Universidad de las Ciencias Informáticas

Facultad 1

Procedimiento para la gestión del mantenimiento de software del sub-sistema Planificación del Sistema Integral de Gestión, Cedrux.

Trabajo de diploma para optar por el título de:

Ingeniero en Ciencias Informáticas.

Autor: Danay Serrano González

Tutor: Ing. Raykenler Yzquierdo Herrera

Co-tutora: Ing. Dinia Zayas Romero

Ciudad de la Habana, Junio 2010

Declaración de autoría

Declaro que soy la única autora de este trabajo y autorizo al Centro de Informatización para la Gestión de Entidades (CEIGE) de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

_____

Autora: Danay Serrano González

_____ _____

Tutor: Ing. Raykenler Yzquierdo Herrera Co-tutora: Ing. Dinia Zayas Romero

Resumen

Un proceso de mantenimiento de software comienza cuando existe una petición de modificación a propuesta de un usuario, es decir, el mantenimiento de software se entiende de manera general como las actividades de cambio de un producto software, desde fases tempranas de su ciclo de vida hasta después de su puesta en marcha. La Universidad de las Ciencias Informáticas (UCI) está desarrollando varios proyectos de desarrollo de software de gestión. Las estimaciones, la gestión de los riesgos y los cambios son procesos que no se están desarrollando óptimamente en el sub-proyecto Planificación del proyecto ERP-Cuba, perteneciente al Centro de Informatización para la Gestión de Entidades (CEIGE), provocando insatisfacciones en los clientes. De ahí la necesidad de crear un procedimiento para estandarizar y optimizar la fase de mantenimiento de software del sub-sistema Planificación.

En este trabajo se hace un estudio de los métodos y técnicas, más usados internacionalmente, referentes al mantenimiento de software. Se analiza este proceso en la UCI, sus principales características, ventajas y deficiencias. Los sub-proyectos del Departamento de Desarrollo de Productos, del CEIGE, serán el espacio para la obtención de información y el levantamiento de las necesidades relacionadas con el tema. A partir de este estudio se definen las principales actividades y tareas que debe tener un procedimiento para realizar un correcto mantenimiento de software. Es por ello que el objetivo de este trabajo es diseñar un procedimiento para la gestión del mantenimiento de software del sub-sistema Planificación.

Palabras claves: Mantenimiento, Modificación, Petición, Procedimiento, Software.

Índice

Introducción

La actividad productiva de la Universidad de las Ciencias Informáticas (UCI) está concentrada actualmente en centros de producción que se encargan de temas como: salud y educación, identificación y seguridad digital, almacenamiento de datos, sistemas de gestión, etc. El Centro de Informatización para la Gestión de Entidades (CEIGE) tiene a cargo entre sus proyectos el ERP-Cuba cuyo objetivo es desarrollar el Sistema Integral de Gestión, Cedrux.

Un ERP (por sus siglas en inglés: *Enterprise Resource Planning*) es un sistema integral de gestión empresarial diseñado para modelar y automatizar la mayoría de procesos en una empresa (Contabilidad General, Caja, Banco, Costos y Procesos, Cobros y Pagos, Inventario, Auditoría, Planificación, etc.). Su misión es facilitar la planificación y el control de todos los recursos de la empresa. Un sistema de este tipo gestiona y automatiza muchos procesos con la meta de integrar información a lo largo de la empresa y elimina los complejos enlaces entre los sistemas de las diferentes áreas del negocio.

El proyecto, ERP-Cuba, surge por la necesidad del país de crear un mecanismo para el control de los recursos económicos a nivel nacional, el cual presenta dificultades debido a la fragmentación y diversidad con que transcurre actualmente. En el departamento Desarrollo de Productos del CEIGE existen cinco sub-proyectos encargados del desarrollo, fundamentalmente del sistema Cedrux, y entre ellos Planificación, que se encarga de desarrollar el sub-sistema llamado por el mismo nombre, que servirá para la realización de los procesos de planificación de las entidades del país.

La tarea no finaliza una vez implantado el producto, sino que se requerirán una serie de actividades que garanticen el correcto funcionamiento del mismo. Después de implantado un sistema se hace necesario supervisar que funcione adecuadamente durante un tiempo dado en el cual el sistema puede estar sometido a circunstancias no previstas. Asimismo, 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).

Una simple definición de mantenimiento es: el proceso de registro y seguimiento de problemas y 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, de la utilización incorrecta del flujo de trabajo, de una documentación pobre, de cambios en las directrices de la empresa o de otras tantas razones. 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)

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 y el costo no exceda el planificado.

Existen diversos estándares para el desarrollo de la fase de mantenimiento, estos son generales para cualquier tipo de proyecto y no definen las actividades específicas a ejecutar, lo cual es muy necesario en proyectos como este donde los desarrolladores son estudiantes con pocos años de experiencia y dirigidos por profesores que prácticamente comienzan su vida laboral.

Definir un procedimiento para la gestión del MS del sub-sistema Panificación de Cedrux, constituye un avance importante que permitirá al sub-proyecto cumplir con el cronograma propuesto y seguir trabajando sin grandes trabas en las futuras versiones del producto, ya que desplegar y darle mantenimiento a una solución mientras que se continúan desarrollando otras versiones de la misma, puede resultar muy complicado.

La planificación y el control del cumplimiento de las actividades necesarias contribuirán a que la fase de MS culmine con la eficiencia y los resultados esperados.

La fase de MS de los sub-sistemas de Cedrux que se han implantado ya en las entidades, ha sido desorganizada y por lo tanto más extensa de lo planificado, ha requerido mayor esfuerzo del equipo de desarrollo y se ha retrasado el trabajo del mismo en nuevas versiones, lo cual provoca insatisfacción en el cliente. El sub-sistema Planificación aún no se ha desplegado, pero tampoco cuenta con las actividades a desarrollar para una eficiente gestión del MS. Solo existe a nivel de proyecto un procedimiento para gestión de incidencias que, a pesar de ser pequeño, no se aplica completamente. Durante esta fase se deben tomar en cuenta muchos aspectos como son: la planificación previa de las tareas más significativas, la instalación en las entidades, el ambiente en el que se usará el sistema, las peticiones de cambio tanto en la documentación como en el código, las pruebas a realizar, etc.

La descripción de la problemática anterior contribuyó a la formulación del siguiente *problema científico*: En el sub-proyecto Planificación, perteneciente al proyecto ERP-Cuba, no están definidas las actividades necesarias para llevar a cabo el mantenimiento de software de manera correcta, por lo que se ve afectada la facilidad de mantenimiento, el cumplimiento de los plazos de tiempo y, como consecuencia, la satisfacción del cliente.

El *objeto de estudio* de la presente investigación es: la gestión del mantenimiento de software; enmarcado en el siguiente *campo de acción*: gestión del mantenimiento de software en el sub-proyecto Planificación del proyecto ERP-Cuba.

Teniendo como *objetivo de la investigación*: Diseñar una propuesta de procedimiento para gestionar las actividades de mantenimiento de software del sub-sistema Planificación del sistema Cedrux. De ahí se derivan los siguientes *objetivos específicos*:

- Valorar los métodos existentes que se pueden usar para la fase de mantenimiento de software.
- Diseñar una propuesta de procedimiento que permita la gestión de las actividades de mantenimiento de software del sub-sistema Planificación.
- Validar la propuesta de procedimiento para la gestión del mantenimiento de software del sub-sistema Planificación.

Para guiar la investigación se plantea la siguiente *idea a defender:*

El desarrollo y empleo de un procedimiento para la gestión del mantenimiento de software del sub-sistema Planificación, de Cedrux, facilitará el desarrollo de las tareas de mantenimiento del mismo en cuanto a la facilidad del proceso, el cumplimiento de los compromisos de tiempo y la satisfacción de los requerimientos del cliente.

La estrategia de investigación a seguir es la descriptiva y se desarrollará en el sub-proyecto Planificación del proyecto ERP-Cuba. Con el uso de la misma se logrará describir el procedimiento para la gestión del MS y reflejar lo esencial y más significativo del mismo.

Para lograr el cumplimiento de los objetivos se proponen las siguientes *tareas investigativas:*

- Realización de entrevistas al personal del Departamento de Desarrollo de Productos del Centro de Informatización para la Gestión de Entidades para realizar un diagnóstico de la situación real existente.
- Caracterización de los procedimientos existentes para la fase de mantenimiento de software en el ámbito nacional e internacional.
- Diseño de una propuesta de procedimiento para la gestión de las actividades de mantenimiento de software del sub-sistema Planificación, de Cedrux.
- Evaluación de la propuesta de procedimiento para la gestión de las actividades de mantenimiento de software del sub-sistema Planificación, de Cedrux.
- Documentación de los resultados obtenidos.

Para dar cumplimiento a las tareas de la investigación se emplean varios *métodos científicos.* Entre ellos: métodos teóricos, empíricos y particulares. Dentro de los teóricos, el histórico y el lógico, y de este último el hipotético deductivo, la modelación y el sistémico. De los empíricos se utilizó la observación, y de los particulares se usó la entrevista individual y la encuesta. A continuación se brinda una breve explicación del uso de cada uno de ellos.

Método histórico: Se comienza la investigación con un análisis del surgimiento y trayectoria de las actividades del MS a nivel mundial y nacional.

Método hipotético deductivo: Este método se utiliza porque a partir de la investigación realizada y las experiencias obtenidas en el proyecto ERP-Cuba, siguiendo las reglas lógicas de deducción, se obtienen nuevos conocimientos sobre el MS.

Método de la modelación: Proceso mediante el cual se pueden crear modelos (propuestas, alternativas, estrategias), con vista a aproximaciones de la realidad. Este método se aplica para definir un procedimiento para la ejecución efectiva de las tareas de mantenimiento en el sub-sistema de planificación de Cedrux.

Método sistémico: El estudio de la fase de MS se realiza por partes con el objetivo de lograr una mejor interpretación y obtener la mayor cantidad de información posible, analizándolas después en conjunto y estableciendo las relaciones entre ellas que determinan la estructura y la jerarquía de cada una de ellas dentro del mantenimiento. Este método es decisivo en la investigación ya que facilita la definición del procedimiento.

Método de la observación: La investigación se realiza de forma selectiva, se seleccionan algunos procesos de mantenimiento para realizar una valoración de cómo ocurre realmente el fenómeno y cuáles son los principales problemas en los cuales se deben trabajar.

La entrevista: Este método se utiliza para realizar las entrevistas a especialistas relacionados de alguna forma con temas de MS para obtener criterios. El principal objetivo de estas es conocer cuáles son las proyecciones sobre MS en la UCI y en los casos que se haya aplicado algún estándar o procedimiento, saber por qué se escogió ese, los errores cometidos y cómo los entrevistados consideran que se contribuye al mejoramiento de dichos procesos.

La encuesta: La encuesta es semejante a la entrevista pero escrita, donde a través de un conjunto de preguntas se pretende obtener una información sobre la percepción del encuestado del fenómeno que se investiga, por lo que no puede ser obtenida por observación. Este método se utiliza con el objetivo de obtener una valoración sobre el estado de determinados elementos en la ejecución de algunos de los proyectos del CEIGE.

El presente trabajo está estructurado en 3 capítulos.

En el capítulo 1, que constituye la fundamentación teórica, se presentan algunas generalidades sobre el MS. Se explican los estándares para la gestión del MS utilizados en el mundo y en la UCI; y se propone la realización de un nuevo procedimiento que se ajuste a las necesidades detectadas.

En el capítulo 2, que consta de la descripción de la solución propuesta, se describen las actividades del procedimiento. Se identifican los principios y prioridades que deben seguirse durante el MS; y se propone un grupo de artefactos a generar en la fase MS.

En el capítulo 3, donde se hará una evaluación del procedimiento, se analiza el procedimiento diseñado por varios especialistas, emitiendo sus criterios y valoraciones, que sirven de retroalimentación.

Capítulo 1: Fundamentación teórica

1.1 Introducción

Los distintos estándares relacionados con el desarrollo de software incluyen a lo largo de su ciclo de vida actividades asociadas al MS. Algunas de las grandes compañías como la IEEE[1], han llegado a desarrollar normas orientadas completamente a guiar las actividades necesarias para la realización del MS de diferentes productos.

A lo largo de este capítulo se analizará cómo se realizan las distintas actividades de mantenimiento según varias de las metodologías más utilizadas, algunos aspectos referentes al MS como son: definiciones, tipos, aspectos relacionados al alto costo del MS, así como las técnicas que sirven de apoyo al mismo.

El objetivo de este capítulo es mostrar cómo enfocan el proceso de MS las diferentes instituciones en el mundo para ver cuál o cuáles de los estándares pudiesen ser utilizados en la confección de un procedimiento para la gestión de las actividades de MS en el sub-proyecto Planificación del proyecto ERP-Cuba.

1.2 Generalidades sobre el mantenimiento de software

El Diccionario de la Lengua Española en su vigésima segunda edición define el mantenimiento como: conjunto de operaciones y cuidados necesarios para que instalaciones, edificios, industrias, etc., puedan seguir funcionando adecuadamente.

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, no sufre de un deterioro físico. No obstante, se suele considerar que el software tiene un deterioro en su estructura, cuando a lo largo del tiempo se van incluyendo 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”.

[1] IEEE: Instituto de Ingenieros Electricistas y Electrónicos, una asociación técnico-profesional mundial dedicada a la estandarización, entre otras cosas. Su creación se remonta al año 1884 y en 1963 adoptó el nombre de IEEE al fusionarse asociaciones como el AIEE (*American Institute of Electrical Engineers*) y el IRE (*Institute of Radio Engineers*).

Esta idea del deterioro de la estructura da lugar a la idea de 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. (Sicilia, 2008)

1.3 Definiciones de mantenimiento de software

El ciclo de vida del software es, básicamente, el proceso que sigue un software desde que es un simple proyecto hasta que deja de utilizarse, pasando por el estudio de su origen, sus funcionalidades, sus restricciones, realizar su diseño, fabricarlo, probarlo, instalarlo, utilizarlo, mantenerlo, etc.

Cada actividad involucrada en la producción del software se ubica en uno de estos bloques:

- Análisis: tener claro qué hay que hacer.
- Diseño: decidir cómo se va a hacer.
- Producción: hacerlo.
- Mantenimiento: mejorar y actualizar lo que se hizo.

Mantener y mejorar el software para enfrentar errores descubiertos y nuevos requisitos, puede llevar más tiempo incluso que el desarrollo inicial del software. La mayoría de las actividades desarrolladas en el ciclo de vida del software tiene que ver con dar mantenimiento. Una pequeña parte de este trabajo consiste en arreglar errores y la mayor parte se utiliza en extender el sistema para aumentar las funcionalidades.

