Universidad de la Ciencias Informáticas

Facultad 15

Centro para la Informatización de la Gestión de Entidades

Trabajo para optar por el grado de Máster en Ciencias Técnicas

Título: Procedimiento para la gestión del mantenimiento de los sistemas de gestión en el CEIGE.

Autor: Ing. Raykenler Yzquierdo Herrera

Tutora: MSc. Yadenis Piñero Pérez

Ciudad de la Habana

Septiembre de 2010

DEDICATORIA

A mi familia, amigos y especialmente a las personas que no se rinden nunca.

AGRADECIMIENTOS

Agradezco a todos los que me impulsaron siempre. No se puede dejar de mencionar a mi tutora que tanto me apoyó.

También quisiera agradecerle a: Alfredo, Dinia, Adieren, Carlos, Temis, Leidy, Ponce, Lilian, Dailyn, Cesar, Joisel, Iliana y muchos otros que por no mencionarlos no los llevo menos presentes.

RESUMEN

Según recientes estudios para el 2010 cerca del 64% de los sistemas informáticos estarán en mantenimiento. El mantenimiento de software puede estar consumiendo el 65% de los costos del proyecto y hasta el 80% del tiempo del proyecto. Hay una serie de aspectos que hacen del mantenimiento una tarea sumamente complicada, se puede mencionar la existencia de código heredado, los problemas inherentes al mantenimiento y los efectos secundarios o laterales no previstos ni deseados. La Universidad de las Ciencias Informáticas no está exenta de estas problemáticas debido a que la cantidad de proyectos desarrollados ya es relativamente grande. El CEIGE es un centro perteneciente a la universidad que tiene la misión de desarrollar sistemas de gestión para las entidades. En este marco se impone la necesidad de hacer un estudio de las metodologías, estándares y técnicas existentes de manera tal que se pueda identificar cual puede ser la guía más adecuada para materializar un procedimiento que garantice la calidad de los procesos de mantenimiento para los productos resultantes. En el presente trabajo se desarrolla un procedimiento que está dirigido a solucionar los problemas asociados al mantenimiento de manera general y al mismo tiempo, cubrir las peculiaridades del proceso de mantenimiento de los sistemas de gestión desarrollados por el CEIGE. Se propone a lo largo de la investigación un grupo de tareas, roles y artefactos que conforman el proceso de mantenimiento de software. Finalmente se realiza una validación a partir de la aplicación del procedimiento propuesto.

Palabras claves: gestión, mantenimiento, software.

INDICE

INTRODUCCIÓN

Situación problemática

La creencia de un equipo de trabajo de que su tarea ha finalizado cuando instala y pone en funcionamiento el software en las instalaciones del cliente no puede ser más errónea. Aunque afirmaciones como estas han dejado de prevalecer debido a la experiencia que adquieren los equipos que desarrollan actualmente software con determinada estabilidad. Después de implantado un sistema se hace necesario supervisar su correcto funcionamiento durante un tiempo dado que el sistema puede estar sometido a circunstancias no previstas. Así mismo, es posible que el software necesite ser modificado, ya sea consecuencia de la detección de errores o bien ante nuevas exigencias y/o necesidades del usuario del sistema. A esta fase se le conoce como fase de Mantenimiento de Software (MS).

Según el IEEE, el Mantenimiento del Software es la modificación de un producto software después de su entrega al cliente o usuario para corregir defectos, para mejorar el rendimiento u otras propiedades deseables, o para adaptarlo a un cambio de entorno. Sin embargo, y aunque no se aprecia a lo largo del estándar IEEE 1219, el proceso de Mantenimiento del Software comienza con las primeras fases del ciclo de vida, puesto que el coste de Mantenimiento va a estar tremendamente influido por las decisiones que se tomen en cada una de estas fases (Párraga, 1999).

Antes del desarrollo, la visibilidad de los problemas del sistema para los usuarios y el grupo de pruebas podría ser relativamente baja, ya que la reparación de algunos problemas puede ser manejada de una manera informal por el grupo de desarrollo. Sin embargo, una vez que el sistema ha sido liberado, es necesario un proceso de seguimiento más formal (Párraga, 1999).

Una simple definición de mantenimiento es: el proceso de registro y seguimiento de problemas, así como la petición y gestión de respuestas. Un problema no necesariamente implica un defecto del software. Los problemas aparecen simplemente porque el comportamiento esperado del sistema no coincide con el comportamiento actual. Esto puede ser el resultado de un defecto del software, pero puede ser fácilmente el resultado de un desconocimiento de las capacidades del sistema, o de la utilización incorrecta del flujo de trabajo, o de una documentación pobre, o de cambios en las directrices de la empresa, o de otras tantas razones. La gestión de las respuestas a los problemas del sistema cubre un número grande de posibles actividades. Algunas de las posibles respuestas a los problemas del sistema podrían incluir: liberación de nuevas versiones, sesiones de

planificación, identificación de la formación necesaria, actualización de la documentación, etc. (Párraga, 1999) (Dhillon, 2006).

Desde el punto de vista del grupo de desarrollo, el aspecto más significativo de un problema será saber si este está originado por un defecto del software. La calidad de la información comunicada desde el usuario al grupo de desarrollo es crucial para poder determinarlo y para su rápida resolución. (Párraga, 1999)

La problemática del mantenimiento se resume en realizar el mantenimiento del software de forma tan rigurosa que la calidad no se deteriore como resultado de este proceso.

Las principales dificultades en el MS son:

- La existencia de los llamados “legacy code” (código heredado).
- Problemas inherentes al mantenimiento del software.
- Efectos secundarios o laterales no previstos ni deseados.

Código heredado.

Con el paso de los años se ha ido produciendo un volumen muy grande de software. En la actualidad, la mayor parte de este software está formado por código antiguo "heredado"; es decir, código de aplicaciones desarrolladas hace algún tiempo, con técnicas y herramientas en desuso y probablemente por personas que ya no pertenecen al colectivo responsable en este momento del mantenimiento del software concreto. En muchas ocasiones, la situación se complica porque el código heredado fue objeto de múltiples actividades de mantenimiento. La opción de desechar este software y reescribirlo para adaptarlo a las nuevas necesidades tecnológicas o a los cambios en la especificación es muchas veces inadecuada por la gran carga financiera que supuso el desarrollo del software original y la necesidad económica de su amortización (Feathers, 2002).

Los problemas específicos del mantenimiento de código heredado han sido caracterizados en las llamadas Leyes del Mantenimiento del Software, propuestas por Lehman (Ruiz, y otros, 2005):

Continuidad del Cambio: Un programa utilizado en un entorno del mundo real está destinado a cambiar, ya que, en caso contrario, será utilizado cada vez menos en dicho entorno.

Las razones que conducen a esta afirmación son varias:

- A los usuarios se les ocurren nuevas funcionalidades cuando comienzan a utilizar el software.
- Nuevas características en el hardware pueden permitir mejoras en el software.
- Se encuentran defectos en el software que deben ser corregidos.
- El software debe instalarse en otro sistema operativo o máquina.
- El software necesita ser más eficiente.

Incremento de la Complejidad: A la par que los cambios transforman los programas, su estructura se hará progresivamente más compleja salvo que se haga un esfuerzo activo para evitar este fenómeno.

Evolución del Programa: La evolución de un programa es un proceso autorregulado. Las medidas de determinadas propiedades (tamaño, tiempo entre versiones y número de errores) revelan estadísticamente determinadas tendencias e invariantes.

Conservación de la Estabilidad Organizacional: A lo largo del tiempo de vida de un programa, la carga que supone el desarrollo de dicho programa es aproximadamente constante e independiente de los recursos dedicados.

Conservación de la Familiaridad: Durante todo el tiempo de vida de un producto software, el incremento en el número de cambios incluidos con cada versión es aproximadamente constante.

Problemas inherentes al mantenimiento del software.

Además de las dificultades de mantenimiento que señalan las leyes anteriores, existen otros problemas que complican el mantenimiento y que son debidos a la propia naturaleza del proceso:

- De carácter técnico:
    - Ausencia metodológica: A menudo, el mantenimiento es realizado de una manera *ad hoc* en un estilo libre establecido por el propio programador.
    - Tendencia a la desestructuración: Cambio tras cambio, los programas tienden a ser menos estructurados. Esto se manifiesta en una documentación desfasada, código que no cumple los estándares, incremento en el tiempo que los programadores necesitan para entender y comprender los programas o en el incremento en los efectos secundarios producidos por los cambios.

- Disminución de la comprensibilidad: Es muy habitual que los sistemas que están siendo sometidos a mantenimiento sean cada vez más difíciles de cambiar. Esto se debe al hecho de que los cambios en un programa por actividades de mantenimiento dificultan la posterior comprensión de la funcionalidad del programa.
- Poca participación de los usuarios: La falta de una metodología adecuada suele conducir a que los usuarios participen poco durante el desarrollo del sistema software. Esto tiene como consecuencia que, cuando el producto se entrega a los usuarios, no satisface sus necesidades y se tienen que producir esfuerzos de mantenimiento mayores en el futuro.

- De gestión: Muchos programadores consideran el trabajo de mantenimiento como una actividad inferior -menos creativa- que les distrae del trabajo -mucho más interesante- del desarrollo de software. Esta visión puede verse reforzada por las condiciones laborales y salariales, y crea una baja moral entre las personas dedicadas al mantenimiento.

Como resultado de lo anterior, cuando se hace necesario realizar mantenimiento, en vez de emplear una estrategia sistemática, las correcciones tienden a ser realizadas con precipitación, sin pensarse de forma suficiente, no documentadas adecuadamente y pobremente integradas con el código existente. No es extraño, pues, que el propio mantenimiento conduzca a la introducción de nuevos errores e ineficiencias que conducen a nuevos esfuerzos de mantenimiento con posterioridad.

Todos estos problemas se pueden atribuir parcialmente al gran número de programas existentes que han sido desarrollados sin utilizar la ingeniería del software (Francisco Ruiz, 2001).

Efectos secundarios.

La posibilidad de error al cambiar un procedimiento lógico tan complejo como el que constituyen la mayor parte de los programas actuales es muy grande. Por esta razón, una de las principales dificultades del MS es el riesgo del llamado efecto *bola de nieve*, de manera que los cambios producidos por una petición de mantenimiento introducen efectos secundarios que implicarán nuevas peticiones de mantenimiento en el futuro. Estos efectos secundarios suponen nuevos defectos que aparecen como consecuencia de las modificaciones realizadas. Según las consecuencias que se derivan, los efectos secundarios del MS son de tres clases: efectos sobre el código, efectos sobre los datos y efectos sobre la documentación (Francisco Ruiz, 2001).

Para tener una idea más clara de cómo estos problemas asociados al mantenimiento de software pueden impactar en la industria de software es necesario que se repasen algunas cifras. Según Capers Jones durante el año 1990 se encontraban en desarrollo aproximadamente 3 millones de sistemas mientras que estaban en mantenimiento 4 millones de sistemas, durante el 2000 se desarrollaban 4 millones y se le daba mantenimiento a 6 millones y se estima que durante el año 2010 se estén desarrollando 5 millones de sistemas y se le darán mantenimiento a 9 millones, lo que representa el 64% de los sistemas existentes o en desarrollo (Sneed, 2008) (Jones, 2008) (Jones, 2009).

Si se hace un análisis a partir de los aspectos que se han expuesto se evidencia que el mantenimiento de software requiere de un gran esfuerzo por parte del proveedor del sistema, por tanto los costos asociados a esta etapa son altos y generalmente mayores que los incurridos durante el desarrollo. También se puede señalar que el tiempo necesario para corregir, adaptar, perfeccionar o prevenir problemas en un sistema puede ser lo suficientemente grande. Especialistas como Bennett y Rajlik plantean que la etapa de mantenimiento puede estar consumiendo el 65 % de los costos del proyecto y el 80 % del tiempo destinado al mismo (Sneed, 2008).

En nuestro país según resientes investigaciones realizadas en 12 empresas desarrolladoras de software se evidencia la desestructuración del proceso de MS. Se evidencian problemas como la falta de una adecuada planificación del trabajo. Además no se hace un registro de la información de los cambios realizados en el sistema o cuando se hace no es suficiente para poder llevar una trazabilidad de la evolución del mismo. La planificación y ejecución del MS se basan en la experiencia y el buen juicio de una persona que esté a cargo de ello, en su generalidad, se adolece del conocimiento, de procedimientos o modelos que indiquen o guíen al mantenedor en cómo realizar el mantenimiento a estos sistemas desarrollados a la medida para un proceso particular vigente en la organización (Fábrega, 2010).

La Universidad de las Ciencias Informáticas (UCI) no está ajena a los problemas que acarrea el mantenimiento de software aunque la mayoría de los proyectos desarrollados o en desarrollo se apoyaron o apoyan en alguna metodología de desarrollo de software que permite garantizar algunos de los aspectos relacionados con el MS. Hay que señalar que la cantidad de proyectos que han sido desplegados hasta el momento es relativamente poca por lo cual las experiencias en etapas de mantenimiento son escasas.

La universidad desarrolla un grupo importante de sistemas de gestión, los cuales están dirigidos a facilitar la gestión empresarial. Entre las características de estos sistemas está que poseen una estructura piramidal (Wikipedia, 2008):

1. La parte inferior de la pirámide está comprendida por la información relacionada con el procesamiento de las transacciones.
2. El siguiente nivel comprende los recursos de información para apoyar las operaciones diarias de control.
3. El tercer nivel agrupa los recursos del sistema de información para ayudar a la planificación táctica y la toma de decisiones relacionadas con el control administrativo.
4. El nivel más alto comprende los recursos de información necesarios para apoyar la planificación estratégica y la definición de políticas de los niveles más altos de la administración.

Los programas que desarrolla el Centro para la Informatización de la Gestión de Entidades (CEIGE) son considerados como sistemas de gestión, un ejemplo claro lo constituye el sistema Cedrux, que está siendo implantado como parte de la etapa de Piloto en 6 entidades cubanas. Desplegar y darle mantenimiento a una solución mientras que se continúan desarrollando otras versiones de la misma puede resultar sumamente complicado tratándose especialmente de los sistemas de gestión desarrollados en el centro antes mencionado. Algunos de los argumentos que respaldan la anterior afirmación se muestran a continuación para poder dimensionar cómo se puede ver afectada la facilidad de mantenimiento, el tiempo destinado al mantenimiento y la satisfacción del cliente:

- Uso de un modelo de desarrollo con un alto nivel de formación. Los recursos humanos que se emplean deben dividirse en función de darle mantenimiento a la solución implantada y desarrollar la nueva versión, mientras que los estudiantes (mayoría en los proyectos de la universidad) reciben la formación correspondiente al año que cursan. Los profesionales también tienen responsabilidad con la formación. Esto evidencia la complejidad de la gestión de los procesos en los que participan estudiantes y profesores.
- Código heredado. Cedrux lleva aproximadamente dos años de desarrollo y el propio modelo de formación y producción de la universidad influye en que las personas permanezcan en su mayoría solo durante un período de tiempo en un proyecto, lo que deviene en una considerable porción de código heredado.

- La solución es sumamente compleja. En la primera versión de Cedrux se implementaron alrededor de 2000 funcionalidades y entre los módulos desarrollados existe un grado de acoplamiento medio. La posibilidad de incluir efectos secundarios producto del mantenimiento es grande.
- Repercusión sobre la próxima versión. Hay que considerar que las próximas versiones de Cedrux estarán soportadas por la línea base de la solución actualmente en despliegue, cualquier modificación en esta última solución pudiese implicar efectos sobre las próximas versiones del producto.
- Otros recursos utilizados. No tener claridad de los procesos de mantenimiento y/o implementarlos incorrectamente puede significar uso de recursos no planificados como puede ser pagos salariales, uso de computadoras y uso excesivo de la transportación asignada para la etapa de soporte de la solución.

Planteamiento del problema de investigación

La no existencia de un procedimiento que gestione adecuadamente los procesos del mantenimiento de software en los sistemas de gestión del CEIGE está provocando que se afecte la facilidad de mantenimiento, el tiempo destinado al mantenimiento y la satisfacción del cliente.

Objeto de estudio

La gestión del mantenimiento de software y modelos de desarrollo de software.

Campo de acción

Gestión del mantenimiento para software de gestión.

Objetivo de la investigación

Desarrollar una propuesta de procedimiento para gestionar los procesos del mantenimiento de los sistemas de gestión en el CEIGE.

Objetivos específicos

- Realizar estudio del estado del arte de los principales enfoques acerca de la gestión de los procesos del mantenimiento de software y un análisis valorativo de las ventajas y desventajas de cada enfoque.
- Definición de una propuesta de procedimiento para gestionar los procesos del mantenimiento en los sistemas de gestión en el CEIGE.

- Evaluar la propuesta de procedimiento en una muestra seleccionada de proyectos que den mantenimiento a los sistemas de gestión en el CEIGE, respecto a la facilidad de mantenimiento, cumplimiento de los compromisos de tiempo destinado al mantenimiento y en la satisfacción de los requerimientos del cliente.

Hipótesis

Si se desarrolla e implanta una propuesta de procedimiento para gestionar los procesos del mantenimiento para los sistemas de gestión del CEIGE, debe mejorar el proceso de mantenimiento de los sistemas de gestión en cuanto a la facilidad de mantenimiento, al cumplimiento de los compromisos de tiempo y en la satisfacción de los requerimientos del cliente.

Aporte teórico y práctico

El procedimiento para la gestión del mantenimiento de los sistemas de gestión que se propone como solución a esta problemática tiene un alto valor práctico que se manifiesta en su aplicación:

- En los sistemas de gestión en etapa de mantenimiento en el CEIGE para mejorar significativamente su eficacia.
- Es una base importante para la organización de los servicios asociados al soporte de software como modelo de negocio para la UCI.
- Como base teórica para el desarrollo de herramientas y sistemas de control que permitan gestionar adecuadamente los procesos cubiertos y apoyar así también el proceso de toma de decisiones durante la organización del desarrollo y mantenimiento de software.

Se pretende proveer a los líderes de los proyectos de mantenimiento de sistemas de gestión en el CEIGE y a otros especialistas de una herramienta informativa sobre el procedimiento que contribuirá a su mejor desempeño.

Novedad científica

La novedad científica del presente trabajo se resume en el desarrollo de un nuevo procedimiento para la gestión del mantenimiento de los sistemas de gestión del CEIGE.

Estructura del trabajo

El presente trabajo está conformado por tres capítulos.

Capítulo 1: Se hace un análisis del estado del arte y se discuten las ideas acerca de los modelos de gestión de los procesos de mantenimiento de software. Además, se hace una evaluación crítica de las ventajas y desventajas de los diferentes enfoques.

Capítulo 2: Se presenta el procedimiento, describiéndose las diferentes fases que se proponen, las actividades, artefactos e implicados en el proceso de mantenimiento.

Capítulo 3: Se realiza un análisis de los resultados obtenidos a partir de la aplicación del procedimiento en un entorno de mantenimiento, describiéndose los instrumentos de medición utilizados y la forma de aplicación del procedimiento.

CAPÍTULO 1. FUNDAMENTO TEÓRICO

Definiciones generales sobre mantenimiento

La definición posiblemente más utilizada de la Ingeniería del Software es la siguiente (IEEE, 1990): “La aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, la operación y el MS; esto es, la aplicación de la ingeniería al software.”

En la propia definición aparece el mantenimiento como una de las actividades de la Ingeniería del Software. Ahora bien, ¿en qué se diferencia el mantenimiento de otras actividades de la Ingeniería del Software? – concretamente, ¿en qué se diferencia de lo que se suele denominar “desarrollo”?. Para clarificar esta delimitación, hay que buscar un criterio que separe unas actividades de las otras.

Como primera aproximación, se puede decir que las actividades de MS son actividades de Ingeniería del Software orientadas a la modificación o cambio del mismo (por diferentes motivos, que se verán más adelante). El cambio tiene como característica fundamental el hecho de que primero se necesita una comprensión del objeto que se ha de cambiar, para poder hacer efectiva la modificación. Esto hace que la comprensión del software como actividad humana, sea un elemento esencial en la Ingeniería del Software. Es decir, es conveniente que el software tenga una estructura y una documentación asociada que facilite su comprensión.

En un sistema que es fácilmente mantenible, se puede implementar un cambio con un menor esfuerzo que en un sistema que es menos mantenible.

El mantenimiento del software como un caso especial de mantenimiento

El software no se deteriora con el uso ni con el paso del tiempo, a diferencia de los materiales mecánicos que son producto de otras actividades de ingeniería. Mejor dicho, el software no sufre de un deterioro físico. No obstante, se suele considerar que los sistemas informáticos sufren un deterioro en su estructura, cuando a lo largo del tiempo se van incluyendo más y más cambios que hacen que su estructura interna sea cada vez más difícil de entender. En ocasiones se ha denominado a este fenómeno como “erosión del diseño”.

Esta idea del deterioro de la estructura da lugar a la idea relacionada a la mantenibilidad. La mantenibilidad es una propiedad del diseño del software relativa a su facilidad de

mantenimiento. Esto lleva a que en ocasiones se introduzcan cambios en el software sólo para hacerlo más mantenible, lo cual rara vez se encuentra en otras ramas de la ingeniería.

El mantenimiento como actividad post-entrega

Cualquier esfuerzo de Ingeniería del Software – si termina con éxito – acaba por producir un determinado producto software, orientado a satisfacer ciertos requisitos previamente establecidos. El mantenimiento en este contexto se entiende de manera general como las actividades de cambio de ese producto. Ahora bien, el cambio se puede entender de diferentes maneras. Comenzando por lo básico, la definición de "Mantenimiento del Software" del estándar IEEE 1219 es: "El mantenimiento del software es la modificación de un producto software después de la entrega para corregir fallos, para mejorar el rendimiento u otros atributos, o para adaptar el producto a un entorno modificado".

Esta definición implica que las actividades de mantenimiento de un producto comienzan en el tiempo sólo después de que el producto se ha entregado, es decir, después de que el producto está en operación. No obstante, en ocasiones se considera que algunas actividades de mantenimiento pueden comenzar antes de la entrega del producto.

Algunas de estas actividades están dirigidas a la planificación del mantenimiento, otras simplemente están orientadas a facilitar el mantenimiento, como puede ser la revisión de la documentación. No obstante, estas pueden considerarse actividades de preparación para el mantenimiento, más que de mantenimiento en sí.

