UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MÉXICO

CENTRO UNIVERSITARIO UAEM TEXCOCO
INGENIERÍA EN COMPUTACIÓN

MANTENIMIENTO A LOS SISTEMAS DE INFORMACIÓN

TESINA

Que para obtener el Título de

Ingeniero en computación

Presenta

Luis Alberto Magallan Yacuta

Director de tesis

M en ISC Irene Aguilar Juárez

Revisor de tesina

Dr. Joel Ayala de la Vega
Dr. Alfonso Zarco Hidalgo

Texcoco, Estado de México, a 7 de Noviembre de 2016

RESUMEN

La finalidad de este trabajo es realizar un estudio sobre cómo se aplica el mantenimiento a los sistemas de información.

Actualmente la forma de darle mantenimiento a los sistemas de información (SI) es bastante controlada y metodológica, dependiendo de qué tipo de mantenimiento quiera o deba dársele al sistema. Las técnicas, métodos y procesos que se utilizan siempre tienen como finalidad el desarrollo de software de calidad.

Se explican de forma general que son los sistemas de información y cuáles son sus funciones en la actualidad, la importancia que representan en las corporaciones para el manejo de datos e información de gran importancia, de la misma forma hace mención a los tipos de sistemas más comunes, cuál es su función o para que fueron diseñados y cuál es la diferencia de otros sistemas, iniciando con los sistemas generales y los de una utilidad particular, también se hace una descripción de cómo es la estructura principal dentro de un SI, que sucede en cada fase y cuál es su función, posteriormente se describen como fueron creados mediante paradigmas de desarrollo y cuáles son los ciclos de vida de un sistema, desde un sistema clásico en cascada hasta los modelos más recientes diseñados por prototipos.

El manejo del sistema o la administración del mismo debe ser bastante detallada, se debe realizar documentación y registro de todo lo que es modificado desde su origen hasta la fecha en la que se encuentra funcionando, para esto existen varios métodos de medición; métricas, modelos, estándares y procesos. En este trabajo se describen los principales métodos que se utilizan para garantizar la calidad del software, los estándares de calidad que llevan a la mejora continua de un sistema de información, así como métodos estadísticos que en base a sus métricas nos indican cual es la calidad que actualmente se maneja en el sistema.

Cuando se elaboran planes para la estrategia de información, las organizaciones no pueden dejar de considerar que el mantenimiento de sistemas es la fase más prolongada y costosa del ciclo de

vida de los sistemas. Por eso también se hace mención a los tipos de mantenimiento que tienen una función específica para poder darle al sistema un mejor desempeño optimizando los procesos que tienen algún defecto o bien que no tienen defectos pero se puede realizar de una mejor forma, el de mantenimiento es continuo nunca termina hasta que el ciclo de vida del sistema concluye, por eso la importancia del mantenimiento que puede prevenir, corregir y mejorar siempre la forma de operar en un sistema de información.

DEDICATORIA

A mis padres

Irma Yacuta Campuzano y **Salvador Magallan Rojas**, por apoyarme siempre e inculcarme los valores que ahora me forman como una persona profesional, por estar siempre conmigo ante cualquier adversidad. Este logro es dedicado humildemente con toda mi admiración y respeto, puesto que ustedes al igual que yo forman parte del éxito que esto representa.

A mi familia

Para **Patricia, Leticia, Gabriela y Diego** que son parte de mí, por brindarme el ejemplo de que todo se puede con esfuerzo, por apoyarme en las buenas y en los momentos difíciles.

A Patricia Lopez

Por el apoyo incondicional que me brindaste para que pudiera realizar este trabajo y más que cualquier cosa, por estar conmigo siempre en los mejores momentos así como en lo que no lo son tanto.

AGRADECIMIENTOS

Agradezco profundamente a: La **Universidad Autónoma Del Estado de México** y al **Centro Universitario UAEM Texcoco** por ser orgullosamente la cuna de mi formación profesional.

Agradezco de manera cordial a mi Directora de Tesis**: MISC. Irene Aguilar Juarez** por habarme dado la oportunidad de recurrir a su experiencia en el ambito profesional, asu capacidad y amplios conocimientos, por brindarme al apoyo y la atención necesaria aun cuando las circunstancias en las que se desarrollo este trabajo fueron bastante dificiles.

Agradezco al **Dr. Alfonso Zarco Hidalgo** por ser revisor de este trabajo, por siempre ser amigo y maestro, por compartir sus conocimientos y apoyo para que pudiera terminar mis estudios profesionales.

Gracias a la ayuda del **DR. Joel Ayala de la Vega** por ser revisor de este trabajo y por todas las materias impartidas durante mi formación en el centro universitario.

CONTENIDO

Resumen i
Dedicatoria iii
Agradecimientos iv
Contenido v
Listado de imágenes 1
Introducción 2
Capítulo 1 Planteamiento del problema 4
1.1 JUSTIFICACIÓN 5
1.2 OBJETIVO 6
1.3 Objetivos Particulares 6
Capítulo 2 Sistemas de información 7
2.1 SISTEMA DE INFORMACIÓN 7
2.1.1 ACTIVIDADES DE UN SISTEMA DE INFORMACIÓN 8
2.2 TIPOS DE SISTEMAS DE INFORMACIÓN 12
2.2.1.3 SISTEMAS DE SOPORTE A LA TOMA DE DECISIONES EN GRUPO (GDSS 16
2.3 CICLO DE VIDA DE UN SISTEMA DE INFORMACIÓN 22
Capítulo 3 Métricas del software 31
3.1 Medidas y calidad de software 31
3.2 Mejoramiento de procesos de software a través de técnicas y métodos estadísticos 41
Capítulo 4 FASE DE MANTENIMIENTO EN LOS SISTEMAS DE INFORMACIÓN 51
4.1 Mantenimiento de los sistemas de información 51
4.1.2 Controladores de versiones para el desarrollo de código dentro del mantenimiento 61
Capítulo 5 Trabajos citados 69

LISTADO DE IMÁGENES

Figura 1 Actividades básicas de in SI (Soroka, 2012) 8
Figura 2 Modelo de vida clásico de un SI (Weitzenfeld, 2004) 23
Figura 3 Modelo en espiral (Weitzenfeld, 2004) 25
Figura 4 Niveles de madurez organizativos (Microsoft.com, 2015) 34
Figura 5 CMMI Niveles de funcionalidad practica (Weitzenfeld, 2004) 35
Figura 6 PEMM Niveles de integración (Weitzenfeld, 2004) 40
Figura 7 Elementos del PSP en el CMM (HUMPHREY, 2001) 45
Figura 8 Metodología d procesos (HUMPHREY, 2001) 46
Figura 9 Etapas del Six Sigma (Lean Solutions, 2011-2015) 48
Figura 10 Fase del mantenimiento en el ciclo de vida de un sistema (HUMPHREY, 2001) . 51
Figura 11 Esfuerzo sobre tipo de mantenimiento (HUMPHREY, 2001) 58
Figura 12 Diagrama de control de versiones local (--fast-version-control, 2016) 62
Figura 13 Diagrama de control de versiones centralizado (--fast-version-control, 2016) 63
Figura 14 sistemas de control de versiones distribuidos (--fast-version-control, 2016) 65

INTRODUCCIÓN

Un sistema de información nos permite organizar, administrar y distribuir, la información de una organización o entidad. Es el conjunto de partes interrelacionadas: hardware, software y personal informático, siendo un sistema de información en definitiva un conjunto de elementos orientados al tratamiento y administración de datos e información, organizados y listos para su posterior uso, generados para cubrir una necesidad.

Los sistemas informáticos cumplen con el desempeño para el que fueron diseñados y posteriormente desarrollados, pero están expuestos a muchos cambios, conectividad entre bases de datos, otros sistemas en diferentes plataformas, conectividad con otros servidores, actualizaciones de reglas del negocio, estas actualizaciones que deben hacerse sobre la marcha sin afectar la operación productiva del sistema en cuestión. Para efectuar estos cambios siempre existe personal de mantenimiento que se encarga de regular, desarrollar y reparar cualquier complicación que se presente en el sistema.

Es por eso que en cualquier sistema de información por mínimo que sea es necesario realizar algún mantenimiento. Llamamos mantenimiento a las modificaciones que se realizan en el producto software después de la entrega al usuario. Estas modificaciones podrán ser realizadas para mejorar el rendimiento, corregir defectos, adaptar el sistema o las aplicaciones utilizadas a un nuevo entorno, software o hardware, u otras propiedades deseables en el producto.

Considerando la importancia de la función de mantenimiento y su responsabilidad en las organizaciones se ha hecho necesario la implementación de equipos de soporte a los cuales se les asigna la responsabilidad del mantenimiento en donde deben hacer labor para que el sistema

maneje de forma eficiente y eficaz los métodos de operación que están de acuerdo con el desarrollo tecnológico evitando cualquier complicación para la operación productiva.

En este escrito se habla sobre los retos a los que se enfrentan los equipos de mantenimiento al recibir un requerimiento y la forma en la que se administra y mide la calidad del mantenimiento en cualquier aplicativo.

Capítulo 1

PLANTEAMIENTO DEL PROBLEMA

El mantenimiento es un aspecto más del desarrollo de sistemas de información. Sin embargo, efectuar cambios y ajustes no necesariamente indica la corrección de errores o la ocurrencia de problemas. Se pueden revisar los requerimientos de un sistema cómo consecuencia de su uso o del cambio de las necesidades de operación o quizás sea necesario corregir algún descuido que ocurrió durante el proceso de desarrollo. A menudo, surge la necesidad de capturar más datos y almacenarlos en la base de datos, o quizás sea necesario añadir características para la detección de errores con la finalidad de evitar que los usuarios del sistema emprendan por equivocación una acción no deseada. También existe el agregado de nuevas funciones, cómo mejoras en la seguridad y en los procesos, o la migración del sistema aun servidor más moderno o con mejor rendimiento, creación de nuevas reglas en la ejecución de procesos por cambios en los negocios. Todas estas y otras situaciones son realidades del mantenimiento de aplicaciones. Cuando se presentan, sin embargo, son un buen indicador de que el sistema se está utilizando, de que tiene una función útil, que cumple el objetivo para el cual fue diseñado y posteriormente desarrollado, de que los usuarios entienden las aplicaciones, pero también es un indicador de que existe la necesidad de adaptar, evolucionar y extender para todo esto es necesario el mantenimiento a los sistemas de información.

El área de mantenimiento debe estar alineado con las reglas del negocio, el impacto que cualquier cambio o falla en el sistema tiene sobre el negocio, cualquier afectación al sistema, las afectaciones que un cambio tiene sobre el ambiente productivo, la seguridad de la información, envió correcto de información, entre otras. Dependiendo del sistema al que se le está dando mantenimiento. Todo esto tiene como finalidad conseguir que los sistemas sean operativos el mayor tiempo posible y que, durante ese tiempo, funcionen de la manera más eficaz y con el máximo de seguridad para el personal que los utiliza.

Para esto nos planteamos ¿Cómo se le da el mantenimiento necesario a un sistema de información?, ¿A qué se enfrenta el personal de mantenimiento cuando ya existe un sistema en pleno funcionamiento?

1.1 JUSTIFICACIÓN

La presente investigación se realiza con la finalidad de encontrar cuales son las necesidades y problemas, además de los métodos más eficientes de mantenimiento en los sistemas de información. Es común que cuando la parte de mantenimiento comienza a ejercer dentro de un sistema, casi en todas las ocasiones este ya ha sido implementado al 100 % y se encuentra operando dentro de su respectiva organización adaptado a la forma y a la estructura que le dieron los desarrolladores, pero esto no garantiza que el sistema cumpla de forma eficaz con su trabajo o que posteriormente no se presenten modificaciones o incidentes que obliguen a realizar modificaciones sobre la marcha; el área de mantenimiento debe ejercer esta labor, apoyado por las metodologías y estándares de calidad de software necesarios para garantizar la funcionalidad del sistema de información.

Por los motivos anteriores se realiza esta investigación para identificar cómo se desempeña la fase de mantenimiento en un sistema de información, involucrando las metodologías y cómo se adaptan los modelos y los estándares de calidad de software para ejercer esta fase de una manera correcta y garantizar la seguridad de la información que se maneja así como la funcionalidad del sistema.

Por lo anterior comentado, al ser desarrollado este proyecto se está generando una investigación del funcionamiento empleado en la fase de mantenimiento de los sistemas de

información que puede ser tomada posteriormente por otras personas, identificando las problemáticas más comunes y esto traería consigo un sin fin de beneficios comunitarios para todos los elementos involucrados en ejercer el mantenimiento.

1.2 OBJETIVO

Elaborar un documento que nos ayude a identifica cómo se les da mantenimiento a los sistemas de información en la actualidad y las maneras más eficientes para llevar a cabo esta fase

1.3 Objetivos Particulares

1.- Identificar cómo se aplica de manera correcta la fase de mantenimiento a los sistemas de información.

2.- Identificar las formas de mantenimiento y los métodos que se utilizan para garantizar el buen funcionamiento de un sistema de información.

Capítulo 2

SISTEMAS DE INFORMACIÓN

Durante los últimos años los sistemas de información constituyen uno de los principales ámbitos en el área de organización de empresas. El entorno donde las compañías desarrollan sus actividades se vuelve cada vez más complejo. La creciente globalización, el proceso de internacionalización de las empresas, el incremento de la competencia en los mercados de bienes y servicios, la rapidez en el desarrollo de las tecnologías de información, el aumento de la incertidumbre en el entorno y la reducción de los ciclos de vida de los productos originan que la información se convierta en un elemento clave para la gestión, así como los sistemas que son capaces de manejarla. Un sistema de información forma parte esencial en el manejo y funcionamiento de cualquier organización y su evolución va de la mano con la evolución de los sistemas.

2.1 SISTEMA DE INFORMACIÓN

"Un Sistema de Información (S.I.) es un conjunto de procedimientos, manuales y automatizados, y de funciones dirigidas a la recogida, elaboración, evaluación, almacenamiento, recuperación, condensación y distribución de informaciones dentro de una organización, orientado a promover el flujo de las mismas desde el punto en el que se generan hasta el destinatario final de las mismas". **(Berzal, 2005-2006)**

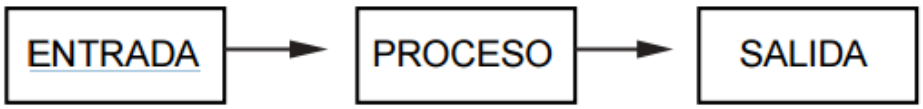
2.1.1 ACTIVIDADES DE UN SISTEMA DE INFORMACIÓN

Hay tres actividades en un sistema de información que producen la información que las organizaciones necesitan para tomar decisiones, controlar operaciones, analizar problemas y crear nuevos productos o servicios. Estas actividades son:

ENTRADA: captura o recolecta datos en bruto tanto del interior de la organización cómo de su entorno externo.

PROCESAMIENTO: convierte esa entrada de datos en una forma más significativa.

SALIDA: transfiere la información procesada a la gente que la usará o a las actividades para las que se utilizará.



Esquema del modelo de sistema.

Figura 1 Actividades básicas de in SI (Soroka, 2012)

Los sistemas de información también requieren retroalimentación, que es la salida que se devuelve al personal adecuado de la organización para ayudarle a evaluar o corregir la etapa de entrada.