Algunas de las clasificaciones para las actividades relacionadas con el mantenimiento pueden ser:

- Mejora: Cuando el sistema está en explotación, a veces es necesario introducir mejoras.
- Ampliación: En ocasiones es necesario introducir nuevas funcionalidades (requisitos) en el producto de software cuando ya está en marcha.
- Corrección de errores: Los errores suelen aparecer con frecuencia en el software cuando está en explotación, aun cuando se dedique gran cantidad de esfuerzo a las pruebas. Aunque en general, la palabra "error" cubre cualquier mal funcionamiento del software, se puede afinar un poco más, y localizar al menos tres orígenes de mal funcionamiento, y así hablar de:

- Defecto: cuando alguna parte del producto del software no se ajusta a su diseño. En ese caso es apropiada la palabra "defecto", porque el software fue bien diseñado, pero al codificarlo, algo no se programó tal como fue diseñado.
- Fallo: cuando alguna parte del producto de software no funciona convenientemente, debido a alguna circunstancia no prevista en el diseño.
- Error: propiamente dicho, se puede dejar para aquellos malos funcionamientos de origen, en principio, desconocido. (ESCAPE, 2008)

Incluso cuando son las últimas en el ciclo de vida del software, las actividades de mantenimiento no son las menos importantes. Muy al contrario el mantenimiento del software se ha convertido en una actividad principal en cuanto a recursos necesarios y costos.

1.3.1 El mantenimiento de software como actividad post-entrega

Un producto software está 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. 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 asume 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 explotación por el cliente.

Otra definición es la dada por Pressman que dice que: “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)

La cual ubica las actividades MS solo como una tarea a realizar después de entregar el producto al cliente. No obstante, en ocasiones se considera que algunas actividades de mantenimiento pueden comenzar antes de la entrega del producto como son la planificación de las actividades posteriores a la entrega, así como toda actividad orientada a facilitar el mantenimiento, por ejemplo la revisión de la

documentación. Aunque, estas pueden considerarse actividades de preparación para el mantenimiento, más que de mantenimiento en sí.

1.3.2 Otros enfoques sobre mantenimiento de software

Varios autores resaltan que hay una necesidad de comenzar a considerar el MS desde el mismo momento en que comienza el desarrollo.

El MS es la totalidad de las actividades necesarias para proporcionar soporte económico 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 actualización de la documentación y la formación de usuarios.

La guía SWEBOK[2] considera que el mantenimiento ocurre durante todo el ciclo de vida. (Abran, 2001)

No son pocas las universidades o instituciones en el mundo que reconocen la fase de MS dentro del ciclo de vida del software como parte del desarrollo del mismo. Reconociéndolo como un proceso de suma importancia por su repercusión económica, temporal y de recursos. Por ejemplo:

- La norma ISO/IEC[3] 12207:1995 reconoce que el Proceso de mantenimiento contiene las actividades y tareas que defina la organización que las llevará a cabo. Señala que el proceso se activa cuando el producto software sufre modificaciones en el código y la documentación asociada, debido a un problema o a la necesidad de mejora o adaptación. Con el objetivo de modificar el producto software existente preservando su integridad. (Comité, 1995)
- El criterio recogido en Métrica v3 es que el objetivo del MS es la obtención de una nueva versión 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.

Con lo visto anteriormente se puede concluir que el MS es “un conjunto de actividades orientadas a facilitar la gestión y solución de los requerimientos del usuario de la manera más correcta posible.

[2] SWEBOK: Software Engineering Body of Knowledge, es un documento creado por la Software Engineering Coordinating Committee, promovido por la IEEE Computer Society, que se define como una guía al conocimiento presente en el área de la Ingeniería del Software.

[3] ISO/IEC: La Comisión Electrotécnica Internacional (IEC) es la organización líder a nivel mundial encargada de preparar y publicar Normas Internacionales para todas las tecnologías eléctricas, electrónicas y afines.

Estas actividades se definen y desarrollan tanto antes como después de la entrega del producto al cliente".

1.4 Tipos de mantenimiento de software

El MS es clasificado generalmente según el objetivo de la modificación que se necesite realizar. A continuación se exponen algunas de las clasificaciones de diversos autores.

En MANTEMA (metodología que será estudiada en el epígrafe 1.6.6) se trabaja con los siguientes tipos de MS:

- No Planificable (NP):
  - Correctivo Urgente (UC): localizar y eliminar los posibles defectos que bloquean el programa o los procesos de funcionamiento de la empresa.
- Planificable (P):
  - Correctivo No Urgente (NUC): localizar y eliminar los posibles defectos de los programas que no son bloqueantes.
  - Perfectivo (PER): añadir al software nuevas funcionalidades solicitadas por los usuarios.
  - Adaptativo (A): modificar el software para adaptarlo a cambios en el entorno de trabajo (hardware o software).
  - Preventivo (PRE): modificar el software para mejorar sus propiedades (calidad, mantenibilidad, etc.).

El estándar IEEE 1219 establece que existen diversos tipos de Mantenimiento del software dependiendo de las demandas de los usuarios del producto software a mantener:

- Mantenimiento Adaptativo
- Mantenimiento Correctivo
- Mantenimiento Perfectivo

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, ya sean cambios en el hardware o en el software, en el que se ejecuta.

Este tipo de mantenimiento es el más usual debido a los rápidos cambios que se producen en la tecnología informática, que en la mayoría de las ocasiones dejan obsoletos los productos software desarrollados, no por su inoperancia, sino por la competitividad entre las empresas, en las que cada vez influye más el software que se utiliza. Por ejemplo, dar acceso a los productos a través de Internet, etc.

Mantenimiento Correctivo

Es la modificación de un producto software después de su puesta en producción y para corregir los fallos descubiertos.

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 podría ser la causa de un fallo. El fallo se produce cuando el comportamiento del sistema es diferente del 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 Mantenimiento del Software.

- *Mantenimiento de emergencia*

  Es un mantenimiento correctivo realizado sin planificación previa, utilizado para mantener operativo el sistema.

Mantenimiento Perfectivo

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

- *Mantenimiento Preventivo*

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, la reestructuración de los programas para mejorar su comprensión, etc.

1.5 Aspectos relacionados al costo del mantenimiento de software

En muchas ocasiones no se puede prescindir de los sistemas en uso por lo que tienen que ser mantenidos, aunque debido a su escasa calidad se disparen los gastos de mantenimiento.

Algunas de las causas por las que las organizaciones actuales requieren mucho trabajo de mantenimiento son:

- Gran cantidad del software actual es muy antiguo y a pesar de la buena calidad del proceso de desarrollo, están limitados por las restricciones de espacio de almacenamiento y tamaño, sin contar con el desfase de las herramientas.
- Sucesivas migraciones a distintas plataformas o sistemas operativos.
- Múltiples modificaciones para mejorarlos y adaptarlos a las necesidades de los usuarios.
- Cambios incontrolados que no tienen en cuenta la arquitectura del sistema, no se realizó ingeniería inversa o reingeniería.

Esto ha provocado que se hayan ido obteniendo sistemas con una escasa calidad debido a: estructuras de datos con un diseño pobre, mala codificación, lógica defectuosa y una documentación escasa o errónea.

Otra causa es que el costo de solucionar un defecto en las etapas finales de desarrollo es mucho mayor que en las etapas iniciales debido a que:

- Es más sencillo cambiar la documentación (la documentación de especificación y diseño) que el propio código fuente.
- Un cambio en una fase avanzada puede implicar cambios en la documentación de las fases anteriores.
- Es más sencillo detectar un defecto en la misma fase que se produce, que corregir los efectos derivados en fases posteriores.

- Otra posible causa es la inexistencia o falta de actualización de los documentos de especificación o diseño.

Sería importante señalar entonces que comenzar con actividades de MS desde el mismo comienzo del desarrollo de software es uno de los aspectos que podrían contribuir en gran medida a cumplir con los plazos de tiempo pactados y a no provocar un esfuerzo extra del equipo de trabajo.

1.6 Modelos, estándares y normas relacionadas con el mantenimiento de software

1.6.1 ISO/IEC 12207

Esta norma agrupa las actividades que pueden llevarse a cabo durante el ciclo de vida del software en cinco procesos principales, ocho procesos de apoyo y cuatro procesos organizativos. Cada proceso del ciclo de vida está dividido en un conjunto de actividades; cada actividad tiene a su vez un conjunto de 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.

Los principales procesos dan servicio a las partes principales (adquisidor, suministrador, desarrollador, operador y mantenedor) 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. Los procesos de apoyo apoyan a otro proceso como parte esencial del mismo, con un propósito bien definido, y contribuye al éxito y calidad del proyecto software. Los procesos organizativos no constituyen objeto de esta investigación.

Los procesos principales del ciclo de vida 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 adquisidor.
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. Lo cual logra mediante las siguientes actividades:

    - Implementación del proceso.
    - Análisis de problemas y modificaciones.
    - Implementación de las modificaciones.
    - Revisión/aceptación del mantenimiento.
    - Migración.
    - Retirada del software.

    Cada una de estas define una serie de tareas con el objetivo de garantizar el éxito de las mismas.

Los procesos de apoyo del ciclo de vida 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. Se pueden emplear Revisiones Conjuntas, Auditorías, Verificación y Validación como técnicas de Aseguramiento de la Calidad.
4. Proceso de verificación. Define las actividades (para el adquisidor, suministrador o una parte independiente) para verificar los productos software según el nivel de detalle acordado en el proyecto.
5. Proceso de validación. Define las actividades (para el adquisidor, suministrador o parte independiente) para validar los productos software del proyecto.
6. Proceso de revisiones conjuntas. Define las actividades para evaluar el estado y productos de una actividad. Este proceso puede ser empleado por dos partes cualesquiera, donde una de las partes (la revisora) revisa a la otra parte (la revisada), de una manera conjunta.

7. Proceso de auditoría. Define las actividades para determinar el cumplimiento de los requisitos, planes y contrato. Este proceso puede ser empleado por dos partes cualesquiera, donde una parte (la auditora) audita los productos software o actividades de otra parte (la auditada).
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.

Se puede concluir que la norma muestra un proceso bien organizado para el desarrollo de un producto software, pero que sólo da una idea de cómo se debe trabajar en el mantenimiento, dejando las especificaciones a disposición de quien lo realice, siendo esta su principal desventaja para su aplicación en el sub-proyecto Planificación, del proyecto ERP-Cuba.

1.6.2 ISO 14764

De esta norma se puede decir que es una especificación de la ISO/IEC 12207 ya que aborda el MS de forma más detallada. En ella se precisan terminología y procesos para el MS e identifica cómo se puede realizar durante la adquisición y operación.

Recomienda que las variaciones en el diseño deben ser estudiadas en todo momento durante el desarrollo para ver el impacto en la mantenibilidad ya que la considera un aspecto muy importante del producto software.

Considera que la estrategia de MS debería consistir de un concepto de mantenimiento, un plan de mantenimiento y el análisis de recursos, con vista a lograr una buena planificación del MS.

Define como actividades para el proceso de MS las siguientes:

- Implementación del Proceso
- Análisis de Modificaciones y Problemas
- Implementación de Modificaciones
- Revisión/Aceptación del Mantenimiento
- Migración
- Retiro

Definiendo para ellas un grupo de tareas que dejan claro que las actividades de mantenimiento deben desarrollarse paralelamente al proceso de desarrollo de software.

A pesar de ello esta norma no es lo suficientemente explícita en cómo se deben implementar las tareas que define.

1.6.3 COBIT

El COBIT[4] Framework es un conjunto de objetivos de control que ayudan (dando unas pautas de actuación) en la realización de una Auditoría Informática, enfocado a procesos de negocio. La primera edición del COBIT fue publicada en 1996 por la ISACF[5], organización creadora de esta norma.

COBIT presenta diferentes niveles, en el más alto se encuentran los dominios. Su agrupamiento natural es confirmado frecuentemente como dominios de responsabilidad en una estructura organizacional, y está en línea con el ciclo administrativo o ciclo de vida aplicable a los procesos de tecnologías de la información. En el nivel inferior se encuentran los procesos como una serie de actividades o tareas conjuntas. Por último, se encuentran las actividades y tareas necesarias para alcanzar un resultado medible. Las actividades cuentan con un concepto de ciclo de vida, mientras que las tareas son consideradas más discretas.

Los dominios antes mencionados son:

- Planificación y Organización
- Adquisición e Implementación
- Entrega y Soporte
- Monitorización

En el que se hace referencia al MS es Adquisición e Implementación que se define como sigue: Para llevar a cabo la estrategia de tecnología de información (TI), las soluciones de TI deben ser identificadas, desarrolladas o adquiridas, así como implementadas e integradas dentro del proceso de

[4] COBIT: Control Objectives for Information and Related Technologies. Objetivos de Control para la información y tecnologías relacionadas: es un conjunto de mejores prácticas (marco) para tecnología de la información.

[5] ISACF: Information Systems Audit and Control Foundation.

negocio. Además, este dominio cubre los cambios y el mantenimiento realizados a sistemas existentes. En él se definen varios procesos de los cuales solo se explicarán en este documento el de adquisición y mantenimiento de aplicaciones y el de gestión de cambios por ser los relacionados con el tema que nos ocupa. Los procesos son:

AI1 Identificar Soluciones

AI2 Adquisición y Mantenimiento de Aplicaciones Software

Control sobre el proceso de TI de: adquisición y mantenimiento de aplicaciones software, que satisface el requisito de negocio suministrar funciones automáticas que soporten de forma efectiva los procesos de negocio, es facilitado por la definición de estados específicos de los requisitos funcionales y operativos, y una implementación estructurada con dictámenes claros y toma en consideración:

- Requisitos de usuario.
- Archivo, gasto, proceso y requisitos externos.
- Interfaz de la máquina-usuario.
- Embalajes habituales.
- Testeo funcional.
- Controles de aplicación y requisitos de seguridad.
- Documentación.

AI3 Adquisición y Mantenimiento de la Infraestructura de la Tecnología

AI4 Desarrollar y Mantener Procedimientos de TI

AI5 Instalación y Acreditación de Sistemas

AI6 Gestión de Cambios

Control sobre el proceso de TI de: administrar el cambio, que satisface el requisito de negocio minimizar la posibilidad de ruptura, alteraciones no autorizadas y errores, es facilitado por un sistema de gestión que se suministre para el análisis, implementación y obtención de todos los

cambios solicitados y realizados para las infraestructuras de TI existentes y toma en consideración:

- Identificación de los cambios.
- Categorizar, priorizar y procedimientos de emergencia.
- Asentamiento de impactos.
- Cambio de autoridad.
- Gestión de venta.
- Distribución de software.