Pressman señaló: "la fase de mantenimiento se centra en el cambio que va a asociado a la corrección de errores, a las adaptaciones requeridas a medida que evoluciona el entorno del software, y a cambios debidos a las mejoras producidas por los requisitos cambiantes del cliente". (Pressman, 2002)

Una visión más amplia del mantenimiento del software

Por ejemplo, varios autores resaltan que hay una necesidad de comenzar a considerar el mantenimiento desde el mismo momento en que comienza el desarrollo:

El MS es la totalidad de las actividades necesarias para proporcionar soporte económico (cost-effective) al sistema software. Estas actividades se desarrollan tanto antes como después de la entrega. Las actividades previas a la entrega incluyen la planificación de las operaciones

posteriores a la entrega, planificación del soporte y determinación de la logística. Las actividades posteriores a la entrega incluyen la modificación del software, la formación de usuarios, y la operación de un help desk.

La guía SWEBOK considera que el mantenimiento ocurre durante todo el ciclo de vida (Abran, y otros, 2001).

Es importante resaltar que el concepto de MS difiere de la concepción de mantenimiento en otras disciplinas de ingeniería. Esto es debido a que el software no se "deteriora" con el uso. En la ingeniería mecánica, el mantenimiento consiste en las acciones de reparación necesarias para que la máquina o sistema mecánico siga funcionando. En la Ingeniería del Software, el mantenimiento tiene un significado más amplio, cubriendo su adaptación a necesidades.

La evolución del software

El término "evolución" del software se utiliza desde los sesenta para denominar la dinámica de crecimiento del software. Una definición atribuida a Lehman y Ramil dice que la evolución del software es: "todas las actividades de programación que se orientan a generar una nueva versión de un software a partir de una versión anterior operativa. Ned Chapin (1999) lo definió como "la aplicación de las actividades y procesos de MS que generan una nueva versión operativa de un software con una funcionalidad de usuario o propiedades cambiadas a partir de una versión anterior, junto con los procesos y actividades de garantía de calidad y con la gestión de esos procesos". De estas definiciones se desprende que la evolución cubre el ajuste a funcionalidades adicionales.

La guía SWEBOK considera que la causa del mantenimiento está tanto en la necesidad de "cambios" como de "evolución" en el software.

Leyes de la Evolución del Software

Lehman (1974) formuló las primeras "leyes de la evolución del software" por primera vez a partir de un estudio del proceso de programación en IBM (Lehman, y otros, 2003).

A continuación se resume la formulación de las leyes del mantenimiento. Todas ellas hacen referencias a programas "de tipo E", es decir, a aquellos programas destinados a solucionar un determinado problema del mundo real.

Ley I. Cambio Continuo: Un programa de tipo-E que se utiliza debe adaptarse continuamente, en caso contrario, el programa se hace progresivamente menos satisfactorio. Estas adaptaciones son el resultado del cambio en la operación del entorno en el cual la aplicación cumple una función.

Ley II. Complejidad creciente: A medida que evoluciona un programa, su complejidad se incrementa, a menos que se trabaje para mantenerla o reducirla. Esta ley implica un tipo de "degradación" o "entropía" en la estructura del programa. Esto a su vez implica un aumento progresivo del esfuerzo de mantenimiento, a menos que se realice algún tipo de mantenimiento perfectivo a este respecto.

Ley III. Autorregulación: El proceso de evolución del programa se autorregula con una distribución de medidas de atributos de producto y procesos cercana a la normal. La evolución de programas industriales tipo-E se lleva a cabo por un equipo que opera en una organización más grande. Las decisiones de gestión respecto a los cambios en el programa constituyen una dinámica que determina las características de crecimiento del producto.

Ley IV. Conservación de la estabilidad organizativa: La velocidad de actividad global efectiva media en un sistema en evolución es invariante a lo largo del ciclo de vida del producto. Usualmente se considera que el esfuerzo gastado en la evolución del sistema se determina por decisiones de dirección. Esto es por supuesto así en un cierto grado, pero su influencia está limitada por factores externos respecto al empleo, la disponibilidad de personal competente, etc. No obstante, también influyen los atributos del sistema. Los datos empíricos sugieren que la actividad lleva a una estabilización de actividad aproximadamente constante.

Ley V. Conservación de la familiaridad: Durante la vida activa de un programa en evolución, el contenido de las versiones sucesivas es estadísticamente invariante. Uno de los factores que determina el progreso de un desarrollo de software es la familiaridad de todos los implicados. Cuantos más cambios y adiciones se hacen a una versión, es más difícil que todos los implicados la conozcan.

Ley VI. Crecimiento continuo: El contenido funcional de un programa debe incrementarse continuamente para mantener la satisfacción del usuario durante su ciclo de vida. Habitualmente, los sistemas se crean con una limitación en cuanto a la funcionalidad del dominio cubierta, por motivos de tiempo o recursos. Esto hace que con el tiempo, los requisitos que se descartaron vuelvan a aparecer como necesidades.

Ley VII. Calidad decreciente: Los programas de tipo-E serán percibidos como de calidad decreciente a menos que se mantengan de manera rigurosa y se adapten al entorno operativo cambiante. Esta percepción de la calidad decreciente tiene que ver con los cambios en los criterios de aceptabilidad de los usuarios.

Modelos y estándares de desarrollo de software y mantenimiento

Estándar IEEE 1219 de mantenimiento del software

Este estándar describe un proceso iterativo para la gestión y ejecución de las actividades de MS. Los criterios establecidos se aplican tanto a la planificación del MS mientras este está en desarrollo, como a la planificación y ejecución de las actividades de Mantenimiento para productos software existentes (Párraga, 1999) (Kajko-Mattsson, 2006).

Las fases del ciclo de vida mediante las que se dirige el estándar son: Identificación del problema, Análisis, Diseño, Implementación, Pruebas del sistema, Pruebas de aceptación y Puesta en producción o liberación de versión. Dentro de cada una de estas fases, el estándar define una serie de procedimientos que se han de llevar a cabo y con los que se identifican la documentación, personas y productos software que intervienen. Las etapas que se han de cubrir son: Entradas, Procesos, Controles y Salidas.

Se cubren algunos de los diferentes tipos de MS, estos se clasifican dependiendo de las demandas de los usuarios del producto Software a mantener: Mantenimiento adaptativo, Mantenimiento correctivo, Mantenimiento perfectivo y Mantenimiento preventivo (Aggarwal, 2007).

Mantenimiento adaptativo: Es la modificación de un producto software, después de su puesta en producción, para mantener operativo un programa mientras se realiza un cambio en el entorno de producción. Tiene por objetivo la modificación de un programa debido a cambios en el entorno, bien cambios en el hardware o en el software, en el que se ejecuta. En cuanto a los cambios en el hardware cabe destacar, por ejemplo, las modificaciones en la plataforma de instalación, pasar de un sistema de 16 bits a uno de 32 bits; los cambios en el sistema de comunicaciones, pasar de un sistema de trabajo stand-alone a un sistema de red; etc.

En cuanto a los cambios en el software se pueden mencionar algunos ejemplos como: cuando se desea pasar un sistema Cliente-Servidor típico a un sistema Three-Tier; los cambios

de un cliente clásico desarrollado en Visual Basic, Delphi o C/C++ a un cliente "ligero" para ejecutarse sobre un navegador de Internet.

Este tipo de mantenimiento es el más usual debido a los rápidos cambios que se producen en la tecnología informática.

Mantenimiento correctivo: Es la modificación de un producto software después de su puesta en producción y para corregir los fallos descubiertos. El Mantenimiento Correctivo tiene por objetivo localizar y eliminar los posibles defectos de los programas. A pesar de las pruebas y verificaciones que aparecen en etapas anteriores del ciclo de vida del software, los programas pueden tener defectos. Un defecto en un sistema es una característica del sistema que podría ser la causa de un fallo. El fallo se produce cuando el comportamiento del sistema es diferente al esperado por su especificación. Estos fallos pueden ser de procesamiento, de rendimiento, de programación o de documentación. Según estudios realizados la mayoría de los defectos se originan en las fases de especificación de requisitos y de codificación, por lo que también son importantes las primeras fases del ciclo de vida para el MS.

Mantenimiento de emergencia: Es un mantenimiento correctivo realizado sin planificación previa y utilizado para mantener operativo el sistema.

Mantenimiento preventivo: Este tipo de mantenimiento no se menciona como tal en el estándar IEEE 1219, sino que se incluye como Mantenimiento Perfectivo, pero que es conveniente tener en cuenta debido a los costes que puede ahorrar frente a otro tipo de mantenimientos. Consiste en la modificación del producto Software sin alterar las especificaciones del mismo, para mejorar las propiedades de mantenimiento del producto y facilitar así las futuras tareas de mantenimiento. Los cambios que se llevan a cabo son en cuanto a los comentarios del código y la reestructuración de los programas para mejorar su comprensión.

Mantenimiento perfectivo: Es la modificación de un producto software, después de su puesta en producción y para mejorar el rendimiento o la mantenibilidad, que es la facilidad de mantenimiento que tiene un software, es decir, la facilidad de un sistema para ser modificado, lo que influye directamente en los costes del mantenimiento.

Además de las fases que se incluyen se describen de manera básica actividades de Planificación y estimación de esfuerzo, Gestión de riesgos y Gestión de la Configuración. Estas

actividades soportan las fases definidas y garantizan un mayor grado de efectividad en la aplicación del estándar.

ISO/IEC 12207:1995

Esta norma establece un marco de referencia común para los procesos del ciclo de vida del software, con una terminología bien definida a la que puede hacer referencia la industria del software. Contiene procesos, actividades y tareas para aplicar durante la adquisición de un sistema que contiene software, un producto software puro o un servicio software, y durante el suministro, desarrollo, operación y mantenimiento de productos software (Comité técnico AEN/CTN 71, 1995).

Esta norma incluye también un proceso que puede emplearse para definir, controlar y mejorar los procesos del ciclo de vida del software.

Procesos principales del ciclo de vida. Los procesos principales del ciclo de vida son cinco, estos dan servicio a las partes principales durante el ciclo de vida del software. Una parte principal es la que inicia o lleva a cabo el desarrollo, operación o mantenimiento de productos software. Estas partes principales son el adquisidor, el suministrador, el desarrollador, el operador y el mantenedor de productos software. Los procesos principales son:

1) Proceso de adquisición. Define las actividades del adquisidor, organización que adquiere un sistema, producto software o servicio software.

2) Proceso de suministro. Define las actividades del suministrador, organización que proporciona el sistema, producto software o servicio software al adquiridor.

3) Proceso de desarrollo. Define las actividades del desarrollador, organización que define y desarrolla el producto software.

4) Proceso de operación. Define las actividades del operador, organización que proporciona el servicio de operar un sistema informático en su entorno real, para sus usuarios.

5) Proceso de mantenimiento. Define las actividades del mantenedor, organización que proporciona el servicio de mantenimiento del producto software; esto es, la gestión de las modificaciones al producto software para mantenerlo actualizado y operativo. Este proceso incluye la migración y retirada del producto software.

Procesos de apoyo del ciclo de vida. Hay ocho procesos de apoyo del ciclo de vida. Un proceso de apoyo es el que apoya a otro proceso como parte esencial del mismo, con un

propósito bien definido, y contribuye al éxito y calidad del proyecto software. Un proceso de apoyo se emplea y ejecuta por otro proceso según sus necesidades. Los procesos de apoyo son:

1) Proceso de documentación: Define las actividades para el registro de la información producida por un proceso del ciclo de vida.

2) Proceso de gestión de la configuración: Define las actividades de gestión de la configuración.

3) Proceso de aseguramiento de la calidad: Define las actividades para asegurar, de una manera objetiva, que los productos software y los procesos son conformes a sus requisitos especificados y se ajustan a sus planes establecidos.

4) Proceso de verificación: Define las actividades para verificar hasta un nivel de detalle dependiente del proyecto software, los productos de software.

5) Proceso de validación: Define las actividades para validar los productos software del proyecto software.

6) Proceso de revisiones conjuntas: Define las actividades para evaluar el estado y productos de una actividad.

7) Proceso de auditoría: Define las actividades para determinar el cumplimiento de los requisitos, planes y contrato.

8) Proceso de solución de problemas: Define un proceso para analizar y eliminar los problemas (incluyendo las no conformidades) que sean descubiertos durante la ejecución del proceso de desarrollo, operación, mantenimiento u otros procesos, cualesquiera que sea su naturaleza o causa.

Entre las limitaciones de la norma tenemos:

Esta norma describe la arquitectura de los procesos del ciclo de vida del software, pero no especifica los detalles de cómo implementar o llevar a cabo las actividades y tareas incluidas en los procesos.

No pretende prescribir el nombre, el formato o el contenido explícito de la documentación que se genere. Si bien esta norma puede requerir la elaboración de diversos documentos de parecido tipo o clase, esto no implica que dichos documentos se desarrollen, agrupen o se

mantengan separados de alguna manera. Estas decisiones se dejan para el usuario de esta norma.

No prescribe un método o un modelo de ciclo de vida concreto para el desarrollo del software. Las partes en esta norma son las responsables de seleccionar un modelo de ciclo de vida para el proyecto software, y de elaborar una correspondencia entre los procesos, actividades y tareas de esta norma y los de dicho modelo. Las partes son también responsables de seleccionar y aplicar los métodos de desarrollo del software, y de llevar a cabo las actividades y tareas adecuadas para el proyecto software.

ISO 14764

Este estándar internacional aclara los requerimientos para el Proceso de MS. El MS es un proceso primario en el ciclo de vida de un producto software tal como se describe en ISO/IEC 12207. Este estándar internacional es parte de la familia de documentos ISO/IEC 12207 (Ruiz, 1999).

Este estándar internacional describe en gran detalle la gestión del Proceso de Mantenimiento enunciado de manera general en ISO/IEC 12207, además establece definiciones para los distintos tipos de mantenimiento y proporciona una guía aplicable a la planificación, ejecución, control, revisión y evaluación del proceso de mantenimiento. También se incluye el mantenimiento para múltiples productos software con los mismos recursos de mantenimiento.

Entre los aspectos más significativos del estándar se tienen los factores que pueden afectar la mantenibilidad. Un ejemplo son los siguientes aspectos que afectan a la mantenibilidad y deberían tenerse en cuenta al elegir el lenguaje de programación: portabilidad del lenguaje, legibilidad del lenguaje, estabilidad del lenguaje, auto-documentación, tolerancia a “trucos” que reducen la claridad, posibilidades de estructuración, facilidad para el desarrollo de nuevas versiones, posibilidades para la estructuración de datos, existencia de compiladores, estabilidad del compilador, posibilidades para la comprobación durante la compilación y ejecución, entre otros elementos.

En el estándar se describe la forma en la que las actividades del ciclo de vida tienen en cuenta desde el nacimiento del proyecto los elementos necesarios para garantizar una facilidad de mantenimiento o mantenibilidad del sistema en desarrollo.

Sin embargo entre las limitaciones del estándar está que las actividades se muestran estructuradas pero no se describen en detalle.

Métrica III

La metodología MÉTRICA Versión 3 ofrece a las organizaciones un instrumento útil para la sistematización de las actividades que dan soporte al ciclo de vida del software dentro del marco que permite alcanzar los siguientes objetivos (Gobierno de España, 2008):

- Proporcionar o definir Sistemas de Información que ayuden a conseguir los fines de la organización mediante la definición de un marco estratégico para el desarrollo de los mismos.
- Dotar a la organización de productos software que satisfagan las necesidades de los usuarios dando una mayor importancia al análisis de requisitos.
- Mejorar la productividad de los departamentos de Sistemas y Tecnologías de la Información y las Comunicaciones, permitiendo una mayor capacidad de adaptación a los cambios y teniendo en cuenta la reutilización en la medida de lo posible.
- Facilitar la comunicación y entendimiento entre los distintos participantes en la producción de software a lo largo del ciclo de vida del proyecto, teniendo en cuenta su papel y responsabilidad, así como las necesidades de todos y cada uno de ellos.
- Facilitar la operación, mantenimiento y uso de los productos de software obtenidos.

MÉTRICA Versión 3 tiene un enfoque orientado al proceso, ya que la tendencia general en los estándares se encamina en este sentido y por ello, como ya se ha dicho, se ha enmarcado dentro de la norma ISO 12.207, que se centra en la clasificación y definición de los procesos del ciclo de vida del software. Como punto de partida y atendiendo a dicha norma, MÉTRICA Versión 3 cubre el Proceso de Desarrollo y el Proceso de Mantenimiento de Sistemas de Información. La metodología descompone cada uno de los procesos en actividades y éstas a su vez en tareas. Para cada tarea se describe su contenido haciendo referencia a sus principales acciones, productos, técnicas, prácticas y participantes.

Así los procesos de la estructura principal de MÉTRICA Versión 3 son los siguientes:

- Planificación de sistemas de información.
- Desarrollo de sistemas de información.

- Mantenimiento de sistemas de información.

Proceso de mantenimiento en MÉTRICA: El objetivo de este proceso es la obtención de una nueva versión de un sistema de información desarrollado con MÉTRICA, a partir de las peticiones de mantenimiento que los usuarios realizan con motivo de un problema detectado en el sistema o por la necesidad de una mejora del mismo.

Como consecuencia de esto, sólo se considerarán en MÉTRICA Versión 3 los tipos de Mantenimiento Correctivo y Evolutivo. Se excluyen los tipos de Mantenimiento Adaptativo y Perfectivo, que abarcan actividades tales como la migración y la retirada de software que precisarían el desarrollo de un tipo de metodología específica para resolver su cometido.

Ante una petición de cambio de un sistema de información ya en producción, se realiza un registro de las peticiones, se diagnostica el tipo de mantenimiento y se decide si se le da respuesta o no, en función del plan de mantenimiento asociado al sistema afectado por la petición y se establece con qué prioridad. La definición de la solución al problema o necesidad planteada por el usuario que realiza el responsable de mantenimiento, incluye un estudio del impacto, la valoración del esfuerzo y coste, las actividades y tareas del proceso de desarrollo a realizar y el plan de pruebas de regresión.

La metodología recoge las actividades fundamentales del proceso de mantenimiento.

Mantema

Mantema es una metodología para el mantenimiento del software basada en el estándar ISO/IEC 12 207 y que fue desarrollada en la Universidad Castilla La Mancha, España (Polo, y otros, 2001).

En Mantema se trabaja con los siguientes tipos de mantenimiento:

- No Planificable (NP): Correctivo Urgente (UC).
- Planificable (P): Correctivo No Urgente (NUC), Perfectivo (PER), Adaptativo (A) y Preventivo (PRE).

La metodología define un grupo de tareas iniciales que se realizan independientemente del tipo de mantenimiento que se desee ejecutar. Se incluyen las actividades y tareas que deben ejecutarse al comienzo de proyecto de mantenimiento (y que, por lo general, se realizan solamente una vez), así como otra serie de actividades y tareas que se ejecutan cada vez que

llega una petición de modificación del software. Posteriormente se particulariza en las acciones propias de cada tipo de mantenimiento y finalmente se realizan una serie de tareas finales o de cierre donde se registran las soluciones dadas a las peticiones de mantenimiento.

La metodología toma en cuenta las actividades de modificación, gestión de configuración y redocumentación.

Propuesta cubana de procedimiento para el mantenimiento de software

La propuesta va dirigida a pequeñas organizaciones que tengan aplicaciones de gestión y requieran hacer evolucionar los mismos (Fábrega, 2010).

El procedimiento comienza con la solicitud de modificación y termina con la migración del sistema. Para su organización se ha estructurado en 5 etapas claves: Solicitud del mantenimiento, Análisis del sistema, Diseño de las modificaciones, Ejecución de las modificaciones, Migración y retirada. La propuesta se centra en equipos de MS pequeños de 2 a 6 personas y se incluyen todos los tipos de mantenimiento de software descritos hasta el momento.

Este procedimiento no se ajusta a las condiciones del centro dado que no prevé una etapa de planificación del mantenimiento, necesaria en el caso de grandes proyectos como los que posee el CEIGE. Tampoco gestiona el proceso de MS considerando los grandes equipos que son necesarios para gestionar grandes sistemas. No considera aspectos como la integración entre el desarrollo y el mantenimiento. Sin embargo hay que señalar que el procedimiento está basado en algunos de los estándares analizados hasta este punto, lo cual aporta elementos importantes en las etapas recogidas.

Técnicas de mantenimiento del software

Ingeniería inversa

La ingeniería inversa se ha definido como el proceso de construir especificaciones de un mayor nivel de abstracción partiendo del código fuente de un sistema software o cualquier otro producto (se puede utilizar como punto de partida cualquier otro elemento de diseño, etc.).

Estas especificaciones pueden volver ser utilizadas para construir una nueva implementación del sistema utilizando, por ejemplo, técnicas de ingeniería directa (Sicilia, 2008) (Abelson, y otros, 2001) (Chikofsky, y otros, 1990) (Bernd Bruegge, 2009).

Beneficios de Ingeniería Inversa

La aplicación de ingeniería inversa nunca cambia la funcionalidad del software sino que permite obtener productos que indican cómo se ha construido el mismo. Su realización permite obtener los siguientes beneficios:

- Reducir la complejidad del sistema: al intentar comprender el software se facilita su mantenimiento y la complejidad existente disminuye.
- Generar diferentes alternativas: del punto de partida del proceso, principalmente código fuente, se generan representaciones gráficas lo que facilita su comprensión.
- Recuperar y/o actualizar la información perdida (cambios que no se documentaron en su momento): en la evolución del sistema se realizan cambios que no se suele actualizar en las representaciones de nivel de abstracción más alto, para lo cual se utiliza la recuperación de diseño.
- Detectar efectos laterales: los cambios que se puedan realizar en un sistema puede conducirnos a que surjan efectos no deseados, esta serie de anomalías pueden ser detectadas por la ingeniería inversa.
- Facilitar la reutilización: por medio de la ingeniería inversa se pueden detectar componentes de posible reutilización de sistemas existentes, pudiendo aumentar la productividad, reducir los costes y los riesgos de mantenimiento.

La finalidad de la ingeniería inversa es la de desentrañar los misterios y secretos de los sistemas en uso a partir del código. Para ello, se emplean una serie de herramientas que extraen información de los datos, procedimientos y arquitectura del sistema existente.

Tipos de Ingeniería Inversa

La ingeniería inversa puede ser de varios tipos:

- Ingeniería inversa de datos: Se aplica sobre algún código de bases datos (aplicación, código SQL) para obtener los modelos relacionales o sobre el modelo relacional para obtener el diagrama entidad-relación.
- Ingeniería inversa de lógica o de proceso: Cuando la ingeniería inversa se aplica sobre código de un programa para averiguar su lógica o sobre cualquier documento de diseño para obtener documentos de análisis o de requisitos.