Estas tres actividades de desglosan de la siguiente manera:

- **RECOLECCIÓN:**
  Esta función implica la captura y el registro de datos. Actúa cómo el órgano sensorio de la organización. Es una función costosa (con frecuencia es la más cara del sistema de información) y muy expuesta a la generación de errores, aunque este último aspecto está

siendo atenuado en grado creciente por la aplicación de nuevas tecnologías de captura de datos, cómo la lectura de caracteres ópticos o magnéticos y la lectura de código de barras. Un criterio que disminuye tanto los costos cómo los errores es el de capturar los datos tan cerca de la fuente (es decir, del lugar donde se generan) cómo sea posible.

- **CLASIFICACIÓN:**

  Esta función consiste en identificar los datos, agruparlos en conjuntos homogéneos, y ordenarlos teniendo en cuenta la manera en que será necesario recuperarlos. Vale decir que los datos se agrupan en estructuras diseñadas conforme a las necesidades del uso que se hará de ellos. El almacenamiento de datos en archivos computadorizados dispone de técnicas que han permitido alcanzar un elevado nivel de refinamiento en este sentido. Sin embargo, ya que el diseño del sistema de clasificación debe hacerse de acuerdo con la forma en que el usuario recuperará la información, tal diseño no puede ser adecuadamente definido si no se posee una clara comprensión de los procesos de decisión.

- **COMPRESIÓN:**

  La compresión es la función por la cual se reduce el volumen de los datos sin disminuir necesariamente la información que suministrarán a su destinatario; muy por el contrario, la compresión generalmente aumenta o hace más expresivo el contenido informativo de los datos. La compresión puede realizarse mediante varios métodos. Uno de ellos es la agregación, por el cual se van acumulando informaciones de detalle para obtener información consolidada de más alto nivel. Otro método es el filtrado. Mediante el mismo, se elimina información no significativa. Actúa como un cedazo o cernedor que sólo deja pasar los datos que tendrán valor para el destinatario. Un típico ejemplo es el de la información por excepción, en la que sólo se consignan los casos que se desvían de una norma, en lugar de informar todos los casos, incluso los que cumplen con la norma. Otro método de compresión es el uso de medidas estadísticas (tales como la media, la moda, la mediana, los cuartiles, el rango, etc.) que describen el comportamiento, real o pronosticado,

de variables probabilísticas. Es frecuente que sea más ilustrativo, por ejemplo, suministrar el promedio mensual de ventas de un año que la lista de las ventas de cada uno de los doce meses de ese año. Teniendo en cuenta que más importante que la existencia de información es la capacidad de procesarla, es fundamental que el sistema de información cuente con componentes que actúen cómo "compresores", es decir, proyectados para recibir más información de la que transmiten.

- **ALMACENAMIENTO:**

  Esta función se vincula con la conservación física de los datos y con su adecuada protección. Aunque no todos los datos que procesa un sistema de información se conservan en dispositivos de computación, éstos constituyen el soporte prácticamente obligado del banco de datos de las organizaciones. Aun en las empresas de mayor envergadura en el mundo, la tecnología de computación disponible permite una capacidad virtualmente ilimitada para mantener este banco de datos en condiciones de ser consultado en forma inmediata. En materia de archivos computadorizados, la teoría y la práctica del diseño, la generación, el mantenimiento, la reorganización y la consulta de las estructuras de datos han alcanzado un alto grado de sofisticación y eficiencia. Cómo una definición general, puede decirse que se denomina "base de datos" a un conjunto de archivos que responde a la aplicación de herramientas lógicas orientadas específicamente al logro de esa eficiencia. A través de la función de almacenamiento, el sistema de información hace las veces de memoria de la organización. Al mismo tiempo, la permanente puesta al día de esa memoria convierte a la base de datos, mediante un modelo simbólico descriptivo, en la imagen actualizada de la organización.

- **RECUPERACIÓN:**

  Esta función tiene el propósito de suministrar el acceso a la base de datos. Cómo se dijo más arriba, depende de un apropiado sistema de clasificación. Cada día están más difundidas las aplicaciones de computación en las que la recuperación de los datos (y, muchas veces, su actualización) debe hacerse en tiempo real, es decir, en el mismo momento en que sucede el hecho que genera la necesidad de la recuperación o la

actualización. En estos casos, la computadora interviene en alguna parte de la ejecución de la propia transacción que demanda el uso o actualización de los datos.

- **PROCESAMIENTO**:

  El sistema de información (cómo todo sistema) es un transformador de entradas en salidas a través de un proceso. Esta transformación se realiza mediante cómputos, clasificaciones, cálculos, agregaciones, relaciones, transcripciones y, en general, operaciones que, no importa qué recursos humanos o tecnológicos empleen, persiguen el objetivo de convertir datos en información, es decir, en datos que habrán de tener valor y significado para un usuario. La función de procesamiento implica, principalmente, la modificación de la base de datos para mantenerla actualizada.

- **TRANSMISIÓN**:

  Esta función comparte la comunicación entre puntos geográ- ficos distantes, sea por el traslado físico del sostén de los datos (papeles, dispositivos de archivos computadorizados, cintas de audio o video, microfichas, etc.) o por la transmisión de señales (comunicación entre equipos de computación, transmisión de facsímiles, teléfono, etc.). Este aspecto del sistema de información se vincula con la tecnología de comunicaciones, la que se halla tan asociada con la de la computación, e igualmente tan desarrollada, que resulta muy difícil trazar una línea de separación entre ellas. De ahí que suele aplicarse la denominación de telemática a la disciplina o ambiente tecnológico que surge de la combinación de las telecomunicaciones y la informática. Las facilidades disponibles para transmitir datos entre distintos puntos físicos, así como la amplísima gama de capacidades de equipos de computación, permiten descentralizar los recursos de computación y las bases de datos. Esto puede hacerse sin caer en costosas redundancias ni perder la integración de sistemas y archivos, ya que todos los puntos pueden estar interconectados, compartiendo recursos y datos, y manteniendo similares grados de actualización de las bases de datos. Así, se conforman las llamadas redes de procesamiento distribuido, mediante las que se lleva la "inteligencia" de computación al mismo lugar en que se la necesita, sin caer en los costosos aislamientos de la descentralización sin comunicación. Además, las posibilidades de transmisión de datos a través de redes de comunicaciones tienen un impacto fundamental

en el planteo estratégico de las empresas y están produciendo cambios trascendentales en la naturaleza y la operación de los negocios.

- **EXHIBICIÓN:**

  Mediante esta función, se proporciona una salida de información preparada de modo tal que resulte legible y útil a su destinatario. En un sistema de información basado en el uso de computadoras, esta función es la que implica la interfaz con el ser humano. Todas las funciones descriptas hasta aquí realizan diversos tratamientos de la información, pero no producen resultados visibles para el usuario. De ello se encarga esta función de exhibición, la que expone la información en forma impresa, en una pantalla de representación visual o en otros dispositivos. La presentación de los resultados tiene particular importancia para que los mismos revistan el carácter de información, para que aparezcan con significado ante los ojos del usuario, para que reduzcan la ignorancia del mismo, y para que lo induzcan a la acción. En la mayor parte de los sistemas de información ineficientes, el problema central no reside en la ausencia de información, sino en el ocultamiento o enmascaramiento de la misma bajo una maraña de datos en las que el usuario debe "hurgar" para encontrar aquellos que, para él, constituyen información. Esto pone en evidencia la importancia de la función de compresión, por un lado, y la de la precisa determinación de las necesidades informativas de cada puesto de la organización, por el otro**. (Soroka, 2012)**

2.2 TIPOS DE SISTEMAS DE INFORMACIÓN

2.2.1.1 SISTEMAS PARA EL PROCESAMIENTO DE TRANSACCIONES (TPS)

Es un sistema básico de contabilidad y mantenimiento de registros que hace un seguimiento de las transacciones diarias rutinarias necesarias para dirigir el negocio.

Históricamente, los sistemas de información transaccionales fueron los primeros (y, durante muchos años, casi los únicos) en ser incorporados al procesamiento computadorizado. En el contexto de los sistemas de información, una transacción es un intercambio entre un usuario que opera una terminal y un sistema de procesamiento de datos, en el que se concreta un determinado resultado. Implica la captura y validación de los datos ingresados por el usuario, la consulta y/o actualización de archivos, y una salida o respuesta. Esta definición connota en la transacción su carácter de operación individual, relativamente breve e indivisible.

Los sistemas de información transaccionales, por lo tanto, están destinados a satisfacer las necesidades del nivel operativo: explotan la capacidad y velocidad de las computadoras para almacenar y procesar grandes volúmenes de datos; realizan operaciones repetitivas y relativamente sencillas; y contribuyen a automatizar las tareas más rutinarias y tediosas, a eliminar el "papeleo", a acelerar los trámites, a disminuir la cantidad de mano de obra, a minimizar los errores, a facilitar la registración y recuperación de datos desagregados y, en general, a reducir o aligerar las actividades que desarrollan los empleados u operarios de las organizaciones. Los sistemas transaccionales son conocidos también con las siglas TPS (Transaction Processing Systems), y cuando el procesamiento se realiza en tiempo real (es decir, cuando el procesamiento de los datos es simultáneo a los hechos) se los conoce cómo OLTP (On Line Transaction Processing).

En este tipo de sistemas, se encuentran los que son prácticamente comunes a todas las organizaciones, tales como los de contabilidad, facturación, inventarios, ventas, proveedores, cuentas corrientes, cobranzas, caja, bancos, sueldos, finanzas, compras, planeamiento y control de la producción, etc. También pertenecen a esta clase muchos otros sistemas (llamados "sistemas para mercados verticales") que resultan más específicos de una rama de actividad, cómo, por ejemplo, Administración de Obras Sociales, Administración de Sistemas de Medicina Prepaga,

Administración de AFJP, Servicios Financieros, Reserva de Pasajes, Administración Hospitalaria, Administración Hotelera, Administración de Propiedades, Administración de Instituciones Educativas, Producción de Seguros, etc. Si no para todos, para la mayoría de estos sistemas existe una variada oferta de paquetes de programas estandarizados. Los más numerosos son los diseñados para las organizaciones más pequeñas, y su costo, su grado de estandarización y su sencillez de manejo los hacen muy accesibles, así como aptos para su empleo con los más económicos modelos de computadoras personales. En el otro extremo, se encuentran las versiones más potentes y costosas, las que suelen tener mayores exigencias de implantación; generalmente, requieren personal especialmente entrenado, recursos de computación relativamente caros y sofisticados, y la adaptación de los programas a las necesidades particulares de la organización. Sobre todo en el caso de esta categoría superior de paquetes, se plantea la alternativa estratégica de optar por estas soluciones de terceros o encarar el desarrollo de sistemas "a medida", es decir, especialmente diseñados y construidos para la organización en que serán utilizados.

2.2.1.2 SISTEMAS DE SOPORTE PARA LA DECISIÓN (DSS)

Decisión Support System: Ayuda a un director a tomar decisiones semiestructuradas, cómo la planificación de un presupuesto y la previsión de ventas, y decisiones no estructuradas, cómo el desarrollo de un nuevo producto y la negociación de un contrato.

Los sistemas de apoyo a la toma de decisiones, o para mayor sencillez, Sistemas de Apoyo a la Decisión (SAD), son sistemas computadorizados, casi siempre interactivos, que están diseñados para asistir a un ejecutivo en la toma de decisiones. Los SAD (también conocidos como DSS, del inglés, Decision Support Systems) incorporan datos y modelos para ayudar a resolver un problema que no está totalmente estructurado. Los datos suelen provenir de los sistemas transaccionales o de un repositorio de datos (conocido cómo data warehouse, concepto que se explicará más adelante), y/ o de alguna fuente o base de datos externa. Un modelo puede ser desde un sencillo

análisis de rentabilidad realizado con nuestra familiar planilla de cálculo, en el cual se calcula un probable resultado (beneficio o pérdida), hasta un modelo complejo de optimización de carga de máquinas de una línea de producción que requiere un complejo programa de base matemática.

Consideremos la popular planilla de cálculo (Excel o Lotus), aplicada a un estado financiero proyectado. En este caso, se utilizan datos históricos y supuestos acerca de la futura tendencia en los ingresos y los egresos. Después de evaluar los resultados del modelo base, el ejecutivo genera lo que se denomina un análisis "¿qué pasaría si...?" ("what if..." analysis), modificando uno o más supuestos para analizar el impacto de los mismos en el resultado final. Por ejemplo, puede explorar la forma en que el flujo de ingresos afecta el crecimiento pronosticado de las ventas, ya sea en un porcentaje superior o inferior al actual. De la misma manera, puede explorar cómo afecta el cambio en el costo de las materias primas, el cambio de la paridad cambiaria, las modificaciones en la tasa de interés, etc

Objetivos de un SAD/ DSS

Los objetivos son los siguientes:

• Apoyar (no reemplazar) el juicio humano, de tal modo que el potencial de los procesos del hombre y de la máquina sea utilizado al máximo.

• Crear herramientas de apoyo bajo el control de los usuarios, sin automatizar la totalidad del proceso decisorio predefiniendo objetivos o imponiendo soluciones.

• Ayudar a incorporar la creatividad y el juicio del tomador de decisiones en las fases de formulación del problema, selección de los datos, y generación y evaluación de alternativas.

• Apoyar a los ejecutivos de alto nivel en la solución de problemas prácticos no totalmente estructurados y en los que, hallándose presente algún grado de estructura, el juicio sea esencial.

Componentes de un SAD Los subsistemas principales de un SAD son: la base de datos, la base de modelos, el generador de diálogo (o interfaz con el usuario) y el tomador de decisiones.

El sistema de administración de modelos es el que constituye la característica distintiva de los SAD. La modelización es la función fundamental de todo SAD, ya que permite crear modelos y

escenarios que representen situaciones reales. Esos escenarios ayudan al gerente a explorar alternativas y examinar las consecuencias de su decisión, antes de ponerla realmente en práctica.

En resumen, en general los SAD tienen las siguientes características:

• Se enfocan en procesos de decisión y no en procesamiento de transacciones.

• Se implantan y modifican rápidamente.

• Suelen ser construidos por los propios usuarios utilizando herramientas muy difundidas, cómo son las planillas electrónicas (Excel, Lotus).

• Aportan información útil para la toma de decisiones, pero ésta finalmente es responsabilidad del ejecutivo.

2.2.1.3 SISTEMAS DE SOPORTE A LA TOMA DE DECISIONES EN GRUPO (GDSS)

Ayuda a que la toma de decisiones sea más eficaz para todos los niveles de usuarios individuales. Ofrecen muchas herramientas útiles para el trabajo en grupo. Permiten que los documentos compuestos incluyan aplicaciones de diferentes compañías de software. El SW de GDSS, ayuda a la programación, comunicación y administración conjunta de grupos de trabajo.

Características:

- Diseño especial
- Facilidad de uso
- Flexibilidad
- Apoyo a la toma de decisiones
- Aportaciones anónimas
- Reducción del comportamiento negativo del grupo

- Mantenimiento de registros automáticos

Elementos:

- Base de datos
- Base de modelos
- Gerente de diálogo

Alternativas:

- Salón de decisiones
- Red de decisiones de área local
- Tele conferencias
- Red de decisión de área extensa

2.2.1.4 SISTEMAS DE TRABAJO CON CONOCIMIENTOS (KWS)

Los sistemas de oficina y las estaciones de trabajo de diseño. Su principal cometido es integrar los conocimientos en el conjunto de la organización y canalizar los flujos de información asociados a puestos intensivos en información. Son denominados K.W.S (Knowledge Work Systems).