Como se puede observar COBIT está enfocado a apoyar los procesos de auditoría informática con la misión de promover un conjunto de objetivos de control para ser usados por gestores de negocio y auditores. Y aunque especifica algunas tareas referentes al MS, no es lo que se necesita para resolver el problema por el que se realiza esta investigación. Aunque sería conveniente tener en cuenta el aspecto de poder auditar las actividades de mantenimiento, siendo esta su característica más significativa.

1.6.4 IEEE 1219

Este estándar define cambios en un producto software a través de un proceso de mantenimiento dividido en fases. Este proceso es iterativo y en cascada, con una gran semejanza al ciclo de vida del desarrollo clásico. Estas fases son:

Identificación y Clasificación del Problema o de la Modificación. En esta fase 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 software que posteriormente se agruparán en bloques de implementación. Cuando finalice esta primera fase, se podrá disponer de una solicitud de modificación validada y las tareas que se van a llevar a cabo para realizar los cambios solicitados.

Análisis. En esta fase 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. Se realiza un análisis de la viabilidad y un análisis detallado.

Diseño. Se realiza el diseño de las modificaciones del sistema sobre la base de la documentación generada en la fase de Análisis. Para ello se identificarán los módulos software 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.

Implementación. En esta fase, se seguirán determinados procesos 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 procesos son:

- Codificación y pruebas de unidad.
- Integración.
- Análisis de riesgo y revisión.
- Revisión de disponibilidad de pruebas.

Pruebas del Sistema. Esta fase debe llevarse como si se tratara de un sistema completamente integrado. Los procesos a realizar durante las pruebas de sistema son:

- Pruebas de funcionalidad del sistema
- Pruebas de interfaz
- Pruebas de regresión
- Revisión de la disponibilidad de pruebas del sistema para valorar el nivel de preparación para las pruebas de aceptación.

Pruebas de Aceptación. Al igual que las pruebas de sistema, las pruebas de aceptación serán realizadas sobre un sistema completamente integrado. Estas pruebas son realizadas 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.

Liberación del Producto. Una vez probado completamente el sistema, se libera de la versión modificada.

El estándar se centra completamente en el proceso de MS señalando la importancia de la planificación previa de las actividades de MS y el control a las modificaciones, mediante las pruebas de sistema y aceptación, exponiendo la Gestión de la configuración como un elemento crítico dentro del proceso.

1.6.5 Métrica III

MÉTRICA Versión 3 está enfocada a procesos enmarcada en ISO/IEC 12207.

MÉTRICA Versión 3 ha sido concebida para abarcar el desarrollo completo de Sistemas de Información sea cual sea su complejidad y magnitud, por lo cual su estructura responde a desarrollos máximos y deberá adaptarse y dimensionarse en cada momento de acuerdo con las características particulares de cada proyecto.

Esta metodología cubre distintos tipos de desarrollo: estructurado y orientado a objetos, mediante los procesos: Planificación de sistemas de información, Desarrollo de sistemas de información y Mantenimiento de sistemas de información. Además tiene en cuenta la gestión de la configuración, el aseguramiento de la calidad de software, con el fin de prevenir deficiencias al obtener un producto software, como procesos de apoyo.

Para el proceso de Mantenimiento MÉTRICA Versión 3 comprende las siguientes actividades:

- Registro de la Petición.
- Análisis de la Petición.
- Preparación de la Implementación de la Modificación.
- Seguimiento y Evaluación de los cambios hasta la Aceptación.

Como ya se ha visto antes en otras metodologías las actividades están conformadas por una serie de tareas que se ejecutan con el fin de lograr el éxito en la aplicación de misma. Es importante, sin embargo, señalar que permite de manera integral insertar las tareas de mantenimiento como parte del desarrollo del software, es decir, que no es visto solamente como un proceso que tiene lugar posterior a la implantación del producto.

1.6.6 MANTEMA

Metodología para el MS desarrollada por la Universidad Castilla-La Mancha, basada en el estándar ISO/IEC 12207.

Define tres grupos de actividades y tareas: conjunto de actividades y tareas "iniciales" comunes a todos los tipos de mantenimiento, conjunto de actividades y tareas "finales" comunes a todos los tipos

de mantenimiento y las actividades "intermedias" de los tipos de mantenimiento planificables, que son bastante coincidentes, y del mantenimiento no planificable, según su propia definición de mantenimiento vista anteriormente.

Utiliza técnicas como la Ingeniería inversa y la reingeniería, además tiene en cuenta las actividades y elementos de la Gestión de la configuración.

La Metodología MANTEMA versión 2 incluye una propuesta para abordar la auditoría del proceso de MS basada en: el modelo de procesos del software definido en la norma ISO 12207, el estándar ISO 14764 para el proceso de mantenimiento del software, y la metodología COBIT para la auditoría de sistemas de información. (Ruiz, et al., 2005)

Esta metodología es la más amplia ya que agrupa muchos aspectos que en las anteriores se encuentran por separado por lo que será de gran utilidad para la elaboración del procedimiento resultante de esta investigación.

1.6.7 Procedimiento para la gestión de incidencias

Este procedimiento se usa hoy para la gestión de incidencias de cada uno de los sub-sistemas de Cedrux que han sido implantados en las entidades. El mismo define roles con sus responsabilidades, artefactos, así como los estados y las prioridades que pueden tener las incidencias. Define el flujo por el que puede pasar una no conformidad (pasando de un estado a otro) y cómo serán atendidas según la prioridad asignada, lo cual se realiza según el impacto que tenga desde el punto de vista del usuario.

Para el control de este proceso el procedimiento propone que se generen informes de control que incluyan aspectos como: análisis de los tiempos de resolución de incidentes organizados según su urgencia e impacto, número de incidentes clasificados temporalmente y por prioridades, tiempos de resolución clasificados en función del impacto y la urgencia de los incidentes y grado de satisfacción del cliente; haciendo énfasis en la importancia de saber cuán satisfecho quedó el cliente.

Esta última parte no se pone en práctica, sirviendo este procedimiento sólo para la gestión y resolución de un problema del software.

De manera concluyente se pude decir que después de estudiar las normas, estándares y modelos vistos anteriormente ninguno se ajusta a las necesidades del sub-proyecto Planificación debido a:

- La escasa especificidad de los procesos que describen.
- Que no tratan todos los aspectos necesarios a tener en cuenta en la fase de MS.
- Que los artefactos que se proponen, por lo general, no recogen suficiente información.

1.7 Técnicas o métodos usados en el mantenimiento de software

1.7.1 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. (Sicilia, 2008)

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

- Reduce los riesgos 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 mantenimiento del software.
- 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 e ingeniería directa; con la finalidad de crear versiones de programas ya existentes que sean de mejor calidad y los mismos tengan una mayor facilidad de mantenimiento.

1.7.2 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.). (Sicilia, 2008)

Beneficios de ingeniería inversa:

- 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 suelen 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 pueden conducir a que surjan efectos no deseados, estas 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 costos y los riesgos de mantenimiento.

La ingeniería inversa puede ser de varios tipos:

- Ingeniería inversa de datos: Se aplica sobre algún código de bases datos (aplicación, código SQL, etc.) 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 el 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 el objetivo de mantener la lógica interna del programa para obtener los modelos y especificaciones que sirvieron de base para la construcción de la misma, con el objetivo de tomarlas como punto de partida en procesos de ingeniería directa que permitan modificar dicha interfaz.

La ingeniería inversa cuenta con un grupo de herramientas que hacen posible la comprensión del código mediante la extracción de la información que este posee.

1.7.3 Reestructuración del software

La reestructuración 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. Es cambio de representación de un producto software, pero dentro del mismo nivel de abstracción.

Los beneficios de la reestructuración son:

- Programas de mayor calidad, con mejor documentación, 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, aun 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 dos: del código y de datos.

1. 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. El objetivo es tomar el código y derivar un diseño de procedimientos que se ajuste a la filosofía de la programación estructurada.

2. 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 y descripciones de interfaz. A esta actividad también se le denomina análisis de datos.

Una vez finalizado el análisis de datos, comienza el rediseño. La forma más sencilla de realizarlo es empleando un paso de estandarización de datos que clarifica sus definiciones para lograr una consistencia entre los nombres de objetos o entre formatos de registro físico en una estructura de datos o en archivos existentes. Otra forma de realizar dicho rediseño es la denominada racionalización (de la información) que garantiza que todas las convenciones de denominación 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)

1.7.4 Técnica basada en Valor

Añadir puntos de flexibilidad no es gratis. Si un diseño es muy flexible (típicamente implementando muchos patrones de diseño), tendrá una gran cambiabilidad y estabilidad, pero perderá en analizabilidad y además el costo inicial de construcción será mayor. Si no se implementa ningún patrón, la situación será justo la contraria. (Piattini, et al., 2006)

Es por ello que la técnica para la mejora del MS basada en valor propone ciertas pautas que expresan cómo ganar en mantenibilidad. Lo primero sería identificar el valor de cada una de las mejoras a realizarse, para lo que se propone un modelo lo más simple posible e incrementar su complejidad allí donde las características del proyecto lo requieran; teniendo en cuenta que a mayor probabilidad de cambio y mayor importancia, mayor será la mantenibilidad que aporte la mejora. Para llevar a cabo lo expuesto se debe:

- Localizar los candidatos de cambio.

  Se deben marcar como posibilidad de mejora aquellos lugares donde se estime que el mantenimiento podría ser caro. Para lo que se usará un catálogo de reglas que deben cumplir todas las aplicaciones software para que los costos de mantenimiento permanezcan bajo control. La idea es que allí donde una regla sea violada, hay una posibilidad de mejora de diseño para hacer el sistema más estable y flexible frente a posibles cambios y ampliaciones. (Piattini, et al., 2006)

- Estimar la probabilidad de cambio de cada artefacto relacionado con la mejora.

Las técnicas de "*predicción del cambio*" pretenden estimar la probabilidad de cambio futura de cada uno de los artefactos de un diseño software. (Piattini, et al., 2006)

Esta técnica basada en valor se refiere a dos posibles técnicas a utilizar seleccionando la primera de las relacionadas a continuación:

  - Técnicas basadas en información histórica, que revisan los cambios que sucedieron en el pasado, y auguran un comportamiento similar en el futuro más cercano.
  - Técnicas de predicción basadas en el análisis de la estructura estática de los diseños. Un ejemplo de estas técnicas es el bien conocido anti-patrón "*God Object*", donde un objeto se relaciona con muchos otros objetos. Este objeto tendrá una mayor probabilidad de cambio porque si un objeto referenciado cambia su interfaz, el cambio puede ser propagado al primer objeto. (Piattini, et al., 2006)

- Estimar la importancia relativa de cada artefacto.

La solución que propone es estimar la importancia de los requisitos usando la opinión de los usuarios, y posteriormente, trasladar mediante técnicas de trazabilidad dichos datos a los artefactos de diseño que tengan una valoración más alta.

De manera general las técnicas vistas en este epígrafe, relacionadas al MS, sirven de apoyo a los procesos a desarrollar en esta fase sobre todo a la hora de actualizar los cambios realizados en el código y su impacto en la documentación. Por lo cual es necesario tomarlas en cuenta en la definición del procedimiento que se propone en el próximo capítulo.

1.8 Conclusiones del capítulo

En este capítulo se ilustraron aspectos referentes al MS: los modelos, las técnicas, normas y metodologías, así como los tipos de MS existentes y sus características. Se investigó acerca de las actividades del proceso de MS, haciendo énfasis en las que proponen los estándares internacionales. Una vez realizado este estudio se decidió elaborar un nuevo procedimiento en el que se tengan en cuenta los elementos más importantes, de manera que posibilite resolver las necesidades identificadas.

Capítulo 2: Procedimiento para la gestión del Mantenimiento de software.

2.1 Introducción

El procedimiento a desarrollar en esta investigación tiene dos momentos básicos: el primero, en el capítulo anterior se realizó una revisión documental de diferentes fuentes de información: artículos de revistas, libros, fuentes electrónicas de Internet, tratando sobre diferentes aspectos de la fase de MS. En el segundo momento concerniente al presente capítulo se expondrá el procedimiento ya elaborado como resultado de un profundo estudio y reflexiones sobre el tema en cuestión.

El objetivo fundamental de este capítulo es documentar todos los pasos que conforman la propuesta de procedimiento para la gestión de las actividades de MS en el sub-proyecto Planificación, teniendo en cuenta los procesos que actualmente se llevan a cabo en la misma y la introducción de nuevas técnicas y elementos que son de suma importancia para que un software tenga la calidad requerida.

Al terminar este capítulo se contará con una serie de elementos no sólo para evitar los errores comunes en el MS, sino también para manejar de manera eficiente los recursos empleados en esta fase.

2.2 Descripción del procedimiento

Este procedimiento se centra en la gestión de las PM al software ya que sus actividades se basan fundamentalmente en la resolución de las mismas, por parte del equipo de desarrollo, satisfaciendo las expectativas que se trazaron en la etapa de contratación. Comprende la distribución de los release[6] con las nuevas versiones del producto a los especialistas del “Grupo de Negocios, Consultoría, Implantación y Soporte, (GNCIS)” del CEIGE, para su posterior ejecución en las entidades. También, incluye toda la planificación del proceso de MS y un grupo de recomendaciones a ejecutar durante el desarrollo que facilitarán la mantenibilidad del producto.

El procedimiento tiene como objetivo proporcionar una línea de trabajo a los desarrolladores de los proyectos donde se utilice, por la necesidad de obtener plazos reales y aceptables para la corrección

[6] Release: Una versión aprobada formalmente de un elemento de configuración, independientemente del medio, diseñado y fijado en un momento específico del ciclo de vida de ese elemento de configuración. Versión completa de un producto software para salir al mercado.

del software y mayores garantías referentes a la organización del trabajo en equipo y la calidad de los entregables.

Se pretende que el procedimiento desarrollado sea aplicable a todos los sub-sistemas del proyecto ERP-Cuba ya que estos requieren de la ejecución del mismo para facilitar y hacer más eficientes las actividades definidas, para lograr la calidad deseada en el producto software, aunque es posible que, en casos específicos, la manera de ejecutarlas varíe de acuerdo con los objetivos y características de cada sub-proyecto. A continuación se detallarán las actividades que conforman el procedimiento desarrollado, agrupadas en tres etapas: Iniciación del mantenimiento de software, Ejecución del mantenimiento de software y Finalización del mantenimiento de software.

2.1.1 Iniciación del mantenimiento de software

Al comenzar un proceso de mantenimiento, e incluso antes de comenzar un proyecto software, es necesario realizar una planificación de las futuras etapas 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.