- Ingeniería inversa de interfaces de usuario: Se aplica con objeto de mantener la lógica interna del programa para obtener los modelos y especificaciones que sirvieron de base para la construcción de la misma, con objeto de tomarlas como punto de partida en procesos de ingeniería directa que permitan modificar dicha interfaz.

Reingeniería

Reingeniería del software se puede definir como: "modificación de un producto software, o de ciertos componentes, usando para el análisis del sistema existente técnicas de Ingeniería Inversa y, para la etapa de reconstrucción, herramientas de Ingeniería Directa, de tal manera que se oriente este cambio hacia mayores niveles de facilidad en cuanto a mantenimiento, reutilización, comprensión o evaluación."

Cuando una aplicación lleva siendo usada años, es fácil que esta aplicación se vuelva inestable como fruto de las múltiples correcciones, adaptaciones o mejoras que han podido surgir a lo largo del tiempo. Esto deriva en que cada vez que se pretende realizar un cambio se producen efectos colaterales inesperados y hasta de gravedad, por lo que se hace necesario, si se prevé que la aplicación seguirá siendo de utilidad, aplicar reingeniería a la misma (Sicilia, 2008) (Ebert, y otros, 2005) (B. B. Agarwal, 2009).

Entre los beneficios de aplicar reingeniería a un producto se puede incluir:

- Reduce los riegos evolutivos de una organización.
- Ayuda a las organizaciones a recuperar sus inversiones en software.
- Hace el software más fácilmente modificable.
- Amplía las capacidades de las herramientas CASE.
- Es un catalizador para la automatización del MS.
- Actúa como catalizador para la aplicación de técnicas de inteligencia artificial para resolver problemas de reingeniería.

La reingeniería del software involucra diferentes actividades como son:

- Análisis de inventarios.
- Reestructuración de documentos.
- Ingeniería inversa.

- Reestructuración de programas y datos.
- Ingeniería directa.

Restructuración del software

La restructuración del software modifica el código fuente y/o los datos en un intento de adecuarlo a futuros cambios. Tiende a centrarse en los detalles de diseño de módulos individuales y en estructuras de datos locales definidas dentro de los módulos.

Los beneficios de la reestructuración son:

- Programas de mayor calidad con mejor documentación y menos complejidad, y ajustados a las prácticas y estándares de la ingeniería del software moderno.
- Reduce la frustración entre ingenieros del software que deban trabajar con el programa, mejorando por tanto la productividad y haciendo más sencillo el aprendizaje.
- Reduce el esfuerzo requerido para llevar a cabo las actividades de mantenimiento.
- Hace que el software sea más sencillo de comprobar y depurar.

La reestructuración se produce cuando la arquitectura básica de la aplicación es sólida, aún cuando sus interioridades técnicas necesiten un retoque. Comienza cuando existen partes considerables del software que son útiles todavía y solamente existe un subconjunto de todos los módulos y datos que requieren una extensa modificación.

Los tipos de reestructuración, básicamente son 2: del código y de datos.

Reestructuración del código

La reestructuración del código se lleva a cabo para conseguir un diseño que produzca la misma función pero con mayor calidad que el programa original.

Un ejemplo de sería: tomar el código de forma de "plato de espaguetis" y derivar un diseño de procedimientos que se ajuste a la filosofía de la programación estructurada.

Reestructuración de datos

Análisis del código fuente, en primer lugar se evaluarán todas las sentencias del lenguaje de programación con definiciones de datos, descripciones de archivos, de E/S y descripciones de interfaz. Esta actividad a veces se denomina análisis de datos.

Una vez finalizado el análisis de datos, comienza el rediseño de datos. En su forma más sencilla se emplea un paso de estandarización de rediseño de datos que clarifica las definiciones de datos para lograr una consistencia entre nombres de objetos de datos, o entre formatos de registros físicos en el seno de la estructura de datos o formato de archivos existentes. Otra forma de rediseño, denominada racionalización de nombres de datos, garantiza que todas las convenciones de denominación de datos se ajusten a los estándares locales, y que se eliminen las irregularidades a medida que los datos fluyen por el sistema.

Cuando la reestructuración sobrepasa la estandarización y la racionalización, se efectúan modificaciones físicas en las estructuras de datos ya existentes con objeto de hacer que el diseño de datos sea más efectivo.

Esto puede significar una conversión de un formato de archivo a otro, o, en algunos casos, una conversión de un tipo de base de datos a otra (Sicilia, 2008) (Ramirez, y otros, 2008).

Conclusiones

A partir de las metodologías, estándares y técnicas estudiados se puede revisar un grupo de elementos que identifican más allá de los problemas que acarrea el MS de manera general, las peculiaridades del MS para los sistemas de gestión desarrollados en el CEIGE.

Se puede comenzar señalando que todos las metodologías o estándares estudiados pueden ser útiles en la definición de una guía o procedimiento dado que pueden soportar el mantenimiento de sistemas de gran complejidad como los que se realizan en el centro. Sin embargo se irán resaltando algunos aspectos en los que determinadas metodologías o estándares pueden ser más útiles y convenientes.

Es conveniente que el proceso de planificación del MS sea un factor que se tenga en cuenta buscando garantizar la estimación del tiempo y el esfuerzo necesario para cumplir con los Acuerdos de Nivel de Servicio pactados con los clientes. En este punto puede ser conveniente analizar el proceso correspondiente descrito en la norma ISO 14 764.

En correspondencia también con lo antes analizado es importante considerar una metodología o norma que permita de manera integral insertar las tareas de mantenimiento

como parte del desarrollo del software y que no sea visto solamente como un proceso que tiene lugar posterior a la implantación del producto. En este sentido especial atención se le puede prestar a la norma ISO 12207 y Métrica v3, el resto de las metodologías o estándares estudiados no consideran de manera general que las actividades de mantenimiento también tienen lugar durante el desarrollo del software.

La definición clara de roles y responsabilidades es un aspecto que debe ser considerando, dado que para organizar el proceso de MS en el CEIGE, es importante tener claridad de qué responsabilidades pueden ocupar los estudiantes y profesores. Estas decisiones deben apoyarse en el fondo de tiempo de cada cual. En este punto todas las metodologías y normas estudiadas lo consideran, algunas de ellas con un mayor grado de definición que otras. Este aspecto toma mayor importancia cuando se determine cuales serán concretamente las actividades que se incluirán en la guía o procedimiento resultante.

En cuanto a las técnicas de MS es importante considerarlas todas y ante todo enmarcarlas en tareas concretas que proponga la guía a realizar durante las tareas de análisis y diseño de la solución y que están presentes en todas las metodologías y estándares estudiados.

Existen áreas de procesos que son frecuentemente consideradas en el proceso de desarrollo de software sin embargo no siempre se considera necesario considerarlas durante proceso el MS, estas áreas son la de Gestión de Riesgos, Gestión de la Configuración y Gestión de las Versiones. Convenientemente se consideran en este punto las metodologías y estándares IEEE 1219, ISO 12 207 (concretamente a través de la ISO 14764) y Mantema.

La división del mantenimiento considerando la urgencia es un factor transcendental debido a que el centro puede definir en consecuencia que tareas tienen mayor prioridad y por tanto a cuáles dedica determinados recursos y a cuáles no. En este punto la metodología Mantema hace una distinción clara y conveniente para las peculiaridades del MS de los sistemas de gestión.

Métricas como las usadas en Métrica v3 y Mantema pueden ser útiles para garantizar el control del proceso y la posterior retroalimentación durante el MS. Algunas de estas métricas descritas como parte de Mantema son: Tiempo de respuesta a las anomalías críticas, Desviación de la cantidad de anomalías críticas, entre otras.

De manera concluyente se pude decir que Mantema es la metodología que de manera integral da cubrimiento a la mayoría de los aspectos analizados como se ha podido apreciar, y si a la guía o procedimiento a desarrollar en consecuencia se le adiciona los demás beneficios identificados pudiese tenerse una herramienta más completa que garantice la correcta implementación del proceso de mantenimiento.

CAPÍTULO 2. PROCESO DE GESTIÓN DEL MS EN SISTEMAS DE GESTIÓN

Introducción

El presente procedimiento tiene el propósito de guiar las actividades de mantenimiento de software de gestión en el CEIGE. Para la estructuración de dicho procedimiento se ha dividido el proceso de MS en tres etapas fundamentales: Actividades iniciales, Ejecución del mantenimiento, Actividades finales. Se describe el flujo de trabajo, los artefactos que se generan y los roles que intervienen en cada una de las etapas descritas. Los artefactos que se generan en cada momento a lo largo del proceso se reflejan en el (Anexo 1. Actividades y artefactos), y los responsables de cada una de las actividades en el (Anexo 2. Actividades y responsables). Durante el procedimiento se hace referencia a un grupo de roles que han sido formalizados por la universidad como parte de su proceso productivo, sin embargo, se adiciona como parte de la propuesta el rol de jefe de MS y miembro del equipo de MS. El rol de jefe de MS puede ser ocupado por una persona que posee las competencias definidas para el rol Jefe de proyecto (según el documento Roles y responsabilidades establecido como parte del proceso de mejoras en la UCI). El rol miembro del equipo de MS puede ser ocupado por una persona que cumpla con las competencias establecidas para el rol Programador (definido igualmente por la UCI). Ambos roles además de poseer las competencias establecidas deben dominar el procedimiento y las técnicas propuestas en esta investigación.

2.1Descripción del procedimiento

El presente procedimiento pretende enmarcarse dentro del proceso de soporte de los sistemas de gestión partiendo del principio que el soporte permite dar solución a todo tipo de incidentes, mientras que el procedimiento se concentra en la forma de resolver las incidencias referidas a las necesarias transformaciones del software. Es importante señalar que el procedimiento no solo se enmarca temporalmente en el período de resolución de las inconformidades, sino que incluye toda la planificación del proceso de mantenimiento de software y un grupo de recomendaciones a ejecutar durante el desarrollo que facilitarán el posterior MS.

El procedimiento se dividió en tres grandes etapas:

Actividades iniciales: cubre las actividades necesarias para la concepción y organización del mantenimiento.

Ejecución del mantenimiento: cubre las siguientes actividades: análisis, diseño, implementación y pruebas del sistema. Se hace una clara distinción entre los tipos de mantenimiento y las peculiaridades a considerar en cada caso.

Actividades finales: cubre un grupo de actividades asociadas con la migración y retirada del sistema.

A continuación se detallarán las actividades que conforman el procedimiento desarrollado.

2.1.1 Actividades iniciales

Antes de comenzar a darle mantenimiento a un producto es necesario realizar una planificación de las futuras actividades de mantenimiento, para determinar el esfuerzo humano, material y económico necesario para llevar a cabo un mantenimiento eficaz. También es importante dejar claramente establecidas las pautas que durante el desarrollo del producto pueden influir en el posterior mantenimiento del sistema, garantizándose desde etapas tempranas la mantenibilidad del mismo. Las actividades que deben realizarse son:

1. Tareas a ejecutar durante el desarrollo.
2. Familiarización con el software al que se le dará mantenimiento.
3. Determinar requisitos del proyecto de MS.
4. Determinar el esfuerzo de MS.
5. Actividades de gestión de la configuración.
6. Elaboración del Plan de MS.
7. Preparar entornos de producción para probar el procedimiento de MS.
8. Preparar el nuevo personal encargado del MS.
9. Recepción de las peticiones de modificación.
10. Decidir el tipo de mantenimiento de cada petición de modificación.

La correlación que existe entre las actividades de esta etapa se ilustra en la Figura 1.

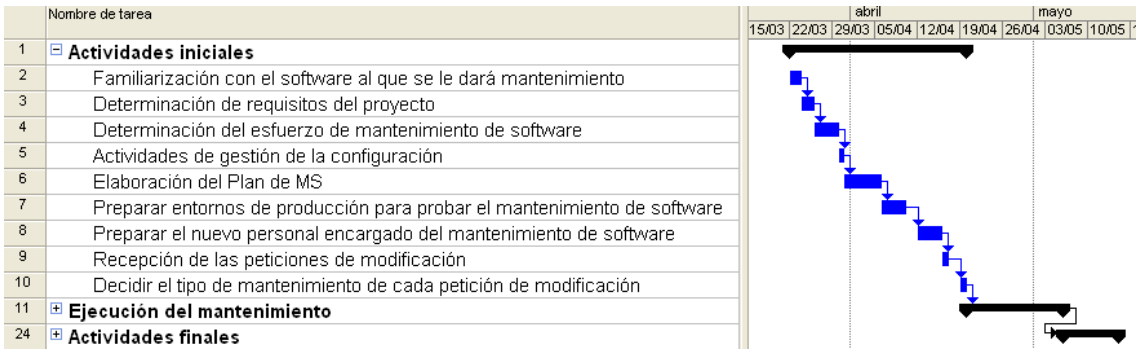

Figura 1. Relación entra las actividades de la etapa Actividades iniciales.

2.1.1.1 Tareas a ejecutar durante el desarrollo

Entre las tareas que se pueden acometer durante el desarrollo del sistema y que posteriormente pueden influir positivamente en el mantenimiento tenemos:

- Trabajar organizadamente a partir de un modelo de desarrollo de software que tenga claramente definidos las actividades, los artefactos y roles que intervienen en cada momento.
- Establecer y difundir con claridad la forma de estandarización del código y como parte de esto exigir los comentarios asociados al mismo.
- Aplicar convenientemente estilos de arquitectura y patrones de diseño que garanticen la mantenibilidad.

El máximo responsable de esta actividad es el jefe de proyecto, aunque generalmente las tareas técnicas recaen sobre el arquitecto principal del sistema.

2.1.1.2 Familiarización con el software que se le dará mantenimiento

El equipo que asumirá el MS debe estudiar detalladamente la documentación existente del sistema a mantener y el código correspondiente, si no se cuenta con el equipo completo que se encargará del MS al menos sí debe ser esta una actividad que realicen las personas que más tarde definirán los requerimientos de mantenimiento y realizarán las estimaciones necesarias. Sería muy conveniente emplear en las tareas de mantenimiento a personal que haya formado parte del desarrollo del sistema. El responsable de esta tarea es el jefe de MS.

2.1.1.3 Definición de los requisitos del proyecto de MS

A este nivel, el proceso de mantenimiento necesita ser asociado al entorno de negocio. Se debería completar una revisión de las expectativas que podría incluir los aspectos que se

mencionan a continuación. La especificación de dichos aspectos se hace usando como base la plantilla de Especificación de Requisitos de Software (Yzquierdo, 2010), esta plantilla recoge la especificación tanto de los requisitos funcionales como no funcionales. Hay que tener presente que, aunque se usa esta plantilla de Especificación de Requisitos de Software, a la misma se le hacen algunos ajustes para poder incluir la totalidad de los elementos que son de interés gestionar durante el MS. El documento generado como salida de esta actividad recogerá las situaciones y requisitos esperados, y no necesariamente los requisitos que tienen que implementarse. Se explica a continuación lo antes mencionado:

- Cambios del sistema esperados, externos o regulatorios (Sección que se le agrega a la plantilla). Este aspecto debe ser del dominio del jefe de MS.
- Cambios internos esperados para soportar los nuevos requisitos (Sección que se le agrega a la plantilla). Este aspecto debe ser del dominio del jefe de MS y del arquitecto.
- Lista de nuevas funciones y características deseadas (Se describe en la sección de Requisitos funcionales). Este aspecto debe ser especificado por los analistas del proyecto.
- Mejoras esperadas en el rendimiento, adaptabilidad, conectividad, etc. (Se describe en la sección de Requisitos no funcionales). Este aspecto debe ser especificado por los analistas del proyecto.
- Nuevas líneas de negocio que necesitan ser soportadas (Sección que se le agrega a la plantilla). Este aspecto debe ser del dominio del jefe de MS.
- Nuevas tecnologías que necesitan ser incorporadas (Sección que se le agrega a la plantilla). Este aspecto debe ser del dominio del arquitecto.

Al documento generado se le llamará Especificación de Requisitos de MS Esperados (Yzquierdo, 2010).

Los encargados de la definición de los requisitos del MS son: el jefe de MS, el arquitecto y los analistas del proyecto. La utilidad del artefacto generado está dada por la posibilidad de poder cuantificar o dimensionar el alcance para determinar la carga de mantenimiento futuro para la organización. Hay que considerar que los datos recopilados son una estimación y no necesariamente un reflejo de lo que será la realidad del MS.

Hasta este momento se ha analizado el sistema a mantener y se ha realizado la primera definición de lo que pudiese ser el alcance del MS, sin embargo pueden surgir dudas sobre qué hacer con determinadas mejoras que se prevén para el sistema.

Diferenciar entre continuar con el MS o emprender un nuevo desarrollo.

Decidir correctamente cuándo continuar con el MS o cuándo encaminarse en un nuevo desarrollo es una decisión en la que influyen un grupo de aspectos tales como:

- Correcta clasificación de las solicitudes de modificación. En este aspecto se pueden resaltar dos tipos de mantenimiento que tienden a generar las principales dudas con respecto a qué decisión tomar, el mantenimiento perfectivo y el mantenimiento adaptativo.
    - Mantenimiento perfectivo: como parte de este tipo MS se puede estar considerando adicionar nuevas funcionalidades (requisitos) y mejorar el rendimiento de la versión actual. Sin lugar a dudas las mejoras dirigidas al rendimiento deben mantenerse dentro del proceso de MS mientras que cuando se valora incluir nuevas funcionalidades al sistema hay que considerar lo que se explica en la Planificación de las versiones.
    - Mantenimiento adaptativo: como parte de este tipo de mantenimiento se pueden realizar modificaciones a las herramientas y framework de trabajo usados para el desarrollo del software. En consecuencia, cualquier mejora que esté dirigida a adaptarse a una nueva tecnología debe mantenerse como MS, siempre y cuando esta tecnología no haya sido sustituida para dar soporte a nuevas funcionalidades del negocio no incluidas en la versión actual del sistema.
- Planificación de las versiones. Esta es una tarea que puede resultar compleja dado que no siempre se tiene la claridad absoluta de qué funcionalidades deben cubrirse del negocio a informatizar. Es necesario que se haga un análisis profundo del alcance de cada una de las versiones previstas, mapeándose cada funcionalidad prevista en las diferentes versiones del sistema. Una versión del sistema puede incluir tanto el desarrollo de nuevas funcionalidades como la corrección de las funcionalidades en una versión anterior del sistema. El desarrollo de esas nuevas funcionalidades se debe asumir desde el proceso de desarrollo de software considerando las normas

establecidas para la integración con las soluciones dadas por el MS. Aunque el presente procedimiento incluye un grupo de actividades que pudiesen dar soporte al desarrollo de nuevas funcionalidades se considera que no es adecuado hacerlo, dado que no se garantiza, entre otros aspectos, el control del proceso de desarrollo de software como se debería.

2.1.1.4 Determinar el esfuerzo de mantenimiento

A la hora de estimar el esfuerzo requerido para el mantenimiento de software puede partirse de dos posibles situaciones:

1. Se cuenta con experiencia y datos referentes a mediciones realizadas durante las actividades de MS ejecutadas hasta el momento.
2. No existe ninguna o existe muy poca experiencia por parte del equipo en tareas de mantenimiento de software.

El objetivo principal de esta actividad, sin importar la situación, es obtener la descripción del esfuerzo global y las necesidades del área de mantenimiento.

En la primera de las situaciones se recomienda comenzar haciendo un análisis de los niveles de servicio y capacidades existentes. Esto incluye un análisis del Plan de mantenimiento desarrollado anteriormente y el estado de cada uno de los sistemas dentro de este plan. Las experiencias anteriores pueden dar una base para la estimación del nuevo mantenimiento.

El análisis a este nivel es simplemente para reunir aquellas medidas necesarias para determinar lo siguiente:

- La relación de gastos y las posibles razones para abandonar el mantenimiento. Se deben valorar los siguientes aspectos a la hora de definir los gastos asociados al MS:
    - Cantidad de personas dedicadas al MS (diferenciando las responsabilidades de cada una).
    - Costos del equipamiento y otros recursos materiales utilizados.
    - Tiempo estimado para el MS de cada sistema.

  Las posibles razones para abandonar el mantenimiento necesitan ser documentadas para apoyar cualquier decisión en este sentido.

- Nivel de experiencia del grupo de mantenimiento. Los conocimientos necesarios deben considerar 3 áreas: conocimiento del negocio, conocimiento de la tecnología a mantener y conocimiento del procedimiento de MS.

- Los métodos documentados actualmente para mantenimiento a nivel de sistema. Esto influye en la facilidad de mantenimiento, por tanto es un factor que puede disminuir los tiempos de análisis de las incidencias a resolver.

- Complejidad de la solución a mantener.

- Los métodos utilizados actualmente por el grupo de programación. Conocer las normas de codificación.

- Definición de los requisitos del proyecto de MS. Se obtendrían los niveles de servicios que hay que satisfacer durante el mantenimiento de software.

- Las herramientas utilizadas para el soporte del mantenimiento y cómo se utilizan. Contar con una serie de herramientas que faciliten el trabajo durante el MS disminuiría considerablemente el esfuerzo necesario para resolver los problemas que se presenten. Se recomiendan en el (Anexo 3. Herramientas de apoyo al MS) un grupo de herramientas útiles para el MS (Centro de Gestión Avanzado, 2009) (Verónica Farach, 2005) (case-tools, 2009) (Software Informer, 2010) (Jeffrey A. Hoffer, 2008) (Susan J Chinn, Scott J Lloyd, Eric Kyper, 2005) (GRUPO DE INVESTIGACIÓN ARQUISOFT, 2007) (Object Mentor, 2009).

Según Capers Jones (Jones, 2009) y otros autores, las estimaciones más efectivas pudiesen hacerse a partir de la utilización de métricas de puntos de función, por lo que se proponen un grupo de ellas que pueden ayudar en la realización de esta tarea y de las que se describen a continuación. Ver (Anexo 4. Descripción y rango de valores de las medidas propuestas

La información a este nivel es utilizada para definir la línea de referencia para la organización del mantenimiento y para proveer un entorno de valoración de las necesidades de cambio.

Se han recomendado un grupo de aspectos que no pueden dejarse de tener en cuenta a la hora de determinar el esfuerzo necesario para las tareas de mantenimiento, sin embargo hay que señalar que cada organización tiene peculiaridades por lo que no se propone ninguna

función específica que pudiese ajustarse genéricamente a las características de cualquier organización que realice MS.