Cómo se observa el conocimiento cómo activo intangible es difícil de administrar, incluso apenas se está comprendiendo cómo se puede administrar; el conocimiento es parte fundamental para las organizaciones y su forma de hacer negocios y tener ventajas competitivas. Dichas herramientas nos ayudan a identificar o clasificar nuestros activos de conocimiento y a llevar los procesos de su gestión de una manera más efectiva, todas las habilidades y los conocimientos deben de ser identificados y valorados, accesibles desde cualquier sitio, deben ser capturados o almacenados, para que a su vez se puedan desarrollar y mejorar.

Para CREAR conocimiento: Knowledge Work Systems (KWS), apoyan las actividades de los empleados y profesionistas de alto desempeño y los ayudan a crear nuevos conocimientos e integrarlos a la empresa (CAD, sistemas de modelación y simulación).

Para COMPARTIR conocimiento: Grupos de trabajo, donde se comparte el conocimiento, este puede ser presencial o a distancia (e-mail, teleconferencias, groupware).

Para DISTRIBUIR conocimiento: Office Automation Systems, ayuda a controlar el flujo de información a través de la organización (procesamiento de datos, calendarios electrónicos).

2.2.1.5 SISTEMAS DE AUTOMATIZACIÓN DE OFICINAS (OAS)

Es una aplicación de Tecnología de información diseñada para aumentar la productividad de los trabajadores de datos en la oficina, apoyando las actividades de coordinación y comunicación de la oficina típica.

Coordinan a diversos trabajadores de información, unidades geográficas y áreas funcionales. Manejan y controlan documentos. Programan actividades. Comunican.

Sirven a las necesidades de información en los niveles de conocimientos en la institución:

- Coordinan y administran
- Enlazan el trabajo
- Acoplan a la institución

Para cumplir con las funciones ya descritas, las oficinas en general llevan a cabo cinco actividades de oficinas principales:

Administración de documentos:

- Programación de las actividades de las personas y grupos
- Comunicación con personas y grupos
- Administración de los datos
- Administración de proyectos

Administración de Documentos

Son las tecnologías que se utilizan para crear, procesar y administrar documentos. (Procesamiento de palabra, las publicaciones de escritorio, imágenes de documentos y administración del flujo de trabajo).

- **Trabajo de Colaboración (Groupware)**

Es el software que reconoce el significado de los grupos en las oficinas al proporcionar funciones y servicios que dan soporte a las actividades de colaboración de los grupos de trabajo.

- **Administración de la Información (Base de Datos de Escritorio**)

Herramienta en paquetes para bases de datos diseñadas para dar soporte a tareas de administración de datos específicos de la oficina para el trabajador de la información.

- **Administración de Proyectos**

Es el software que facilita el desarrollo, programación y administración de un proyecto complejo en subtareas más sencillas, cada una con su propio tiempo de terminación y sus requerimientos de recursos.

2.2.1.6 SISTEMAS DE INFORMACIÓN PARA LA ADMINISTRACIÓN (SIA)

(MIS Management Information System) son un conjunto organizado de personas, procedimientos, software, bases de datos y dispositivos para suministrar la información rutinaria a administradores y tomadores de decisiones.

Proporcionan informes periódicos para la planeación, el control y la toma de decisiones. Son sistemas que se sustentan en la relación que surge entre las personas y las computadoras. Su interés principal es la eficiencia operativa.

Objetivo

Ofrecer a la administración la información necesaria de manera habitual y continua.

No sólo ofrece datos, sino el conjunto de éstos analizados y procesados.

Ayudar en el proceso de planeación cómo una herramienta en el desarrollo de estrategias para dar ventajas competitivas a la empresa.

Disminuir la necesidad de dependencia de un ejecutivo en el mecanismo de control en una empresa.

Permitir una comunicación más lateral y cruzada sobre una base formal en una organización.

Y principalmente, dar soporte en la toma de decisiones en los altos mandos administrativos de una organización mediante el uso de la información recabada.

2.2.1.7 SISTEMAS DE INFORMACIÓN ESTRATÉGICOS

Son los que ayudan a los administradores del nivel superior (o alta gerencia) a abordar y resolver cuestiones estratégicas y tendencias a largo plazo, tanto en la compañía cómo en su entorno exterior.

Características:

Suelen desarrollarse “in house”, es decir, dentro de la organización, por lo tanto no pueden adaptarse fácilmente a paquetes disponibles en el mercado.

Su forma de desarrollo es la base de incrementos y a través de su evolución dentro de la organización. Se inicia con un proceso o función en particular y a partir de ahí se van agregando nuevas funciones o procesos.

Apoyan en el proceso de innovación de productos y proceso dentro de la empresa Cambian significativamente el desempeño de un negocio al medirse por uno o más indicadores clave, entre ellos, la magnitud del impacto.

Contribuyen al logro de una meta estratégica.

Generan cambios fundamentales en la forma de dirigir una compañía, la forma en que compite o en la que interactúa con clientes y proveedores.

Su función es lograr ventajas que los competidores no posean, tales como ventajas en costos y servicios diferenciados con clientes y proveedores.

2.2.1.8 ESTRUCTURA DE UN SISTEMA DE INFORMACIÓN

Whitten, Bentley y Barlow 1996 proponen un modelo basado en cinco bloques elementales para definir un sistema de información, personas, actividades, datos, redes y tecnología: **(Whitten, 1996)**

Personas: Engloba a los propietarios del sistema (entendiendo cómo tales aquellas personas que patrocinan y promueven el desarrollo de los sistemas de información), a los usuarios (directivos ejecutivos, directivos medios, jefes de equipo, personal administrativo) a los diseñadores y a los que implementan el sistema.

Datos: constituyen la materia prima empleada para crear información útil.

Actividades: se incluyen las actividades (procesos) que se llevan a cabo en la empresa y las actividades y las actividades de proceso de datos y generación de información que sirven de soporte a las primeras

Redes: se analiza la descentralización de la empresa y la distribución de los restantes bloques elementales en los lugares más útiles (centros de producción, oficinas, delegaciones) así como la comunicación y coordinación de dichos lugares

Tecnología: Hace referencia tanto al hardware cómo al software que sirven de apoyo a los restantes bloques integrantes del sistema de información. **(Soroka, 2012)**

2.3 CICLO DE VIDA DE UN SISTEMA DE INFORMACIÓN

El ciclo de vida de un sistema de información es un enfoque por fases del análisis y diseño que sostiene que los sistemas son desarrollados de mejor manera mediante el uso de un ciclo especifico de actividades del analista y del usuario.

Según James Senn, existen tres estrategias para el desarrollo de sistemas: el método clásico del ciclo de vida de desarrollo de sistemas, el método de desarrollo por análisis estructurado y el método de construcción de prototipos de sistemas. Cada una de estas estrategias tiene un uso amplio en cada una de los diversos tipos de empresas que existen, y resultan efectivas si son aplicadas de manera adecuada. **(Senn, 2015)**

2.3.1.1 MODELO DE CICLO DE VIDA CLÁSICO

El modelo de cascada clásico data de la década de los 60s y 70s (Royce 1970, Boehm 1981). El modelo de cascada se define como una secuencia de actividades a ser seguidas en orden, donde la estrategia principal es definir y seguir el progreso del desarrollo de software hacia puntos de revisión bien definidos (“milestones” o “checkpoints”). En su época de esplendor, este modelo tuvo gran aceptación en la comunidad de contratistas gubernamentales estadounidenses, ya que éstos recibían sus pagos del gobierno en base a entregas basadas en horarios (“schedule”) predefinidos. El desarrollo de software implicaba una secuencia de actividades a realizarse y cuyo seguimiento era verificar que cada actividad haya sido completada. La ejecución del modelo era muy lineal, por lo cual el modelo fue sencillo y atractivo; donde se especificaba las actividades para luego hacerlas de principio a fin. Se consideraba que una vez terminada una actividad se continuaba con la siguiente. La Figura muestra un diagrama conceptual del modelo describiendo el orden a seguir de las actividades del desarrollo de software. No se muestra una etapa explícita de “documentación” dado que ésta se llevaba a cabo durante el transcurso de todo el desarrollo.

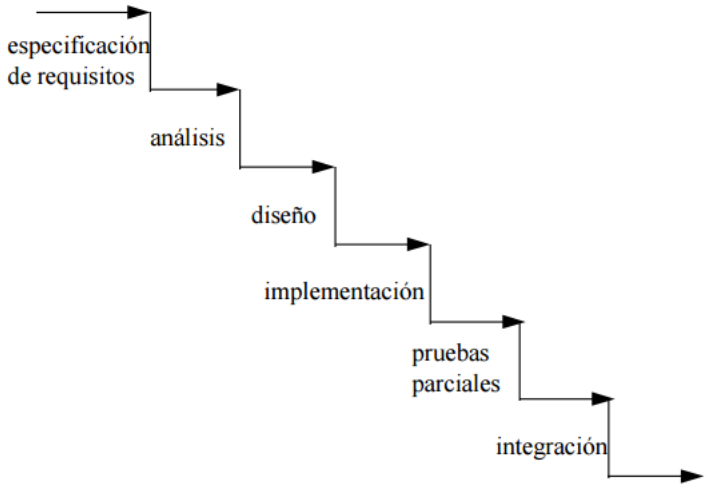


Figura 2 Modelo de vida clásico de un SI (Weitzenfeld, 2004)

2.3.1.2 Modelo Espiral

El modelo de espiral es una modificación al modelo de cascada desarrollado durante la década de los 80s (Boehm1988). El modelo de espiral se basa en una estrategia para reducir riesgo, al contrario del modelo de cascada que es dirigido por documentos. Cómo parte del manejo de riesgo el modelo incorpora una estrategia de uso de prototipos, algo muy aceptado en la actualidad. El modelo enfatiza ciclos de trabajo, cada uno de los cuales estudia el riesgo antes de proceder al siguiente ciclo. Cada ciclo comienza con la identificación de los objetivos para una parte del producto, formas alternativas de lograr los objetivos, restricciones asociadas con cada alternativa, y finalmente procediendo a una evaluación de las alternativas. Cuando se identifica incertidumbre, se utilizan diversas técnicas para reducir el riesgo en escoger entre las diferentes alternativas. Cada ciclo del modelo de espiral termina con una revisión que discute los logros actuales y los planes para el siguiente ciclo, con el propósito de lograr la incorporación de todos los miembros del grupo para su continuación. La revisión puede determinar si desarrollos posteriores no van a satisfacer las metas definidas y los objetivos del proyecto. En tal caso, se terminaría el espiral.

Para utilizar este modelo se debe ser particularmente bueno en identificar y manejar riesgos. La Figura muestra un diagrama conceptual del modelo de cascada describiendo los distintos ciclos del espiral. Nuevamente, no se muestra una etapa explícita de “documentación” dado que ésta se llevaba a cabo durante el transcurso de todo el desarrollo.

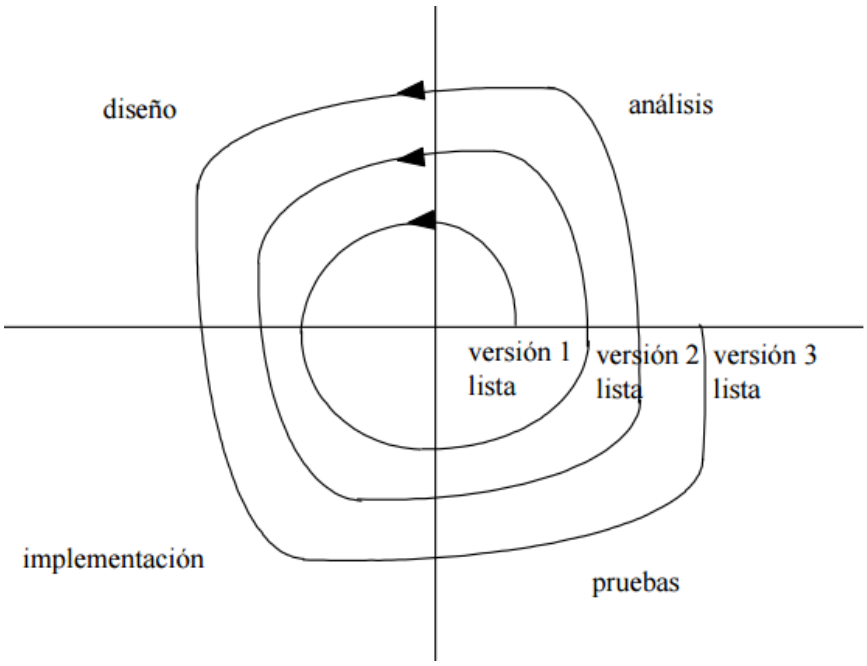


Figura 3 Modelo en espiral (Weitzenfeld, 2004)

El modelo de espiral contempla que el desarrollo de sistemas es un proceso de cambios progresivos. Un sistema normalmente se desarrolla mediante cambios en la especificación de la versión anterior del sistema que son incorporados a nuevas versiones, donde un cambio se conoce cómo un delta en la especificación de requisitos o versión. **(Weitzenfeld, 2004)**

2.3.1.3 MÉTODO DE DESARROLLO POR ANÁLISIS ESTRUCTURADO

Muchos especialistas en sistemas de información reconocen la dificultad de comprender de manera completa sistemas grandes y complejos. El método de desarrollo del análisis estructurado tiene como finalidad superar esta dificultad por medio de:

1). La división del sistema en componentes

2). La construcción de un modelo del sistema.

El análisis estructurado se concentra en especificar lo que se requiere que haga el sistema o la aplicación. Permite que las personas observen los elementos lógicos (lo que hará el sistema) separados de los componentes físicos (computadora, terminales, sistemas de almacenamiento, etc.). Después de esto se puede desarrollar un diseño físico eficiente para la situación donde será utilizado.

El análisis estructurado es un método para el análisis de sistemas manuales o automatizados, que conduce al desarrollo de especificaciones para sistemas nuevos o para efectuar modificaciones a los ya existentes. Éste análisis permite al analista conocer un sistema o proceso en una forma lógica y manejable al mismo tiempo que proporciona la base para asegurar que no se omite ningún detalle pertinente.

Componentes

Símbolos gráficos: Iconos y convenciones para identificar y describir los componentes de un sistema junto con las relaciones entre estos componentes.

Diccionario de datos: descripción de todos los datos usados en el sistema. Puede ser manual o automatizado.

Descripciones de procesos y procedimientos: declaraciones formales que usan técnicas y lenguajes que permiten a los analistas describir actividades importantes que forman parte del sistema.

Reglas: estándares para describir y documentar el sistema en forma correcta y completa.

Diseño Estructurado.

El diseño Estructurado es otro elemento del Método de Desarrollo por Análisis Estructurado que emplea la descripción gráfica, se enfoca en el desarrollo de especificaciones del software.

El objetivo del Diseño Estructurado es programas formados por módulos independientes unos de otros desde el punto de vista funcional.

La herramienta fundamental del Diseño Estructurado es el diagrama estructurado que es de naturaleza gráfica y evitan cualquier referencia relacionada con el hardware o detalles físicos. Su finalidad no es mostrar la lógica de los programas (que es la tarea de los diagramas de flujo).

Los Diagramas Estructurados describen la interacción entre módulos independientes junto con los datos que un módulo pasa a otro cuando interacciona con él.

ANÁLISIS DE FLUJO DE DATOS

Estudia el empleo de los datos para llevar a cabo procesos específicos de la empresa dentro del ámbito de una investigación de sistemas usa los diagrama de flujos de datos y los diccionarios de datos.

HERRAMIENTAS

Las herramientas muestran todas las características esenciales del sistema y la forma en que se ajustan entre sí, cómo es muy difícil entender todo un proceso de la empresa en forma verbal, las herramientas ayudan a ilustrar los componentes esenciales de un sistema, junto con sus acciones.