En esta etapa de Inicialización del MS se llevan a cabo un grupo de actividades con vista a garantizar la organización del proceso de mantenimiento. Estas actividades son:

- Tareas a ejecutar durante el desarrollo.
- Familiarización con el software al que se le dará mantenimiento.
- Definición de los requisitos del proyecto de MS.
- Determinar el esfuerzo de MS.
- Actividades de Gestión de la configuración.
- Elaboración del Plan de MS.
- Preparar entornos de producción para probar el procedimiento de MS.
- Preparar el nuevo personal encargado del MS.
- Recepción de las peticiones de modificación.
- 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.

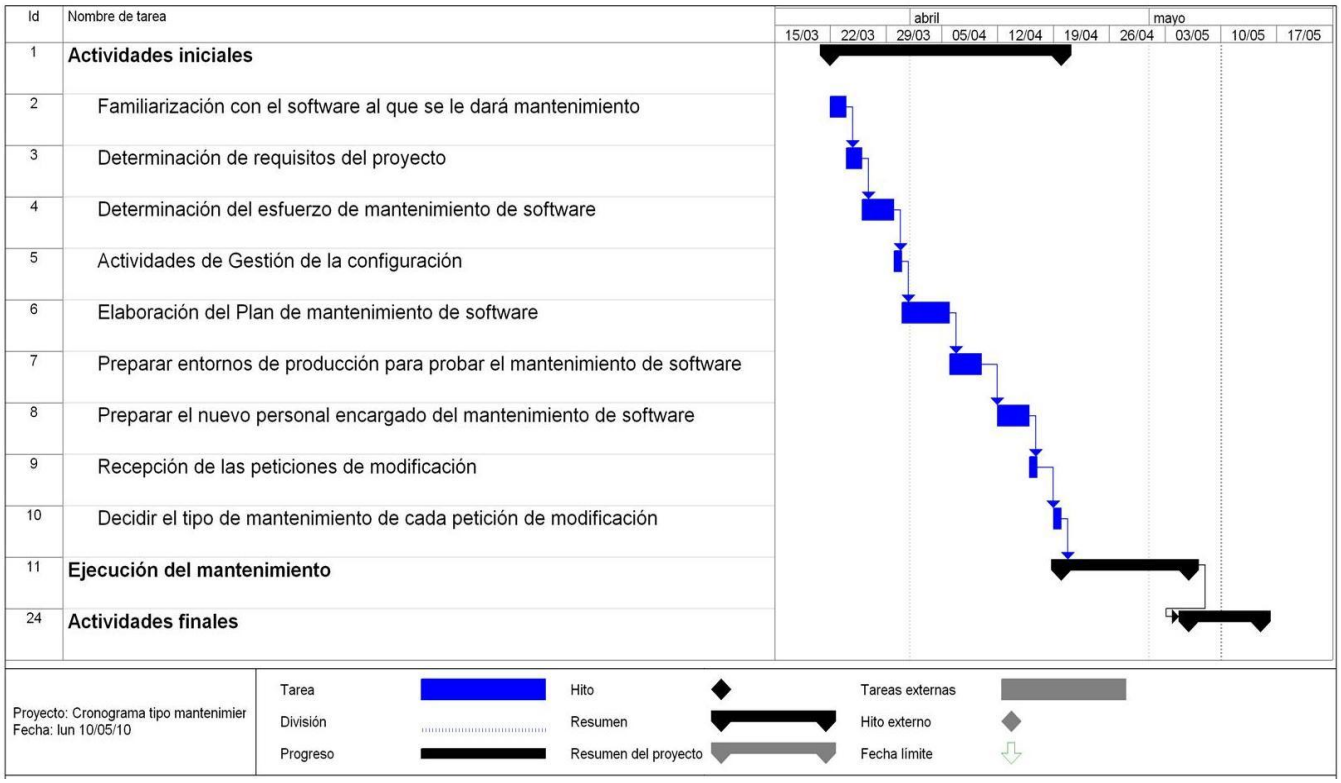

Figura 1: Relación entre las actividades de la etapa Inicialización del MS.

A continuación se describen detalladamente cada una de las actividades para esta primera etapa del procedimiento.

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 durante el mantenimiento se encuentran:

- Trabajar organizadamente a partir de un procedimiento de desarrollo de software que tenga claramente definidos las actividades, 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 código.
- Usar nombres descriptivos para las variables, procedimientos, clases, etc. que sean fáciles de entender después.

- Reducir la complejidad ciclomática de los métodos partiéndolos en otros más pequeños.
- Aplicar convenientemente estilos de arquitectura y patrones de diseño que garanticen la mantenibilidad, sin que se pierda la analizabilidad del diseño.

El máximo responsable del cumplimiento de estas tareas es el Jefe de proyecto, aunque generalmente las tareas más técnicas recaen sobre el Arquitecto principal del sistema.

2. Familiarización con el software al que se le dará mantenimiento

Para poder determinar el esfuerzo del MS, es necesario conocer el software al 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. Sería muy conveniente emplear en las tareas de mantenimiento personal que haya formado parte del desarrollo del sistema, garantizándose un dominio mayor de la solución. En la medida que se incorporen personas al equipo debe garantizarse una preparación adecuada en cuanto al conocimiento del sistema; por lo que es aconsejable la realización de un taller una vez que se incorporen nuevos miembros al equipo donde se explique el sistema de trabajo y se comenten las características fundamentales del sistema. El responsable de la realización de los talleres es el Jefe de mantenimiento.

3. Definición de los requisitos del proyecto de mantenimiento de software

A este nivel, el proceso de mantenimiento necesita ser asociado al entorno de negocio. Se debería completar una revisión de las futuras expectativas que podrían incluir los aspectos que se mencionan a continuación. La especificación de dichos aspectos se hace usando como base la plantilla 0113_Especificación de Requisitos de Software (Universidad de las Ciencias Informáticas, 2010). Esta plantilla recoge la especificación tanto de los requisitos funcionales como no funcionales. Hay que tener presente que aunque se usa esta plantilla, 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 los requisitos esperados y situaciones esperadas, y no necesariamente los requisitos que tienen que implementarse.

- Cambios del sistema esperados, externos o regulatorios (Sección que se le agrega a la plantilla). Este aspecto debe ser del dominio del Jefe de mantenimiento.

- 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 mantenimiento y del Arquitecto de sistema.
- 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 mantenimiento.
- Nuevas tecnologías que necesitan ser incorporadas (Sección que se le agrega a la plantilla). Este aspecto debe ser del dominio del Arquitecto de sistema.

El documento generado debe nombrarse Especificación de requisitos de MS esperados (Yzquierdo, 2010). La utilidad del artefacto generado es que todo esto necesita ser cuantificado o dimensionado para determinar la carga de mantenimiento futuro para la organización.

Los encargados de la definición de los requisitos del MS son: Jefe de mantenimiento, Arquitecto de sistema y Analistas de requisitos del proyecto.

4. Determinación del esfuerzo de mantenimiento de software

Se han desarrollado varias técnicas de estimación para el desarrollo de software, aunque cada una tiene sus puntos fuertes y sus puntos débiles, todas tienen en común los siguientes atributos:

- Se ha de establecer de antemano el ámbito del proyecto.
- Como bases para la realización de estimaciones se usan métricas del software de proyectos pasados.
- El proyecto se desglosa en partes más pequeñas que se estiman individualmente.

El objetivo principal es obtener la descripción del esfuerzo global y las necesidades del área de mantenimiento. Esto incluye las cantidades y clases de deficiencias en la funcionalidad del sistema. Además, las revisiones de los procedimientos de mantenimiento son necesarias para determinar el esfuerzo de mantenimiento global.

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.

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. En consecuencia, 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, tanto para la aplicación actual como en la industria en general. Se debe valorar la posibilidad de contar con un equipo para el MS que haya (en la medida de las posibilidades) participado en el desarrollo del sistema a mantener y que tenga experiencia en el mantenimiento de sistemas. En caso de no contar con el personal idóneo en el procedimiento se proponen un grupo de acciones en los siguientes epígrafes para mitigar los riesgos que esto puede acarrear.

- 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 y por tanto el esfuerzo necesario para dar una solución efectiva.
- Los métodos de programación utilizados actualmente por el equipo de desarrollo.
- Complejidad de cada módulo del sistema y del sistema como un todo.
- Complejidad de un problema y su impacto para el usuario.
- Tiempo estimado de solución de un problema.
- 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. De acuerdo con las peculiaridades del proyecto pueden valorarse los siguientes tipos de herramientas para el mantenimiento:
    - Herramientas de gestión de las incidencias.
        - HelpDesk - Gestor de Incidencias, GMF (Módulo de Gestión de Incidencias), Mantis, sistema de gestión de incidencias, NetSupport DNA Helpdesk, Bugzilla, Itracker, FootPrints, SIGILA. Sistema Integral de Gestión de Incidencias y Localización de Averías, etc.
    - Herramientas CASE.
        - DESIGNER/2000 de ORACLE, EASY CASE de Evergreen, Rational ROSE / RequisitePro, EXCELERATOR de Intersolv, OBJECT MAKER de MarkIV, OMToolde GTE, PARADIGM Plus de Platinum, SILVERRUN de CSA Research, System Architect de Telelogic, etc.
    - Herramientas de perfeccionamiento del código.
        - Para este propósito se pueden emplear los IDE utilizados para el desarrollo como pueden ser: Zend Studio, NetBeans, NotePad ++, etc.
    - Herramientas de ingeniería inversa.
        - Visual Paradigm, Visual Studio, Eclipse, NetBeans, phpDocumentor, etc.
    - Herramientas de gestión de la configuración.
        - CVS, Subversion, SourceSafe, ClearCase, Darcs, Bazaar, Plastic SCM, Git, Mercurial, IBM Rational Clearcase, Perforce, Borland Starteam, TFS, etc.
    - Herramientas de prueba.

- ObjectGen, TestTools del VisualStudio de Microsoft, Rational Robot, TestGen4J, JUnit, etc.

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.

Algunas de las métricas que se pudieran estar utilizando para ayudar en la realización de esta tarea se pueden ver en el Anexo II. Según Capers Jones y otros autores, las estimaciones más efectivas pudiesen estarse haciendo a partir de la utilización de métricas de puntos de función, las que deben ser usadas para el cálculo del esfuerzo de la fase de MS.

La meta debe ser tener una estimación de los recursos y el tiempo que 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 (pueden 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.

También hay que considerar para una correcta estimación el resultado de la actividad Definición de los requisitos del proyecto de MS, dado que se obtendrían los niveles de servicios que hay que satisfacer durante el mantenimiento.

El responsable de determinar el esfuerzo del MS es Jefe de mantenimiento. La información generada se especificará en la sección correspondiente en el Plan de mantenimiento de software (versión preliminar) (Yzquierdo, 2010).

5. Actividades de Gestión de la configuración

La Gestión de configuración (GC) es una disciplina cuya misión es controlar la evolución de un sistema software en su definición más abarcadora. Las actividades relacionadas con la GC de la fase de MS deben ajustarse a lo planteado en el Plan de soporte, epígrafe Gestión de configuraciones definido

para el proyecto ERP-Cuba. Así mismo el control de los cambios como se define en el “Procedimiento para la gestión de cambios” definido en el GNCIS.

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. El líder de gestión del proyecto es el responsable del control de las actividades de GC.

6. Elaboración del Plan de mantenimiento de software

La información recogida dará una base para el Plan de mantenimiento. El plan deberá cubrir cuatro áreas principales:

- Proceso de Mantenimiento
- Organización
- Reserva de Recursos
- Auditoría del Rendimiento.

Proceso de Mantenimiento

El primer proceso deberá ser descrito en términos de su alcance, la secuencia de procesamiento, y la manera de controlar este proceso.

Alcance del proceso

El plan necesita definir las fronteras del proceso de mantenimiento. El proceso comienza con la recepción de una *petición de modificación* (a partir de ahora, en este documento, PM) y termina con la puesta en producción. La diferencia entre el mantenimiento y el desarrollo se verá claramente en este punto: ¿Una mejora se considera un nuevo desarrollo o un mantenimiento? ¿En qué punto un sistema de nuevo desarrollo entra en el proceso de mantenimiento? Otro elemento que debería ser definido dentro del alcance es si debe categorizarse el proceso de mantenimiento y cómo debe hacerse. ¿Habrá diferencias entre notificaciones y otros tipos de mantenimiento? ¿Se considerarán las adaptaciones y mejoras dentro del mismo proceso o se gestionarán de forma diferente?

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:

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

Control del proceso

Cada uno de los pasos en el proceso debe ser controlado y medido. Deberían ser definidos los niveles de rendimiento esperados. Los mecanismos de control deberían ser automatizados en lo posible.

Organización

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

Reserva de recursos

Se reflejan 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 mantenimiento.

7. **Preparación de entornos de producción para probar el procedimiento de mantenimiento de software**

Se deben preparar las condiciones para que posterior a la liberación del producto, el mismo pueda probarse 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 el CEIGE, 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 utilizará el sistema desarrollado. Durante esta etapa se generan todo tipo de incidencias y las que representan una PM son tratadas por el equipo de desarrollo, así mismo funcionará para el sub-sistema Planificación. En consecuencia, el equipo solucionará los problemas encontrados siguiendo el mismo sistema de trabajo que venía utilizando durante el desarrollo ignorando que las condiciones son otras y que por tanto influyen otro tipo de factores propios del MS y ya explicados anteriormente. Por tanto, es conveniente durante este período poner en funcionamiento el procedimiento de MS definido. La cantidad de recursos dependerá de la dimensión y complejidad del Piloto.

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 mantenimiento.

8. **Preparación del nuevo personal encargado del mantenimiento de software**

Las personas que se encargarán del MS deben estar preparadas no sólo en el manejo de la solución de software sobre la que se incidirá sino que se debe 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. Dicho programa debe contar con una descripción de la planificación de los cursos a impartirse, el objetivo que se persigue con cada uno y el método de evaluación de los contenidos abordados.

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), en dicho plan queda claramente establecidas las actividades, artefactos y roles que forman parte del proceso de MS. Es

conveniente que se deje claramente establecido el cronograma las fechas en las que se impartirán los cursos correspondientes, también debe definirse los recursos necesarios.

Los Analistas de procesos y los de Requisitos son los responsables de la realización de esta actividad.

9. Recepción de las peticiones de modificación

La admisión y registro del incidente es el primer paso necesario para una correcta gestión del mismo. Este registro debe hacerse por parte del equipo de soporte del GNCIS y solo llegan al Equipo de mantenimiento aquellas peticiones que requieren una modificación del sistema desplegado. Además, las incidencias pueden provenir del Equipo de mantenimiento.

El proceso de registro debe realizarse inmediatamente que se detecte la anomalía pues resulta mucho más costoso hacerlo posteriormente y se corre el riesgo de que la aparición de nuevas incidencias demore indefinidamente el proceso.