Si la organización no cuenta con experiencia en actividades de mantenimiento, se recomienda un grupo de elementos que pudiesen guiar básicamente las acciones durante la estimación del esfuerzo global necesario. No se debe olvidar que, como se mencionó durante la introducción de este trabajo, según estudios recientes de especialistas como Bennett y Rajlik plantean que la etapa de mantenimiento puede estar consumiendo el 65 % de los costos del proyecto y el 80 % del tiempo destinado al mismo (Sneed, 2008). Estudios como estos no pueden perderse de vista aún cuando las características de la organización, del producto a mantener y las circunstancias sean diferentes para cada caso.

Hay que partir de que se desea estimar estos tres aspectos de manera básica, como se había mencionado anteriormente: cantidad de personas dedicadas al MS, costos del equipamiento y otros recursos materiales utilizados y tiempo estimado para el MS de cada sistema.

Lo primero es que la organización debe empezar a medir todos los aspectos a los que se hizo referencia anteriormente y que de alguna forma las métricas propuestas pueden ayudar. La meta debe ser que exista una estimación de los recursos y el tiempo que se requiere para cada una de las actividades atómicas que se proponen en este procedimiento: recepción de la solicitud de modificación, clasificación de la misma, análisis, diseño, codificación y prueba del sistema. Sin embargo, se puede señalar que las actividades atómicas asociadas al análisis, diseño, codificación y prueba del sistema se realizaron durante el desarrollo del producto, y las estimaciones realizadas en esos momentos pueden dar una idea de cómo manejar, muy a groso modo, las solicitudes de modificación. Es decir, se haría una analogía entre el esfuerzo necesario para desarrollar y probar un requisito (puede haberse utilizado para la estimación casos de uso o cualquier otro concepto unitario) durante el desarrollo y el esfuerzo necesario para modificar y corregir un requisito durante el MS. En el análisis de una solicitud de modificación puede consumirse mucho tiempo y recursos, dado que, casi nunca es sencillo entender una funcionalidad desarrollada por otra persona y más si el software no está correctamente documentado e implementado. La estimación realizada puede afectarse por estos y otros aspectos ya mencionados (comentarios del software, experiencia del equipo, entre otros aspectos).

El responsable de esta actividad es el jefe de mantenimiento. La información generada se especificará en la sección correspondiente en el Plan de Mantenimiento de Software (versión preliminar) (Yzquierdo, 2010).

2.1.1.5 Actividades de Gestión de la Configuración

La Gestión de Configuración (GC), es una disciplina que tiene como misión controlar la evolución de un sistema software en su definición más abarcadora. Las actividades relacionadas con la GC durante el MS deben ajustarse a las pautas definidas por la organización en este sentido.

Un aspecto importante es que la GC debe estar centrada en un esquema orientado al producto y no al proyecto, dado que solo con este esquema se estaría garantizando la correcta gestión de la evolución del software, la cual no se detendrá al terminar un proyecto de desarrollo. Este tema no es abordado con mayor profundidad porque no constituye un objetivo de esta investigación.

A pesar de lo antes expuesto cabe señalar que la principal tarea de la GC es llevar un registro actualizado de todos los elementos de configuración de la infraestructura de las Tecnologías de la Información (TI) junto con sus interrelaciones. Esto no resulta una tarea sencilla considerándose la influencia que se ejerce sobre los elementos de configuración durante la gestión de los cambios y la estructuración de las diferentes versiones del sistema. Para realizar una correcta gestión del MS se hace necesario que la GC permita acceder a información fiable. Además se tiene que controlar los elementos que configuran la infraestructura de las TI y mantener actualizada la Base de Datos de Configuraciones (BDC). Garantizándose que se puedan soportar los procesos de mantenimiento de manera ágil y precisa. Los elementos de configuración afectados durante las tareas de MS deben ser cuidadosamente tratados y se debe registrar el estado actual de cada uno en la BDC del proyecto.

El encargado de hacer cumplir las definiciones correspondientes a la GC es el responsable del área en el proyecto y el jefe de MS debe también velar por el cumplimiento de las regulaciones establecidas.

2.1.1.6 Elaboración del Plan de MS

La información recogida proporciona una base para el nuevo Plan de MS. El plan deberá cubrir cuatro áreas principales: proceso de mantenimiento, organización, reserva de recursos y auditoría del rendimiento. Cada uno de estos elementos será incluido en el Plan de MS final.

Alcance del proceso: El plan necesita definir las fronteras del proceso de mantenimiento. El proceso comienza en algún punto (recepción de la Solicitud de Modificación) y termina con alguna acción (puesta en producción). La diferencia entre el mantenimiento y el desarrollo se verá claramente en este punto. Otro elemento que debería ser definido dentro del alcance es si debe categorizarse el proceso de mantenimiento y cómo.

Secuencia del proceso: La secuencia debería utilizar el proceso descrito en este documento como una guía. Es necesario describir el flujo global de trabajo incluyendo lo siguiente:

- Descripciones de cada uno de los pasos del proceso y sus interfaces.
- Entrada en la gestión de configuración del software y los sistemas de gestión de proyectos.
- Flujo de datos entre los procesos.

Control del proceso: Cada uno de los pasos en el proceso debería ser controlado y medido. Se tienen que definir los niveles de rendimiento esperados en la medida de lo posible. Los mecanismos de control pueden ser automatizados en lo posible.

Organización: Se reflejan las estimaciones realizadas para la definición del tamaño del grupo de trabajo y también se reflejan los roles correspondientes del equipo conformado.

Reserva de recursos: Se describen a partir de las estimaciones realizadas todos los recursos necesarios. Se debe detallar el momento en el que se hará uso de cada recurso para que pueda ser gestionado correctamente.

Auditoría del rendimiento: Una vez iniciado el proceso de mantenimiento, se debería monitorizar y evaluar para juzgar su efectividad. Si cada uno de los pasos en el proceso tiene sus criterios de medida, debería ser sencillo obtener y evaluar el rendimiento.

Para esta tarea se propone un artefacto llamado Plan de Mantenimiento de Software (Yzquierdo, 2010) y debe ser completado por el jefe de MS.

2.1.1.7 Preparar entornos de producción para probar el procedimiento de MS

Es necesario preparar las condiciones para que después a la liberación del producto pueda probarse el procedimiento diseñado en un entorno representativo del real. Esto permitirá engranar el procedimiento de MS y por tanto garantizará el éxito de este proceso después de implantado el sistema.

En la UCI, como parte del proceso de desarrollo de software, se incluye una etapa que se denomina Piloto, en la que se realiza un despliegue de la solución a una menor escala y para un grupo de clientes que constituyen una muestra representativa de la población que explotará el sistema desarrollado. Durante esta etapa se generan todo tipo de incidencias, por tanto, es conveniente poner en funcionamiento el procedimiento de MS definido. La cantidad de recursos dependerá de la dimensión y complejidad del Piloto.

En este momento es conveniente usar técnicas de comunicación como pueden ser las sesiones de Lecciones Aprendidas en la que con la participación de la totalidad del equipo de MS pueden reunirse las mejores experiencias para ajustar el procedimiento en función de las peculiaridades del entorno.

Como salida a esta actividad se debe actualizar el Plan de Mantenimiento de Software (Yzquierdo, 2010) en correspondencia con las condiciones detectadas y los ajustes al procedimiento identificados en cuestión. El responsable de esta actividad es el jefe de MS.

2.1.1.8 Preparar el nuevo personal encargado del MS

Las personas que se encargarán del mantenimiento de software deben estar preparadas no solo en el manejo de la solución de software sobre la que se incidirá sino que deben dominar el procedimiento de MS a implementar, y por consiguiente, un grupo de técnicas de MS. Debe quedar claramente establecido un programa docente y los recursos destinados a la capacitación. Para cada uno de los cursos de capacitación necesarios se hará uso de la documentación técnica del sistema desarrollado y del Plan de Mantenimiento de Software (Yzquierdo, 2010). Es necesario dejar establecido un cronograma con las fechas y los recursos necesarios para los cursos que se impartirán. Estas acciones pueden ser coordinadas por un integrante del equipo de MS y supervisadas por el jefe de MS. No necesariamente tienen que ser los miembros del equipo de mantenimiento quienes impartan la totalidad de la preparación.

2.1.1.9 Recepción de las peticiones de modificación

La admisión y registro del incidente es el primer y necesario paso para una correcta gestión del mismo. Este registro debe hacerse por parte del equipo de soporte del centro y solo llegan al equipo de mantenimiento aquellas solicitudes que requieren de una modificación del sistema desplegado. Las incidencias pueden provenir de diversas fuentes tales como: el centro de soporte, el equipo de mantenimiento o el equipo de desarrollo, entre otros.

Entre los aspectos que el equipo de MS debe tener presente se encuentran:

La admisión a trámite del incidente: Es decir, si el problema reportado no está entre las posibles incidencias recogidas en los Acuerdos de Nivel de Servicios (SLAs) entonces se consulta con la dirección del proyecto qué acción seguir y se le informa al cliente.

Comprobación de que ese incidente aún no ha sido registrado: es moneda corriente que más de un usuario notifique la misma incidencia y por lo tanto han de evitarse duplicaciones innecesarias.

Registro inicial: se ha de introducir, haciendo uso del sistema seleccionado, la información básica necesaria para el procesamiento del incidente (hora, descripción del incidente, sistemas afectados, etc.). Aunque esta información se recopila por el equipo de soporte, el equipo de MS puede enriquecerla convenientemente.

Información de apoyo: se incluirá cualquier información relevante para la resolución del incidente la cual puede ser solicitada al cliente a través de un formulario específico (el formulario se le hará llegar al equipo de soporte para que sea tramitado con el cliente), o que pueda ser obtenida de la propia BDC.

Notificación del incidente: en los casos en que el incidente pueda, según el criterio del equipo de MS, afectar a otros usuarios, estos deben ser notificados para que conozcan cómo esta incidencia puede afectar su flujo habitual de trabajo. De esta notificación se encarga el equipo de soporte.

Como resultado de esta actividad se actualiza la descripción del incidente que tiene asociado una modificación del sistema y de ello se encarga el miembro del equipo de MS que haya sido designado para la recepción del incidente.

2.1.1.10 Decidir el tipo de mantenimiento de cada petición de modificación

En esta actividad se identifican, clasifican y asignan una prioridad inicial a las modificaciones del software. Cada solicitud de modificación será evaluada para determinar su clasificación y prioridad. Esta contiene diversos cambios a realizar sobre el producto de software que posteriormente se agruparán en bloques de implementación y darán lugar a las diferentes versiones. La clasificación será identificada según los tipos de mantenimiento: correctivo urgente o no urgente, adaptativo y perfectivo. El mantenimiento preventivo no se incluye en este momento, dado que este tipo de mantenimiento surge como una necesidad del equipo de MS. Aún así sí se considera en la planificación del MS del proyecto.

Para mantener en todo momento una correcta organización y control del MS es aconsejable hacer revisiones periódicas de los elementos de modificación recibidos, de esta manera se pueden observar los más críticos y solicitados, y ayudaría a prevenir cualquier bloqueo de trabajo o parada en alguna de las fases del MS por la falta de orientación.

Una vez recibida la solicitud de modificación, comienza el MS. En este momento las acciones a seguir son:

- Clasificación del tipo de mantenimiento.
- Análisis de la modificación para determinar si se acepta, se deniega o se evalúa.
- Realizar una estimación preliminar de la magnitud de la modificación.
- Priorizar la modificación.
- Asignar solicitudes de modificación a bloques de tareas planificadas para su implementación.

Se deben realizar reuniones de revisión en las que participará el jefe de proyecto, el arquitecto o los arquitectos, el analista principal, funcionales y jefe de MS. Pueden participar otros implicados en dependencia de las peculiaridades del producto. La frecuencia y duración de las reuniones de revisión de la clasificación deberán ser dependientes de las circunstancias. Como guía, estas reuniones deberían ser consideradas como revisiones de estado en vez de revisiones técnicas. Si después de unas pocas sesiones las revisiones duran más de una hora, debería aumentarse su frecuencia, que inicialmente puede ser semanal. Si esto sigue siendo insuficiente es posible que se deba a tres razones principales:

- Si la discusión se centra en mejoras del sistema (mantenimiento perfectivo) será porque es necesario analizar la magnitud del desarrollo antes que llegar a un nivel de mantenimiento sostenido.
- Si el sistema es de nuevo desarrollo y requiere un importante esfuerzo de mantenimiento justo después de su puesta en producción, la estrategia a seguir será clasificar las solicitudes de modificación por tipo de mantenimiento e integrarlas en conjuntos para compartir las mismas áreas de diseño. De esta forma se priorizará por el conjunto, en vez de por solicitud de modificación individual, con lo que conseguirá minimizar la repetición de las tareas de diseño, codificación, pruebas y liberación.
- Si el sistema es antiguo, es posible que se considere reemplazarlo o rediseñarlo. De esta forma vemos la importancia de clasificar las modificaciones a realizar para que puedan ser aplicadas a un sistema particular de la mejor manera posible sin que afecte al sistema existente o a otras posibles modificaciones.

La prioridad de las solicitudes de modificación deberá ser asignada considerando los siguientes criterios:

- Recursos estimados inicialmente según la facilidad/dificultad de implementación y el tiempo aproximado para llevarla a cabo según los recursos disponibles.
- Impacto esperado a los usuarios actuales y futuros, indicando las ventajas y desventajas.
- Asignación de un bloque de modificaciones planificadas para minimizar el impacto a los usuarios.

Este es un momento en el que debe hacerse especial esfuerzo en gestionar adecuadamente la comunicación, dado que, el éxito de procesos como la integración entre los resultados del desarrollo y los del mantenimiento dependen casi totalmente de la correcta comunicación que se establezca entre los miembros de los equipos implicados. En consecuencia hay que dejar claramente establecido:

- Elemento de comunicaciones: La información que será distribuida a los interesados.
- Finalidad: Motivo de la distribución de dicha información.
- Frecuencia: Cada cuánto tiempo se distribuirá la información.
- Fechas de inicio / finalización: Plazo para la distribución de la información.

- Formato / medio: El diseño de la información y el método de transmisión.
- Responsabilidad: El miembro del equipo encargado de la distribución de la información.

Cuando finalice esta primera etapa, se podrá disponer de una solicitud de modificación validada y las tareas que se van a llevar a cabo para realizar los cambios solicitados. Para evidenciar la conclusión de esta etapa, tendrá que recopilar en el repositorio lo siguiente:

- Relación de los problemas o nuevos requerimientos.
- Evaluación de los problemas o nuevos requerimientos.
- Clasificación del tipo de mantenimiento solicitado.
- Prioridad inicial.
- Datos de verificación (para el mantenimiento correctivo).
- Estimación inicial de los recursos necesarios para modificar el sistema existente.

En la etapa de Análisis se realizarán los estudios de impacto y costos a bajo nivel, pero para realizar la clasificación inicial es deseable comprobar los costos generales estimados. Ya que esto es un proceso iterativo, cabe la posibilidad de que la gestión de la prioridad cambie durante las siguientes etapas. En consecuencia, se describirán los aspectos señalados en el artefacto propuesto para la descripción de los requisitos de MS, Especificación de Requisitos de MS Esperados (Yzquierdo, 2010) y en el artefacto usado para el análisis de viabilidad, Análisis de Viabilidad (Yzquierdo, 2010). Se modificará la solicitud de modificación actualizando información como la prioridad, se actualizará el cronograma del proyecto en correspondencia con las estimaciones preliminares.

2.1.2 Ejecución del mantenimiento

Este procedimiento define cambios en un producto software a través de un proceso de mantenimiento dividido en fases. Sin importar el tipo de mantenimiento que se realice, siempre se transita por las siguientes cuatro fases: Análisis, Diseño, Implementación y Pruebas del Sistema. Este proceso es iterativo y en cascada, con una gran semejanza al ciclo de vida del desarrollo clásico.

Según se ha diseñado, el proceso de mantenimiento se inicia con una Solicitud de Modificación, que incluye los formularios asociados con los informes de problemas o fallos, documentos para el control de cambios en la configuración, entre otros.

Primero se expondrán las actividades generales a las que se hace referencia recogiendo los aspectos más importantes y posteriormente se describirá para cada tipo de MS las acciones particulares. La relación entre las actividades que se realizan durante la etapa de Ejecución del MS se especifica en la Figura 2.

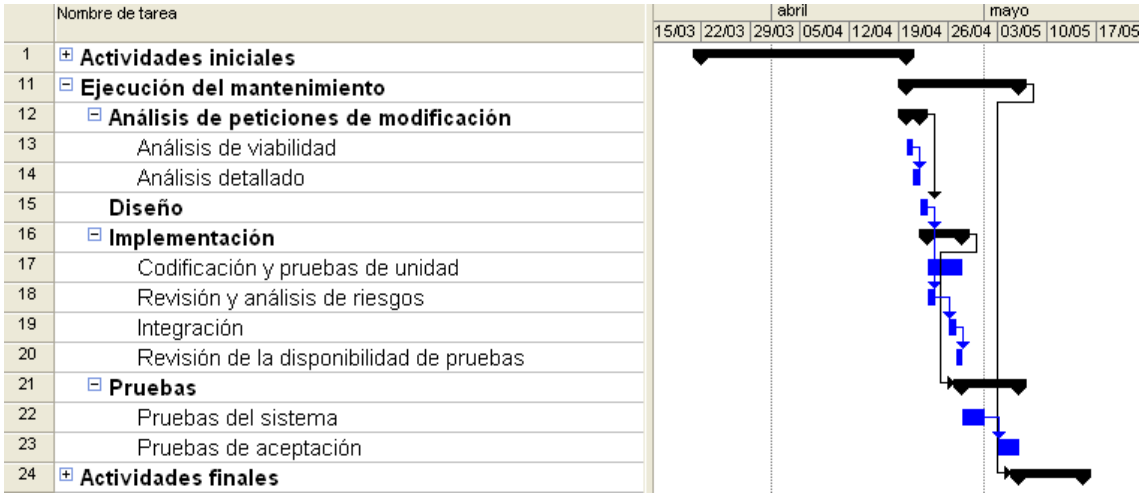

Figura 2. Relación entra las actividades durante la Ejecución del MS.

2.1.2.1 Análisis

En este punto se estudia la viabilidad y el alcance de las modificaciones, que ya se tienen clasificadas y priorizadas, así como la generación de un plan preliminar de diseño, implementación, pruebas y liberación del software. La información necesaria proviene del repositorio y de la Solicitud de Modificación validada en la etapa anterior, además de la documentación del sistema existente.

Una Solicitud de Modificación podría generar varios requisitos de funcionalidad, rendimiento, usabilidad, fiabilidad, comprensibilidad y mantenibilidad, que podrán ser descompuestos en varios requisitos de software, base de datos, interfaces, documentación y hardware. Por lo tanto es necesario que participen tanto los funcionales y desarrolladores (miembro del equipo de MS) para asegurar que los elementos de la solicitud no sean ambiguos. Si la documentación no está disponible o no es suficiente, y el código fuente es la única representación fiable del sistema, la ingeniería inversa es el método más recomendado (Ebert, y otros, 2005). Así mismo en dependencia de la situación pueden estarse recomendando que durante el diseño se haga uso de las restantes técnicas para el MS descritas en el primer capítulo: Reingeniería y Restructuración del software.

El análisis es un proceso iterativo que tiene al menos dos componentes: el análisis de viabilidad y el análisis detallado.

Análisis de viabilidad

El análisis de viabilidad debe desembocar en un Informe de Viabilidad, que contiene los siguientes elementos:

- Impacto de las modificaciones. Identificar los efectos bilaterales potenciales.
- Soluciones alternativas, incluyendo el prototipado de la solución.
- Análisis de los requisitos de conversión.
- Implicaciones de seguridad.
- Implicaciones de robustez.
- Factores humanos.
- Costes a corto y largo plazo.
- Beneficios de realizar las modificaciones

Análisis detallado

Al identificar los elementos susceptibles de modificación en el análisis, se examinan todos los elementos implicados en las etapas del MS que pueden ser afectados. En consecuencia se hacen las estimaciones necesarias para obtener una solución en cuanto a recursos y tiempo.

En cuanto a la estrategia de pruebas se tendrán que definir los requisitos para al menos tres niveles de pruebas: las pruebas de elementos individuales, las de integración y las de aceptación del usuario final. En estos también se ha de incluir, para cada uno de estos niveles, los requisitos de las pruebas de regresión, que se realizan para validar que el código que se ha modificado no introduce errores que no existían antes de la actividad de mantenimiento.

Por último, se tiene que desarrollar un plan de implementación preliminar.

En resumen, el análisis detallado contempla los siguientes pasos:

- Definición de los requisitos fijados para la modificación. Se describen en el artefacto Especificación de Requisitos de MS Esperados (Yzquierdo, 2010).

- Identificación de los elementos a modificar. Se describen en el artefacto propuesto Análisis de Viabilidad (Yzquierdo, 2010).
- Identificación de los elementos de seguridad. Se describen en el artefacto propuesto Análisis de Viabilidad.
- Identificación de los elementos de robustez. Se describen en el artefacto propuesto Análisis de Viabilidad.
- Definición de la estrategia de pruebas. Se describen en el artefacto Plan de Pruebas v2.0 utilizado por la universidad (Universidad de las Ciencias Informáticas, 2009).
- Desarrollo de un plan para la fase de implementación. Se describen en el artefacto Cronograma del Proyecto definido por el área afectada.

La gestión durante el análisis tiene que realizarse metódicamente, obteniendo, en primer lugar, la última versión de la documentación del proyecto y el sistema ha modificarse. Se verificará que toda la documentación del análisis y del proyecto sea actualizada. En cuanto a las pruebas, hay que verificar que exista la estrategia apropiada para la realización de estas y que la planificación de los cambios pueda soportar la realización de las mismas.

Los analistas (jefe de MS, arquitecto y equipo de mantenimiento) tendrán que revisar los cambios propuestos para poder documentar la contribución técnica necesaria y la viabilidad económica de las modificaciones. Una parte importante será la identificación de los elementos que afectan al control de la seguridad y la robustez del sistema, para comprobar cómo influyen los cambios en estos atributos. Todo esto se detalla en el artefacto Análisis de Viabilidad propuesto.

En el MS, el principal efecto se produce por realizar modificaciones en un sistema software en producción, por lo que habrá que considerar la integración de los cambios propuestos dentro del sistema existente, como uno de los principales factores de riesgo. Para una correcta integración entre el resultado del MS y los nuevos requisitos en desarrollo se tienen que asumir varias medidas:

- Garantizar que se realicen reuniones de planificación y confección de las versiones; de forma tal que se garantice que las funcionalidades que se implementarán en el sistema estén correctamente estructuradas y organizadas en versiones progresivas. Concretamente se estaría organizando el trabajo del equipo de MS y de desarrollo al proyectar sobre las versiones la integración entre funcionalidades. Así el equipo

de desarrollo estaría consiente sobre qué áreas críticas se está trabajando en el mantenimiento y se garantiza que mediante el flujo natural del proceso de desarrollo y mantenimiento se pruebe correctamente la integración. La Figura 3 que se muestra a continuación pretende ilustrar la solución al problema.

