validación.

Luego de esto se les envió a los especialistas seleccionados una segunda encuesta acompañada de un resumen del modelo propuesto, esta encuesta está en el Anexo 4.

La siguiente tabla muestra los resultados obtenidos en la segunda encuesta:

Pregunta	Sí	No	No sé
¿Es necesario implantar procesos para garantizar niveles adecuados de Gestión de Cambios en la Facultad 3?	100%	0%	0%
¿Considera posible la implantación y ejecución de la referencia para la Gestión de Cambios en los proyectos productivos de la Facultad 3?	100%	0%	0%
¿Considera importante la implantación y ejecución de la referencia para la gestión de cambio en los proyectos productivos de la Facultad 3?	100%	0%	0%
¿Considera imprescindible para ejecutar los procesos de Gestión de Cambios la existencia de un Comité de Control de Cambios en los proyectos productivos de la Facultad 3?	83.3%	16.7%	0%

Tabla 3.7. Resultados del cuestionario de validación.

En la tabla se muestra una comparación de los aspectos que esta investigación pretende cambiar en los proyectos de desarrollo de software de la Facultad 3.

Aspecto	Antes	Después
Existencia de proceso de GC	Algunos proyectos	Todos los proyectos
Proceso claramente definido	No	Sí
Uniformidad del proceso	No existía	Proceso definido y repetible
Uso de herramientas	No	Bugzilla 3.0

Tabla 3.8 Posibles cambios de la propuesta en el proceso de Gestión de Cambios actual

Con la puesta en práctica del proceso y la herramienta propuesta en el Capítulo II, se podría garantizar la implementación de un proceso de Gestión de Cambios en los proyectos de la Facultad 3, que como consecuencia inmediata tendría una mejor organización y productividad, aspecto deseable en todo proyecto de desarrollo de software. Para aquellos proyectos donde se estaba haciendo Gestión de Cambios, pero que la persona encargada de ello carecía de experiencia (en la mayoría de los casos el líder), se brinda un proceso claramente definido, que posibilita su puesta en práctica de manera aceptable aún cuando no se tengan conocimientos previos en estas actividades. Se garantizaría que todos los proyectos gestionen los cambios de manera uniforme, con la misma herramienta; el uso de esta ayuda en gran medida a que el proceso de Gestión de Cambios sea una carga menos pesada para aquellos que lo consideran así, pues permite asignar el trabajo que se deriva de una Solicitud de Cambio aceptada, de manera automatizada, por medio de la funcionalidad que brinda el Bugzilla de enviar correo electrónico de manera automática, a los desarrolladores a quienes vaya dirigida un Orden de Cambio de Ingeniería.

3.5 Conclusiones

Es posible configurar el Bugzilla-3.0 para gestionar las Órdenes de Cambio de Ingeniería, obteniendo de él, ventajas como notificación automática por correo electrónico, generación de informes, con posibilidades de ver gráficamente los resultados del cálculo de algunas métricas.

Se propusieron un conjunto de métricas que permitirán evaluar el resultado de la aplicación del modelo propuesto.

Se obtuvo un modelo según las particularidades de los procesos productivos de la Facultad 3. Este modelo puede mejorar los procesos productivos de la Facultad 3. El modelo fue validado por especialistas.

Conclusiones

- El cumplimiento de las tareas de la investigación propuestas en el trabajo de diploma permitió la realización de un análisis del proceso de Gestión de Configuración así como la comprensión de la importancia que se le atribuye al proceso de Gestión de Cambios.
- Después de analizado el proceso de desarrollo de software en varios de los proyectos de la Facultad 3 se concluye la necesidad que tienen los mismos de un modelo que les facilite una aplicación adecuada para controlar los cambios que pueden ocurrir durante el ciclo de vida del software.
- Se desarrolló un modelo para la aplicación de la Gestión de Cambios en los proyectos de la Facultad 3 que incluye las actividades, métricas y técnicas a seguir durante el proceso. El sistema de métricas permitirá evaluar la aplicación del modelo y las técnicas influirán en las decisiones tomadas por Comité de Control de Cambios durante las reuniones de análisis de las Solicitudes de Cambio.
- Se propone una Guía Práctica que detalla la aplicación del modelo así como una herramienta de software libre que soporta parte del proceso de Gestión de Cambios.

Recomendaciones

- Que se impartan contenidos de Gestión de Cambios a los primeros años de la Facultad 3.
- Implementar módulos que permitan el uso del Bugzilla para la emisión de Solicitudes de Cambio prescindiendo de la utilización de votos que utiliza por defecto el Bugzilla.
- Que aplique el modelo propuesto en los procesos productivos de la Facultad 3.
- Definir e implementar un sistema de indicadores que permita calcular el impacto del cambio en un proyecto.

Referencias Bibliográficas

1. Bellagio, David E. y Milligan, Tom J. *Software Configuration Management Strategies and IBM® Rational® ClearCase® Second Edition A Practical Introduction*. s.l. : Addison Wesley Professional, May 23, 2005. 0-321-20019-5.
2. Pressman, Roger. *Ingeniería del Software. Un enfoque práctico*. España : Mc Graw Hill/ Interamericana de España S.A., 2002.
3. de Antonio, Angélica. *La Gestión de la Configuración del Software*. Chile : SPIN, 2001.
4. Babich, Wayne A. *Software configuration management: coordination for team productivity*. Boston : Addison-Wesley Longman Publishing, 1986.
5. Crawford, Neil. CM Crossroads LLC. [En línea] 2002 de Diciembre de 30. [Citado el: 20 de Marzo de 2007.] <http://www.cmcrossroads.com/cgi-bin/cmwiki/view/CM/WebHome>.
6. Samaras, Thomas T. y Czerwinski, Frank L. *Fundamentals of Configuration Management*. s.l. : John Wiley & Sons Inc, 1971.
7. Software Engineering Conference sponsored by the NATO Science Comitee. Garmisch, Germany : s.n., 1968.
8. Rational Software Corporation. *Rational Unified Process*. 2003. 2003.06.00.65.
9. Asklund, Ulf, Bendix, Lars y Ekman, Torbjörn. *Software Configuration Management Practices for eXtreme Programming Teams*. [Documento] Sweden : Department of Computer Science, Lund Institute of Technology.
10. IEEE. *IEEE Standard for Software Configuration Management Plans*. New York : Institute of Electrical and Electronics Engineers, 1998.