Cuando las PM provienen del sub-proyecto Soporte se registrarán con el identificador otorgado por el mismo, las que no el Equipo de mantenimiento le otorgará un identificador.

El responsable de esta actividad es el Jefe de mantenimiento y para que exista constancia debe llenar la tabla mostrada en el Anexo III por cada PM.

10. Definición del tipo de mantenimiento de cada petición de modificación

En esta actividad se identifica y asigna una prioridad inicial a las PM al software. Cada PM 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, adaptativo o perfectivo.

Para mantener en todo momento una correcta organización y control del MS es aconsejable hacer una revisión periódica de las PM recibidas, de esta manera se pueden observar las más críticas y solicitadas, y ayudaría a prevenir cualquier bloqueo de trabajo o parada en alguna de las etapas del MS por la falta de orientación.

Una vez recibida la PM, comienza el MS. En este momento las tareas a realizarse son:

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

Serán identificados tanto las PM como los procesos que se determinen realizar. Estos identificadores se mantendrán en un repositorio para su control.

Se deben realizar reuniones de revisión en la que participará el Jefe de proyecto, Arquitectos, Analistas, Funcionales y Jefe de mantenimiento. Pueden participar otros implicados en dependencia de las peculiaridades del proyecto. La frecuencia y duración de las reuniones de revisión de la clasificación deberán ser dependientes del proyecto. 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 PM 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 PM individual, con lo que se 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, se aprecia la importancia de clasificar las modificaciones a realizar para que puedan ser aplicadas a un sistema particular de la mejor manera posible y sin que afecte al sistema existente o a otras posibles modificaciones.

La prioridad de las PM 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.

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

- Relación de los problemas o nuevos requerimientos.
- Evaluación de los problemas o nuevos requerimientos.
- Datos de verificación (para el mantenimiento correctivo).
- Registro de las PM como se muestra en la tabla del Anexo III.
- Estimación inicial de los recursos necesarios para modificar el sistema existente.

En esta actividad participan todos los involucrados en el mantenimiento, siendo el Jefe de mantenimiento el máximo responsable de la misma.

2.1.2 Ejecución del mantenimiento de software

Este procedimiento define cambios en un producto software a través de un proceso de mantenimiento dividido en fases y tipos de mantenimiento. Este proceso es iterativo y en cascada, con una gran semejanza al ciclo de vida del desarrollo clásico. Estas fases se detallan a lo largo del procedimiento, indicando en cada una los elementos de los que se dispone al empezar, por ejemplo la documentación, las tareas que se han de realizar, y por último el resultado de la fase, como documentación generada, código, etc. Estas fases son:

- Análisis.
- Diseño.
- Implementación.
- Pruebas de sistema.

Cada una de estas fases tiene asociadas métricas de calidad que permiten de manera eficaz realizar el seguimiento y gestión del proceso de MS.

Además de las métricas, en cada fase se realizará un seguimiento o control de los procedimientos que se llevan a cabo, con el fin de documentar y monitorizar las modificaciones que se lleven a cabo. Para ello se mantendrá un repositorio con la identificación y descripción de los cambios.

Primero se expondrán las fases a las que se hace referencia, recogiendo los aspectos generales, 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.

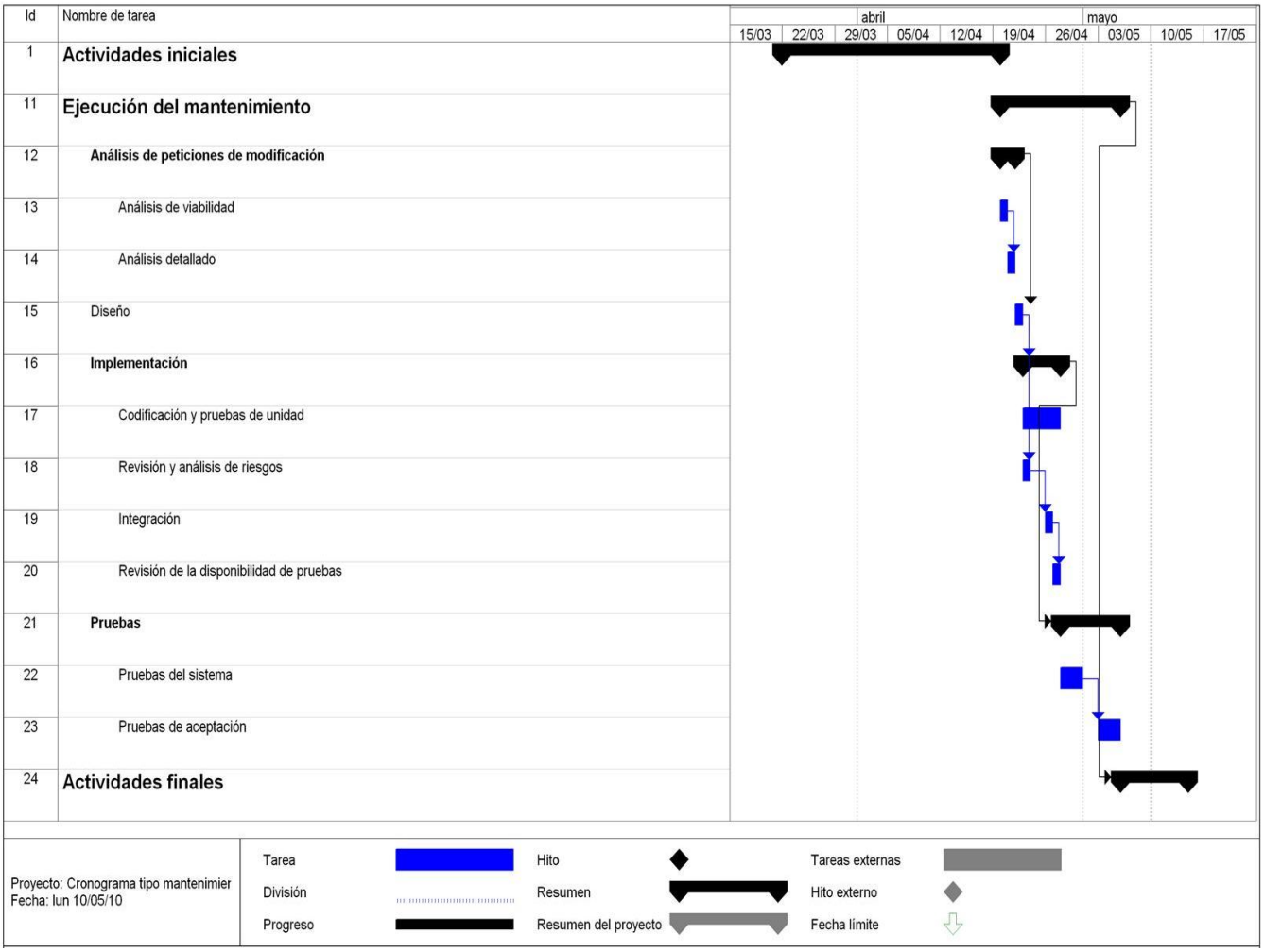

Figura 2: Relación entre las actividades de la etapa Ejecución del MS.

1. Análisis

En esta fase 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 que se va a utilizar en esta fase proviene del repositorio, además de la documentación del proyecto y del sistema existente.

Una PM podría generar varios requisitos de funcionalidad, rendimiento, usabilidad, fiabilidad, comprensibilidad y mantenibilidad, que podrán ser descompuestos en varios requisitos de software, base de datos, interfaz, documentación y hardware. Por lo tanto, es necesario que participen los funcionales y desarrolladores 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). Este es el procedimiento por el cual a partir del análisis de un sistema software existente, se identifican sus componentes y las interrelaciones que existen entre ellos, así como su representación en un nivel de abstracción más elevado. Para este procedimiento, 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 prototipado.
- Análisis de los requisitos de conversión.
- Implicaciones de seguridad.
- Implicaciones de robustez.
- Factores humanos.
- Beneficios de realizar las modificaciones.

Análisis detallado

Al identificar los elementos susceptibles de modificación en el Análisis, se examinan todos los elementos generados en las etapas del MS que están afectados, por ejemplo el software, las especificaciones, las bases de datos, la documentación, etc. Cada uno de estos elementos será identificado y generado si fuera necesario, especificando las partes del producto que van a ser modificadas, las interfaces afectadas, los cambios esperados por el usuario, experiencia necesaria del personal para hacer los cambios, y el tiempo estimado para completar la modificación.

En cuanto a la estrategia de pruebas, está basada en los elementos de modificación. 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. Al ser modificaciones hechas sobre un producto software ya desarrollado hay que revalidar las pruebas básicas de éste, por lo que se tendrán que generar y utilizar nuevos casos de prueba para el mismo.

Por último, se tiene que desarrollar un plan de implementación preliminar que tendrá que prever el menor impacto posible para los usuarios del sistema software, tanto desde el punto de vista del futuro trabajo con el sistema, como durante la fase de desarrollo, para que esta no sea con métodos intrusivos en el entorno en producción.

En resumen, se puede decir, que 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.
- Identificación de los elementos de seguridad.
- Identificación de los elementos de robustez. Al igual que las dos anteriores esta actividad se describe en el artefacto propuesto Análisis de viabilidad (Yzquierdo, 2010).
- 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 de implementación. Se describen en el artefacto Cronograma del proyecto definido por el área afectada.

La gestión en la fase de Análisis tiene que realizarse metódicamente, obteniendo en primer lugar, la última versión de la documentación del proyecto y del sistema a modificarse que debe tener la propia organización, y verificando que toda la documentación del Análisis y del proyecto sea actualizada y controlada. En cuanto a las funciones de pruebas asignadas a la organización, hay que verificar que exista la estrategia apropiada para la realización de estas pruebas y que la planificación de los cambios pueda soportar la realización de las mismas.

Los Analistas en conjunto con el Jefe de mantenimiento, el Arquitecto de sistema y el 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 afectan los cambios en ellos. También, como parte de este control, se revisarán los recursos estimados y la planificación temporal, para poder verificar su perfecto cumplimiento.

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 el 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. Para ello se debe considerar en relación con los Acuerdos de niveles de servicios (SLAs) pactados con qué frecuencia se deben generar nuevas versiones con la solución de las solicitudes de cambio. Concretamente se estaría organizando el trabajo del Equipo de mantenimiento y del equipo de desarrollo al proyectar sobre las versiones la integración entre funcionalidades. Así el equipo de desarrollo estaría consciente de 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.
- En el caso de que el desarrollo de una funcionalidad nueva requiera invertir un período de tiempo superior al destinado para el 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.

El objetivo final del Análisis será la generación de la documentación necesaria, por lo que se debe realizar una revisión técnica y realizar los informes con los problemas existentes y las mejoras propuestas que van a ser implementadas en la nueva versión del producto software.

Como resumen se relacionan los documentos que deben incluirse en el repositorio tras la fase de Análisis:

- Informes de Análisis de viabilidad para las PM 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_Especificación 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 la fase 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). Tendría la siguiente estructura:
    - Identificador de la versión
    - Fecha de salida
    - Funcionalidades que cubrirá, diferenciado las que provienen del MS y las del desarrollo
    - Elementos de configuración que incluye, diferenciado las que provienen del MS y las del desarrollo

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 sobre la base de la documentación generada en la fase de Análisis (Análisis detallado, Informe de requisitos, Identificación de los elementos afectados, Estrategia de pruebas y Plan de implementación).

El primer paso en la fase de diseño será identificar los módulos software 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.

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

En la documentación que se va generando en la fase de diseño, se tendrán que identificar y documentar los cambios que se realicen sobre los requisitos, y mantener al día una lista con las modificaciones que se van a llevar a cabo.

Como forma de asegurar el cumplimiento de los estándares definidos por la organización se deben ir verificando las modificaciones en la documentación y el desarrollo del proceso de diseño. Además, se ha de verificar la inclusión en el repositorio del nuevo material de diseño, sin olvidar los elementos de seguridad y robustez, cuidando que la documentación de pruebas haya sido actualizada.

Como último paso en la verificación se ha de comprobar el cumplimiento y coherencia de los requisitos en la fase de diseño.

La fase de diseño no se dará por finalizada 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 fase de implementación debería comenzar durante la fase de diseño para comprobar la factibilidad de los cambios propuestos. Es posible que el grupo de ingeniería no entienda completamente el impacto y magnitud de los cambios hasta que no finalice la fase de diseño, o puede que un cambio específico sea demasiado complejo de implementar.

El Jefe de mantenimiento 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 del sistema desarrollado y la accesibilidad a los usuarios.

Para asegurar la fiabilidad y mantenibilidad del sistema software, hay que tener presente desde las primeras fases del ciclo de desarrollo todas las características del producto, ya que las decisiones de diseño, sobre las modificaciones de módulos software, van a afectar la calidad del propio software.

3. Implementación

Para dirigir apropiadamente el esfuerzo en la fase de implementación será necesario disponer, como en las otras fases, de la documentación actualizada del proyecto y del sistema, del código fuente actual y los resultados de la fase de diseño. Otros elementos necesarios para el éxito de esta fase serán:

- Documentación de diseño y requisitos aprobados y controlados.
- Estándar de codificación acordado para ser utilizado por el Equipo de mantenimiento.
- Métricas de diseño que podrían ser aplicables en la fase de 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 fases anteriores y que serán aplicables en la fase de pruebas.

En esta fase, se van a seguir determinados procesos 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 procesos son:

- Codificación y pruebas de unidad.
- Integración.
- Revisión y análisis de riesgo.
- Revisión de la disponibilidad de pruebas.

Codificación y pruebas de unidad

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

Integración

Tras la codificación y las pruebas de unidad, o incluso durante la fase de 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 fase de Implementación, al igual que en las fases de Análisis y Diseño, la Revisión y análisis del riesgo será realizada periódicamente. Es lo más recomendado ya que un alto porcentaje de los riesgos y problemas de diseño, costos y rendimiento aparecen mientras se realiza la modificación del sistema.

Para cuantificar el análisis del riesgo, se utilizarán las métricas oportunas según los estándares. Llega a ser especialmente importante si el número de iteraciones de la codificación y las pruebas de unidad, así como la integración, están fuera del límite establecido al iniciar las modificaciones. Si es así, será necesario revaluar la factibilidad del diseño y/o modificación de los cambios solicitados, y volver a la fase de Diseño, Análisis o incluso de identificación y clasificación.

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 del configuración del software.
- Criterio para los resultados de las pruebas de sistema.
- Resultados de las pruebas a bajo nivel.

Cuando se esté en la fase 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 en la fase de implementación, y de esta manera aumentar la comprensibilidad del código.