- En el caso de que el desarrollo de una funcionalidad nueva requiera invertir un período de tiempo superior al destinado al desarrollo de las versiones se procederá de la siguiente manera: cada desarrollador se mantiene actualizando la aplicación con las nuevas modificaciones planificadas y el código generado no se incorpora a la versión en desarrollo sin que exista la seguridad de que podrá completarse la funcionalidad y probarse coherentemente.

Como resumen se relacionan los documentos que deben incluirse en el repositorio después de realizado el análisis:

- Informes de Análisis de Viabilidad para las Solicitudes de Modificación recibidas.
- Artefactos correspondientes al análisis detallado.
- Actualización de los requisitos. Para este artefacto se propone usar la plantilla definida por la universidad para este propósito: 0113_Especificacion de Requisitos de Software (Universidad de las Ciencias Informáticas, 2010).
- Lista de modificaciones preliminares. Descritas en el documento de Análisis de Viabilidad propuesto.
- Estrategia de pruebas. Para este artefacto se propone usar la plantilla definida por la universidad para este propósito: Plan de Pruebas v2.0 (Universidad de las Ciencias Informáticas, 2009).
- Plan de implementación. Este artefacto es conocido como el Cronograma del Proyecto donde se refleja las actividades de implementación.
- Plan de versiones actualizado. Para este artefacto se propone usar la plantilla definida por la universidad para este propósito: Plan del producto 1.0 (Universidad de las Ciencias Informáticas, 2009).

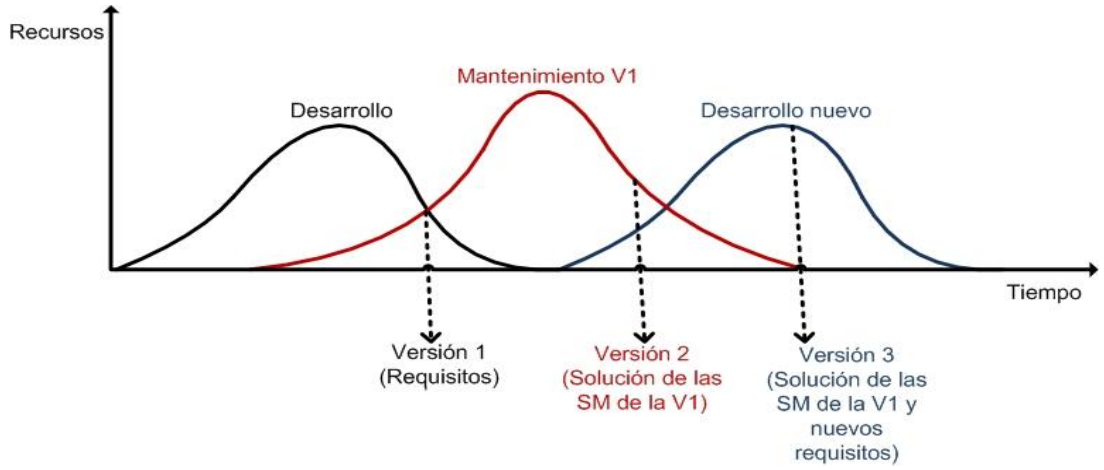

Figura 3. Representación del consumo de recursos y tiempo de los procesos de desarrollo y mantenimiento de software.

2.1.2.2 Diseño

A partir de toda la documentación existente del proyecto y del sistema que esté en producción, así como todo el código fuente y las bases de datos de la última versión, se va a realizar el diseño de las modificaciones del sistema en base a la documentación generada en el análisis antes realizado (Análisis de Viabilidad, Especificación de Requisitos de MS Esperados, identificación de los elementos afectados, Plan de Pruebas y Cronograma de implementación).

El primer paso del diseño será identificar los módulos o componentes que van a ser objeto de modificación, con el fin de hacer constar la planificación de tareas y ver la previsión de las mejoras a introducir. Según se avanza en los módulos se realizarán las modificaciones oportunas a la documentación de los mismos. En la documentación que se va generando se tendrán que identificar y documentar los cambios que se realicen sobre los requisitos afectados.

Para las modificaciones a realizar, se generarán casos de pruebas que incluyan los elementos de seguridad y robustez. Además es necesario identificar e incluir las pruebas de regresión necesarias.

El diseño no se dará por finalizado hasta no disponer o asegurar los siguientes elementos:

- Lista de modificaciones revisada y actualizada.
- Artefactos definidos por el proyecto para el diseño.
- Planes de pruebas actualizados.

- Análisis de Viabilidad actualizado.
- Requisitos verificados.
- Plan de implementación revisado.
- Lista de restricciones y riesgos bien documentados. Se utilizará el artefacto definido en la universidad con tal propósito: 0223_Problemas, Desviaciones y Acciones 2.0 (Universidad de las Ciencias Informáticas, 2010).

La implementación debería comenzar durante la etapa de diseño para comprobar la factibilidad de los cambios propuestos.

El jefe de MS es el responsable de velar porque el equipo de mantenimiento encargado actualice y complete la documentación definida en esta actividad de diseño.

El proceso de diseño puede variar de un proyecto a otro y dependerá de: las herramientas utilizadas, el tamaño de las modificaciones, el tamaño del sistema existente, la disponibilidad o no de un sistema desarrollado, y la accesibilidad a los usuarios y la organización que ha solicitado el mantenimiento.

2.1.2.3 Implementación

Para dirigir apropiadamente el esfuerzo durante la implementación será necesario disponer, como en las otras actividades, de la documentación actualizada del proyecto y del sistema, y los resultados de la actividad de diseño. Los elementos necesarios para el éxito de esta actividad serán:

- Documentación de diseño y requisitos aprobados y controlados.
- Estándar de codificación acordado para ser utilizado por el grupo de mantenimiento. Este estándar se establece por la dirección de arquitectura del proyecto.
- Métricas de diseño que podrían ser aplicables durante la implementación. Pueden ser empleadas las métricas identificadas para el desarrollo del proyecto o utilizar algunas de las recomendadas por autores como Pressman (Pressman, 2002).
- Una planificación de desarrollo detallada, incluyendo todas las revisiones de código que se harán y a qué nivel.
- Un conjunto de respuestas a los riesgos identificados en las etapas anteriores.

En esta actividad, se van a seguir determinados pasos que serán iterativos y gradualmente incrementales, es decir, se irán repitiendo y desarrollando en mayor detalle, hasta obtener el resultado previsto por la organización. Estos pasos son: codificación y pruebas de unidad, integración, análisis y revisión de los riesgos y la revisión de disponibilidad de pruebas.

Codificación y pruebas de unidad: Se implementan los cambios en el código y se realizan las pruebas de unidad, así como otros procesos de verificación, validación y aseguramiento de la calidad.

Integración: Tras la codificación y las pruebas de unidad, o incluso durante la codificación, el software modificado puede ser integrado con el sistema existente, pudiendo a su vez, realizar el refinamiento de las pruebas de integración y regresión. También, como objetivo, se pretenden valorar todos los efectos producidos tras la modificación del software existente, ya que cualquier impacto inaceptable debe ser conocido, para poder volver a la fase de codificación y pruebas y corregir su efecto.

Revisión y análisis de riesgos: Durante la implementación, la revisión y análisis de los riesgos será realizada periódicamente, siempre durante esta actividad general, preferiblemente al final, al igual que en las actividades de análisis y diseño. Es lo más recomendado ya que un alto porcentaje de los riesgos y problemas de diseño aparecen mientras se realiza la modificación del sistema.

Revisión de la disponibilidad de pruebas: Para valorar la disponibilidad de las pruebas del sistema, se mantendrá una revisión de acuerdo con los aspectos definidos en cada uno de los entornos de proyecto. Antes de comenzar la revisión, se tendrá que preparar y facilitar al personal del proyecto la siguiente información:

- Criterios iniciales para las pruebas del sistema.
- Recursos y tiempo necesario para su realización.
- Plan de pruebas detallado.
- Plantillas de documentación de las pruebas.
- Procedimientos para resolver anomalías.
- Procedimientos para la gestión de la configuración del software.
- Criterio para los resultados de las pruebas de sistema.

- Resultados de las pruebas a bajo nivel.

Cuando se esté en la actividad de implementación se ha de inspeccionar periódicamente el software que se va desarrollando con el fin de comprobar que el código se ajusta al estándar de codificación definido, y de esta manera aumentar la comprensibilidad del código.

Todas las pruebas de integración y de unidad deben estar documentadas. Además se debe comprobar que la documentación técnica y de usuario también han sido actualizada.

Por último, el resultado de la implementación, será un nuevo software que deberá estar bajo el control de la Gestión de Configuración del Software, al que se acompañará de toda la documentación actualizada de diseño, documentación de pruebas, documentación de usuario y material de aprendizaje, sin olvidarse de la declaración de riesgos e impacto para los usuarios y el informe de revisión de la disponibilidad de pruebas.

2.1.2.4 Pruebas del sistema

Las pruebas de sistema se realizan según las normas establecidas por la Universidad de las Ciencias Informáticas y el CEIGE. En un primer momento el Departamento de Calidad del CEIGE realiza las pruebas correspondientes y se corrigen por parte del equipo de MS las no conformidades detectadas. Después de la liberación del producto por parte del Departamento de Calidad del centro el sistema es probado por Calisoft. La interacción con estas entidades externas al equipo de MS se hace de la misma forma en que se realiza actualmente con el equipo de desarrollo, considerando que, el equipo de MS valora la solución a la no conformidad emitida y de ser necesario llevar a cabo una modificación en el sistema se realizará una iteración que incluye el tránsito por las fases previas.

Para realizar las pruebas de sistema se debe disponer de toda la documentación generada a tal efecto en las actividades anteriores: Informe de revisión de disponibilidad de pruebas, Plan de pruebas de sistema, Casos de pruebas de sistema, Procedimientos de prueba del sistema, Manuales de usuario y Diseño. Un aspecto que no se debe olvidar es la realización de pruebas de regresión.

Después de terminadas las pruebas todo el código fuente, las solicitudes de modificación y la documentación de pruebas, será puesta bajo la Gestión de Configuración del Software.

Para el mantenimiento de futuras versiones, es posible que se realicen otras pruebas para satisfacer requisitos en los que se necesite interactuar con otros sistemas o subsistemas. En

este caso será necesario controlar y validar que no se introduzcan nuevos fallos como resultado de los cambios.

Pruebas de aceptación

Al igual que las pruebas de sistema, las pruebas de aceptación serán realizadas sobre un sistema completamente integrado. Estas pruebas se realizan por el cliente, por el usuario o por un tercero designado por el cliente. Se llevan a cabo para asegurar que el resultado de las modificaciones es satisfactorio para el cliente, tanto del software como de la documentación generada. La universidad y el centro tienen experiencia en este sentido y al igual que en las pruebas anteriores estas se realizarán siguiendo lo establecido por la organización.

Tras las pruebas se establecerá el nuevo sistema de referencia y se situará la documentación generada durante las pruebas de aceptación bajo el control de Gestión de Configuración del Software. Se almacenará tanto el código como la totalidad de la documentación en directorios diferentes a los de desarrollo. El artefacto que se actualizará en correspondencia es el Plan del producto 1.0 (Yzquierdo, 2010). Es importante también que como parte de estas actividades de prueba se actualice toda la información que se le entrega al cliente, como es el caso del Manual de Usuario.

2.1.2.5 Mantenimiento no Planificable. Mantenimiento Correctivo Urgente

Las modificaciones que se identifiquen como de Mantenimiento Correctivo Urgente seguirán cuidadosamente cada una de las etapas descritas anteriormente, sin que la premura implique omitir o deformar la ejecución de alguna actividad. En la medida en la que se vaya trabajando de manera ordenada según orienta este procedimiento se podrán ir disminuyendo los tiempos de respuesta del equipo de mantenimiento.

Las modificaciones enmarcadas en este tipo de mantenimiento tendrán toda la prioridad en cuanto al uso de los recursos asociados al mantenimiento. Poco tiempo después de la implantación de una nueva versión del producto se producirá un aumento en la cantidad de solicitudes de modificación asociadas al Mantenimiento Correctivo Urgente.

2.1.2.6 Mantenimiento Planificable. Mantenimiento Correctivo no Urgente

Las modificaciones que se identifiquen como de Mantenimiento Correctivo no Urgente seguirán cuidadosamente cada una de las etapas descritas anteriormente. Este tipo de modificaciones tienen el segundo grado de prioridad, antecedidas solamente por las

modificaciones clasificadas como de Mantenimiento Correctivo Urgente. Se debe planificar coherentemente el momento en que se le darán solución a este tipo de problemas.

2.1.2.7 Mantenimiento Planificable. Mantenimiento Perfectivo

Los requisitos de un sistema software no tienen por qué ser siempre los mismos. Los requisitos pueden cambiar con el tiempo para ajustarse a nuevas necesidades o para mejorar las prestaciones actuales, por lo que pueden ir desde una pequeña modificación en un módulo (formato de impresión) hasta la adición de nuevos módulos.

En conclusión, podemos definirlo como: el conjunto de actividades para mejorar o añadir nuevas funcionalidades requeridas por el usuario. Algunos autores han dividido este tipo de mantenimiento en:

- Mantenimiento de Ampliación (Incorporar nuevas funcionalidades). Esta situación debe tratarse con cuidado en el caso de sistemas que continúan en desarrollo, dado que los nuevos requisitos pasarán a desarrollo y no requieren ser manejados por el MS.
- Mantenimiento de Eficiencia (Mejorar la eficiencia de ejecución). Mejorar la eficiencia de la solución de software puede garantizar no solo un mejor aprovechamiento de los recursos tecnológicos o un mejoramiento en los indicadores de calidad relacionados con mantenibilidad, reusabilidad, entre otros ya mencionados, sino que puede acarrear un aumento en la satisfacción del usuario final al elevarse la usabilidad del sistema.

Para enfrentar este tipo de mantenimiento durante el análisis deben realizarse un grupo de pruebas fundamentalmente de stress (GRUPO DE INVESTIGACIÓN ARQUISOFT, 2007) que permitan identificar las zonas críticas. También pueden revisarse minuciosamente la implementación de las funcionalidades más complejas y usadas. A partir de este análisis se diseñarán las mejoras a implementar. Durante el diseño se pueden identificar patrones que pudiesen hacer más eficiente la aplicación.

2.1.2.8 Mantenimiento Planificable. Mantenimiento Preventivo

Con este tipo de mantenimiento no se pretende alterar las especificaciones funcionales iniciales del sistema software. Consiste en modificar el software de forma que se mejoren propiedades tales como: la calidad, la mantenibilidad, entre otros atributos), pero sin cambiar las especificaciones iniciales.

También se seguirán en el caso del Mantenimiento Preventivo las actividades generales descritas pero prestando atención a las peculiaridades que se detallan a continuación.

Algunas de las acciones concretas que se recomiendan son las siguientes:

- Comprobación de la validez de los datos de entrada. Lograr una validación exhaustiva de los datos de entrada puede disminuir considerablemente futuros problemas. Las entradas de datos pueden estar asociadas a diferentes tipos de interfaces de entrada, por ejemplo pudiese ser útil validar un XML que contiene datos importados.
- Reestructuración del software para mejorar la legibilidad y su futuro mantenimiento.
- Adición de comentarios que puedan esclarecer las funcionalidades desarrolladas. Esto debe realizarse según lo normado por el equipo de arquitectura del proyecto a través de los estándares de codificación.
- El mantenimiento enfocado a la reutilización es realmente mantenimiento preventivo. Consiste en modificar el software buscando segmentos de código que puedan ser almacenados en bibliotecas para ser reutilizados más fácilmente. En este caso, la propiedad que se pretende mejorar es la propiedad de reusabilidad del software.

2.1.2.9 Mantenimiento Planificable. Mantenimiento Adaptativo

Cuando el software ya está en explotación, puede que se produzca algún cambio en el entorno en el que se ejecuta, este tipo de mantenimiento responde a esta situación. Estos cambios pueden deberse a cambio en el sistema operativo, a cambio del tipo de arquitectura en la que se ejecuta (red local a Internet/Intranet) o a cambios en el entorno de desarrollo del software (nuevos elementos y herramientas).

Los cambios pueden ir desde un pequeño retoque a nivel de módulo hasta la casi reescritura de todo el código. Se pueden realizar cambios en el entorno del software **a nivel de datos** (migración de un sistema de ficheros a un sistema basado en bases de datos relacionales) o **de procesos** (pasar de una tecnología a otra, como una plataforma de desarrollo con componentes distribuidos).

El mantenimiento adaptativo es cada vez más frecuente debido a la velocidad a la que está evolucionando el hardware, los nuevos sistemas operativos y las continuas actualizaciones de los existentes (actualizaciones de seguridad, Service Packs). Por término general, la vida de un sistema software suele ser superior a la frecuencia de estos cambios, por lo que debe adaptarse.

2.1.3 Actividades finales

La relación entre las tareas de esta etapa de Actividades finales se describe en la Figura 4.

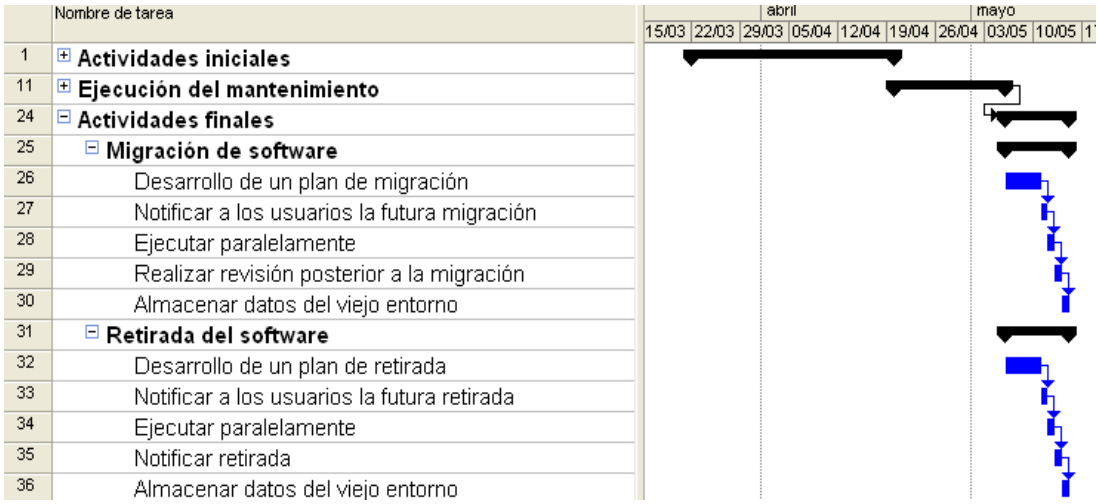

Figura 4. Relación entra las actividades de la etapa de Actividades finales.

2.1.3.1 Migración de software

2.1.3.1.1 Desarrollo de un plan de migración

Esta actividad debería desarrollarse por el equipo de soporte del proyecto y aunque puede estar apoyada por el equipo de mantenimiento no es responsabilidad de este. Aún así describen algunos principios que pudiesen facilitar el trabajo en los proyectos que no tienen un equipo de soporte claramente definido.

Como parte del Plan de Migración debe estar:

- Descripción de los requisitos corregidos o incluidos.
- Organización de la capacitación en caso de que sea necesario. Esto abarca la relación de todos los recursos imprescindibles para dar cursos y preparaciones. Además se hace referencia al Manual de Usuario actualizado.
- Versión del sistema a sustituir y versión a instalar.

- Procedimiento de instalación, configuración y carga inicial.
- Para reducir los riesgos asociados con la instalación de la nueva versión del sistema software, el jefe de soporte (con el apoyo del equipo de MS) debería planificarlo y documentar los procedimientos de instalación alternativos, que podrían asegurar el mínimo impacto sobre los usuarios del sistema debido a fallos del software imprevistos, no detectados durante las pruebas. El plan deberá incluir factores críticos en el tiempo, por ejemplo, las fechas disponibles para la instalación, así como los procedimientos de recuperación o vuelta atrás.
- Accesibilidad de copias de datos referentes al sistema sustituido.

Esta información es la salida de esta actividad y se detalla en el artefacto Plan de Migración de Software (Yzquierdo, 2010).

2.1.3.1.2 Notificar a los usuarios la futura migración

Se debe mantener al usuario actualizado con relación a las fechas en las que se ha planificado la futura migración del software, informando además cuales serán los problemas que se han solucionado o las nuevas funcionalidades incluidas.

2.1.3.1.3 Ejecutar paralelamente

Los pasos a seguir para instalar la nueva versión serán los siguientes:

- Desarrollar una versión de archivo del sistema para salvaguarda del mismo.
- Desinstalar la vieja versión del sistema.
- Realizar la instalación y formación para el cliente.
- Realizar actividades de carga inicial en caso de ser necesario.
- Se ha de completar la documentación de descripción de la versión, se propone usar el artefacto definido por la universidad para tal propósito: Plan del producto 1.0 (Universidad de las Ciencias Informáticas, 2009).
- Finalmente, hay que poner todo bajo el control de la Gestión de Configuración del Software.

2.1.3.1.4 Realizar revisión posterior a la migración

Se debe realizar una revisión del correcto funcionamiento del sistema en el entorno real, verificando la efectividad de la instalación y puesta en funcionamiento.

2.1.3.1.5 Almacenar datos del viejo entorno

Al estar bajo la GC se realizará una copia de seguridad de la versión del sistema existente, incluyendo los datos que se manejaban. Esto garantizaría poder recuperar el sistema en caso de presentarse alguna dificultad con la nueva versión.

2.1.3.2 Retirada del software

Saber cuándo se tiene que retirar un software es tan importante como saber cuándo adquirirlo. Es primordial saber cuándo hay que deshacerse de lo viejo y pasar a lo nuevo. El esfuerzo que se ahorra en mantenimiento se puede enfocar en nuevas aplicaciones. El software antiguo puede tener un mantenimiento muy caro, en términos de esfuerzo, y puede que no se integre adecuadamente en el entorno donde existen nuevas aplicaciones. Por lo mismo, el costo de actualización puede reducirse sensiblemente si eliminamos el software obsoleto y a este ahorro podemos agregar un mayor valor asociado generalmente al software de última generación, ya que introduce tecnologías y funcionalidades que aumentan la productividad notablemente.