Es el modelo del sistema. Es la herramienta más importante y la base sobre la cual se desarrollan otros componentes. El modelo original se detalla en diagramas de bajo nivel que muestran características adicionales del sistema. Cada proceso puede desglosarse en diagramas de flujos de

datos cada vez más detallados. Repitiéndose esta secuencia hasta que se obtienen suficientes detalles para que el analista comprenda la parte del sistema que se encuentra bajo investigación.

El diagrama físico de datos da un panorama del sistema en uso, dependiente de la implantación, mostrando cuales tareas se hacen y cómo son hechas. Incluyen nombres de personas, nombres o números de formato y documento, nombres de departamentos, archivos maestro y de transacciones, equipo y dispositivos utilizados, ubicaciones, nombres de procedimientos.

El diagrama lógico de datos da un panorama del sistema, pero a diferencia del físico es independiente de la implantación, que se centra en el flujo de datos entre los procesos, sin considerar los dispositivos específicos y la localización de los almacenes de datos o personas en el sistema. Sin indicarse las características físicas.

Flujo de datos: son movimientos de datos en una determinada dirección, desde un origen hasta un destino. Es un paquete de datos.

2.3.1.4 MÉTODO DEL PROTOTIPO DE SISTEMAS

La construcción de prototipos representa una estrategia de desarrollo, cuando no es posible determinar todos los requerimientos del usuario. Es por ello que incluye el desarrollo interactivo o en continua evolución, donde el usuario participa de forma directa en el proceso.

Este método contiene condiciones únicas de aplicación, en donde los encargados del desarrollo tienen poca experiencia o información, o donde los costos y riesgos de que se cometa un error pueden ser altos.

Así mismo este método resulta útil para probar la facilidad del sistema e identificar los requerimientos del usuario, evaluar el diseño de un sistema o examinar el uso de una aplicación. El método del prototipo de sistemas consta de 5 etapas:

1). **Identificación de requerimientos conocidos:** La determinación de los requerimientos de una aplicación es tan importante para el método de desarrollo de prototipos cómo lo es para el ciclo de desarrollo de sistemas o análisis estructurado. Por consiguiente, antes de crear un prototipo, los analistas y usuario deben de trabajar juntos para identificar los requerimientos conocidos que tienen que satisfacer.

2). **Desarrollo de un modelo de trabajo:** Es fácil comenzar los procesos de construcción del prototipo con el desarrollo de un plan general que permita a los usuarios conocer lo que se espera de ellas y del proceso de desarrollo. Un cronograma para el inicio y el fin de la primera interacción es de gran ayuda. En el desarrollo del prototipo se preparan los siguientes componentes:

a). El lenguaje para el dialogo o conversación entre el usuario y el sistema.

b). Pantallas y formatos para la entrada de datos.

c). Módulos esenciales de procesamiento.

d). Salida del sistema

3). **Utilización del prototipo:** Es responsabilidad del usuario trabajar con el prototipo y evaluar sus características y operación. La experiencia del sistema bajo condiciones reales permite obtener la familiaridad indispensable para determinar los cambios o mejoras que sean necesarios, así como las características inadecuadas

4). **Revisión del prototipo:** Durante la evaluación los analistas de sistemas desean capturar información sobre los que les gusta y lo que les desagrada a los usuarios.

Los cambios al prototipo son planificados con los usuarios antes de llevarlos a cabo, sin embargo es el analista responsable de tales modificaciones.

5). **Repetición del proceso las veces que sea necesarias:** El proceso antes descrito se repite varias veces, el proceso finaliza cuando los usuarios y analistas están de acuerdo en que el sistema ha evolucionado lo suficiente como para incluir todas las características necesarias. **(SENN, 1992)**

Capítulo 3 MÉTRICAS DEL SOFTWARE

3.1 Medidas y calidad de software

La calidad del software es un aspecto a evaluar dentro de cualquier sistema de información, para esto siempre es necesario identificar y llevar algún tipo de medición. La necesidad de medir está ligada fundamentalmente a la mejora para esto existen técnicas y métodos para mejorar la calidad y la productividad

Adopción de modelos y estándares

– CMMI

– SPICE

– ISO 9001

– Moprosoft

3.1.1.1 CMMI

El modelo CMMI vio la luz en 1987 cómo Capability Maturity Model (CMM), un proyecto del Software Engineering Institute, que es un centro de investigación de la Universidad Carnegie-Mellon. Este centro lo fundó y lo financia el Departamento de Defensa de los Estados Unidos. En 1991, se publicó por primera vez el modelo CMM for Software, que está basado en una lista de comprobación de los principales factores de éxito de los proyectos de desarrollo de software realizados a finales de los años setenta y principios de los años ochenta. El modelo también se fundamenta en las investigaciones realizadas por International Business Machines (IBM) Corporation y por Philip Crosby y W. Edwards Deming, destacados representantes del ámbito de control de calidad del siglo XX. Tanto el nombre, Capability Maturity Model, cómo los cinco niveles de la representación por etapas (que se abordará más adelante en este tema) están inspirados

en el modelo de madurez Manufacturing Maturity Model de Crosby.Aplicado principalmente a programas de defensa, el modelo CMM ha logrado una aceptación considerable y se ha sometido a varias revisiones e iteraciones. Su éxito condujo al desarrollo de modelos CMM para diversos ámbitos más allá del ámbito de software. La proliferación de nuevos modelos dio lugar a confusión, por lo que el gobierno financió un proyecto de dos años en el que participaban más de 200 expertos del mundo industrial y académico a fin de crear un solo marco extensible para la ingeniería de sistemas, la ingeniería de software y el desarrollo de productos. El resultado fue CMMI.

Lo más importante que se debe saber de CMMI-DEV es que se trata de un modelo. No se trata de un proceso ni de una prescripción que se deba seguir. Es un conjunto de comportamientos organizativos de méritos demostrados en el marco del desarrollo de software y la ingeniería de sistemas. ¿Por qué debe usarse este tipo de modelo? ¿Cuál es su propósito? ¿Y cuál es la mejor forma de utilizarlo? Son preguntas esenciales que hacen referencia a aspectos de CMMI que han originado el mayor número de malentendidos.

¿Por qué debe usarse un modelo?

Si no se dispone de un modelo de cómo funcionan las organizaciones, qué funciones necesitan y cómo interactúan estas funciones, es difícil encauzar los esfuerzos de mejora. Un modelo nos permite comprender los elementos específicos de las organizaciones y ayuda a formular y a hablar de lo que hay que mejorar y de cómo se pueden lograr dichas mejoras. Un modelo ofrece las siguientes ventajas:

- proporciona un marco y un lenguaje comunes que ayudan a comunicarse,
- aporta años de experiencia,
- ayuda a los usuarios a no perder de vista la idea global cuando se enfocan específicamente en la mejora,
- suele tener el respaldo de instructores y consultores,
- puede proporcionar un estándar para ayudar a salvar las discrepancias.

¿Cuál es el propósito del modelo CMMI?

El libro de texto indicará que el propósito del modelo es evaluar la madurez de los procesos de una organización y proporcionar una orientación referente a cómo mejorar los procesos que darán lugar a mejores productos. Cuando se habla directamente con personas del Software Engineering Institute, es posible que digan que CMMI es un modelo para la administración de riesgos y que indica la capacidad de una organización para administrar los riesgos. Esta indicación es un indicio de la probabilidad con la que una organización puede cumplir sus promesas o proporcionar productos de alta calidad que sean atractivos para el mercado. Otro enfoque es que el modelo proporciona un buen indicador de cómo actuará una organización en situaciones de estrés. Una organización de gran madurez y altas capacidades afrontará con calma las situaciones inesperadas y de estrés, reaccionará, realizará cambios y seguirá adelante.

Una organización con un reducido nivel de madurez y pocas capacidades tenderá a dejarse llevar por el pánico en situaciones de estrés, seguirá a ciegas los procedimientos obviados, o bien, desbaratará todos los procesos y volverá al caos.

El modelo CMMI no es un buen indicador del rendimiento económico de una organización. Si bien las organizaciones de gran madurez pueden administrar mejor el riesgo y ser más predecibles, está demostrada la aversión de estas organizaciones hacia el riesgo. Esta aversión puede conducir a una falta de innovación o un mayor grado de burocracia que da lugar a plazos de producción significativos y una falta de competitividad. Las empresas con un reducido nivel de madurez suelen ser más innovadoras y creativas pero caóticas e impredecibles. Cuando se logran resultados, suelen ser el fruto del esfuerzo heroico de algunas personas individuales o administradores.

¿Cuál es la mejor forma de usar el modelo CMMI?

El modelo se diseñó para que se use cómo base de las iniciativas enfocadas a mejorar los procesos y, en el ámbito de la evaluación, únicamente cómo ayuda para medir las mejoras. Este enfoque ha dado lugar a resultados mixtos. Resulta demasiado fácil confundir el modelo con una definición de proceso e intentar seguirlo en lugar de considerarlo como un mapa que identifica las lagunas en los procesos existentes que habría que rellenar. El bloque de creación fundamental del modelo CMMI es un área de proceso que define los objetivos y varias de las actividades que se suelen realizar para lograr dichos objetivos. Un ejemplo de un área de proceso es el control de calidad de los procesos y productos. Otro ejemplo es la administración de las configuraciones. Es importante

entender que un área de proceso no es un proceso. Un solo proceso puede atravesar varias áreas de proceso y una sola área de proceso puede abarcar varios procesos.

En realidad, CMMI-DEV representa dos modelos que comparten los mismos elementos subyacentes. El primero y el más conocido es el modelo de la representación por etapas, que presenta 22 áreas de proceso asignadas a uno de los cinco niveles de madurez organizativa. Al valorar una organización, se evaluaría su nivel de funcionamiento y este nivel sería un indicador de su capacidad para administrar los riesgos y, por consiguiente, cumplir con sus promesas.

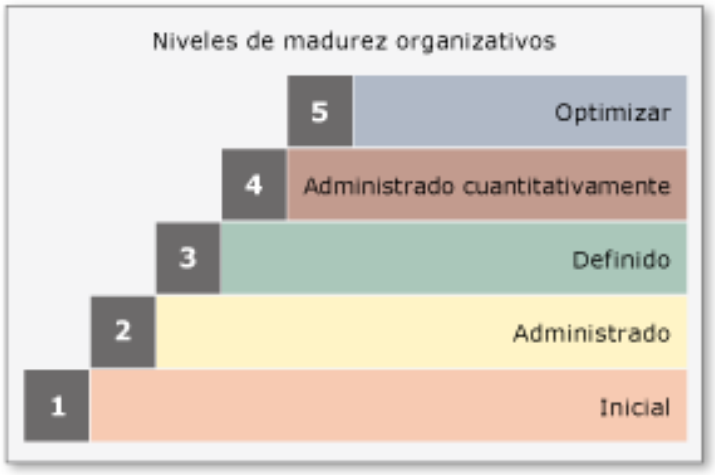


Figura 4 Niveles de madurez organizativos (Microsoft.com, 2015)

Los niveles 4 y 5 suelen denominarse los niveles de gran madurez. Suele haber una diferencia clara entre las organizaciones de gran madurez, que manifiestan comportamientos de administración cuantitativa y optimización, y las organizaciones con bajo nivel de madurez, que simplemente se administran o siguen los procesos definidos. Las organizaciones de gran madurez tienen una menor variabilidad en los procesos y suelen utilizar importantes indicadores cómo parte de un método de administración basado en estadísticas. Como resultado, estas organizaciones tienden a ser más predecibles y a responder con mayor rapidez a información nueva, suponiendo que la burocracia no se lo impida. Las organizaciones con un reducido grado de madurez tienden a realizar esfuerzos heroicos mientras que las organizaciones de gran madurez siguen a ciegas los procesos en

situaciones de estrés y no reconocen que un cambio en los procesos podría ser una respuesta más adecuada.

El segundo, la representación continua, modela la capacidad de proceso en cada una de las 22 áreas de proceso y permite a la organización ajustar sus esfuerzos de mejora a los procesos que aporten el mayor valor de negocio. Esta representación está más en línea con el modelo original de Crosby. Las valoraciones según este modelo dan lugar a perfiles de capacidad en lugar de un mero número. Por supuesto, dado que el nivel de madurez organizativa es el nivel que la mayoría de los directivos y ejecutivos entienden, es posible asignar los resultados de una evaluación según el modelo continuo a las cinco etapas.

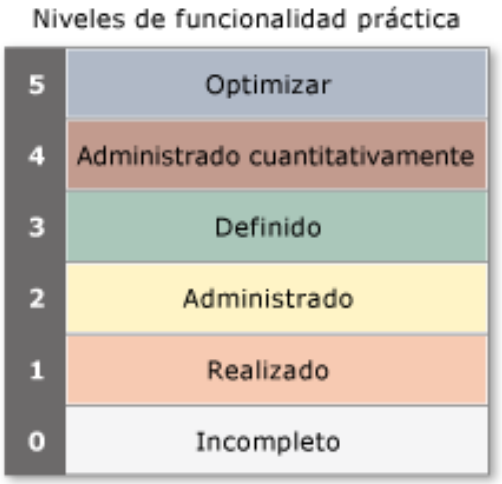


Figura 5 CMMI Niveles de funcionalidad practica (Weitzenfeld, 2004)

Usar el modelo por etapas cómo base de un programa enfocado a mejorar los procesos puede ser peligrosa porque los implementadores podrían olvidarse de que el modelo CMMI no es un proceso ni un flujo de trabajo sino que proporciona los objetivos que los procesos y flujos de trabajo deben alcanzar. Si se cumplen esos objetivos, mejorará la madurez de la organización y aumentará la probabilidad de que todo transcurra según lo previsto. Quizás el mayor error sea convertir el hecho de alcanzar un nivel en un objetivo y crear procesos y una infraestructura simplemente para superar la valoración. El objetivo de cualquier actividad orientada a mejorar los procesos debe ser una mejora mensurable, no un número.

Parece que el modelo continuo tiene más éxito cómo guía para mejorar los procesos y algunas consultorías optan por ofrecer únicamente asesoramiento en torno al modelo continuo. La diferencia más obvia reside en que un programa orientado a mejorar los procesos que se ha diseñado basándose en el modelo continuo no tiene objetivos artificiales que vengan determinados por los niveles de madurez. Asimismo, el modelo continuo se presta de manera más natural a aplicar las mejoras de proceso en las áreas donde proporcione con mayor probabilidad un beneficio económico a la organización. Por consiguiente, las organizaciones que siguen el modelo continuo son las que obtienen con mayor probabilidad una respuesta positiva a una iniciativa basada en el modelo CMMI. Es más, las respuestas positivas son las que conducen con mayor probabilidad al desarrollo de un ciclo virtuoso de mejoras. **(Microsoft.com, 2015)**

3.1.1.2 MODELO SPICE

La ISO/IEC TR 15504, conocida como SPICE (Software Process Improvement and Capability dEtermination) es un modelo de evaluación y mejora de los procesos de desarrollo y mantenimiento de sistemas y productos de software. El estándar ISO 15504 es una herramienta que ayuda a reducir costes y mejorar la calidad evitando problemas.

La ISO/IEC TR 15504 es un marco de valoración de procesos, que puede ser empleado por las organizaciones involucradas en la planificación, gestión, monitorización, control y mejora de la adquisición, suministro, desarrollo, operación, evolución y soporte de software.

Lo puedes encontrar también cómo: Software Process Improvement and Capability Determination .

La ISO/IEC TR 15504 está diseñada para facilitar una aproximación común para realizar valoraciones de procesos, haciendo posible comparaciones entre los resultados de las mismas.