11. Rational Software Corporation. *Rational ClearCase Help*. 2003.
12. *Sitio Web del Software Engineering Institute*. [En línea] <http://www.sei.cmu.edu>.
13. Bersoff, Edward H., Siegel, Stanley y Henderson, Vilas. *Software Configuration Management: An Investment in Product Integrity*. s.l. : Prentice Hall Professional Technical Reference, 1980. 0138217696.
14. *Bug Tracking and Defect Tracking Resource*. [En línea] [Citado el: 12 de 5 de 2007.] <http://www.bug-tracking.info/software-change-management-tools.html>.
15. *BMC Software*. [En línea] [Citado el: 26 de Abril de 2007.] <http://www.bmc.com/>.
16. *Universidad de Illinois*. [En línea] [Citado el: 6 de Junio de 2007.] <http://www.uic.edu/sph/prepare/courses/ph440/mods/bpr.htm>.
17. *Sitio Web del proyecto Mozilla*. [En línea] [Citado el: 5 de Mayo de 2007.] <https://bugzilla.mozilla.org/>.
18. *Sitio Web del proyecto EPF*. [En línea] [Citado el: 22 de Mayo de 2007.] <http://www.eclipse.org/epf/>.
19. Bugzilla. [En línea] 7 de 5 de 2007. [Citado el: 7 de 5 de 2007.] <http://www.bugzilla.org/installation-list/>.
20. del Valle, Amaury E. Impulsará nuestro país sistema operativo Linux. *Juventud Rebelde*. 2005.
21. *ActiveState*. [En línea] [Citado el: 5 de Mayo de 2007.] <http://www.activestate.com/store/download.aspx?prdGUID=81fbce82-6bd5-49bc-a915-08d58c2648ca>.
22. *AppServ Open Project*. [En línea] [Citado el: 6 de Mayo de 2007.] <http://prdownloads.sourceforge.net/appserv/appserv-win32-2.4.8.exe?download>.
23. *Sitio Web de Bugzilla*. [En línea] Mayo de 2007. <http://www.bugzilla.org>.

Bibliografía

- **Pressman, Roger.** *Ingeniería del Software. Un enfoque práctico.* España : Mc Graw Hill/ Interamericana de España S.A., 2002.
- **de Antonio, Angélica.** *La Gestión de la Configuración del Software.*
- **Jacobson, Ivar; Booch, Grady, Rumbaugh, James.** *El Proceso Unificado de Desarrollo de Software.* Addison Wesley 2000.
- **Febles, Ailyn.** *MConfig.PM, Modelo de referencia para la Gestión de Configuración en la pequeña y mediana empresa de software.* Tesis doctoral inédita. CUJAE. La Habana, Cuba, 2004.
- **Franco Navarro, José Angel.** *Entorno Unificado para la Gestión de Configuración de Software.* Tesis de Maestría. CUJAE. Ciudad Habana, Cuba, 2006.
- **Delgado Martínez, Ramsés; González Soca, Emilio.** *ConfigCASE 3.0 HERRAMIENTA DE APOYO A LA GESTIÓN DE CONFIGURACIÓN. PROPUESTA ARQUITECTÓNICA.* Tesis de Diploma. CUJAE. Ciudad de la Habana, Cuba, 2006.

Glosario de términos

Artefacto: Una parte de la información que (1) es producida, modificada, o usada por un proceso, (2) define un área de responsabilidad, y (3) está sujeta al control de versión. Un artefacto puede ser un modelo, un elemento del modelo, o un documento. Un documento puede adjuntar otros documentos.

Bug: es el resultado de un fallo durante el proceso de creación de programas de ordenador o computadora (software). Dicho fallo puede presentarse en cualquiera de las etapas del ciclo de vida del software aunque los más evidentes se dan en la etapa de desarrollo y programación.

Dar de alta: Una vez acabado el cambio, la copia es almacenada de nuevo en el repositorio mediante una operación llamada dar de alta, que insertará la copia en el repositorio como una nueva versión sucesora de la que se extrajo de él.

Dar de baja: Todas las revisiones se encuentran almacenadas en un repositorio gestionado por la herramienta. Si alguien necesita trabajar en uno de los objetos que se encuentran en el repositorio, debe realizar dale de baja. Con esto está solicitando una copia de la última versión del objeto que se encuentre en el repositorio, y esta copia se almacenará en un directorio de trabajo local. A partir de este momento la copia queda fuera del alcance de la herramienta y puede ser modificada por el usuario.

Modelo: Cosa que ha de servir de objeto de imitación. Objeto, construcción u otra cosa con un diseño del que se reproduce más iguales. Esquema teórico de un sistema o de una realidad compleja que se elabora para facilitar su comprensión y estudio.

Liberación: Es una versión de parte de un software que ha sido publicado. Tiene como objetivo marcar con una etiqueta única una versión de un producto software como estable, listo para su entrega y uso.

Rol: Papel, cometido o función que tiene o desempeña que interpreta un actor.

RUP: (Rational Unified Process): Proceso Unificado de Desarrollo de Software de Rational.