Todas las pruebas de integración y de unidad deben estar documentadas, por lo que será necesario crear un archivo o carpeta en el repositorio, específico para el desarrollo y la documentación de las pruebas. Habrá que asegurarse que efectivamente estas pruebas y su documentación se realicen según los estándares definidos para el proyecto, además de comprobar que la documentación técnica y de usuario haya sido actualizada.

Durante las sucesivas revisiones del software y de las pruebas, como se ha mencionado anteriormente, pueden aparecer riesgos debido a los cambios que se van a producir en el software existente. Estos riesgos deben ser identificados y documentados e incluso resueltos cuando proceda.

Por último, el resultado de la fase de 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, de pruebas, de usuario y material de aprendizaje.

4. Pruebas de sistema

Las pruebas de sistema se realizan sobre un sistema modificado. Deberían ser realizadas por una organización independiente y siempre estar presente el cliente y el usuario final. La organización responsable de las pruebas de sistema deberá ser independiente de los desarrolladores y diseñadores del software, pero podrían utilizarse como recursos para el personal de pruebas.

Una parte de estas pruebas son las 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. Si se han realizado cambios al software o a los casos de prueba una vez que el producto ha sido liberado, será necesario realizar las pruebas de unidad y regresión durante la fase de Análisis para establecer la línea de referencia del producto, que son unos datos básicos utilizados como referencia en diversas métricas, de rendimiento, calidad, etc.

Para realizar las pruebas de sistema se dispondrá de toda la documentación generada en las fases anteriores que son:

- Informe de revisión de disponibilidad de pruebas.
- Plan de pruebas de sistema.
- Casos de pruebas de sistema.
- Procedimientos de pruebas de sistema.
- Manuales de usuario.
- Diseño.

Esta fase ha de llevarse como si se tratara de un sistema completamente integrado. Los procesos que han de realizarse durante las pruebas de sistema son:

- Pruebas de funcionalidad del sistema.
- Pruebas de interfaz.
- Pruebas de regresión.

- Revisión de la disponibilidad de pruebas de sistema para valorar el nivel de preparación para las pruebas de aceptación.

El cliente participará en la revisión para ver que la versión de mantenimiento está lista para empezar las pruebas de aceptación.

Las pruebas de sistema deberán ser conducidas como una función de pruebas independiente, o por la función de aseguramiento de calidad del software. Antes de finalizar esta fase, la función de pruebas será responsable de informar del estado del criterio que se ha establecido en el plan de pruebas para satisfacer la correcta finalización de las pruebas de sistema, es decir, especificar los criterios seguidos para decir que el sistema ha superado o no con éxito las pruebas.

Todo el código fuente, las PM y la documentación de pruebas, será puesta bajo la Gestión de configuración del software. Esta misma función será la que realizará el control de las versiones del software y todos los ficheros durante las pruebas de sistema.

Al acabar la fase de pruebas de sistema se tendrá un sistema completamente integrado y probado, junto a los informes de pruebas y el informe de revisión de la disponibilidad de pruebas.

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 sub-sistemas. 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.

Para comenzar las pruebas de aceptación, se asegurará disponer del informe de revisión de disponibilidad de pruebas, así como del sistema totalmente integrado. En las fases anteriores han de haberse preparado los planes para las pruebas de aceptación y los casos y procedimientos para realizar estas pruebas.

Los resultados de las pruebas realizadas antes del informe de revisión de disponibilidad de pruebas podrían ser utilizados por el cliente para reducir el alcance de las pruebas de aceptación. Si es así, el cliente deberá documentar en el informe de las pruebas de aceptación, los resultados que se tomaron de anteriores pruebas.

Lo primero que se ha de realizar son las pruebas de aceptación a nivel funcional, para proseguir con las pruebas de interoperabilidad y las de regresión. Antes de terminar las pruebas de aceptación, se deberá informar pertinentemente sobre el estado de los criterios que se han establecido en el plan de pruebas para llegar a comprobar que se han superado con éxito las pruebas. El cliente o su representante formará parte del grupo de revisión, al que se le informará de estos criterios para evaluar así el resultado de las pruebas, y asegurarse de que la versión de mantenimiento está lista para ser puesta en producción.

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. Para garantizar la correcta documentación de la versión se puede seguir el estándar IEEE 1042-1987. El artefacto que se actualizará en correspondencia es el Plan del producto 1.0 (Universidad de las Ciencias Informáticas, 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.

De igual forma que con las pruebas de sistema, para las siguientes versiones de mantenimiento es posible que se realicen otras pruebas de aceptación con el fin de satisfacer los requisitos de interfaz con otros sistemas o sub-sistemas y con el fin de validar que no se han introducido fallos como resultado de los cambios.

Mantenimiento no Planificable. Mantenimiento correctivo urgente

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. En este tipo de mantenimiento es necesario especificar, entre otras cosas, el tiempo de respuesta en que el equipo de desarrollo debe acudir a verificar o determinar el problema ante un fallo eventual según el tipo de equipamiento. Así mismo, se

tendrán en cuenta qué tipo de servicios abarcará y se delimitará claramente qué será provisto por el Equipo de mantenimiento y qué no.

La finalidad de este tipo de gestión es atender las necesidades del usuario de la aplicación de la forma más rápida posible dentro de las posibilidades del Equipo de mantenimiento y según la magnitud del problema. Es un mantenimiento desarrollado de una forma peculiar puesto que se realiza sin planificación previa y es utilizado para mantener operativo el sistema.

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 podrá 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.

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 tiene 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.

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, puede ser definido 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 en la fase de Análisis deben realizarse un grupo de pruebas, fundamentalmente de stress que permitan identificar las zonas críticas (GRUPO DE INVESTIGACIÓN ARQUISOFT, 2007). 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.

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, etc., sin cambiar las especificaciones iniciales.

También se seguirán en el caso del Mantenimiento preventivo las fases 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 para 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.

Mantenimiento Planificable. Mantenimiento adaptativo

Cuando el software ya está en explotación, puede que se produzca algún cambio en el software o el hardware del entorno en el que se ejecuta, este tipo de mantenimiento responde a esta situación. Estos cambios pueden deberse a:

- Cambio en el sistema operativo.
- Cambio del tipo de arquitectura en la que se ejecuta (red local a Internet/Intranet).
- 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 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 gran velocidad a la que está evolucionando el hardware, nuevos sistemas operativos (Linux), 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 Finalización del mantenimiento de software

En este punto se definen una serie de actividades descriptivas de lo que debería ser el proceso de cambio de versiones proporcionando así una guía para la realización del mismo.

La relación entre las actividades de esta etapa de Finalización se describe en la Figura 3.

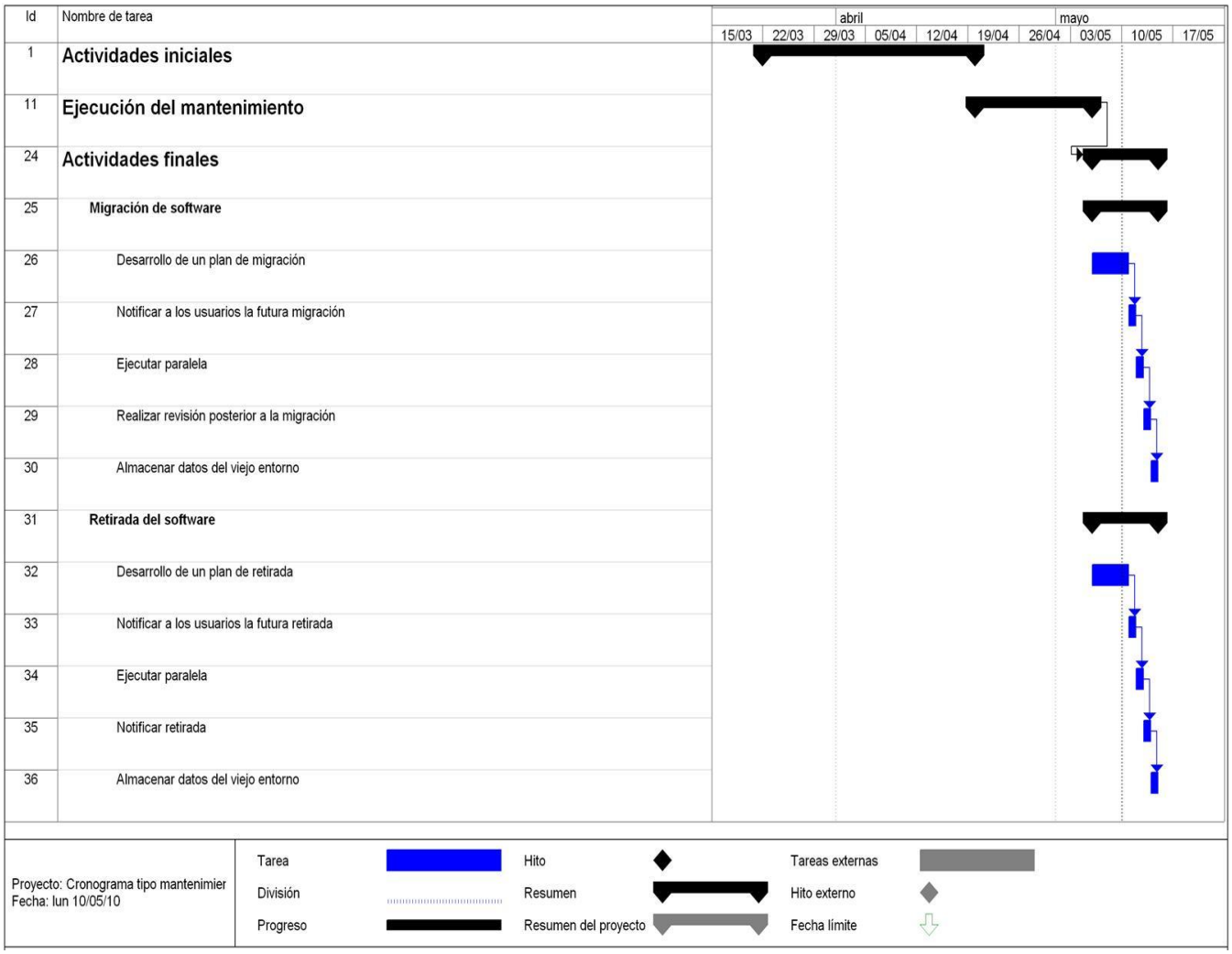

Figura 3: Relación entre las actividades de la etapa Finalización del MS.

1. Migración de software

Durante la vida de un sistema, puede que haya que modificarlo por diversas causas. Para migrar de un sistema a otro, el Equipo de mantenimiento necesita determinar las acciones necesarias para conseguir una migración adecuada y a partir de ahí desarrollar y documentar los pasos necesarios para efectuar la misma.

Desarrollo de un plan de migración

Esta actividad no es responsabilidad del Equipo de mantenimiento aunque puede estar apoyada por este. De cualquier modo se describen algunas premisas que pudiesen proporcionar una guía para el trabajo en los proyectos que no tienen un equipo de soporte claramente definido, el cual debería ser el encargado de la realización de esta actividad.

Como parte del Plan de migración se debe recoger:

- Descripción de los requisitos corregidos o incluidos.
- Organización de la capacitación en caso de que se sea necesario. Esto abarca la relación de todos los recursos necesarios para dar cursos y preparaciones necesarias. 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 mantenimiento) 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 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).

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 cuáles serán los problemas que se han solucionado o las nuevas funcionalidades incluidas.

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.

- 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.

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.

Almacenar datos del viejo entorno

Al estar bajo la Gestión de configuración del software 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. 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, ya que el esfuerzo que se ahorrará en mantenimiento puede ser enfocado en nuevas aplicaciones. El software antiguo puede tener un mantenimiento muy caro en términos de esfuerzo y puede que no funcione bien con otras aplicaciones más nuevas. Por lo mismo, el costo de actualización puede reducirse considerablemente si se elimina el software obsoleto y a este ahorro se puede agregar un mayor valor asociado generalmente al software de última generación, ya que introduce tecnologías y funcionalidades que aumentan la productividad notablemente.

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).

Notificar a los usuarios la futura retirada

Deberán 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.

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.

En caso de que sólo 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 Gestión de configuración del software.

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.

Almacenar datos del viejo entorno

Toda la documentación de desarrollo asociada: registros y código, deberá 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.

2.3 Conclusiones del capítulo

Se tuvieron en cuenta los procedimientos existentes en el GNCIS del centro para la realización del procedimiento. Se desarrolló el procedimiento propuesto para la fase de MS del sub-sistema Planificación. Se explicó paso a paso la composición del procedimiento, exponiendo las actividades y tareas por las que está constituido. Se propusieron métricas que ayudan a medir los principales indicadores a tener en cuenta a la hora de realizar el MS.

Capítulo 3: Validación del procedimiento.

3.1 Introducción

En el presente capítulo se realiza una validación teórica del procedimiento propuesto para el MS al sub-sistema Planificación, debido a la imposibilidad de ponerlo en práctica en el sub-proyecto Planificación esencialmente por dos razones: la primera, el factor tiempo y la segunda, con más influencia aún, el grado de terminación que presentan los componentes.

Con este objetivo se utilizó el método de consulta a especialistas, uno de los métodos subjetivos de pronosticación más usados, el cual consiste en la selección de varios especialistas, que dan su criterio sobre la utilidad de la propuesta presentada.

3.2 Proceso de validación

La consulta a especialistas es un método de pronosticación de un hecho o fenómeno. Esta pronosticación puede ser de dos tipos: pronóstico de previsión y pronóstico de predicción. En el de previsión, los elementos del fenómeno futuro son, en su mayor parte, conocidos, al contrario de lo que ocurre en el pronóstico de predicción, en el que son generalmente desconocidos, debiéndose por tanto determinar las características futuras del comportamiento del fenómeno. (Díaz, 2007)

La consulta a especialistas es un instrumento rápido y eficaz por el potencial que contiene para conformar, valorar y enriquecer criterios, concepciones, modelos, estrategias, metodologías, etc. Existen varias técnicas: encuestas, cuestionarios, entrevistas, estados de opinión, y Positivo-Negativo-Interesante.

En este caso para valorar la pertinencia del procedimiento propuesto se utilizó la técnica de cuestionarios.

3.2.1 Elección de los especialistas

Se entiende por especialista a una persona experimentada, que posee experiencia o habilidad en una actividad, capaz de ofrecer valoraciones conclusivas de un problema en cuestión y hacer

recomendaciones al respecto. En la investigación se entienden por especialistas a profesores que ocupan un cargo de liderazgo dentro del Departamento de Desarrollo de Productos del CEIGE.