Estos resultados se pueden basar en diferentes modelos de valoración (siempre que sean compatibles con el estándar) y métodos de valoración.

¿Para qué?

Proporciona todas las facilidades para la evaluación del proceso y establece los requisitos mínimos para realizar una evaluación que asegure la repetitividad y consistencia de las valoraciones obtenidas.

El objetivo principal de evaluar estos procesos es conocer la capacidad que tienen en una organización.

Después de su ejecución, se debe obtener la información relevante de cada proceso, y el punto hasta el cual estos cumplen con su propósito.

Es un Marco de referencia para:

- Determinar las fortalezas y debilidades de los procesos.
- Mejorar los procesos de software y medir sus mejoras.
- Aquellos que adquieren un sistema para evaluar la capacidad de los proveedores de sistemas.
- Determinar los riesgos de negocio para una empresa que considera desarrollar un nuevo producto de software o servicio. **(BOCCO, 2012)**

3.1.1.3 ISO 9001

¿Qué es ISO 9001?

ISO 9001 es una norma de sistemas de gestión de la calidad (SGC) reconocida internacionalmente. La norma ISO 9001 es un referente mundial en SGC, superando el millón de certificados en todo el mundo.

¿Quién puede aplicar ISO 9001?

La norma ISO 9001 es aplicable a cualquier organización – independientemente de su tamaño y ubicación geográfica. Una de las principales fortalezas de la norma ISO 9001 es su gran atractivo para todo tipo de organizaciones. Al centrarse en los procesos y en la satisfacción del cliente en lugar de en procedimientos, es igualmente aplicable tanto a proveedores de servicios cómo a fabricantes.

Los sectores internacionales siguen centrando sus esfuerzos en la calidad, con SGC específicos derivados de la norma ISO 9001, aplicables a los sectores de la automoción, aeroespacial, defensa y medicina.

Beneficios de ISO 9001 con LRQA Business Assurance

Los sistemas de gestión están cada vez más vinculados con el éxito y supervivencia de las organizaciones. De forma paralela, directores generales y gerentes de todo el mundo enfatizan la importancia que tienen las auditorías independientes para ayudar a asegurar que los sistemas de gestión alcanzan sus objetivos.

3.1.1.4 MoProSoft

MoProSoft se define como un modelo de procesos para el desarrollo y mantenimiento de software dirigido a la pequeña y mediana industria y a las áreas internas de desarrollo de software. Su objetivo principal es incorporar las mejores prácticas en gestión e ingeniería de software. Su incorporación en la industria eventualmente permitirá elevar la capacidad de ofrecer productos y servicios de software con calidad.

MoProSoft fue desarrollado por expertos mexicanos que recopilaron las experiencias exitosas de la industria de software a nivel mundial, y las adaptaron a las necesidades y características de las pequeñas y medianas industrias mexicanas (PYMEs) desarrolladoras de software.
MoProSoft está dividido en 9 procesos, llamados también prácticas, organizados por categorías de acuerdo a sus respectivas áreas de aplicación. Las categorías de procesos coinciden con los tres niveles básicos de la estructura de una organización: alta dirección, gestión y operación.

MoProSoft determina el nivel de madurez de la capacidad de cada proceso a través de una evaluación, que permite colocar a la empresa en uno de los siguientes 5 niveles.Nivel 1: Proceso Realizado

Nivel 2: Proceso Administrado

Nivel 3: Proceso Establecido

Nivel 4: Proceso Predecible

Nivel 5: Optimización del proceso

También existe el nivel 0, que indica que el proceso está incompleto (caos). El nivel de una empresa corresponde al nivel máximo al que están todos sus 9 procesos. Par pasar de un nivel al siguiente, la empresa debe cumplir todos los requisitos de los niveles anteriores más los del nuevo nivel. Los requisitos de cada nivel se encuentran detallados en el modelo. **(Pilar Gómez Gil, 2007)**

3.1.1.5 PEMM

PEMM (Performance Engineering Maturity Model) [Scholz, 1999] presenta un modelo para evaluar los niveles de integración, aplicación, ejecución y diseño, llamado ingeniería de la ejecución del modelo de madurez. Al igual que SPICE se apoya en el modelo de madurez de capacidades CMM. El objetivo de PEMM es poder evaluar la Ejecución de la Ingeniería (EI) así como la integración del proceso. El modelo sirve tanto para evaluar una organización cómo los propios desarrollos de procesos tecnológicos específicos. Sirve también para definir el criterio al escoger un proveedor de software para los productos críticos o semi-críticos de la compañía.

Al igual que el CMM, PENN cuenta con 5 niveles, los cuales determinan la mejora del comportamiento de ejecución y el decremento del riesgo de ejecución a través de estos niveles, cómo se muestra en la Figura 6

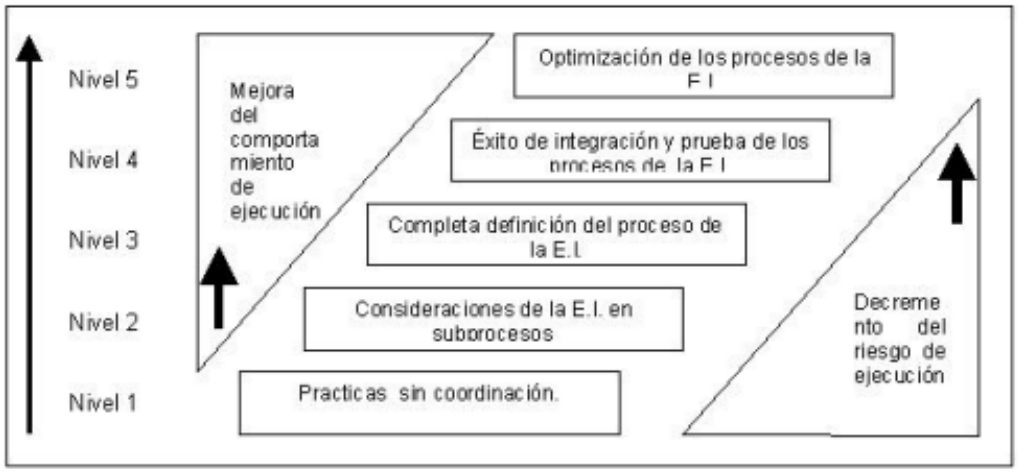


Figura 6 PEMM Niveles de integración (Weitzenfeld, 2004)

La evaluación de una compañía se hace a través de la medición de aspectos generales, la organización, la definición de procesos de ingeniería, el proyecto de la dirección y la tecnología, a través de 34 preguntas, utilizando el método Meta-Pregunta-Métrico (GQM, Goal Question Metric). GQM se usa para encuestas expertas identificando y midiendo los objetivos a través de preguntas con respuestas cuantificables (en la actualidad sólo ' Si' o ' No'). **(Weitzenfeld, 2004)**

3.1.1.6 TickIt

Tick It [Tick It, 1992], desarrollado por el Departamento de Comercio e Industria del Reino Unido, surge por la poca adopción de las normas internacionales de calidad ISO 9000 para el área deesarrollo de software. TickIt es primordialmente una guía que presenta las estrategias para lograr la certificación en la producción de software a través de la interpretación de los estándares ISO.

Los objetivos principales de TickIt son, además de desarrollar un sistema de certificación aceptable en el mercado, estimular a los desarrolladores de software a implementar sistemas de calidad, dando la dirección y guías necesarias para tal efecto. El objetivo de certificación es demostrar que

las prácticas necesarias para asegurar la calidad durante el desarrollo de software existen y son verificables. En general el modelo permite certificar cualquier tipo de proyecto a través de una estructura más flexible.

La guía de auditoria provee la liga necesaria para que la conformación (o no) del sistema auditado respecto al modelo TickIt, pueda ser expresada en función de los criterios de la ISO 9001, logrando así la aplicación de esta última al desarrollo de software. Finalmente, los requerimientos de experiencia y conocimientos que se piden a los auditores hacen posible la aplicación del modelo, suponiendo que la experiencia y conocimientos de los auditores no se vean obsoletos frente a prácticas y técnicas nuevas dentro del medio de desarrollo de software.

Esta guía se compone de (i) un capítulo de conceptos de calidad, (ii) la norma ISO 9000-3, (iii) una serie de guías para proveedores y compradores, (iv) una guía para la auditoria del sistema de calidad, (v) el proceso de certificación y (vi) guías complementarias.

3.2 Mejoramiento de procesos de software a través de técnicas y métodos estadísticos

– PSP (Personal Software Process)
–TSP, TSPI (Team Software Process)
–Six Sigma (seis sigma)

3.2.1.1 PROCESO PERSONAL DE SOFTWARE

PSP se concentra en las prácticas de trabajo de los ingenieros en una forma individual. El principio detrás de PSP es ése, sirve para producir software de calidad, cada ingeniero debe trabajar

en la necesidad de realizar trabajo de calidad. PSP se diseñó para ayudar a profesionales del software para que utilicen constantemente prácticas sanas de ingeniería de software. Así mismo les enseña a cómo planear y darle un seguimiento a su trabajo, a utilizar un proceso bien definido y medido, a establecer metas mesurables, y finalmente a la utilización del rastreo constante para alcanzar dichas metas. PSP les demuestra a los ingenieros a cómo manejar la calidad desde el principio del trabajo, a cómo analizar los resultados de cada trabajo, y a cómo utilizar los resultados para mejorar el proceso del proyecto siguiente.

El diseño de PSP se basa en los siguientes principios de planeación y de calidad

• Cada ingeniero es esencialmente diferente; para ser más precisos, los ingenieros deben planear su trabajo y basar sus planes en sus propios datos personales.

• Para mejorar constantemente su funcionamiento, los ingenieros deben utilizar personalmente procesos bien definidos y medidos.

• Para desarrollar productos de calidad, los ingenieros deben sentirse personalmente comprometidos con la calidad de sus productos.

• Cuesta menos encontrar y arreglar errores en la etapa inicial del proyecto que encontrarlos en las etapas subsecuentes.

• Es más eficiente prevenir defectos que encontrarlos y arreglarlos.

• La manera correcta de hacer las cosas es siempre la manera más rápida y más barata de hacer un trabajo. Para hacer un trabajo de ingeniería de software de la manera correcta, los ingenieros deben planear de la mejor manera su trabajo antes de comenzarlo y deben utilizar un proceso bien definido para realizar de la mejor manera la planeación del trabajo. Para que los desarrolladores lleguen a entender su funcionamiento de manera personal, deben medir el tiempo que pasan en cada proceso, los defectos que inyectan y remueven de cada proyecto y finalmente medir los diferentes tamaños de los productos que llegan a producir. Para producir constantemente productos de calidad, los ingenieros deben planear, medir y rastrear constantemente la calidad del producto y deben centrarse en la calidad desde el principio de un trabajo. Finalmente, deben analizar los resultados de cada trabajo y utilizar estos resultados para mejorar sus procesos personales.

NIVELES DE PSP

PSP tiene un marco de proceso de evolución similar al que tiene CMM. PSP trata parcialmente 12 de las 18 Capas definidas en el CMM. Las Capas son las áreas de procesos clave o Key Process Areas por su significado en inglés, estas áreas ayudan a guiar a los programadores a que exista un mejoramiento notable en el proceso de software.

En CMM un nivel de madurez sólo se alcanza si se logran cumplir todas las Capas que exige cada nivel. Sin embargo PSP solamente cubre de manera parcial estas Capas debido a que es un complemento de CMM y no depende uno del otro en ningún sentido por lo que es considerado cómo material de apoyo. Cómo se ha visto anteriormente el Instituto de la Ingeniería del Software (SEI) ha desarrollado el proceso personal del software para definir y reparar la holgura que existe entre el modelo de la madurez de la capacidad y el individuo. Por lo tanto es ideal utilizarlo junto con CMM pero no es obligatorio ya que es un proceso y no un modelo cómo lo es CMM. Para desarrollar software de alta calidad, cada componente individual también debe de contar con la más alta calidad posible. La estrategia total de PSP es cerciorarse de que todos los componentes individuales se desarrollen con la más alta calidad. PSP logra esto proporcionando un marco de proceso personal ya definido que el programador puede utilizar. Este marco es:

- Desarrollar un plan para cada proyecto y/o componente.
- Registrar su tiempo de desarrollo.
- Registrar sus defectos
- Conservar sus datos en informes del proyecto
- Utilizar sus datos para planear los proyectos y/o los componentes futuros.
- Analizar sus datos para desarrollar sus procesos con más calidad para mejorar su funcionamiento.

El proceso personal de software fue diseñado para ayudar y guiar a los ingenieros de software a realizar bien su trabajo y haciendo esto pueden llegar a cubrir las Capas requeridas. PSP también muestra cómo aplicar métodos avanzados de ingeniería a sus proyectos y/o deberes diarios. Así mismo provee métodos de estimación y de planeación muy bien detallados que son necesarios para dar un seguimiento a su trabajo.

La disciplina del PSP provee un marco estructurado para desarrollar habilidades personales y métodos que se necesitarán más adelante para ir forjando al ingeniero de software. Es importante que la calidad del software desarrollado abarque hasta el más mínimo detalle, por muy pequeño que éste sea, ya que si no se hace así, pueda dañar el sistema entero.

Cabe recalcar que se puede "personalizar" el proceso agregando o removiendo tareas conforme a las exigencias de cada persona o empresa. Esto quiere decir que por lo mismo de que PSP es un proceso y no un modelo, se puede amoldar a las necesidades del programador.

Ahora bien que si lo que se quiere hacer es que el proceso de PSP sea usado para cumplir con el modelo de capacidad de madurez, entonces se deben de tomar en cuenta los puntos que exige CMM de PSP. En cada nivel de CMM existen Capas que pueden ser cumplidas siguiendo el proceso personal de software, de hecho esa sería la mejor manera de cumplir con las exigencias de CMM.

La figura 7 presenta los elementos que tienen en común PSP con CMM. De esta manera puede analizarse cuales se podrían aplicar a cada proyecto.

Elementos del PSP en el CMM

Nivel 1 - Inicial

Nivel 2 - Repetible administración de la configuración de software aseguramiento de la calidad de software administración del subcontrato de software rastreo y visión general del proyecto de software planeación del proyecto de software administración de requerimientos

Nivel 3 - definido revisión de colegas coordinación entre los grupos ingeniería de productos de software administración integrada de software programa de entrenamiento definición del proceso de software enfoque en el proceso de software

Nivel 4 - Administrado administración de la calidad administración del proceso cuantitativo

Nivel 5 - Optimización administración del cambio de proceso administración del cambio de la tecnología prevención de defectos. (HUMPHREY, 2001)

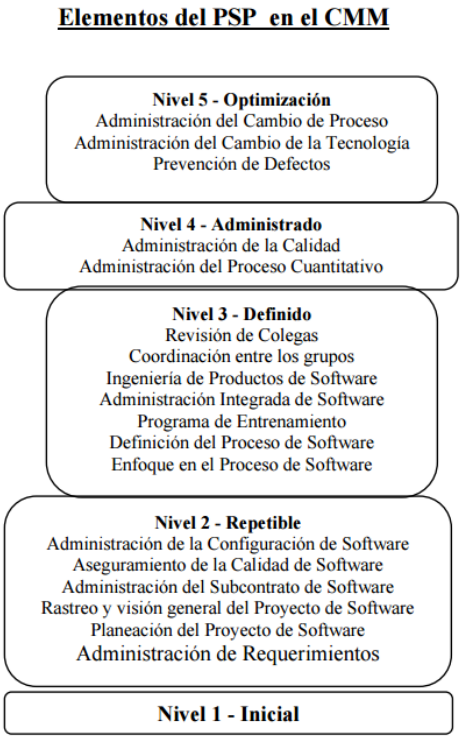


Figura 7 Elementos del PSP en el CMM (HUMPHREY, 2001)

3.2.1.2 TSP, TSPI (Team Software Process)

TSP significa Team Software Process y es esencialmente el conjunto de prácticas de estrategias que debe seguir un administrador para poder aprovechar el valor que le ofrece a una empresa o grupo de trabajo contar con un equipo de personas capacitadas en PSP que es el Personal Software Process.