2.1.3.2.1 Desarrollo de un plan de retirada

La fecha de retirada es el momento en que finalizan las actualizaciones, el soporte y las correcciones del software.

Deberá prepararse y documentarse un plan de retirada para el cese del soporte activo por parte de las organizaciones de operación y mantenimiento. Las actividades de planificación deberán incluir a los usuarios. El plan deberá considerar los elementos enumerados a continuación (Comité técnico AEN/CTN 71, 1995):

- Cese de soporte total o parcial después de un cierto tiempo.
- Archivo del producto software y su documentación asociada.
- Responsabilidad sobre cuestiones de soporte residual futuro.
- Transición al nuevo producto.
- Accesibilidad de copias de datos.

Esta información es la salida de esta actividad y se detalla en el artefacto Plan de Retirada de Software (Yzquierdo, 2010).

2.1.3.2.2 Notificar a los usuarios la futura retirada

Deberá notificarse a los usuarios los planes y actividades de la retirada. Las notificaciones deberán incluir lo siguiente:

- Descripción del sustitutivo o mejora, con su fecha de disponibilidad.
- Descripción de por qué el producto software no va a seguir siendo soportado.
- Descripción de otras opciones de soporte disponibles, una vez que el soporte ha cesado.

2.1.3.2.3 Ejecutar paralelamente

Puede o no valorarse sustituir el software existente por uno nuevo. En el caso positivo para facilitar la transición al nuevo sistema, conviene que se lleve a cabo la operación en paralelo del sistema a retirar y del nuevo producto software. Para esto pueden seguirse los pasos descritos en la actividad “Ejecutar paralelamente” de la etapa de Migración de software.

En caso de que solo se desee retirar el antiguo sistema se realizan las siguientes acciones:

- Desarrollar una versión de archivo del sistema para salvaguarda del mismo.
- Desinstalar la vieja versión del sistema.
- Finalmente, hay que poner todo bajo el control de la GC.

2.1.3.2.4 Notificar retirada

Se deberán mantener informados, a todos los afectados, del estado de cumplimiento del plan de retirada. Cuando llegue la fecha prevista de retirada, se deberá notificar a todos los involucrados.

2.1.3.2.5 Almacenar datos del viejo entorno

Toda la documentación de desarrollo asociada, registros y código deberán archivarse en el momento oportuno. Los datos usados o asociados al producto software retirado deberán ser accesibles de acuerdo con los requisitos del contrato sobre protección de datos y auditorías aplicables.

Conclusiones

En este capítulo se dio cumplimiento al segundo objetivo de la investigación relacionado con la definición de una propuesta de procedimiento para gestionar los procesos del mantenimiento en los sistemas de gestión en el CEIGE. El procedimiento fue descrito considerando las actividades que se debían realizar, los roles que debían involucrarse y los artefactos que se generarían en cada momento. Se debe resaltar que aún cuando el procedimiento se enfocó en la gestión del proceso de mantenimiento de software de gestión en el CEIGE, este constituye una guía que puede ser mejorada considerando las peculiaridades de cada nuevo entorno. Los elementos más significativos del procedimiento son:

- Organización del trabajo en etapas que cubren la planificación del MS, la ejecución del MS y la puesta en producción del producto.
- Descripción de criterios a considerar a la hora de decidir cuándo se debe asumir un nuevo desarrollo de software y cuándo actividades de MS.
- Descripción de elementos relevantes en la estimación de esfuerzo necesario para el MS.
- Descripción de un grupo de métricas que permiten analizar la evolución de un producto (estabilidad) entre otros aspectos.

CAPÍTULO 3. ANÁLISIS DE RESULTADOS

Introducción

En este capítulo se realiza un análisis de la aplicación del procedimiento en la muestra seleccionada. En el primer epígrafe se describe el proceso de la realización del experimento a partir de la descripción del diseño y aplicación del instrumento de captación de datos y la descripción del proceso de aplicación de procedimiento a la muestra seleccionada. En el segundo se realiza un estudio de los resultados obtenidos de la aplicación del experimento. Se realiza también un análisis del procedimiento a partir de la aplicación de otro instrumento. Por último se presentan las conclusiones del capítulo.

3.1 Realización del experimento

Se realizó un experimento en condiciones naturales donde se aplicó el procedimiento desarrollado para la gestión del proceso de mantenimiento de software en proyectos del CEIGE. Se tomaron los valores de los parámetros en el estado inicial de la muestra y una vez aplicados el procedimiento en el entorno del CEIGE con el objetivo de determinar si existían diferencias significativas en los resultados.

La población es bastante homogénea en cuanto a las características de los sistemas desarrollos (se consideran sistemas de gestión), sin embargo se tomó una muestra que cubre el 88% de la población que es de 9 sistemas actualmente en MS. La población estuvo determinada porque de los productos desarrollados en el CEIGE, solo 9 entran en el proceso de mantenimiento, y por tanto, se pueden formalizar las tareas correspondientes como un proyecto de MS. Las 9 aplicaciones son: Estructura y Composición, Contabilidad, Cobros y pagos, Banco, Caja, Costos y Procesos, Logística, Planificación y Configuración. El único que no se seleccionó como parte de la muestra fue el sistema de Planificación.

Se considera que en estas aplicaciones será posible probar la generalidad del procedimiento propuesto y se empleó para la selección de la muestra la técnica de Muestreo Intencionado. Las razones de la selección se exponen a continuación.

En consecuencia se valoraron los sistemas de gestión en etapa de mantenimiento desarrollados por el CEIGE, se consideran aspectos como el tamaño, complejidad y facilidad para obtención detallada de información. La complejidad de estos sistemas es grande o

mediana en todos los casos (en la Tabla 1 se muestra la cantidad de requisitos de cada uno), y a esto se le agrega la necesidad de integración entre los mismos.

Tabla 1. Proyectos desarrollados en CEIGE.

	Proyectos	**Total de requisitos**
1	Contabilidad general	44
2	Cobros y pagos	87
3	Costo y procesos	64
4	Estructura y composición	105
5	Configuración	120
6	Caja	77
7	Banco	55
8	Logística	240
9	Planificación	160

3.1.1 Diseño y aplicación del instrumento de captación de datos

Se aplicará el instrumento diseñado (Anexo 5. Instrumento para el diagnóstico) a la muestra antes seleccionada. El instrumento tiene como objetivo obtener un diagnóstico sobre el estado de determinados elementos en la ejecución de los proyectos.

Tabla 2. Caracterización del instrumento de captación de datos.

Tipo de pregunta	Descripción	Cantidad
A	Preguntas de afirmación o negación	8
B	Preguntas donde se evalúa, caracteriza o califica algún parámetro o elemento en valores	3
C	Preguntas donde se escogen elementos libres de los encuestados	1

Las preguntas de tipo A tienen por objetivo comprobar la existencia de procesos directamente relacionados con la gestión de contratación en los proyectos y que son reflejo de

la correcta ejecución de los mismos. También recogen otros indicadores positivos del proyecto. Algunos de estos son: el cumplimiento de los compromisos de tiempo y la satisfacción del cliente.

Las preguntas de tipo B tienen por objetivo evaluar el nivel en que un determinado parámetro se encuentra. Las preguntas de tipo B están cuantificadas en una escala de valores de cero a tres como máximo. Siendo el valor mayor siempre el que tenga mayor aporte en positivo para la ejecución del proyecto. La redacción y categorías establecidas en las preguntas se concibieron para lograr el máximo de precisión y evitar la subjetividad en las respuestas de los encuestados.

Las preguntas de tipo C tienen por objetivo identificar determinados factores con incidencia directa en el proyecto.

El instrumento se aplicará a tres personas por cada uno de los proyectos, directamente involucradas en la gestión de proyectos y en específico en la gestión del mantenimiento:

1. Responsable del mantenimiento en el proyecto.
2. Participante en el equipo de mantenimiento.
3. Un funcional que en el modelo de trabajo que implementa la UCI representa a las entidades clientes.

También se realizará una encuesta a los participantes en la aplicación del procedimiento y a otros conocedores del tema entre los que estarán líderes de proyectos y especialistas que han participado en la ejecución del desarrollo y mantenimiento de software. El objetivo es evaluar la calidad del procedimiento. La encuesta aplicada para la captura de los datos se encuentra en el (Anexo 6. Evaluación del procedimiento).

La evaluación del procedimiento se realizará a partir de los siguientes criterios:

- Calidad de los procesos.
- Capacidad de los procesos en el análisis de factores esenciales en la ejecución de proyectos de software.
- Evaluación de los instrumentos y artefactos.

La calidad de los procesos se valora a partir de los siguientes criterios: aplicabilidad, claridad y reusabilidad de los procesos.

La capacidad de los procesos en el análisis de factores esenciales en la ejecución de proyectos de software se evalúa a partir de la calificación en cuanto a los siguientes elementos:

- Adaptabilidad a los tipos de Producto.
- Integración al proceso de desarrollo de software de gestión.
- Adaptabilidad a diferentes escenarios según capacidad de los recursos humanos.
- Completitud.

La evaluación de los instrumentos y artefactos se realiza a partir de la calificación en cuanto a los siguientes elementos:

- Claridad y precisión.
- Completitud (Que se recoja toda la información necesaria).
- Adaptabilidad de los instrumentos o Generalidad (Que se pueda aplicar a diferentes escenarios).

En la encuesta se solicita la evaluación de cada criterio en tres niveles: bajo, medio y alto.

3.1.2 Aplicación de la metodología a la muestra seleccionada

El proceso de aplicación del procedimiento se realizó a través de tres actividades:

1. Preparación del personal.
2. Implantación.
3. Control del uso y ejecución metodológica.

La explicación de la ejecución de cada una de estas actividades se hace a continuación.

3.1.2.1 Preparación del personal

Esta actividad se realizó en dos momentos fundamentalmente. El primero fue cuando se le explicó a la dirección del centro el procedimiento propuesto, incluyéndose a los futuros jefes de los proyectos de MS. Esta fue una oportunidad en la que se discutió el procedimiento y la integración del mismo con los procesos definidos por el centro, haciendo en consecuencia los ajustes necesarios.

El segundo momento fue cuando se preparó a parte del equipo de mantenimiento para que estas personas pudieran posteriormente transmitir al resto del equipo el procedimiento

propuesto. Las personas presentes representaban los roles de jefe de MS, miembro del equipo de mantenimiento, arquitecto, analista, responsable de la GC, especialista de calidad.

Para llevar a cabo la preparación del personal se organizó los encuentros en forma de talleres, en un primer momento se expuso el procedimiento. Posteriormente se orientó el estudio del documento donde se describe el procedimiento y finalmente se hizo una discusión que consolidó los conocimientos adquiridos.

Con la dirección del centro se realizaron 4 encuentros:

- Primer encuentro: Se explicó el procedimiento propuesto, se esclarecieron las dudas y se orientó el estudio del documento que describe el procedimiento.
- Restantes encuentros: Se esclarecieron las dudas, se hizo un análisis de la integración del procedimiento con las restantes definiciones del centro, se propusieron y acordaron los ajustes necesarios en cuanto a la organización del trabajo y la administración de los recursos imprescindibles. El análisis se hizo de manera general y posteriormente se particularizó en cada proyecto de MS.

3.1.2.2 Implantación

Como primer paso en la implantación se definió claramente la necesidad de formalizar los proyectos de MS correspondientes a los sistemas seleccionados en la muestra, aplicándose en consecuencia las definiciones establecidas en la sección Definición de los requisitos del proyecto de MS, donde se dejan establecidas las condiciones necesarias para que se pueda decidir asumir un proyecto de MS. Para ello se contó con el conocimiento de la dirección del equipo que desarrollo de los sistemas seleccionados. La decisión estuvo apoyada por la proyección de las futuras versiones de los sistemas y por el hecho de que la mayor parte de las funcionalidades incluidas en las versiones proyectadas estarán dirigidas a:

- Corregir el funcionamiento de la versión actual.
- Mejorar el rendimiento del sistema.
- Adaptarse a mejoras arquitectónicas.

Siendo esta la primera forma de aplicación del procedimiento propuesto.

En un segundo momento se definió un cronograma tipo a partir de las actividades propuestas en el procedimiento, de manera que cada proyecto de MS pudiese guiarse y

ajustarlo a sus peculiaridades. Dicho cronograma se formalizó en el sistema de planificación utilizado por el centro (Redmine).

Se definió más tarde por parte de la dirección del equipo un estimado de los recursos necesarios y posterior asignación de los mismos, se continuó aplicando las actividades descritas en el procedimiento propuesto.

Se trabajó con el departamento de Calidad del centro en la elaboración de una guía de adaptación de este tipo de proyecto. Además se planificaron las auditorias y revisiones técnicas formales necesarias durante la ejecución del mantenimiento. Como consecuencia de este trabajo con el departamento de Calidad del centro se obtuvo lo siguiente:

1. La estructura del expediente para un proyecto de MS.
2. La lista de chequeo necesaria para hacer las auditorias y revisiones técnicas formales.
3. La descripción del procedimiento usando el estándar definido en la plantilla Plan del Proceso de Producción (se alinea a las definiciones del proceso de mejoras de la universidad).

Este resultado representa una internalización de la investigación.

3.1.2.3 Control del uso y ejecución metodológica

Las actividades de control y ejecución metodológica estuvieron orientadas a la elaboración de la documentación necesaria, actividades de chequeo y auditoría a los proyectos para verificar la realización de los procesos orientados en la metodología.

Para cada proyecto se estableció un sistema de chequeo con una frecuencia quincenal, mientras que se le daba seguimiento a través de los reportes obtenidos a partir del uso del Redmine como herramienta de planificación.

También se organizó con la subdirección de calidad del centro un grupo de auditorías que serían aplicadas a los proyectos de MS en diferentes momentos durante la ejecución de los mismos.

3.2 Análisis de los resultados

A continuación se realiza un análisis de los resultados obtenidos en el experimento realizado.

La Figura 5 muestra los resultados alcanzados para las preguntas de tipo A durante el diagnóstico inicial y una vez aplicado el procedimiento.

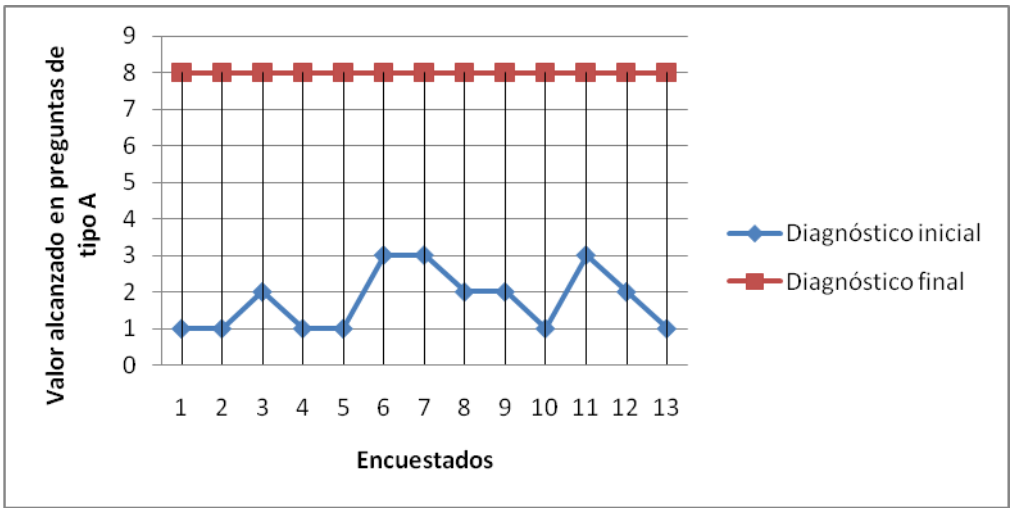

Figura 5. Valores alcanzados para las preguntas de tipo A.

La Figura 6 muestra los resultados alcanzados para las preguntas de tipo B durante el diagnóstico inicial y una vez aplicado el procedimiento.

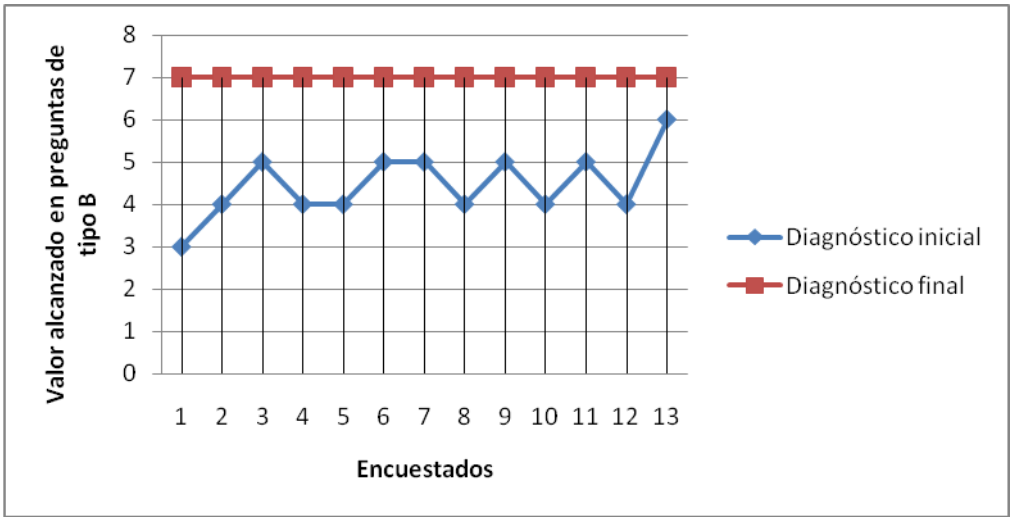

Figura 6. Valores alcanzados para las preguntas de tipo B.

En ambos gráficos se puede apreciar las mejoras posteriores a la aplicación del procedimiento para los tipos de preguntas A y B.

Para un análisis más profundo de comparación de estas muestras se utilizó el Mann-Whitney Test. Se emplea este test para la comparación de dos muestras independientes

porque no se tiene conocimiento sobre la normalidad de los valores de la muestra. En ambos casos para el nivel de significación aplicamos el método de Monte Carlo con intervalos de confianza del 99% consideramos significativa una significación menor de 0.05. En el procesamiento de los resultados de los experimentos se utilizó el SPSS versión 13.0. En el (Anexo 7. Detalles del Test de Mann-Whitney) se muestran los detalles de la aplicación del test de Mann-Whitney.

Como resultado de la aplicación del test se comprobó que existe una diferencia significativa en los valores de las preguntas de tipo A y B, siendo significativamente mejores los valores una vez aplicado el procedimiento propuesto. Se evidenció un dominio de las definiciones establecidas, una mejora en la organización del trabajo correspondiente al proceso de MS y por tanto una mejor gestión del tiempo y los compromisos. Todo esto reflejó una mejora en la mantenibilidad de los sistemas y una mayor satisfacción en los clientes.

Para la pregunta 4 del instrumento (Tipo C), dirigida a conocer el dominio de los elementos que se deben tener en cuenta para estimar el tiempo necesario para el mantenimiento y el esfuerzo correspondiente, se realiza una graficación que permite visualizar los resultados obtenidos durante el diagnóstico inicial y el diagnóstico realizado posterior a la aplicación del procedimiento. Esta representación ubica los elementos identificados en una gráfica de dos dimensiones donde el eje Y representa la dimensión de importancia, esta dimensión indica el número de personas que han enunciado el elemento. El Eje X representa la relevancia, que consiste en la cantidad de veces que el elemento ha sido enunciado entre las tres primeras posiciones de la lista.

La Figura 7 muestra los resultados de este análisis y en el (Anexo 8. Resultados para la pregunta 4 de tipo C para el diagnóstico inicial y final) se puede encontrar los detalles de este análisis.

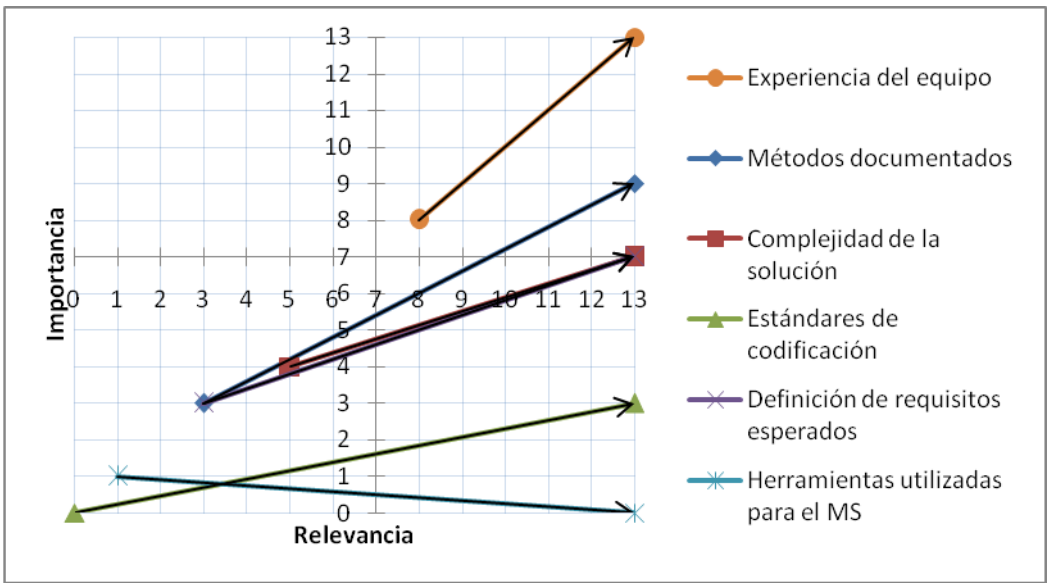

Figura 7. Resultados de la pregunta de tipo C durante el diagnóstico inicial y el final.

El análisis de la figura arroja lo siguiente:

- En el diagnóstico inicial se puede apreciar como existía muy poco dominio de los elementos que pueden estar influyendo en la estimación del esfuerzo y tiempo necesario para el MS, denotándose en el hecho de que elementos fueron mencionados pocas veces o ninguna, como es el caso del uso de Estándares de codificación.
- Después de aplicado el procedimiento se puede apreciar que los todos los elementos mencionados aparecen de una forma u otra entre las opciones de cada encuestado. Manifestando esto un mayor dominio de los elementos que pueden ser relevantes a la hora de hacer una estimación.
- Los resultados fueron satisfactorios para cada uno de los elementos analizados, considerando que la recta que intercepta los puntos $(x_0; y_0)$ y $(x_1; y_1)$ correspondientes al diagnóstico inicial y final respectivamente, tiene una pendiente positiva lo que denota una mejora significativa en los resultados alcanzados. Solo no se cumple esta afirmación para el elemento Herramientas utilizadas para el MS, sin embargo antes de la aplicación del procedimiento solo un encuestado señaló este elemento, mientras que después de la aplicación del procedimiento los 13 encuestados coincidían que era un elemento a considerar aunque no lo

contemplaban entre los tres más importantes. Por tanto, el resultado sigue siendo positivo.