Para la selección del grupo de los especialistas se realizaron las siguientes actividades:

- Determinación de las áreas del conocimiento que deben dominar: Partiendo del problema planteado en la introducción de este trabajo se determinó que los especialistas a consultar debían dominar las siguientes áreas del conocimiento: conocimiento de los principales estándares relacionados con el MS, trabajo en algún proceso de despliegue de un producto software, experiencia de liderazgo en proyectos de producción de software.
- Elaboración del listado de especialistas candidatos: Luego de determinar las áreas del conocimiento se elaboró un listado de especialistas candidatos; se seleccionaron personas que están vinculadas a la producción de software en la universidad y cuentan con los conocimientos necesarios para emitir una valoración. El listado inicial estaba conformado por diez especialistas.
- Selección de especialistas: Para seleccionar los especialistas se tuvieron en cuenta las siguientes cualidades: competencia, creatividad, disposición a participar, espíritu colectivista y autocrítico, capacidad de análisis, responsabilidad, sinceridad y seriedad. Estas características permiten que las opiniones recogidas sean confiables y válidas para valorar el procedimiento propuesto.

  Las características de los especialistas influyen decisivamente en la confiabilidad de los resultados obtenidos. Estas características son: calificación técnica, capacidad de emitir una decisión al respecto, conocimientos sobre el tema a evaluar, disposición a participar, entre otros. De los especialistas candidatos escogidos, seis estuvieron dispuestos a participar en el proceso de validación del procedimiento.

  Especialista # 1

  Nombre: Dailys Díaz Fuentes.

  Con el título de de Ingeniero en Ciencias Informáticas y cinco años de experiencia en proyectos de producción de SOFTWARE se desempeña como Jefa de soporte del Centro de Informatización para la Gestión de Entidades (CEIGE); trabajó anteriormente en el proyecto

SIGEP. Ha desempeñado los roles de Diseñador gráfico, Responsable de Calidad, Analista Principal, Responsable de Soporte. Cuenta con publicaciones en UCIENCIA, en la Serie Científica-UCI y en Fordes, además ha presentado trabajos en la Jornada científica.

Especialista # 2

Nombre: Joiser Bruzón Estrada.

Con el título de Ingeniero en Ciencias Informáticas y cinco años de experiencia en proyectos de producción de SOFTWARE se desempeña como Líder de la Línea Contabilidad y finanzas, perteneciente al proyecto ERP-Cuba. Anteriormente ocupaba el rol de Desarrollador en el proyecto de Contabilidad material.

Especialista # 3

Nombre: Meylin Martínez Chong.

Con el título de Ingeniero Informático y seis años de experiencia en proyectos de producción de SOFTWARE se desempeña como Líder de la Línea Logística, perteneciente al proyecto ERP-Cuba, y Jefe de Automatización en el UCID. Trabajó en varios proyectos de automatización de la Vice-rectoría de Informatización de la CUJAE hasta el 2005. Después ocupó el rol de diseñador en el proyecto Gestión de inventario v.1 en el MINFAR, siendo Jefa del proyecto para su versión dos. Cuenta con numerosas publicaciones en UCIENCIA en el presente año y dos años de experiencia en liderazgo de proyectos de producción de software.

Especialista # 4

Nombre: Saumel Tejeda Díaz.

Con el título de Ingeniero Informático y cinco años de experiencia en proyectos de producción de SOFTWARE se desempeña como Jefe del centro de Mando y dirección en el UCID. Trabajó en varios proyectos como Planificación material y financiera, Planificación de actividades, Gestión de inventarios y Control de recursos humanos del UCID, pasando por los roles de Desarrollador, Desarrollador de Interfaz de usuario y Líder de línea. Cuenta con publicaciones

en la Revista científica Militar. Es acreedor del Premio del Rector 2008 y cuenta con dos años de experiencia en liderazgo de proyectos de producción de software.

Especialista # 5

Nombre: Donel Vázquez Zambrano.

Con el título de Ingeniero en Ciencias Informáticas y cinco años de experiencia en proyectos de producción de SOFTWARE. Anteriormente trabajó de Arquitecto Principal en el Sistema de Gestión de Inventario y Almacenes para el Ministerio del Turismo y en la Plataforma de Comercio Electrónico B2B y de Arquitecto de Seguridad en el ERP-Cuba. De igual manera se ha desempeñado como Jefe de línea de Costos y Procesos, Jefe de línea de Contabilidad y Finanzas y Jefe de línea de Capital Humano; cargo que ocupa actualmente. Ha participado en eventos como: UCIENCIA 2008, III Taller Internacional de Gestión Financiera, I Taller Nacional de Interoperabilidad y Sistemas de Gestión, UCIENCIA 2010, I Taller Nacional PGDay. Cuenta con dos años de experiencia en liderazgo de proyectos de producción de software.

Especialista # 6

Nombre: Dionisdel Ponce Santana.

Con el título de Ingeniero en Ciencias Informáticas y cinco años de experiencia en proyectos de producción de SOFTWARE se desempeña como Líder de la Línea Planificación, perteneciente al proyecto ERP-Cuba; trabajó anteriormente en los proyectos: UCI-TV, Planificación material y presupuestada, Arquitectura UCI y Contabilidad. Ha desempeñado los roles de Diseñador de interfaz de usuario, Líder de desarrollo y Arquitecto de sistema. Cuenta con dos años de experiencia en liderazgo de proyectos de producción de software.

- Confirmar la participación de los especialistas: Una vez conformado el listado, se invitó personalmente a cada especialista elegido para participar en la evaluación. Allí se les explicó en qué consistía el trabajo en general, la propuesta a evaluar y el objetivo de la realización de los cuestionarios, así como el plazo de entrega. Una vez recibida la respuesta positiva, se estableció el listado final de los especialistas, informando a cada especialista su inclusión en el proceso a evaluar y las instrucciones necesarias para contestar las preguntas. Todos confirmaron su participación.

3.2.2 Elaboración del cuestionario

La elaboración del cuestionario tiene en cuenta los principios de la teoría de la comunicación y crea mecanismos que reducen los sesgos en las respuestas. Además, los objetivos que debería cumplir el procedimiento propuesto para su implantación en los demás sub-proyectos del proyecto ERP-Cuba. Se facilita que el especialista valore alternativas a sus respuestas y se solicita que exponga sus argumentos.

El cuestionario diseñado, mostrado en el Anexo IV, establece una serie de preguntas de enfoque investigativo que se centran principalmente en los principios básicos que debe cumplir la propuesta presentada, lo que permite observar la posibilidad real de que la misma pueda ser aplicada según las características actuales del proyecto. Además, brinda una evaluación general del proceso teniendo en cuenta una serie de requisitos. Estas preguntas proporcionan una mayor riqueza en las respuestas que son brindadas por los especialistas, y en todos los casos dan la posibilidad de emitir su criterio personal según su experiencia en proyectos de producción de software.

3.2.3 Desarrollo práctico

Para la evaluación del procedimiento se tuvieron en cuenta los siguientes criterios:

- Calidad de los procesos.
- Capacidad de los procesos en el análisis de factores esenciales en la ejecución de proyectos de software.
- Calidad 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 calidad de los instrumentos y artefactos, se evalúa 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 el cuestionario se solicita la evaluación de cada criterio en tres niveles: bajo, medio y alto.

Se les dio la posibilidad a los especialistas de modificar aspectos que ellos consideraban necesario cambiar y presentar su opinión general a favor o en contra del procedimiento propuesto, con la libertad de expresar todo lo que se pudo obviar en el cuestionario. Los especialistas recibieron la propuesta por vía del correo electrónico o entrega personal, además se les indicó el plazo de entrega de las respuestas del cuestionario sobre el procedimiento y la posibilidad de realizar preguntas que les surgieran al estudiar el documento.

El resultado de la evaluación puede verse en el Anexo VII, donde de manera general los especialistas opinaron que el procedimiento satisface las necesidades de la fase de MS debido a la claridad y reusabilidad de sus procesos, la adaptabilidad a diferentes tipos de productos y la integración al proceso de desarrollo de software.

3.2.4 Análisis de los resultados

En este apartado se analizarán los resultados para arribar a conclusiones sobre el nivel de aceptación del procedimiento.

En la Tabla 2 se muestra la correspondencia entre los criterios cualitativos y los valores cuantitativos que se usarán para la evaluación.

Criterio	Valores
Alta	3
Media	2
Baja	1

Tabla 1: Valores para evaluar cuantitativamente.

La siguiente tabla y gráficos muestran los resultados estadísticos de las evaluaciones.

Especialistas / Criterios	1	2	3	4	5	6	Promedio
Necesidad e importancia	3	3	3	3	3	3	3
Aplicabilidad	3	3	2	2	2	2	2,33
Claridad	3	3	2	3	3	3	2,83
Reusabilidad de los procesos	3	3	3	3	3	3	3
Adaptabilidad a los tipos de Producto	3	3	3	2	3	3	2,83
Integración al proceso de desarrollo de software de gestión	2	3	3	3	3	3	2,83
Adaptabilidad a diferentes escenarios según capacidad de los recursos humanos	2	3	2	3	3	2	2,5
Completitud	3	3	3	3	3	3	3
Claridad y precisión de los artefactos	3	2	2	2	3	3	2,5
Completitud de los artefactos	3	2	3	3	3	3	2,83
Adaptabilidad de los artefactos	2	3	3	2	2	3	2,5
Promedio	2.72	2.81	2,63	2,63	2.81	2.81	

Tabla 2: Muestra estadística del resultado de la valoración del procedimiento.

Criterios	N	Mínimo	Máximo	Media	Desviación típica
Necesidad e importancia	6	3,00	3,00	3,0000	,00000
Aplicabilidad	6	2,00	3,00	2,3333	,51640
Claridad	6	2,00	3,00	2,8333	,40825
Reusabilidad de los procesos	6	3,00	3,00	3,0000	,00000
Adaptabilidad a otros tipos de productos	6	2,00	3,00	2,8333	,40825
Integración al proceso de desarrollo	6	2,00	3,00	2,8333	,40825
Adaptabilidad a diferentes esenarios según la capacidad de los recursos humanos	6	2,00	3,00	2,5000	,54772
Completitud	6	3,00	3,00	3,0000	,00000
Claridad y precisión de los artefactos	6	2,00	3,00	2,5000	,54772
Completitud de los artefactos	6	2,00	3,00	2,8333	,40825
Adaptabilidad de los artefactos	6	2,00	3,00	2,5000	,54772

Figura 4: Tabla de estadísticas descriptivas.

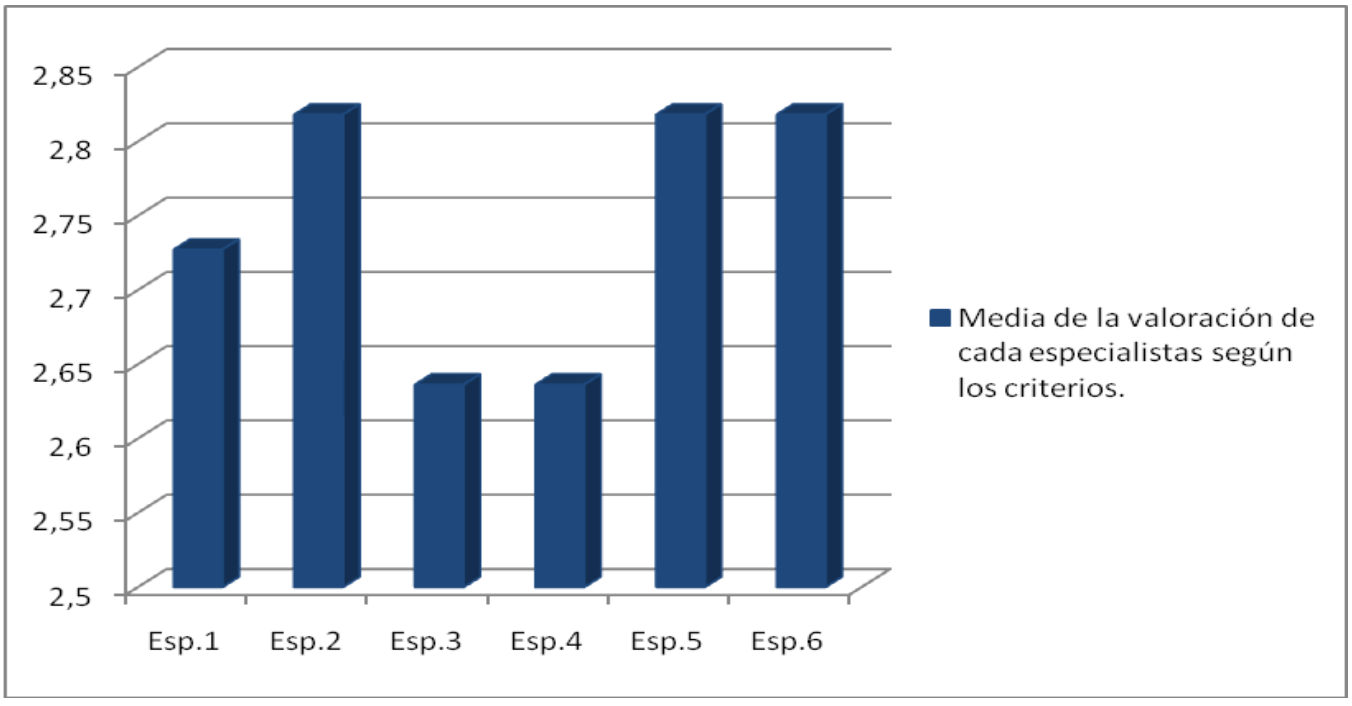

Figura 5: Promedio de evaluación por especialista.