¿Qué es el TSP?

El Team Software Process (TSP) es un proceso de desarrollo para equipos de ingenieros basado en CMMi.

Este modelo es una continuación de la CMM (Capability Maturity Model) ya que al igual que éste, trata de demostrar que es más productivo trabajar con prácticas de ingeniería de software y también es benéfico para su mantenimiento.

A diferencia de otros métodos…

· Mejora el desempeño tanto de equipos como individuos.

· Es disciplinado y ágil.

· Provee beneficios inmediatos y medibles.

· Acelera las iniciativas de mejora de procesos organizacionales.

Como ya se mencionó TSP es una metodología para dirigir el trabajo de mejora y desarrollo de software además de que establece un entorno donde el trabajo efectivo de equipo es normal y natural. La estructura de dicho entorno se muestras en la figura 8

CMMI	ADMINISTRACION
TSP	EQUIPO DE INGENIEROS
PSP	INGENIERO

Figura 8 Metodología d procesos (HUMPHREY, 2001)

ESTA METODOLOGIA PERMITE

· PSP, los desarrolladores utilizan procesos definidos y medibles. Se toma información de tamaño, tiempo y fallas al momento de realizar el trabajo. Se utilizan los datos para: planear y monitorear el trabajo, administrar la calidad de los productos que se producen y medir y mejorar el desempeño.

· TSP permite resolver problemas típicos de negocio: predictibilidad de costo y tiempo, mejorar la productividad y los ciclos de desarrollo, así como mejorar la calidad de los productos.

· PSP/TSP mejoran el desempeño tanto de equipos como individuos; es disciplinado y ágil; provee beneficios inmediatos y medibles; acelera las iniciativas de mejora de procesos organizacionales.

· Con TSP, los equipos encuentran y reparan defectos en etapas tempranas del proceso de desarrollo.

· Esto reduce de manera importante el tiempo de pruebas.

· Con un testing más corto, el ciclo completo se reduce. (HUMPHREY, 2001)

3.2.1.3 SIX SIGMA

Seis Sigma es un método de mejora de procesos que se basa en la reducción de la variabilidad de los mismos, lo que se busca es reducir o eliminar defectos en la entrega de un servicio o producto, tratándose de un proceso. El objetivo de Seis Sigma es reducir al mínimo los errores que se producen en un proceso, para ello es necesario comprender este proceso y profundizar en él hasta desgranar cada una de sus piezas.

Los principios del 6σ son: liderazgo comprometido de arriba abajo, orientación al cliente y enfocada a los procesos, dirigida con datos, metodología estricta, se generan ahorros y aumentas ventas, planificación de proyectos a largo plazo y la comunicación a todos los niveles.

Esta metodología tiene 5 etapas: Definir el problema o defecto, Medir, Analizar, Mejorar o Incrementar y Controlar (D-M-A-M-C) como lo muestra la figura 9

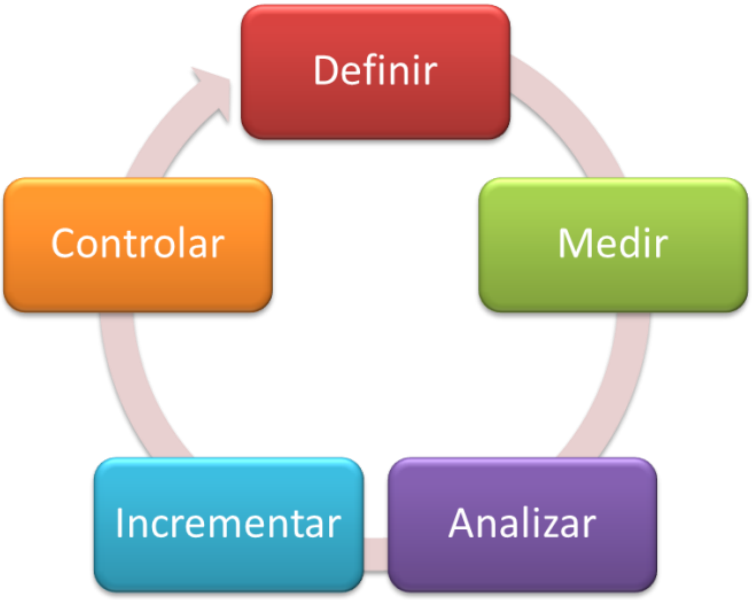


Figura 9 Etapas del Six Sigma (Lean Solutions, 2011-2015)

En la primera de las fases se identifican los posibles proyectos Seis Sigma, evaluados por la dirección para evitar la inadecuada utilización de recursos. Se pueden crear grupos de trabajo. En esta fase se crea una carta donde se explica que se va a realizar en el proyecto, nombre del proyecto, objetivo, voz de cliente (lo que el cliente reclama y necesita en lo referente al proceso con el que tratamos), al responsable del proyecto y a un supervisor que se encargarán de que el proyecto llegue a sus hitos y finalice en el plazo delimitado.

En la segunda de las etapas consiste en la caracterización de los procesos afectados, se hace un análisis de su funcionamiento actual y se determinan los requisitos clave de los clientes de dichos procesos, así como las características de calidad del producto o servicio críticas para el cliente, conocidos como CTQ's (Critical To Quality´s). La segunda parte de la medición se centra en identificar las variables que regulan el funcionamiento del proceso y condicionan su resultado. A partir de esta caracterización, se define el método para recoger datos sobre el funcionamiento actual del proceso, se recolectan dichos datos y se mide la capacidad del proceso en su situación actual, punto de partida para evaluar las posteriores mejoras conseguidas. Así, el equipo identifica

oportunidades de mejora centradas en actividades que, sin añadir valor al resultado, consumen tiempo y recursos.

La tercera (Analizar y Mejorar) de las etapas consiste en el análisis de los datos obtenidos sobre el funcionamiento del proceso, aquí se pasa del problema real al problema estadístico. Para ello el equipo desarrolla y comprueba hipótesis sobre posibles causas de variabilidad de las variables de respuesta y relaciones causa-efecto entre las variables de respuesta y las variables clave de funcionamiento, utilizando las herramientas gráficas y estadísticas pertinentes; Este punto es clave, se necesita comprender que herramienta o tipo de gráfica se va a usar, pues definirá el modo de comprender los datos del estudio. Además dependiente del tipo de datos de los que dispongamos estaremos restringidos a un tipo de gráfico u otro, de este modo podremos extraer un razonamiento lógico de cómo influyen unos parámetros frente a otros, aportando valor al proyecto.

A continuación, el equipo comienza a buscar la solución al problema, determinando las relaciones causa-efecto, lo que se pretende en esta fase es mediante el ingenio y la imaginación, teniendo como base el conocimiento del proceso, es conseguir acciones que al implantarlas de consiga optimizar el proceso y que den un resultado cuantificable positivo, aquí sirve de ayuda el brainstorming, árbol de ideas…se puede juntar un grupo de personas relacionadas con el proyecto y obtener una serie de ideas que

La última etapa de esta fase se centra en la implantación de las soluciones para mejorar y optimizar el funcionamiento del proceso. Por último se determina el rango operacional de los parámetros o variables de funcionamiento en que debía funcionar el proceso, en su régimen habitual, para asegurar los objetivos de mejora.

La última etapa de control, consiste en diseñar y documentar los controles necesarios para asegurar que lo conseguido mediante el proyecto Seis Sigma se mantenga, una vez que se hayan implantado los cambios, y el equipo deje de prestar al proceso la atención que le estuvo prestando durante el proyecto.

Una vez implantadas las acciones de mejora y hacer un control en el tiempo, el suficiente para confirmar que las acciones mantienen el proceso en los rangos deseados. Pasado un tiempo se puede retomar el proyecto y redefinirlo de nuevo pues Lean Six Sigma se basa en una mejora continua de los procesos basándose en las bases del ciclo PDCA (Lean Solutions, 2011-2015)

Capítulo 4 FASE DE MANTENIMIENTO EN LOS SISTEMAS DE INFORMACIÓN

4.1 Mantenimiento de los sistemas de información

Con posterioridad a la fase de implementación de los sistemas, se impone la fase de mantenimiento.

El mantenimiento de sistemas es el mantenimiento continuo después del inicio del funcionamiento.

Cuando se elaboran planes para la estrategia de información, las organizaciones no pueden dejar de considerar que el mantenimiento de sistemas es la fase más prolongada y costosa del ciclo de vida de los sistemas. Las implicaciones del volumen de trabajo para mantenimiento para los planes de estrategia de información en una organización es un tema que merece atención especial. La estructura de organización necesita flexibilidad para apoyar el mantenimiento de los sistemas existentes concurrentemente con la ejecución de nuevas tecnologías.

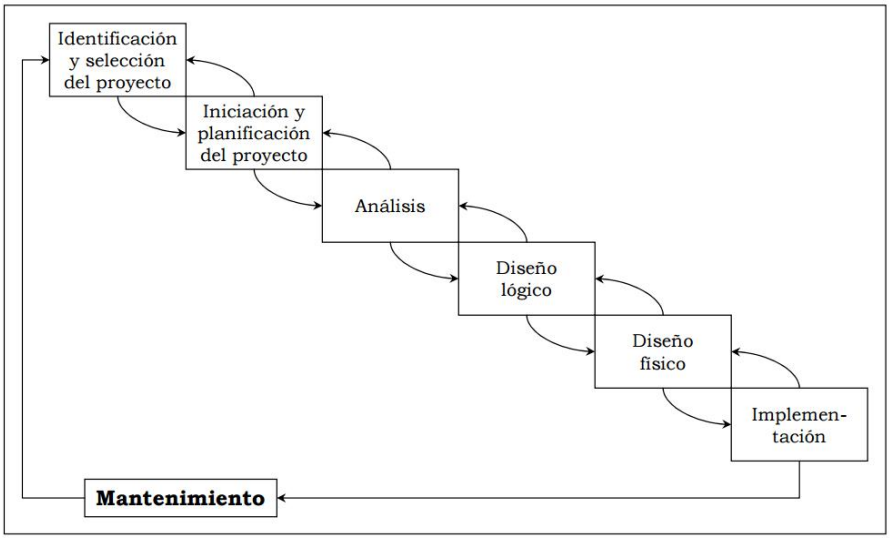


Figura 10 Fase del mantenimiento en el ciclo de vida de un sistema *(HUMPHREY, 2001)*

Es importante considerar la evaluación y el monitoreo de un sistema en términos del mantenimiento necesario y, en consecuencia, reducir o contener los costos implícitos. El

mantenimiento de sistemas puede clasificarse en cuatro grupos, cada uno de los cuales repercute en el plan estratégico de información institucional de diferentes maneras.

4.1.1.1 FUNCIONES BÁSICAS DEL MANTENIMIENTO

El servicio de mantenimiento tiene cinco funciones básicas a saber: reparar, mantener, preservar, mejorar y concebir los sistemas, con los que cualquier organización desarrolla su actividad.

Reparar: Es solucionar las averías que se producen en el sistema, para devolver al mismo el estado de disponibilidad perdido a causa de la avería, en el menor tiempo y con el menor costo posible. Para ello, se debe coordinar el uso de los recursos, establecer los procedimientos y coordinar las prioridades con otros departamentos así como las consecuencias del fallo del sistema.

Mantener: Es planear la forma más adecuada de intervenir en el sistema, para que el costo total del mantenimiento sea mínimo a corto plazo. De esta forma, se evitan las averías y el mal funcionamiento de los sistemas, reduciendo el costo y la cantidad de intervenciones. Para ello, se utilizan todos los medios disponibles, incluso los estadísticos para determinar la frecuencia de revisiones, sustitución de partes claves, probabilidad de aparición de averías, etc. La programación, el análisis de fallas, la relación causa-efecto son herramientas fundamentales.

Preservar: Es realizar las intervenciones que exige el diseño del sistema para su correcta conservación y así, poder alargar la vida útil del código según las reglas de negocio, evitando su desgaste mediante la generación de rutinas de mejoras y liberaciones emergentes.

Mejorar: Es modificar el diseño del sistema a la luz de la experiencia, para reducir el costo del mantenimiento en el futuro. Comprende las actividades de todo tipo, tendientes a eliminar las necesidades de mantenimiento (mejorar para no reparar) para corregir las fallas de manera integral

a mediano plazo mediante la modificación de elementos del sistema, el planteo de nuevas alternativas de proceso o la revisión de los elementos básicos de mantenimiento. Analizar la creación de valor mediante nuevas inversiones.

Concebir: Es participar en el diseño de los sistemas, para transferir al diseñador la experiencia y los conocimientos de las características de mantenimiento de los sistemas actuales. Esto asegura que, en el diseño de un nuevo sistema o la extensión del mismo, se tengan en cuenta los factores que de una manera u otra inciden sobre la mantenibilidad, tanto en lo que trata de evitar las fallas como en lo concerniente a facilitar las operaciones de mantenimiento. (Esteban, 2009)

4.1.1.2 Mantenimiento correctivo

Se trata del mantenimiento que se realiza a cabo para corregir los errores detectados durante la explotación del sistema. Independientemente de cuán bien diseñado, desarrollado y probado está un sistema o aplicación, ocurrirán errores inevitablemente. Este tipo de mantenimiento se relaciona con la solución o la corrección de problemas del sistema. Atañe generalmente a problemas no identificados durante la fase de ejecución. Un ejemplo de mantenimiento correctivo es la falta de una característica requerida por el usuario, o su funcionamiento defectuoso.

No programado: El mantenimiento correctivo no programado o de emergencia, obliga actuar con la mayor rapidez posible para superar los incidentes, evitar costos y en dado caso de que se llegara a parar la operación poner el sistema de nuevo a funcionar. Se efectúa con la urgencia debida, dependiendo de la avería imprevista a reparar lo más pronto posible o por una condición imperativa que hay que satisfacer. Este mantenimiento es aplicable normalmente a sistemas o componentes en los que es imposible predecir las fallas y en los procesos que admiten ser interrumpidos en cualquier momento y durante cualquier tiempo, sin afectar la producción, seguridad u otros factores igualmente importantes en la empresa. También para procesos que cuentan con cierta

antigüedad en donde se debe corregir y hacer una liberación de código a producción de manera urgente. Este tipo de mantenimiento en alto grado es causado por incidencias de otras liberaciones de código en donde no se consideraron algunas condiciones de lo que se estaba modificando.

El inconveniente que tiene este mantenimiento es que la falla puede darse en cualquier momento, muchas veces, en los momentos menos oportunos, es decir en los casos cuando se está en plena producción.

Programado: En el mantenimiento correctivo es factible desarrollar un plan a mediano y largo plazo, si es posible efectuar programas de mantenimiento que se realizan a corto plazo. Para que esto se pueda desarrollar se requiere conocer con anticipación qué es lo que debe hacerse, de modo que cuando se pare la operación pueda efectuar la liberación de un nuevo componente o código para reparación, se disponga del personal, repuestos y documentos técnicos necesarios para realizarlo correctamente. Este tipo de mantenimiento permite programar la parada de la operación o bien realizar el movimiento cuando menos riego esto represente y la ejecución de los trabajos sin ninguna urgencia y sin interferir en la producción, que lo diferencia del mantenimiento por emergencia. La oportunidad para su realización se dará en los espacios de tiempo de paradas, cambios de turnos, fines de jornada o semana, periodos de baja producción, o en vacaciones del personal, etc.