- Los elementos más importantes se pueden apreciar en el cuadrante superior derecho de la gráfica. Estos son:
    - Experiencia del equipo.
    - Métodos documentados.
    - Complejidad de la solución.
    - Definición de requisitos esperados.

Coincidiendo en todo momento el resultado obtenido con lo propuesto en el procedimiento.

3.3 Análisis del procedimiento

Se realizó además una encuesta a los participantes en la aplicación del procedimiento y a otros conocedores del tema para un total de veinte personas. Entre los seleccionados están líderes de proyectos y especialistas que han participado en la ejecución de proyectos de desarrollo y mantenimiento de productos. La encuesta aplicada para la captura de los datos se encuentra en el (Anexo 6. Evaluación del procedimiento).

La evaluación del procedimiento se realizó a partir de los siguientes criterios:

- Calidad de los procesos.
- Capacidad de los procesos en el análisis de factores esenciales en la ejecución de proyectos de software.
- Evaluación de los instrumentos y artefactos.

La calidad de los procesos se valoró a partir de la evaluación de los siguientes criterios:

- Aplicabilidad de los procesos.
- Claridad de los procesos.
- Reusabilidad de los procesos.

La capacidad de los procesos en el análisis de factores esenciales en la ejecución de proyectos de software se evaluó a partir de la calificación en cuanto a los siguientes elementos:

- Integración al proceso de desarrollo de software de gestión.
- Adaptabilidad a los tipos de producto.
- Adaptabilidad a diferentes escenarios según capacidad de los recursos humanos.
- Completitud (alcance a todo el proceso de mantenimiento).

La evaluación de los instrumentos y artefactos se evaluó a partir de la calificación en cuanto a los siguientes elementos:

- Claridad y precisión.
- Completitud (Que se recoja toda la información necesaria).
- Adaptabilidad de los instrumentos o Generalidad (Que se pueda aplicar a diferentes escenarios).

En la encuesta aplicada se solicitó la evaluación de cada criterio en tres niveles: bajo, medio y alto. En la evaluación posterior del instrumento se asignaron valores de uno a tres. Los detalles de estos resultados pueden consultarse en el (Anexo 9. Resultados de la aplicación del instrumento de evaluación del procedimiento. El resumen de los resultados de la aplicación de este instrumento se encuentra en la siguiente tabla.

Tabla 3. Resultados de la aplicación del instrumento de evaluación del procedimiento.

Criterios generales de evaluación	**Elementos de evaluación**	**Puntuación**	**% Con respecto al Total**
Calidad de los procesos	Aplicabilidad	52	86.66%
	Claridad	56	93.33%
	Reusabilidad	56	93.33%
Capacidad de los procesos en el análisis de factores esenciales en la ejecución de proyectos de software	Adaptabilidad al tipo de Producto	51	85.00%
	Integración al proceso de desarrollo	58	96.66%
	Adaptabilidad a diferentes escenarios	49	81.66%
	Completitud	56	93.33%

Calidad de los artefactos e instrumentos definidos	Claridad y precisión	54	90.00%
	Completitud	54	90.00%
	Generalidad	51	85.00**%**

Todos los parámetros de evaluación del procedimiento obtuvieron resultados por encima del 80 %. Siendo el de más bajos resultados el indicador de *Adaptabilidad a diferentes escenarios Adaptabilidad a diferentes escenarios según capacidad de los recursos humanos*, esto implica que el procedimiento para ser correctamente adaptado a otros escenarios depende en gran medida de la preparación y disponibilidad de los recursos humanos implicados.

La Figura 8 muestra los valores que han sido tomados como indicadores de la Calidad de los procesos definidos en el procedimiento.

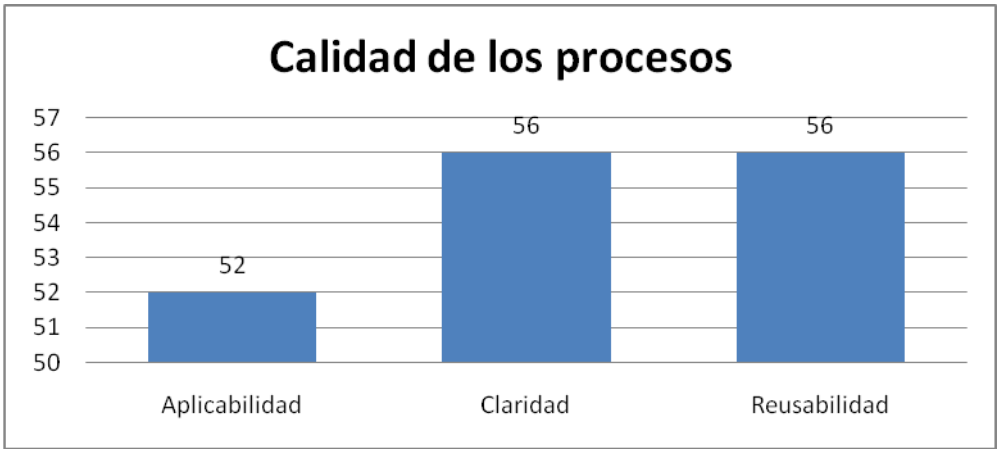

Figura 8. Valores asociados a la calidad de los procesos.

Un análisis descriptivo de estos indicadores (Tabla 4) arrojó que todos los valores de la media para estos indicadores son superiores a 2, que es el valor intermedio en el rango establecido. El valor más bajo con respecto a la media lo obtuvo el indicador *Aplicabilidad* debido a que dada la dimensión del proceso de mantenimiento puede ser complicado para el equipo de MS dominar todo el procedimiento en un plazo relativamente corto.

La desviación estándar oscila entre 0.410 y 0.503, lo que indica un índice medio o bajo de dispersión en los criterios de los entrevistados. Por tanto, se considera que existió consenso entre los encuestados.

Tabla 4. Análisis descriptivo de los indicadores de Calidad de los procesos.

	N	Mínimo	Máximo	Suma	Media	Desviación estándar	Varianza
Aplicabilidad	20	2	3	52	2.6	.503	.253
Claridad	20	2	3	56	2.8	.410	.168
Reusabilidad	20	2	3	56	2.8	.410	.168

En la Figura 9 se puede apreciar una vista gráfica de los valores de los indicadores que describen la Capacidad de los procesos en el análisis de los factores esenciales en la ejecución de los proyectos de software.

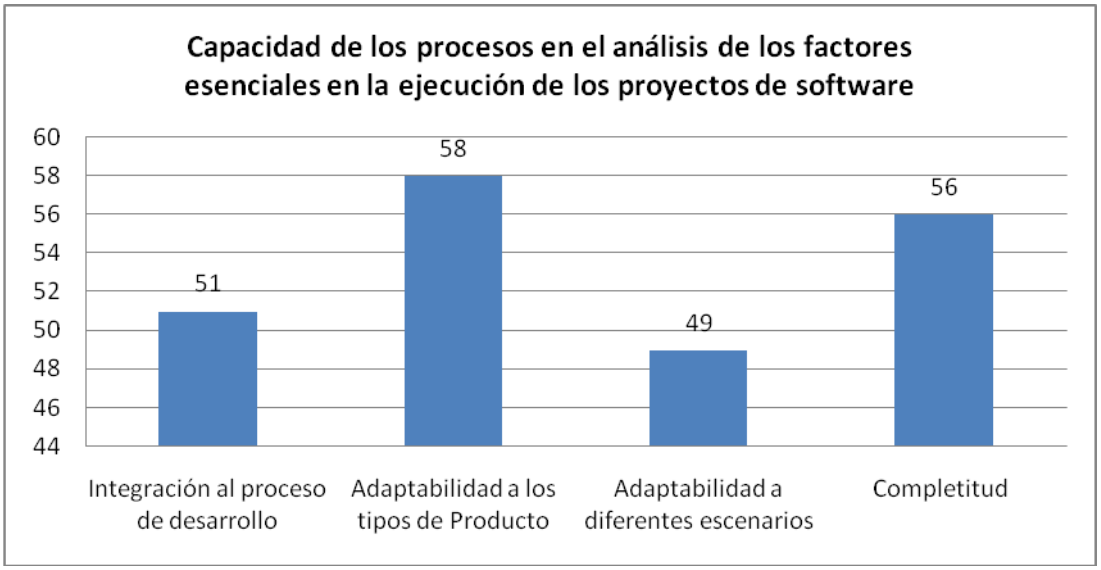

Figura 9. Valores de los indicadores que describen la Capacidad de los procesos en el análisis de los factores esenciales en la ejecución de los proyectos de software.

Un análisis descriptivo de estos indicadores (Tabla 5) arrojó que todos los valores de la media para estos indicadores son superiores a 2, que es el valor intermedio en el rango establecido. El valor más bajo con respecto a la media lo obtuvo el indicador *Adaptabilidad a diferentes escenarios Adaptabilidad a diferentes escenarios según capacidad de los recursos humanos*, este resultado se explicó anteriormente.

La desviación estándar oscila entre 0.308 y 0.510, lo que indica un índice medio o bajo de dispersión en los criterios de los entrevistados. Por tanto se considera que existió consenso entre los encuestados.

Tabla 5. Análisis descriptivo de los indicadores de Capacidad de los procesos en el análisis de los factores esenciales en la ejecución de los proyectos de software.

	N	Mínimo	Máximo	Suma	Media	Desviación estándar	Varianza
Integración al proceso de desarrollo	20	2	3	51	2.6	.510	.261
Adaptabilidad a los tipos de Producto	20	2	3	58	2.9	.308	.095
Adaptabilidad a diferentes escenarios	20	2	3	49	2.5	.510	.261
Completitud	20	2	3	56	2.8	.410	.168

El mejor resultado lo obtuvo el indicador *Adaptabilidad a los tipos de Producto* apuntando esto a que el procedimiento pudiese garantizar el mantenimiento de diferentes tipos de productos dado que las actividades y artefactos propuestos son lo suficientemente genéricos en este sentido.

En la Figura 10 se puede apreciar una vista gráfica de los valores de los indicadores que describen la calidad de los instrumentos definidos.

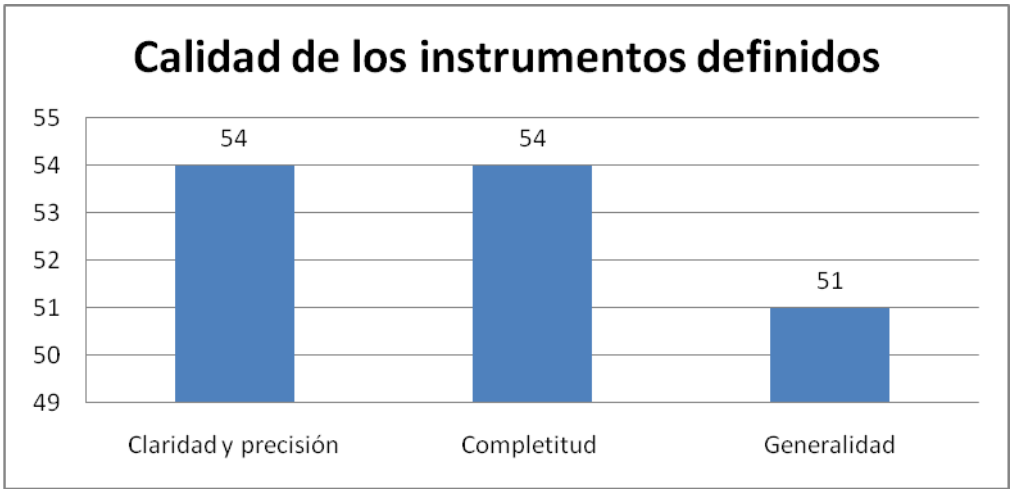

Figura 10. Valores de los indicadores que describen la calidad de los instrumentos definidos.

Un análisis descriptivo de estos indicadores (Tabla 6) arrojó que todos los valores de la media para estos indicadores son superiores a 2, que es el valor intermedio en el rango establecido. El valor más bajo con respecto a la media lo obtuvo el indicador *Generalidad* considerándose que los artefactos generados son específicos para el proceso de MS.

La desviación estándar oscila entre 0.470 y 0.510, lo que indica un índice medio o bajo de dispersión en los criterios de los entrevistados. Por tanto, se considera que existió consenso entre los encuestados.

Tabla 6. Análisis descriptivo de los indicadores de Calidad de los instrumentos definidos.

	N	**Mínimo**	**Máximo**	**Suma**	**Media**	**Desviación estándar**	**Varianza**
Claridad y precisión	20	2	3	54	2.7	.470	.221
Completitud	20	2	3	54	2,7	.470	.221
Generalidad	20	2	3	51	2,6	.510	.261

De manera general se puede señalar que la evaluación del procedimiento fue satisfactoria. El mejor resultado de manera global lo obtuvieron los aspectos relacionados con la Calidad de los procesos definidos. Sin embargo se considera que aspectos como los relacionados con la Gestión de la Configuración deben seguirse profundizando para garantizar correctamente la integración entre el resultado del desarrollo y el resultado del MS. También puede ampliarse el procedimiento en la gestión de los costos, comenzando por la realización de análisis de viabilidad económica. Finalmente se considera que pueden definirse otros indicadores que permitan medir con mayor rigurosidad el proceso MS en todo su ciclo de vida.

Conclusiones

En este capítulo se dio cumplimiento al tercer objetivo trazado para la presente investigación. La evaluación se efectuó a partir de la aplicación del procedimiento propuesto en la muestra seleccionada. A lo largo del capítulo se describió el proceso para la realización del experimento.

Se presentó el método aplicado para la implementación de las definiciones propuestas, se expuso el diseño de los instrumentos de recogida de información y el mecanismo de aplicación de los mismos. Se realizó un análisis de los resultados del experimento desarrollado, a partir de

la comparación de los valores obtenidos entre el estado de los proyectos antes de la aplicación del procedimiento y una vez aplicado utilizando el Mann-Whitney Test.

Como resultado de la aplicación del test se comprobó que existe una diferencia significativa en los valores de las preguntas de tipo A y B, siendo significativamente mejores los valores una vez aplicado el procedimiento.

Los resultados positivos estuvieron denotados por la mejora experimentada después de la aplicación del procedimiento en áreas como:

- La estimación de tiempo y esfuerzo.
- La participación de todas las partes involucradas en el mantenimiento.
- El correcto manejo de los artefactos y la actualización oportuna de los mismos.
- La clasificación y el tratamiento diferenciado de los diferentes tipos de mantenimientos.
- La adecuada utilización durante el MS de técnicas como la ingeniería inversa, reingeniería y restructuración de software.

Finalmente se realizó un análisis de los resultados obtenidos a partir de una encuesta aplicada a un grupo de personas involucradas en la aplicación del procedimiento y otras que cuentan con experiencia en la dirección de proyectos y en el mantenimiento de productos. La encuesta estuvo dirigida a la evaluación de determinados parámetros de calidad del procedimiento y de los artefactos generados, arrojando la misma resultados positivos. En este sentido los mejores resultados evaluativos se concentran en el criterio de la Calidad de los procesos definidos.

Aunque estos resultados son alentadores en general se señalan un grupo de aspectos que se deben mejorar tales como:

- Definiciones relacionadas con la Gestión de la Configuración.
- Definiciones relacionadas con la gestión de los costos.
- Definiciones de otros indicadores de calidad para el control del proceso.

CONCLUSIONES

En un primer momento se realizó un análisis del estado del arte, considerándose los principales estándares y técnicas internacionalmente reconocidos para el mantenimiento de software. A partir de este análisis se arribó a las siguientes conclusiones:

- El mantenimiento no comienza al recibir una solicitud de modificación sino que debe comenzar desde la etapa de desarrollo del producto garantizándose aspectos que influyen en la posterior mantenibilidad, además debe realizarse una correcta planificación de la etapa de MS.
- A partir de las metodologías, estándares y técnicas estudiados se puede detectaron un grupo de elementos que identifican más allá de los problemas que acarrea el MS de manera general, las peculiaridades del MS para los sistemas de gestión desarrollados en la UCI.
- Todos las metodologías o estándares estudiados fueron útiles en la definición de una guía o procedimiento para la gestión del MS en el CEIGE. Sin embargo ninguna de las definiciones estudiadas por si sola podía cubrir las necesidades del centro, por lo que se resaltaron los aspectos en los que determinadas metodologías o estándares fueron más útiles y convenientes.

Dándole cumplimiento al segundo objetivo trazado para investigación se desarrolló un procedimiento para gestionar el proceso de MS en el CEIGE. A manera de resumen se puede señalar que los elementos más significativos del procedimiento son:

- Organización del trabajo en etapas que cubren la planificación del MS, la ejecución del MS y la puesta en producción del producto. Definiéndose actividades, roles y artefactos necesarios en cada momento.
- Descripción de criterios a considerar a la hora de decidir cuándo se debe asumir un nuevo desarrollo de software y cuándo actividades de MS.
- Descripción de elementos relevantes en la estimación de esfuerzo necesario para el MS.
- Descripción de un grupo de métricas que permiten analizar la evolución de un producto (estabilidad) entre otros aspectos.

Finalmente para darle cumplimiento al tercer y último objetivo se realizó una evaluación del procedimiento a partir de la aplicación del mismo en una muestra seleccionada. También se realizó una evaluación del procedimiento por parte del equipo que participó en la aplicación y por un grupo de especialistas relacionados con la temática de MS.

Como parte de la aplicación del procedimiento se trabajó en conjunto con el departamento de Calidad del centro en función internalizar los resultados de la investigación al alinear el procedimiento con las definiciones del proceso de mejoras que lleva a cabo la universidad.

A partir de la evaluación de las dos muestras recogidas durante la aplicación del procedimiento se concluyó que los resultados de la aplicación de la propuesta fueron positivos. Esto estuvo determinado por la mejora experimentada después de la aplicación del procedimiento en áreas como: la estimación de tiempo y esfuerzo, la participación de todas las partes involucradas en el mantenimiento, el correcto manejo de los artefactos y la actualización oportuna de los mismos, la clasificación y el tratamiento diferenciado de los diferentes tipos de mantenimientos; y la adecuada utilización durante el MS de técnicas como la ingeniería inversa, reingeniería y restructuración de software.

El análisis de los resultados obtenidos a partir de la encuesta aplicada denotó aceptación y efectividad en el procedimiento. La encuesta estuvo dirigida a la evaluación de determinados parámetros de calidad del procedimiento y de los artefactos generados. En este sentido los mejores resultados evaluativos se concentraron en el criterio de la Calidad de los procesos definidos. Además se señalaron un grupo de aspectos que se deben mejorar progresivamente en la ejecución del procedimiento y en posteriores investigaciones.

RECOMENDACIONES

Se debe continuar investigando sobre las definiciones necesarias para la correcta Gestión de la Configuración en proyectos de MS. Profundizándose especialmente en la integración entre el resultado del desarrollo y el resultado del MS en una misma versión del producto.

Ampliar el procedimiento propuesto considerando la gestión de los costos y en consecuencia los análisis de viabilidad económica para los proyectos de MS.

Definir otros indicadores que permitan medir con mayor rigurosidad el proceso MS en todo su ciclo de vida.

BIBLIOGRAFÍA

1. **Abelson, H. y Greenspun, P. 2001.** Teaching Software Engineering. [En línea] 2001. http://philip.greenspun.com.

2. **Abran, Alain, y otros. 2001.** Guide to the Software Engineering Body of Knowledge - SWEBOK. s.l. : IEEE Press, 2001.

3. **Aggarwal, K.K. 2007.** *Software engineering.* s.l. : New Age International, 2007. 81-224-1638-1.

4. **B. B. Agarwal, S. P. Tayal,M. Gupta. 2009.** *Software Engineering & Testing: An Introduction.* s.l. : Jones and Bartlett Publishers, 2009. 978-1-934015-55-1.

5. **Bernd Bruegge, Allen H. Dutoit. 2009.** *Object Oriented Software Engineering Using UML, Patterns, and Java.* s.l. : Pearson Education, 2009. 0-13-606125-7.

6. **CanforaHarman, Gerardo y Penta, Massimiliano Di. 2007.** New Frontiers of Reverse Engineering. s.l. : IEEE Computer Society, 2007.

7. **case-tools. 2009.** case-tools. [En línea] 2009. http://case-tools.org/uml.html.

8. **Centro de Gestión Avanzado. 2009.** Nueva Herramienta para la Gestión de Incidencias y Peticiones de Software. 2009.

9. **Chikofsky, E.J. y Cross, J.H. 1990.** Reverse Engineering and Design Recovery: A Taxonomy. *Vol. 7, No. 1, pp. 13-17.* s.l. : IEEE Software, 1990.

10. **Comité técnico AEN/CTN 71. 1995.** *Procesos del ciclo de vida del software (ISO/IEC 12207:1995).* 1995.

11. **Dhillon, Balbir S. 2006.** *Maintainability, maintenance, and reliability for engineers.* s.l. : CRC Press, 2006. 0-8493-7243-7.

12. **Ebert, J., y otros. 2005.** Software Reegineering. *Vol. 19, No. 3, pp. 125-126.* s.l. : Informatik Forschung und Entwicklung, 2005.

13. **Fábrega, D. S. 2010.** Propuesta de procedimiento para el mantenimiento de software. Ciudad Habana. 2010.

14. **Feathers, Michael. 2002.** *Working Effectively With Legacy Code.* 2002.

15. **Gobierno de España. 2008.** MÉTRICA VERSIÓN 3. Metodología de Planificación, Desarrollo y Mantenimiento de sistemas de información. [En línea] 2008. [Citado el: 15 de 5 de 2009.]

16. **—. 2008.** MÉTRICA VERSIÓN 3. Metodología de Planificación, Desarrollo y Mantenimiento de sistemas de información. 2008.

17. **Grupo ARQUISOFT. 2007.** Herramientas de carga y rendimiento. [En línea] 2007. [Citado el: 19 de Abril de 2010.] http://www.udistrital.edu.co/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/node1.html.

18. **GRUPO DE INVESTIGACIÓN ARQUISOFT. 2007.** INVESTIGACIÓN SOBRE ESTADO DEL ARTE EN DISEÑO Y APLICACIÓN DE PRUEBAS DE SOFTWARE. [En línea] 2007. http://www.udistrital.edu.co/comunidad/grupos/arquisoft/fileadmin/Estudiantes/Pruebas/HTML%20-%20Pruebas%20de%20software/node1.html.