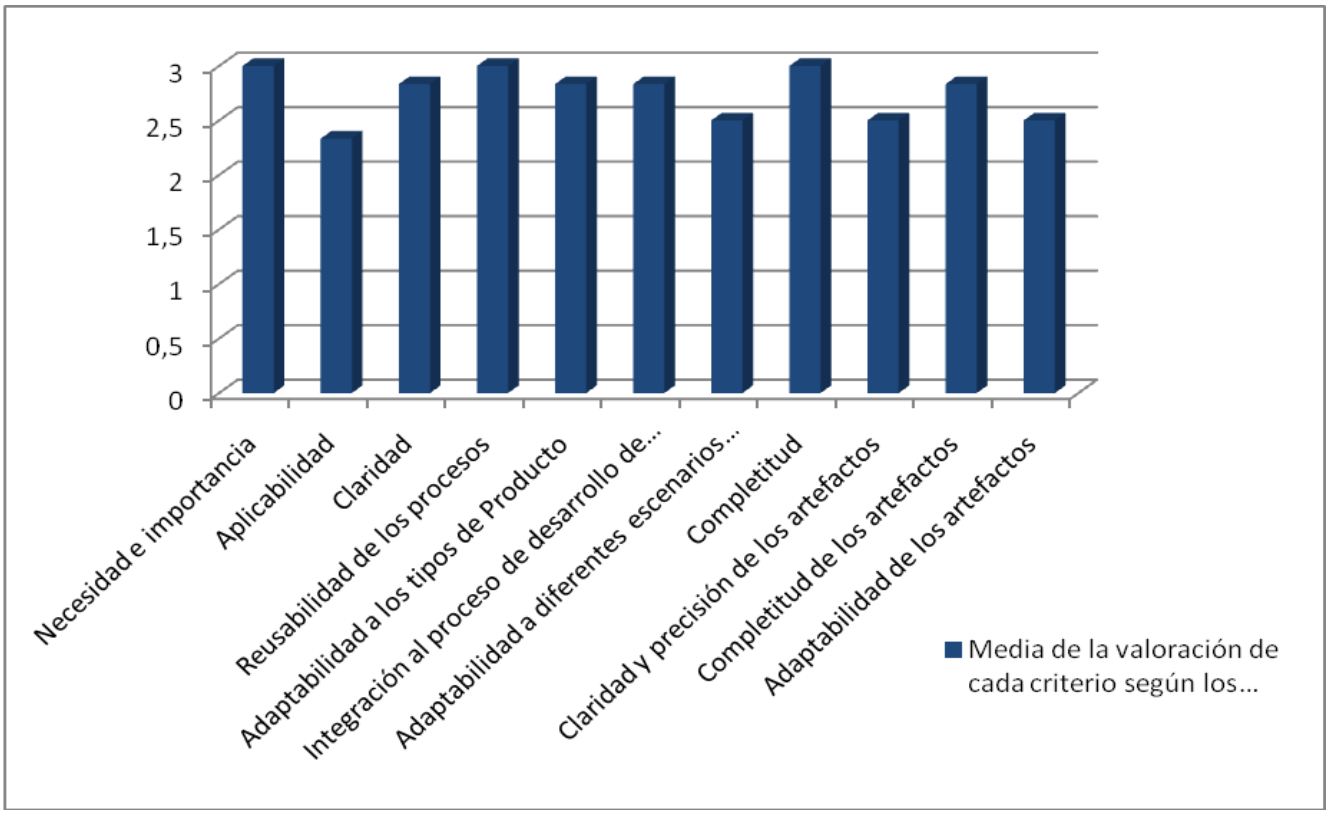

Figura 6: Promedio de evaluación por criterio.

Con el ochenta por ciento del valor máximo para cada criterio se asume que la evaluación es satisfactoria. Como el ochenta por ciento de 3 es 2.4 se puede decir que la mayoría de los criterios fueron evaluados satisfactoriamente y que todos los especialistas tienen una excelente opinión del procedimiento.

Algunos de los comentarios de los especialistas frente a la propuesta fueron los siguientes:

- “El procedimiento se centra en el ciclo de vida del producto por lo que puede ser aplicado en otros contextos.”
- “El estudio casuístico y la recopilación de datos se realizaron en el marco del proyecto.”
- “El procedimiento tiene en cuenta las actividades de MS desde la concepción del proyecto hasta el cierre del mismo, estimando tiempo y esfuerzo de las mismas en todo el ciclo de vida del producto.”

- “El procedimiento no depende del nivel de preparación o capacidad del personal, solo de los que lo aplican de forma tal que se orientan, dirigen y controlan las actividades de MS de acuerdo con lo establecido en el procedimiento.”
- “Cuenta con una correcta organización para la gestión y estandarización del proceso.”
- “Muestra una adecuada organización y claridad de los artefactos generados.”
- “Aporta facilidad al proceso de MS en cuanto a los roles y sus funciones.”
- “Garantiza un espacio para la planificación del MS, lo cual aporta más estabilidad a las actividades realizadas en esta fase.”

Las recomendaciones y sugerencias dadas por los especialistas en los cuestionarios sirvieron para incorporar mejoras a la propuesta inicial. El análisis de los resultados anteriores permite afirmar que de manera general el procedimiento para la gestión de las actividades de MS fue evaluado por los especialistas como útil, correcto y efectivo para su puesta en práctica. Esta evaluación permite concluir que con la aplicación del procedimiento propuesto se facilitará el desarrollo de las tareas de mantenimiento, en el sub-proyecto Planificación, en cuanto a la facilidad del proceso, el cumplimiento de los plazos de tiempo y la satisfacción del cliente.

3.3 Conclusiones del capítulo

Se realizaron todos los pasos que componen el proceso de validación del procedimiento, desde la elaboración del cuestionario hasta la aplicación del mismo a los seis especialistas seleccionados para este propósito. Los resultados obtenidos de las respuestas dadas por los profesionales demuestran de manera general, que el procedimiento de trabajo propuesto es considerado válido.

Conclusiones generales

En este trabajo se analizaron aspectos relacionados con el MS y la necesidad de disminuir los plazos de tiempo en que los proyectos dan soporte a determinado sistema.

Se desarrolló un estudio previo sobre los temas relacionados con el MS y los problemas que actualmente afectan esta fase en la producción de software. Después del análisis de varios modelos y estándares se decidió no utilizar ninguno de los estudiados debido a que no cubren las necesidades del sub-proyecto Planificación. Se desarrolló un procedimiento de trabajo para la gestión de las actividades de mantenimiento que incluye un ciclo de trabajo completo dividido en tres etapas. Se definieron las tareas, artefactos y roles expuestos, para cada etapa del ciclo de trabajo del procedimiento. Se propusieron métricas para la estimación de las medidas más importantes a tener en cuenta en la fase de MS. Se validó el procedimiento por un conjunto de seis especialistas, quedando avalado para ser usado en el sub-proyecto Planificación del proyecto ERP-Cuba.

Por todo lo anteriormente expuesto se cumplieron los objetivos propuestos para esta investigación.

Recomendaciones

Para dar continuidad a la presente investigación, se recomienda:

- Utilizar el procedimiento propuesto en este trabajo de diploma para la fase de MS de los sub-proyectos del departamento Desarrollo de Productos del CEIGE.
- Profundizar en el estudio de la fase de MS de los proyectos de la universidad, con el objetivo de mejorar la gestión de sus actividades, contribuyendo así a la disminución del tiempo destinado al MS.
- Perfeccionar el procedimiento propuesto enfocándolo al aseguramiento de calidad del software durante el proceso.
- Hacer extensiva la propuesta de este trabajo para las universidades y empresas cubanas productoras de software.

Bibliografía consultada

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

Antonio, Angélica de. 2001. *Gestión de Configuración.* Chile : s.n., 2001.

Brau Araujo, Ing. Marlenes y Yzquierdo Herrera, Ing. Raykenler. 2008. *Plan de Soporte.* 2008.

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

Cillero, Manuel. 2009. PROCESOS PRINCIPALES DE MÉTRICA VERSIÓN 3. [En línea] 2009. http://manuel.cillero.es/metrica-3/introduccion/procesos-principales.

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

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

ESCAPE, La tecla. 2008. Las actividades del ciclo de vida del software. [En línea] 2008. http://latecladeescape.com/w0/ingenieria-del-software/las-actividades-del-ciclo-de-vida-del-software.html.

España, Gobierno de. 2008. MÉTRICA VERSIÓN 3. Metodología de Planificación, Desarrollo y Mantenimiento de sistemas de información. [En línea] 2008. [Citado el: 15 de 11 de 2009.]

1987. *ESTÁNDAR IEEE 1042.* 1987.

Farach, Verónica. 2005. HERRAMIENTA DE GESTIÓN DE CAMBIOS E INCIDENCIAS. 2005.

Fuentes, Ing. Dailys Díaz. 2009. *Procedimiento para la gestión de incidencias.* 2009.

GMF: Módulo de gestión de incidencias. **Genos Open Source.** Barcelona : s.n.

Gutiérrez, J.J, y otros. 2006. MODELOS DE PRUEBAS PARA PRUEBAS DEL SISTEMA. 2006.

ISO 9126-3. 2006. Métricas Internas de la Calidad del Producto de Software. [En línea] 2006. http://www.mena.com.mx/gonzalo/maestria/calidad/presenta/iso_9126-3/.

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

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

Mora Castillo, Ing. Marisleidy y Diaz Fuentes, Ing. Dailys. 2009. *Procedimiento para la gestión de cambios.* 2009.

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

Piattini Velthuis, Mario G., y otros. 2007. *ANALISIS Y DISEÑO DETALLADO DE APLICACIONES INFORMATICAS DE GESTION.* 2007.

Piattini, Mario, Garzás, Javier y Cabrero, Daniel. 2006. *Técnica de Mejora del Mantenimiento Software Basada en Valor.* 2006.

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

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

Rojas, María Esther Ruilova. 2008. *Métricas del Producto para el Software.* 2008.

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

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

Saffirio, Mario. 2007. El Mantenimiento o la Administración de Cambios. [En línea] 2007. http://msaffirio.wordpress.com/2007/06/09/el-mantenimiento-o-la-adminisrracion-de-cambios/.

Scalone, Fernanda. 2009. *SACM y RDM en ITIL V3.* 2009.

Service Desk (HelpDesk - Gestor de Incidencias) Gestor de incidencias técnicas siguiendo las mejoras practicas ITIL. *Addlink. Software Científico.* [En línea] http://www.addlink.es/productos.asp?pid=542.

Sicilia, Migel Angel. 2009. *Métricas de Mantenibilidad Orientadas al Producto.* 2009.

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

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

Universidad de las Ciencias Informáticas. 2010. *0113_Especificación de Requisitos de Software.* 2010.

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

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

Wingerd, Laura. 2005. Practical Perforce. 2005.

Bibliografía citada

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

Cillero, Manuel. 2009. PROCESOS PRINCIPALES DE MÉTRICA VERSIÓN 3. [En línea] 2009. http://manuel.cillero.es/metrica-3/introduccion/procesos-principales.

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

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

ESCAPE, La tecla. 2008. Las actividades del ciclo de vida del software. [En línea] 2008. http://latecladeescape.com/w0/ingenieria-del-software/las-actividades-del-ciclo-de-vida-del-software.html.

España, Gobierno de. 2008. MÉTRICA VERSIÓN 3. Metodología de Planificación, Desarrollo y Mantenimiento de sistemas de información. [En línea] 2008. [Citado el: 15 de 11 de 2009.]

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

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

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

Piattini, Mario, Garzás, Javier y Cabrero, Daniel. 2006. *Técnica de Mejora del Mantenimiento Software Basada en Valor.* 2006.

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

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

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

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

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

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

Glosario de términos

Actividades: Conjunto de operaciones o tareas propias de una persona o entidad. Normalmente, se agrupan en un procedimiento para facilitar su gestión.

Analizabilidad: Característica de un producto software referente a su facilidad de análisis.

Artefactos: Productos tangibles del proyecto que son creados, modificados y usados por las actividades. Pueden ser modelos, elementos dentro del modelo, código fuente y ejecutables.

Cambiabilidad: Capacidad de un producto software para adaptarse a los cambios o, dicho de otra forma, facilidad para la aceptación de los cambios.

CASE (*Computer-Aided Software Engineering*): Conjunto de herramientas y métodos asociados que proporcionan asistencia automatizada en el proceso de desarrollo del software a lo largo de su ciclo de vida.

Ciclo de vida del software: ciclo que cubre las fases de desarrollo de un producto software.

Cliente: Persona, organización o grupos de personas que encargan la construcción de un sistema, ya sea empezando desde cero, o mediante el refinamiento de versiones sucesivas.

Defecto: Anomalía del sistema, por ejemplo un síntoma de un error en el software descubierto durante las pruebas, o un problema descubierto durante una reunión de revisión.

Despliegue: Transición del producto del entorno de desarrollo al usuario final.

Fase: Período de tiempo entre dos hitos principales de un proceso de desarrollo.

Gestión de configuración: Es el servicio que tiene por finalidad la implementación de una serie de mecanismos, procedimientos y tecnologías que le permitan a quien lo aplique organizar el proceso de generación de código y documentar un sistema dado.

Implantación: Puesta en marcha del sistema en el entorno donde será usado por los usuarios.

Incidencia: es cualquier evento que no forma parte de la operación estándar de un servicio y que causa, o puede causar una interrupción, o una reducción de la calidad del mismo.

Mantenibilidad: Es una propiedad del diseño del software relativa a su facilidad de mantenimiento.

Método: Conjunto de instrucciones a las que se les da un determinado nombre de tal manera que sea posible ejecutarlas en cualquier momento sin tenerlas que reescribir sino usando sólo su nombre.

Migración: elevar una versión de un producto software a otra de más alto nivel, o bien el movimiento de una arquitectura a otra, por ejemplo, de un sistema centralizado a otro con una estructura basada en el modelo cliente/servidor.

Petición de modificación: es la solicitud de cambio o problema detectado por el cliente, e informado al equipo de Soporte y que implica una modificación en el código del programa por lo que el Equipo de mantenimiento se encarga de solucionarlo.

Procedimiento: Un procedimiento es la acción de proceder o el método de ejecutar algunas cosas. Se trata de una serie común de pasos definidos, que permiten realizar un trabajo de forma correcta. En muchos casos los procedimientos se expresan en documentos que contienen el objeto y el campo de aplicación de una actividad; qué debe hacerse y quién debe hacerlo; cuándo, dónde y cómo se debe llevar a cabo; qué materiales, equipos y documentos deben utilizarse; y cómo debe controlarse y registrarse.

Proceso: Un proceso se define como un conjunto de tareas, actividades o acciones interrelacionadas entre sí que, a partir de una o varias entradas de información, materiales o de salidas de otros procesos, dan lugar a una o varias salidas también de materiales (productos) o información con un valor añadido.

Proceso de desarrollo de software: Conjunto total de actividades necesarias para transformar los requisitos de un cliente en un conjunto consistente de artefactos que representan un producto software y para transformar cambios en dichos requisitos en nuevas versiones del producto software.

Producto software: Está formado por el software y los manuales de usuario y de entrenamiento.

Proyecto: Esfuerzo de desarrollo para llevar un sistema a lo largo del ciclo de vida.

Prueba: Actividad en la cual un sistema o uno de sus componentes se ejecutan en circunstancias previamente especificadas, los resultados se observan y registran y se realiza una evaluación de algún aspecto.

Prueba de aceptación: Son las pruebas finales que se le hacen al sistema, para comprobar que esté listo y si es o no aceptado por el usuario.

Prueba de sistema: Se hacen cuando todos los módulos de un software están funcionando en conjunto. Está dirigida a verificar el programa final.

Requisito: Condición o capacidad que debe cumplir un sistema.

Retirada: Cese del soporte activo por parte del sub-proyecto Soporte, sustitución total o parcial por un nuevo sistema, o instalación de un sistema mejorado.

Sistema: Conjunto de elementos interrelacionados que trabajan juntos para obtener un resultado deseado.

Usuario: Humano que interactúa con un sistema.