Ejecución de Trabajos: Se controla la ejecución de las actividades programadas, órdenes de trabajo y el resultado de los trabajos realizados que servirán para tomar decisiones; esto información como contexto para la liberación de mantenimiento emergente, ya que las actividades programadas pueden detenerse y reorganizarse para dar espacio al mantenimiento emergente que no se puede prevenir.

4.1.1.3 Mantenimiento evolutivo

Un mantenimiento evolutivo es aquel que pretende modificar algo que funcionaba o estaba correcto, con el objeto de aumentar, disminuir o cambiar las funcionalidades del sistema, ya sea por las necesidades del usuario o por otras causas como pueden ser, por ejemplo, cambios normativos.

Diseño: Un primer paso en este modelo es arrancar por deseables de mejora a nivel diseño, que es el primer paso o escalón para definir la misma. Una vez que la mejora es diseñada y aprobada por las partes, se pasa a la parte de Desarrollo.

Desarrollo: Así como se parte del diseño para un modelo de mejora continua, es tambien posible aplicarlo a las áreas de desarrollo directamente, operando en mejoras continuas sobre el agregado de funcionalidades o módulos adicionales. Es decir, las mejoras en desarrollo pueden derivar del diseño, o ser independientes de él.

El problema está en la definición de qué es lo que debería funcionar o estar correcto en el sistema de información, es decir la delimitación clara de la frontera entre el correctivo y evolutivo. Y no es algo que esté nada claro en muchos casos, ya que por ejemplo, eso que esa funcionalidad que dice el cliente que debería estar y no contempla el programa. Se trata del mantenimiento que se realiza para mejorar o añadir alguna característica nueva al programa. Es el mantenimiento que mayor coste supone (hasta un 60% relativo respecto al resto de mantenimientos).

4.1.1.4 Mantenimiento para fines específicos

Este tipo de mantenimiento se refiere a la creación de características nuevas o a la adaptación de las existentes según lo requieren los cambios en la organización o los usuarios, por ejemplo, los cambios en el código tributario o los reglamentos internos de la organización. Se trata del

mantenimiento llevado a cabo para adaptar el sistema a un nuevo entorno tecnológico (software, hardware, etc.).

4.1.1.5 Mantenimiento para mejoras

Se trata de la extensión o el mejoramiento del desempeño del sistema, ya sea mediante el agregado de nuevas características, o el cambio de las existentes. Un ejemplo de este tipo de mantenimiento es la conversión de los sistemas de texto a GUI (interfaz gráfica de usuarios).

4.1.1.6 Mantenimiento preventivo

Este tipo de mantenimiento es probablemente uno de los más eficaces en función de los costos, ya que si se realiza de manera oportuna y adecuada, puede evitar serios problemas en el sistema.

No existe un único tipo de mantenimiento. Según las necesidades extraídas en el proceso de mantenimiento será necesario realizar un tipo de mantenimiento u otro diferente, siendo cada uno independiente del resto. Por ejemplo, si existe un problema con una funcionalidad de la aplicación, será necesario realizar un Mantenimiento Correctivo, y si se desea añadir una nueva funcionalidad, se realizará un Mantenimiento Evolutivo.

Como se puede observar en la figura 10, el esfuerzo que se le da a los tipos de mantenimiento durante esta fase en los sistemas

El mantenimiento preventivo se refiere a las acciones, tales como; Reemplazos, adaptaciones, restauraciones, inspecciones, evaluaciones, etc. Hechas en períodos de tiempos por calendario o uso de los equipos. (Tiempos dirigidos).

El mantenimiento preventivo podría en un futuro ser potencialmente mejorado por medio de la incorporación de un programa de Mantenimiento Predictivo dependiendo de lo que piensen las organizaciones.

Beneficios del mantenimiento preventivo.

Necesitará proyectar los beneficios del mantenimiento preventivo, los más relevantes son los siguientes:

1. - Reduce las fallas y tiempos muertos (incrementa la disponibilidad de rutinas funcionales y programas vigentes).

Obviamente, si tiene muchas fallas que atender menos tiempo puede dedicarle al mantenimiento programado y estará utilizando un mantenimiento reactivo mucho más caro por ser un mantenimiento de "apaga fuegos"

2. - Incrementa la vida de los sistemas.

Si tiene buen cuidado con los sistemas puede ayudar a incrementar su vida. Sin embargo, requiere de involucrar a todos en la idea de la prioridad ineludible de realizar y cumplir fielmente con el programa.

3. - Mejora la utilización de los recursos.

Cuando los trabajos se realizan con calidad y el programa se cumple fielmente. El mantenimiento preventivo incrementa la utilización de los sistemas, equipo e instalaciones, esto tiene una relación directa con:

El programa de mantenimiento preventivo que se hace. Lo que se puede hacer, y como debe hacerse.

4. - Ahorro

Un peso ahorrado en mantenimiento son muchos pesos de utilidad para la compañía. Cuando los equipos trabajan más eficientemente el valor del ahorro es muy significativo y el costo del mantenimiento preventivo es altamente rentable.

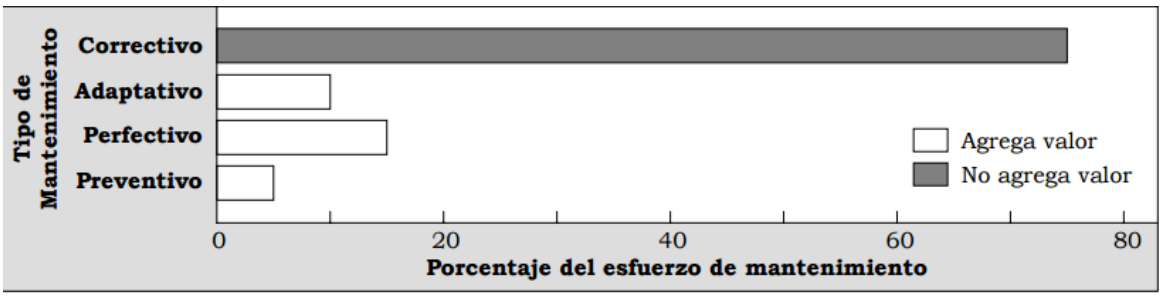


Figura 11 Esfuerzo sobre tipo de mantenimiento *(HUMPHREY, 2001)*

4.1.1.7 Costes del Mantenimiento

Los costes del Mantenimiento se refieren a los costes tanto en tiempo como dinero y recursos de la empresa que son utilizados para la realización del proceso de mantenimiento. Puede suponer hasta el 80%-90% del coste total del ciclo de vida del Software. Existen dos tipos, Directos e Indirectos.

Directos: Son los costes referidos a las actividades de Mantenimiento enumeradas anteriormente. La comprensión del software y los cambios a realizar en el mismo suponen prácticamente el 50% del coste y las pruebas un 28%.

Indirectos: Son los costes ajenos a la realización del mantenimiento y no involucrados directamente en el ciclo de vida del Software, entre los cuales se pueden encontrar los siguientes:

Costes de Mantenimiento Indirectos **(Insatisfacción del Cliente)**: Se refiere a la imposibilidad de realizar en un tiempo razonable una petición de mantenimiento.

Costes de Mantenimiento Indirectos **(Deterioro del Software)**: Al modificar el software en el proceso de mantenimiento se puede producir una disminución de la calidad y aparición de nuevos errores en el software.

Costes de Mantenimiento Indirectos **(Perdida de Oportunidades)**: Al utilizar demasiados recursos en el proceso de mantenimiento es posible que se dejen de realizar nuevos desarrollos al carecer de los recursos necesarios

4.1.1.8 Factores del Mantenimiento

Se tratan de factores que dificultan el mantenimiento del software, entre los cuales se encuentran:

Código Heredado: Código antiguo, el cual la mayoría fue construido para ocupar poco espacio, sin importar la eficiencia, diseño y mantenimiento del mismo. A su vez el código se encuentra muy deteriorado lo que aumenta el coste y la dificultad de su mantenimiento.

Evolución del Software: El software va sufriendo cambios a lo largo del tiempo, lo que disminuye su calidad y lo hace menos eficiente, aumentando su complejidad y deteriorándolo. Si tampoco se ha llevado un control de la documentación su mantenimiento se vuelve más costoso.

Ausencia de herramientas: No se utilizan herramientas, métodos ni técnicas que faciliten la realización del mantenimiento. Por lo tanto la mayoría de las veces el mantenimiento se realiza Ad-hoc

Reingeniería: Alternativa al mantenimiento. Modificación del producto software o componentes del mismo para mejorar su mantenimiento futuro. La reingeniería del software se compone de los siguientes pasos:

Análisis de inventarios: Lista con información de las aplicaciones candidatas a la reingeniería, ordenadas para conocer cual son las mejores candidatas para la misma.

Restructuración de documentos: Realizar la documentación necesaria para llegar comprender el sistema y hacer que el software sea más fácil de entender y cambiar. También se puede definir como una representación funcionalmente equivalente dentro de un mismo nivel de abstracción.

Ingeniería inversa: Es el proceso de analizar un programa con la finalidad de conocer su comportamiento. De esta forma se logra una representación del programa con una mayor abstracción que la ofrecida por el código fuente y proporciona información del diseño y la arquitectura del programa.

Ingeniería directa: También llamada renovación, consiste en utilizar los principios, métodos y conceptos de la ingeniería del software para volver a implementar la aplicación.

Herramientas CASE: Se tratan de Aplicaciones destinadas a facilitar y aumentar la productividad en el desarrollo Software. Uno de los objetivos de la Reingeniería es capturar información en un repositorio que será utilizado posteriormente por estas herramientas.

Migración: Consiste en trasladar la aplicación de un sistema a otro nuevo en condiciones de compatibilidad. La reingeniería facilita esta acción.

Esperanza de vida: Es el tiempo que la aplicación puede estar funcionando sin presentar inconvenientes graves. La reingeniería permite aumentar la esperanza de vida de la aplicación.

Prototipo de Software: Versión inicial de una aplicación Software la cual se va refinando a través de diferentes versiones. Aumenta la productividad al utilizarse Ingeniería Directa

Completitud: Se refiere al nivel de detalle que se obtiene, a medida que aumenta el nivel de abstracción la completitud disminuye.

Interactividad: Se refiere al grado con el cual las personas se integran con las herramientas de ingeniería inversa, para de este modo crear un proceso más efectivo. A medida que crece la abstracción hay que aumentar la interactividad.

Direccionalidad Mono direccional: toda la documentación extraída del código fuente se utilizará para facilitar la actividad de mantenimiento.

Direccionalidad Bidireccional: documentación obtenida del código fuente se utiliza adicionalmente para el proceso de ingeniería directa.

Existen algunas definiciones que son comunes para todo tipo de mantenimiento, como puede ser las actividades de mantenimiento, o proceso de mantenimiento. A su vez se han añadido algunas definiciones sin las cuales sería imposible dar una definición de alguna de las posteriores definiciones enumeradas en el desarrollo conceptual. Por ejemplo, no se podría dar una

definición de mantenimiento correctivo ni de reingeniería sin antes conocer la definición mantenimiento. (Esteban, 2009)

4.1.2 Controladores de versiones para el desarrollo de código dentro del mantenimiento

Las responsabilidades del desarrollador en lo referente al mantenimiento y a la actualización de software y los intereses de mantener normas para la definición de datos, rutinas de programación, información y calidad pueden verse deterioradas por la proliferación de muchas versiones "copia " del producto original, para lo cual la mayoría de las organizaciones responsables de desarrollar o mejorar código, han optado por una herramienta que se encarga de guardar las versiones de código que se tiene dentro del sistema llamados controladores de versiones

4.1.2.1 Acerca del control de versiones

¿Qué es el control de versiones, y por qué debería importarte? El control de versiones es un sistema que registra los cambios realizados sobre un archivo o conjunto de archivos a lo largo del tiempo, de modo que puedas recuperar versiones específicas más adelante. A pesar de que los ejemplos de este libro muestran código fuente como archivos bajo control de versiones, en realidad cualquier tipo de archivo que encuentres en un ordenador puede ponerse bajo control de versiones.

Si eres diseñador gráfico o web, y quieres mantener cada versión de una imagen o diseño (algo que sin duda quieres), un sistema de control de versiones (Version Control System o VCS en inglés) es una elección muy sabia. Te permite revertir archivos a un estado anterior, revertir el proyecto entero a un estado anterior, comparar cambios a lo largo del tiempo, ver quién modificó por última vez algo que puede estar causando un problema, quién introdujo un error y cuándo, y mucho más. Usar un VCS también significa generalmente que si fastidias o pierdes archivos, puedes recuperarlos fácilmente. Además, obtienes todos estos beneficios a un coste muy bajo.

4.1.2.2 Sistemas de control de versiones locales

Un método de control de versiones usado por mucha gente es copiar los archivos a otro directorio (quizás indicando la fecha y hora en que lo hicieron, si son avispados). Este enfoque es muy común porque es muy simple, pero también tremendamente propenso a errores. Es fácil olvidar en qué directorio te encuentras, y guardar accidentalmente en el archivo equivocado o sobrescribir archivos que no querías.

Para hacer frente a este problema, los programadores desarrollaron hace tiempo VCSs locales que contenían una simple base de datos en la que se llevaba registro de todos los cambios realizados sobre los archivos

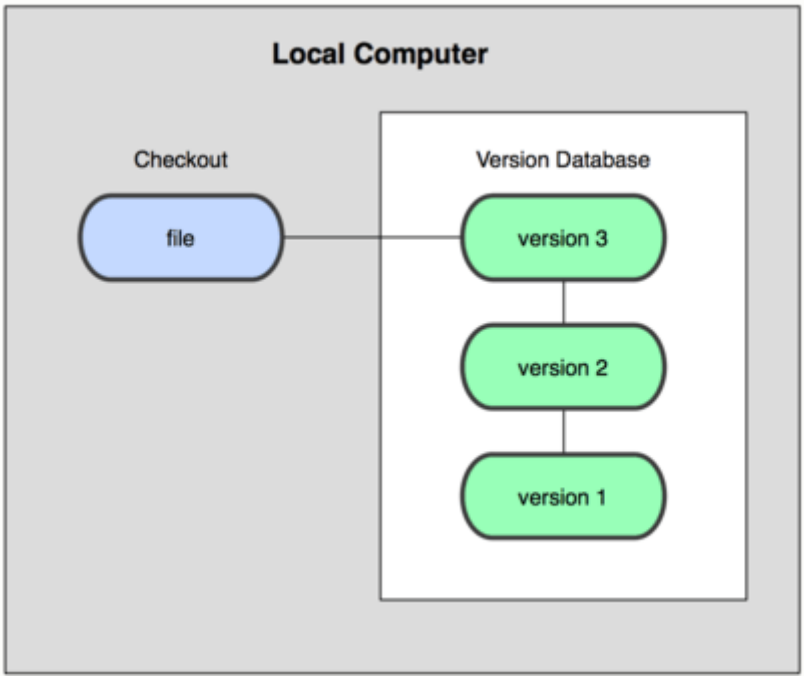


Figura 12 *Diagrama de control de versiones local (--fast-version-control, 2016)*

Una de las herramientas de control de versiones más popular fue un sistema llamado rcs, que todavía podemos encontrar en muchos de los ordenadores actuales. Hasta el famoso sistema operativo Mac OS X incluye el comando rcs cuando instalas las herramientas de desarrollo. Esta herramienta funciona básicamente guardando conjuntos de parches (es decir, las diferencias entre archivos) de una versión a otra en un formato especial en disco; puede entonces recrear cómo era un archivo en cualquier momento sumando los distintos parches.