19. *IEEE 1219-1998 Standard for Software Maintenance.* **Institute of Electrical and Electronics Engineers. 1998.** 1998. ISBN: 0738103365.

20. **Jeffrey A. Hoffer, Joey F. George, Joseph S. Valacich. 2008.** Modern Systems Analysis and Design. 2008.

21. **Jones, Capers. 2009.** El control de la calidad es la clave del éxito en los proyectos de software. *Lider de proyecto.com.* 2009.

22. **—. 2009.** *Software Engineering Best Practices: Lessons from Successful Projects in the Top Companies.* s.l. : McGraw-Hill, 2009. 978-0-07-162161-8.

23. **Kajko-Mattsson, M. 2006.** Applicability of IEEE 1219 within Correctiev Maintenance. s.l. : Software Engineering Advances, International Conference, 2006.

24. **Lehman, Meir M y Ramil, Juan F. 2003.** Software evolution: background, theory, practice. 2003.

25. **León, Rolando Alfredo Hernández. 2002.** *EL PARADIGMA CUANTITATIVO DE LA INVESTIGACIÓN CIENTIFICA.* Ciudad de la Habana : EDUNIV, 2002.

26. *Mantenimiento del Software.* **Francisco Ruiz, Macario Polo. 2001.** ESCUELA SUPERIOR DE INFORMÁTICA UNIVERSIDAD DE CASTILLA-LA MANCHA : s.n., 2001.

27. **McBreen, Pete. 2009.** Calidad desde el Diseño, diseñar para el mantenimiento mejorando el código heredado. [En línea] Diciembre de 2009. InformIT.com.

28. **Object Mentor. 2009.** JUnit. [En línea] 2009. http://www.junit.org/.

29. *Offering Software Maintenance as an Offshore Service.* **Sneed, Harry M. 2008.** Beijing : s.n., 2008. 24th International Conference on Software Maintenance.

30. **Párraga, Juan Angel Martínez. 1999.** *ESTÁNDAR IEEE 1219 DE MANTENIMIENTO DEL SOFTWARE.* 1999.

31. **—. 1999.** *ESTÁNDAR IEEE 1219 DE MANTENIMIENTO DEL SOFTWARE.* 1999.

32. **Polo, Macario, Piattini, Mario y Ruiz, Francisco. 2001.** Using code metrics to predict maintenance of legacy programs: a case study. Escuela Superior de Informática University of Castilla-La Mancha : s.n., 2001.

33. **Pressman, Roger S. 2002.** Ingeniería de Software un enfoque práctico. s.l. : Quinta, 2002.

34. **Ramirez, Reyes Juaréz, Licea, Guillermo y Salas, Alfredo Cristobal. 2008.** Ingeniería Inversa y Reingeniería Aplicadas a Proyectos de Software Desarrollados. Universidad Autónoma de Baja California, Mexico : s.n., 2008.

35. **Ruiz, Francisco. 1999.** *La norma ISO 14764.* 1999.

36. **Ruiz, Francisco y Polo, Macario. 2005.** *Mantenimiento Avanzado de Sistemas de Información.* Ciudad Real : s.n., 2005.

37. **Sicilia, Miguel Angel. 2008.** Técnicas de Mantenimiento de Software. [En línea] 2008. http://cnx.org/content/col10571/1.6/.

38. **—. 2009.** Técnicas de Mantenimiento de Software. [En línea] 2009. http://cnx.org/content/col10571/1.6.

39. **Sneed, H. M. 2008.** Offering Software Maintenance as an Offshore Service. Beijing : 24th International Conference on Software Maintenance, 2008.

40. **Software Informer. 2010.** Comparison UML Case Tools. [En línea] 2010. http://comparison.software.informer.com/download-comparison-uml-case-tools/.

41. *SOFTWARE QUALITY IN 2008: A SURVEY OF THE STATE OF THE ART.* **Jones, Capers. 2008.** 2008.

42. **Susan J Chinn, Scott J Lloyd, Eric Kyper. 2005.** Contemporary Usage of CASE Tools in U. S. Colleges and Universities. s.l. : Journal of Information Systems Education, 2005, Vol. 16.

43. **Universidad de las Ciencias Informáticas. 2010.** 0113_Especificacion de Requisitos de Software. 2010.

44. **—. 2010.** 0223_Problemas, Desviaciones y Acciones 2.0. 2010.

45. **—. 2009.** Plan de Pruebas v2.0. 2009.

46. **—. 2009.** Plan del producto 1.0. 2009.

47. **Verónica Farach. 2005.** HERRAMIENTA DE GESTIÓN DE CAMBIOS E INCIDENCIAS. 2005.

48. **2008.** Wikipedia. [En línea] 15 de 5 de 2008. http://es.wikipedia.org/wiki/Sistemas_de_informaci%C3%B3n_gerencial.

49. **Yzquierdo, Raykenler. 2010.** Memorias de artefactos para el proceso de mantenimiento de software. 2010.

ANEXOS

Anexo 1. Actividades y artefactos

Tabla 7. Relación de actividades y artefactos propuestos.

Actividades	**Artefactos generados**
Actividades iniciales	
Familiarización con el software al que se le dará mantenimiento	
Determinación de requisitos del proyecto	Especificación de Requisitos de MS Esperados
Determinación del esfuerzo de mantenimiento de software	Plan de Mantenimiento de Software (versión preliminar)
Actividades de Gestión de la configuración	Artefactos definidos por el proyecto en este sentido
Elaboración del Plan de mantenimiento de software	Plan de Mantenimiento de Software
Preparar entornos de producción para probar el mantenimiento de software	Plan de Mantenimiento de Software(actualizado)
Preparar el nuevo personal encargado del mantenimiento de software	Cronograma del proyecto(actualizado), Descripción de los cursos a impartir
Recepción de las peticiones de modificación	Descripción del incidente (actualizada)
Decidir el tipo de mantenimiento de cada petición de modificación	Especificación de Requisitos de MS Esperados, Análisis de Viabilidad, Solicitud de modificación(actualizada), cronograma del proyecto (actualizado)
Ejecución del mantenimiento	
Análisis de peticiones de modificación	
Análisis de viabilidad	Análisis de Viabilidad
Análisis detallado	Análisis de Viabilidad, Especificación de Requisitos de MS Esperados(actualizado), Plan de Pruebas v2.0, cronograma (actualizado), Plan del producto 1.0(actualizado)

Diseño	Artefactos definidos para la etapa de análisis durante el desarrollo, Especificación de Requisitos de MS Esperados(actualizado), Análisis de Viabilidad(actualizado), cronograma del proyecto(actualizado), 0223_Problemas, Desviaciones y Acciones 2.0, Plan de Pruebas v2.0(actualizado)
Implementación	
Codificación y pruebas de unidad	Sistema (actualizado), Plan de Pruebas v2.0 (actualizado)
Revisión y análisis de riesgos	Sistema (actualizado), 0223_Problemas, Desviaciones y Acciones 2.0, Plan de Pruebas v2.0(actualizado)
Integración	Sistema (actualizado), Plan de Pruebas v2.0 (actualizado)
Revisión de la disponibilidad de pruebas	Plan de Pruebas v2.0 (actualizado)
Pruebas	
Pruebas del sistema	Sistema (actualizado), Solicitudes de Modificación, Plan de Pruebas v2.0 (actualizado), Plan del producto 1.0 (actualizado), Manual de Usuario(actualizado)
Pruebas de aceptación	Sistema (actualizado), Solicitudes de Modificación, Plan de Pruebas v2.0 (actualizado), Plan del producto 1.0 (actualizado), Manual de Usuario(actualizado)
Actividades finales	
Migración de software	
Desarrollo de un plan de migración	Plan de Migración, Cronograma del proyecto(actualizado)
Notificar a los usuarios la futura migración	
Ejecutar paralela	Plan del producto 1.0(actualizado)
Realizar revisión posterior a la migración	
Almacenar datos del viejo entorno	

Retirada del software	
Desarrollo de un plan de retirada	Plan de Retirada de Software
Notificar a los usuarios la futura retirada	
Ejecutar paralela	
Notificar retirada	
Almacenar datos del viejo entorno	

Anexo 2. Actividades y responsables

Tabla 8. Relación de actividades y roles.

Actividades	**Responsables**
Actividades iniciales	
Familiarización con el software al que se le dará mantenimiento	Jefe de Mantenimiento
Determinación de requisitos del proyecto	Jefe de MS, arquitecto y analistas del proyecto
Determinación del esfuerzo de mantenimiento de software	Jefe de Mantenimiento
Actividades de Gestión de la configuración	Responsable del área de Gestión de la Configuración de Software en el proyecto y el Jefe de MS
Elaboración del Plan de mantenimiento de software	Jefe de MS
Preparar entornos de producción para probar el mantenimiento de software	Miembro del equipo de MS
Preparar el nuevo personal encargado del mantenimiento de software	Miembro del equipo de MS
Recepción de las peticiones de modificación	Miembro del equipo de MS
Decidir el tipo de mantenimiento de cada petición de modificación	Arquitecto, analista, funcionales y Jefe de MS
Ejecución del mantenimiento	
Análisis de peticiones de modificación	
Análisis de viabilidad	Arquitecto, miembro del equipo de MS
Análisis detallado	Arquitecto, miembro del equipo de MS
Diseño	Arquitecto, miembro del equipo de MS
Implementación	Miembro del equipo de MS
Codificación y pruebas de unidad	Miembro del equipo de MS

Revisión y análisis de riesgos	Miembro del equipo de MS
Integración	Miembro del equipo de MS
Revisión de la disponibilidad de pruebas	Miembro del equipo de MS
Pruebas	
Pruebas del sistema	Miembro del equipo de MS, especialista funcional
Pruebas de aceptación	Miembro del equipo de MS
Actividades finales	
Migración de software	
Desarrollo de un plan de migración	Jefe de MS
Notificar a los usuarios la futura migración	Miembro del equipo de MS
Ejecutar paralela	Miembro del equipo de MS
Realizar revisión posterior a la migración	Miembro del equipo de MS
Almacenar datos del viejo entorno	Miembro del equipo de MS, Responsable del área de Gestión de la Configuración de Software en el proyecto
Retirada del software	
Desarrollo de un plan de retirada	Jefe de MS
Notificar a los usuarios la futura retirada	Miembro del equipo de MS
Ejecutar paralela	Miembro del equipo de MS
Notificar retirada	Miembro del equipo de MS
Almacenar datos del viejo entorno	Miembro del equipo de MS, Responsable del área de Gestión de la Configuración de Software en el proyecto

Anexo 3. Herramientas de apoyo al MS

Tabla 9. Herramientas de apoyo al MS

Herramienta	**Ejemplos**
Herramientas de gestión de las incidencias	HelpDesk, GMF, Mantis, NetSupport DNA Helpdesk, Bugzilla y SIGILA.
Herramientas CASE	DESIGNER/2000, EASY CASE, Rational ROSE / RequisitePro, EXCELERATOR y Visual Paradigm.
Herramientas de perfeccionamiento del código	Para este propósito se pueden emplear la mayoría de los IDE que se han utilizado para el desarrollo, algunos de estos pueden ser Zend Studio, NetBeans, NotePad ++, entre

	otros.
Herramientas para la ingeniería inversa	Visual Paradigm, Visual Studio, Eclipse, NetBeans y phpDocumentor.
Herramientas de gestión de la configuración	CVS, Subversion, SourceSafe, ClearCase, Darcs, Bazaar, Plastic SCM, Git, Borland Starteam y TFS.
Herramientas de prueba	ObjectGen, TestTools del VisualStudio de Microsoft, Rational Robot, TestGen4J, JUnit, Rational Performance Tester (Pruebas de carga y rendimiento) y e–Load (Pruebas de carga y rendimiento).

Anexo 4. Descripción y rango de valores de las medidas propuestas

Tabla 9. Descripción de las medidas propuestas.

Medidas	Descripción	Valor
Auto documentación	El grado en que el código fuente proporciona documentación significativa.	0 - 4
Simplicidad	El grado de facilidad con que se puede entender un programa. - Número de sentencias. - Número ciclomático V(G) - Frecuencia de operandos - Longitud de programa	0 - 4
Modularidad	La independencia funcional de componentes de programa.	ModFI/TMod $\rightarrow$1 ModFI: Cantidad de módulos funcionalmente independientes TMod: Total de módulos

Analizabilidad	Capacidad del producto software de diagnosticar deficiencias en las partes identificadas para ser modificadas. - Frecuencia de las excepciones.	0 - 4
Cambiabilidad	Capacidad del modelo para permitir que una modificación sea realizada. - Efectos causados en otras partes del programa.	0 - 4
Independencia del sistema de software	El grado de independencia de programa respecto a las características del lenguaje de programación no estándar, características del sistema operativo y otras restricciones del entorno. - Multiplataforma. - Aplicable en varios entornos. - Capacidad para conectarse con otros sistemas que estén en otros lenguajes.	0 - 4
Capacidad de expansión	El grado con que se pueden ampliar el diseño arquitectónico, de datos o procedimental. - Cambios a realizar debido a la ampliación.	0 - 4
Estabilidad	La capacidad del software para minimizar los efectos inesperados de las modificaciones del software. El índice de madurez del software (IMS) que proporciona una indicación de la estabilidad de un producto software basada en los cambios que ocurren con cada versión del producto.	IMS= [MT - (Fc + Fa + Fe)]/MT MT: Número de módulos en la versión λ Fc: Número de módulos en la versión actual λ que se han cambiado Fa: Número de módulos en la versión actual λ que se han añadido

		Fe: Número de módulos en la versión actual λ que se han eliminado
Facilidad de prueba	El esfuerzo necesario para probar un programa para asegurarse de que realiza su función pretendida. - Impacto y dimensión del cambio realizado. - Medidas de sentencias, condiciones, cobertura, flujo de datos. - Complejidad estructural del sistema. - Complejidad ciclomática.	0 - 4

Anexo 5. Instrumento para el diagnóstico

Nombre del encuestado: _____

Proyecto: _____

Tipo de encuestado: _____

(Líder, Funcional o Responsable del mantenimiento)

1. ¿Tiene conocimiento de los requisitos implementados en la solución desplegada a cada cliente? (Tipo A)

   Si___ No___

2. ¿Existe un proceso definido para el mantenimiento de software en el proyecto? (Tipo A)

   Si___ No___

3. ¿En el mantenimiento participan todas las partes involucradas? (Tipo A)

   Si___ No___

4. ¿Qué elementos tiene usted en cuenta para estimar el tiempo necesario para el mantenimiento y el esfuerzo correspondiente? (Tipo C)

   1. _____ 2. _____

   3. _____ 4. _____

   5. _____ 6. _____

5. ¿Se realiza la estimación de tiempo de desarrollo por algún método o modelo de estimación? (Tipo A)

   Si___ No___

6. ¿En qué medida cree usted que la mala estimación influya en la satisfacción del cliente? (Tipo B)

   Alto___ Medio___ Bajo___ En ninguna medida___

7. ¿Conoce los artefactos que se generan durante la etapa de mantenimiento? (Tipo A)

   Si___ No___

8. ¿Con que frecuencia se actualiza la documentación después de actualizado el software? (Tipo B)

   Siempre___ Algunas veces____ Nunca___

9. ¿Participa el funcional en el proceso de mantenimiento? (Tipo A)

   Si___ No___

10. ¿Qué nivel de satisfacción tiene el cliente con los resultados del proyecto? (Tipo B)

    Alto___ Medio___ Bajo___

11. ¿Se solucionan durante el mantenimiento incidencias caracterizadas como de mantenimiento Adaptativo, Preventivo y Perfectivo? (Tipo A)

    Si___ No___

12. ¿Se aplica en el proyectos alguna de estas técnicas de mantenimiento: Ingeniería inversa, Reingeniería y Restructuración de Software? (Tipo A)

    Si___ No___

Anexo 6. Evaluación del procedimiento

Nombre de la persona: _____

Evalué el procedimiento en cuanto a la calidad de sus procesos según su criterio en:

- Nivel de aplicabilidad:

Alta___ Media___ Baja___

- Claridad en la definición de sus procesos:

Alta___ Media___ Baja___

- Reusabilidad de los procesos en otros contextos:

Alta___ Media___ Baja___

Evalué el procedimiento en cuanto a la capacidad de los procesos en el análisis de factores esenciales en la ejecución de proyectos de software

- Adaptabilidad al tipo de producto que se desarrolla:

Alta___ Media___ Baja___

- Integración al proceso de desarrollo de software de gestión:

Alta___ Media___ Baja___

- Adaptabilidad a diferentes escenarios según capacidad de los recursos humanos:

Alta___ Media___ Baja___

- Completitud (alcance a todo el proceso de mantenimiento):

Alta___ Media___ Baja___

Evalué el procedimiento en cuanto a la Calidad de los instrumentos

- Claridad y precisión de los instrumentos y artefactos:

Alta___ Media___ Baja___

- Completitud (Que se recoja toda la información necesaria):

Alta___ Media___ Baja___

Evalué el procedimiento en cuanto a la adaptabilidad

- Generalidad (Que se pueda aplicar a diferentes escenarios):

Alta___ Media___ Baja___

Anexo 7. Detalles del Test de Mann-Whitney

Descriptive Statistics

	N	Mean	Std. Deviation	Minimum	Maximum
Valor alcanzado en preguntas de tipo A	26	4,88	3,229	1	8
Valor alcanzado en preguntas de tipo B	26	5,73	1,402	3	7
Medición	26	1,50	,510	1	2

Mann-Whitney Test

Ranks

	Medición	N	Mean Rank	Sum of Ranks
Valor alcanzado en preguntas de tipo A	1	13	7,00	91,00
	2	13	20,00	260,00
	Total	26		
Valor alcanzado en preguntas de tipo B	1	13	7,00	91,00
	2	13	20,00	260,00
	Total	26		

Test Statistics[c]

			Valor alcanzado en preguntas de tipo A	Valor alcanzado en preguntas de tipo B
Mann-Whitney U			,000	,000
Wilcoxon W			91,000	91,000
Z			-4,676	-4,682
Asymp. Sig. (2-tailed)			,000	,000
Exact Sig. [2*(1-tailed Sig.)]			,000[a]	,000[a]
Monte Carlo Sig. (2-tailed)	Sig.		,000[b]	,000[b]
	99% Confidence Interval	Lower Bound	,000	,000
		Upper Bound	,000	,000
Monte Carlo Sig. (1-tailed)	Sig.		,000[b]	,000[b]
	99% Confidence Interval	Lower Bound	,000	,000
		Upper Bound	,000	,000

a. Not corrected for ties.

b. Based on 10000 sampled tables with starting seed 2000000.

c. Grouping Variable: Medición

Figura 11. Detalles del Test de Mann-Whitney.

Anexo 8. Resultados para la pregunta 4 de tipo C para el diagnóstico inicial y final

Tabla 10. Pregunta de tipo C Diagnóstico inicial.

	Diagnóstico inicial Cantidad de veces por lugar donde aparece							
Criterio	**1**	**2**	**3**	**4**	**5**	**6**	**Importancia**	**Relevancia**
Experiencia del equipo	5	1	2	0	0	0	8	8
Métodos documentados	0	2	1	0	0	0	3	3
Complejidad de la solución	2	2	0	1	0	0	4	5
Estándares de codificación	0	0	0	0	0	0	0	0
Definición de requisitos esperados	0	1	2	0	0	0	3	3
Herramientas utilizadas para el MS	0	0	1	0	0	0	1	1

Tabla 11. Pregunta de tipo C Diagnóstico inicial.

	Diagnóstico final Cantidad de veces por lugar donde aparece							
Criterio	**1**	**2**	**3**	**4**	**5**	**6**	**Importancia**	**Relevancia**
Experiencia del equipo	10	2	1				13	13
Métodos documentados	1	4	4	3	0	1	9	13
Complejidad de la solución	1	3	3	1	2	3	7	13
Estándares de codificación		1	2	5	3	2	3	13
Definición de requisitos esperados	1	3	3	1	3	2	7	13
Herramientas utilizadas para el MS				3	5	5	0	13

Anexo 9. Resultados de la aplicación del instrumento de evaluación del procedimiento.

Tabla 12. Resultados de las encuestas sobre la calidad de los procesos del procedimiento.

	Calidad de los procesos		
Encuestados	Aplicabilidad	Claridad	Reusabilidad
1	3	3	3
2	3	3	3
3	2	3	3
4	3	3	3
5	2	2	3
6	2	3	3
7	2	3	3
8	3	2	3
9	3	3	2
10	3	3	2
11	3	3	3

12	2	3	3
13	3	3	3
14	2	2	3
15	2	3	3
16	2	3	3
17	3	2	3
18	3	3	2
19	3	3	2
20	3	3	3
Totales	52	56	56
%	86,66666667	93,3333	93,33333333

Tabla 13. Resultados de las encuestas sobre la Capacidad de los procesos en el análisis de los factores esenciales en los proyectos de software.

	Capacidad de los procesos en el análisis de los factores esenciales en los proyectos de software			
Encuestados	Integración al proceso de desarrollo	Adaptabilidad a los tipos de Producto	Adaptabilidad a diferentes escenarios según capacidad de los recursos humanos	Completitud
1	3	3	3	3
2	2	3	2	3
3	3	3	3	3
4	3	3	3	3
5	3	3	2	3
6	3	3	2	3
7	3	2	3	3
8	3	3	2	3
9	3	3	3	2
10	2	3	2	2
11	3	3	2	3
12	2	3	3	3
13	3	3	3	3
14	2	3	2	3
15	2	3	2	3
16	2	2	3	3
17	3	3	2	3
18	2	3	3	2
19	2	3	2	2
20	2	3	2	3
Totales	51	58	49	56
%	85	96,66666667	81,6666667	93,33333333

Tabla 14. Resultados de las encuestas sobre la Calidad de los instrumentos propuestos.

	Calidad de los instrumentos		
Encuestados	Claridad y precisión	Completitud (Que se recoja toda la información necesaria)	Generalidad
1	3	3	3
2	3	3	2
3	3	3	2
4	3	2	2
5	2	3	3
6	3	3	3
7	2	3	2
8	3	2	3
9	2	3	2
10	3	2	3
11	3	3	3
12	3	3	2
13	3	2	2
14	2	3	3
15	3	3	3
16	2	3	2
17	3	2	3
18	2	3	2
19	3	2	3
20	3	3	3
Totales	54	54	51
%	90	90	85