El siguiente gran problema que se encuentra la gente es que necesitan colaborar con desarrolladores en otros sistemas. Para solventar este problema, se desarrollaron los sistemas de control de versiones centralizados (Centralized Version Control Systems o CVCSs en inglés). Estos sistemas, como CVS, Subversion, y Perforce, tienen un único servidor que contiene todos los archivos versionados, y varios clientes que descargan los archivos desde ese lugar central. Durante muchos años éste ha sido el estándar para el control de versiones (véase en la figura 11)

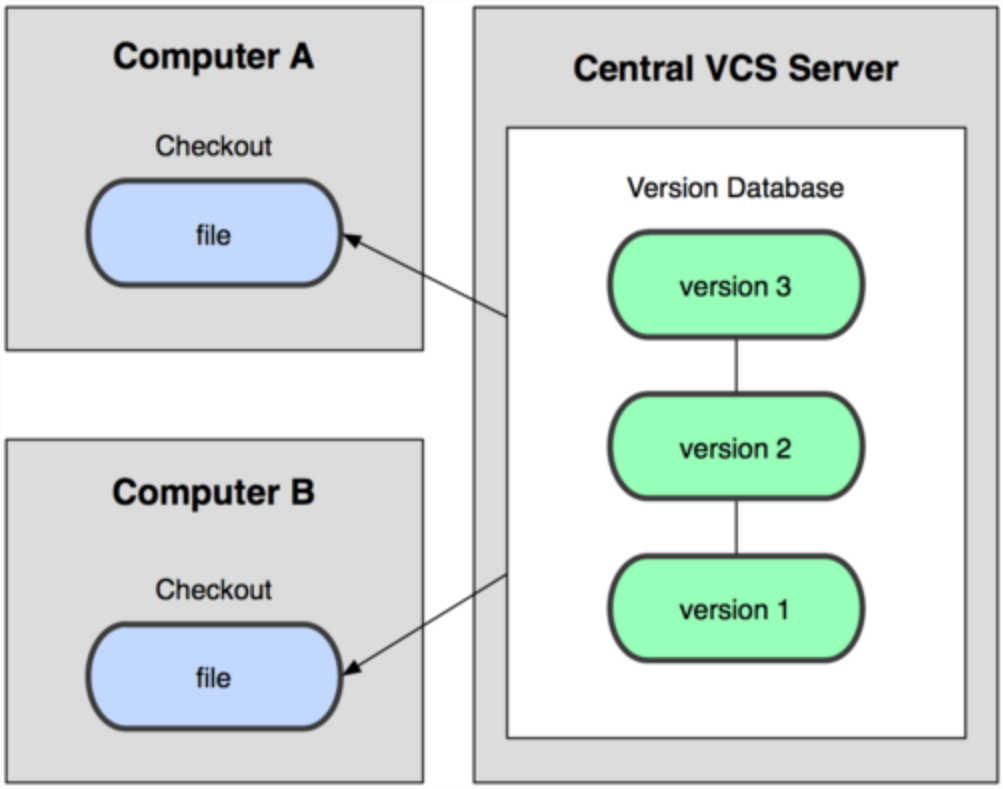


Figura 13 Diagrama de control de versiones centralizado (--fast-version-control, 2016)

Esta configuración ofrece muchas ventajas, especialmente frente a VCSs locales. Por ejemplo, todo el mundo puede saber (hasta cierto punto) en qué están trabajando los otros colaboradores del proyecto. Los administradores tienen control detallado de qué puede hacer cada uno; y es mucho más fácil administrar un CVCS que tener que lidiar con bases de datos locales en cada cliente.

Sin embargo, esta configuración también tiene serias desventajas. La más obvia es el punto único de fallo que representa el servidor centralizado. Si ese servidor se cae durante una hora, entonces

durante esa hora nadie puede colaborar o guardar cambios versionados de aquello en que están trabajando. Si el disco duro en el que se encuentra la base de datos central se corrompe, y no se han llevado copias de seguridad adecuadamente, pierdes absolutamente todo —toda la historia del proyecto salvo aquellas instantáneas que la gente pueda tener en sus máquinas locales. Los VCSs locales sufren de este mismo problema— cuando tienes toda la historia del proyecto en un único lugar, te arriesgas a perderlo todo.

Es aquí donde entran los sistemas de control de versiones distribuidos (Distributed Version Control Systems o DVCSs en inglés). En un DVCS (como Git, Mercurial, Bazaar o Darcs), los clientes no sólo descargan la última instantánea de los archivos: replican completamente el repositorio. Así, si un servidor muere, y estos sistemas estaban colaborando a través de él, cualquiera de los repositorios de los clientes puede copiarse en el servidor para restaurarlo. Cada vez que se descarga una instantánea, en realidad se hace una copia de seguridad completa de todos los datos.

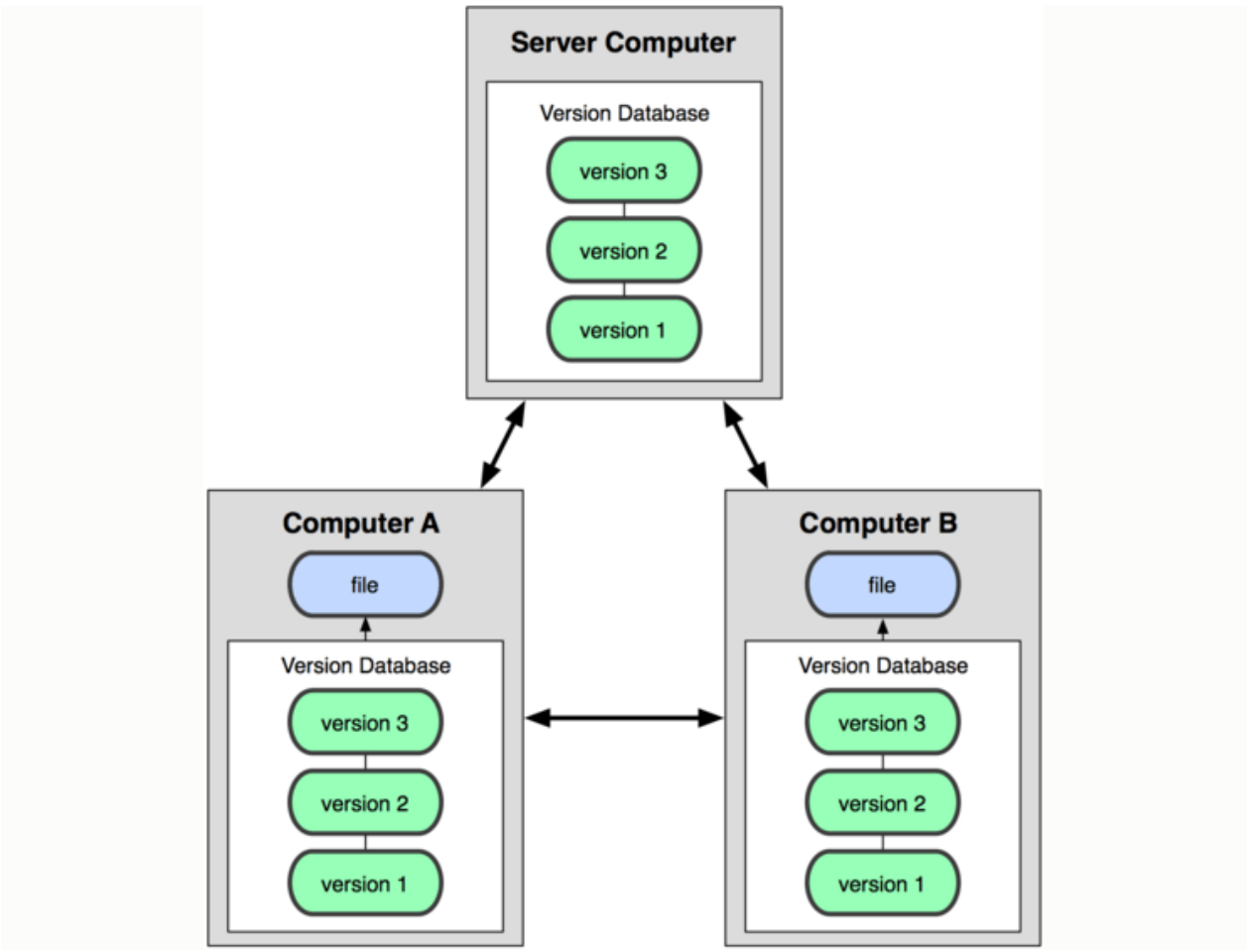


Figura 14 sistemas de control de versiones distribuidos (--fast-version-control, 2016)

Es más, muchos de estos sistemas se las arreglan bastante bien teniendo varios repositorios con los que trabajar, por lo que puedes colaborar con distintos grupos de gente simultáneamente dentro del mismo proyecto. Esto te permite establecer varios flujos de trabajo que no son posibles en sistemas centralizados, como pueden ser los modelos jerárquicos **(--fast-version-control, 2016)**

(lend J. Myers, 2004)

4.1.2.3 Pruebas de Software posteriores al mantenimiento

Posterior al desarrollo o a las modificaciones que se le realizan al código durante las fases de los diferentes tipos de mantenimiento se deben realizar pruebas para verificar la calidad del software que se va a liberar.

Las pruebas de software (en inglés software testing) son las investigaciones empíricas y técnicas cuyo objetivo es proporcionar información objetiva e independiente sobre la calidad del producto a la parte interesada o stakeholder. Es una actividad más en el proceso de control de calidad.

Las pruebas son básicamente un conjunto de actividades dentro del desarrollo de software. Dependiendo del tipo de pruebas, estas actividades podrán ser implementadas en cualquier momento de dicho proceso de desarrollo. Existen distintos modelos de desarrollo de software, así como modelos de pruebas. A cada uno corresponde un nivel distinto de involucramiento en las actividades de desarrollo.

Es el proceso exhaustivo y profundo que determina si, bajo condiciones conocidas, el sistema produce los resultados deseados.

Niveles de pruebas				
Test	**Objetivo**	**Participantes**	**Ambiente**	**Método**
Unitario	Detectar errores en los datos, lógica, algoritmos	Programadores	Desarrollo	Caja Blanca
Integración	Detectar errores de interfaces y relaciones entre componentes	Programadores	Desarrollo	Caja Blanca, Top Down, Bottom Up
Funcional	Detectar errores en la implementación de requerimientos	Testers, Analistas	Desarrollo	Funcional
Sistema	Detectar fallas en el cubrimiento de los requerimientos	Testers, Analistas	Desarrollo	Funcional
Aceptación	Detectar fallas en la implementación del sistema	Testers, Analistas, Cliente	Productivo	Funcional

Figura 15 Pruebas de Software ***(lend J. Myers, 2004)***

Una vez ejecutadas estas pruebas el tester identifica si el producto que será liberado es de buena o de baja calidad y que cumple con las especificaciones para la que fue modificado o desarrollado.

Pruebas por unidades. O pruebas de programas, consisten en probar cada programa por separado en el sistema. Las pruebas deben verse como un medio de localizar errores.

Pruebas al sistema. Prueban el funcionamiento del sistema de información como un todo. Tratan de determinar si los módulos pueden funcionar conjuntamente tal como se planeó.

Pruebas de aceptación. Proporcionan la certificación final de que el sistema está listo para ser usado en un escenario de producción. Las pruebas de sistemas son evaluadas por usuarios y revisadas por la administración.

Todos los aspectos de las pruebas deben ser pensados con sumo cuidado. Para asegurar esto el equipo de desarrollo trabaja con los usuarios para pensar en un plan sistemático de prueba. En el plan de prueba se incluyen todos los preparativos para la serie de prueba previamente descripta.

Aquí se verá la eficiencia de los revisores independientes, ya que si criticaron bien durante todo el proceso tendremos pocos errores en la prueba

Una mala crítica hará un sistema más costoso por un lado y pérdida de tiempo y esfuerzo por el otro. Cuando se critica algo se debe hacer con el ánimo de ver los errores. **(lend J. Myers, 2004)**

4.1.3 Conclusiones

Este proyecto esta vasado en la necesidad de comprender como se realiza el mantenimiento a los sistemas de información, como es que una vez entregado un producto de software este sigue funcionando a través del paso del tiempo y no solamente funcionando si no que funcione de la mejor forma correcta.

Llevando a cabo la recopilación de la información y la investigación e identificando como se desarrolla, cual es ciclo de vida y las funciones de un sistema así como las formas de darle mantenimiento a cada tipo de incidente, la medición y los métodos que se emplean para garantizar la calidad del software, dentro de los sistemas de información el desarrollo y la entrega del producto vienen de la mano, el mantenimiento es la fase más larga dentro del ciclo de vida de un sistema así como la más costosa y la más importante si se pretende que un sistema de información perdure por muchos años, trabajando de la mejor forma posible.

En base a la información obtenida de la investigación se identifica claramente que un modelo de medición y método de mejora de procesos pueden ayudar bastante al mantenimiento de software dándole calidad, sin importar si es mantenimiento preventivo, correctivo o evolutivo, una metodología de liberaciones, estadísticas de los incidentes atendidos y monitoreo del sistema puede ayudarnos a identificar fallos

En base a lo antes mencionado se llega a la conclusión de que el mantenimiento es la fase más importante después de la implementación de un sistema de información

Capítulo 5 **TRABAJOS CITADOS**

A.G., R. (2012). SISTEMAS DE INFORMACIÓN (CUARTA EDICIÓN AMPLIADA YACTUALIZADA). MEXICO D.F.: ALFAOMEGA GRUPO EDITOR S.A. DE C.V.

Berzal, F. (2005-2006). *http://flanagan.ugr.es/docencia/2005-2006/2/apuntes/ciclovida.pdf*. Recuperado el 1 de 3 de 2016

BOCCO, M. G.-F.-J.-M. (2012). MEDICIÓN Y ESTIMACIÓN DEL SOFTWARE. ALFAOMEGA.

Esteban, L. V. (2009). *scrib*. Obtenido de https://es.scribd.com/doc/25467886/Mantenimiento-Sistemas-de-Informacion

--fast-version-control, G. (2016). *Git --fast-version-control*. Obtenido de https://git-scm.com/book/es/v1/Empezando-Acerca-del-control-de-versiones

HUMPHREY, W. S. (2001). INTRODUCCIÓN ALPROCESO SOFTWARE PERSONAL.

Lean Solutions. (2011-2015). *Leansolutions.com*. Obtenido de http://www.leansolutions.co/conceptos/que-es-six-sigma/

lend J. Myers, J. W. (2004). lend J. Myers, Jhon Wiley & sons.

Microsoft.com. (2015). *microsoft.com*. Obtenido de https://msdn.microsoft.com/es-es/library/ee461556.aspx

Pilar Gómez Gil, P. (2007). MOPROSOFT: Un Camino Hacia el Éxito Mundial en el Desarrollo del Software Mexicano. 3-4.

Senn, J. (2015). Análisis y Diseño de Sistemas de Información (2da. Ed.).

SENN, J. A. (1992). Análisis y Diseño De Sistemas De Información segunda edición. McGrawHill.

Soroka, R. H. (2012). SISTEMAS DE INFORMACIÓM EN LA ERA DIGITAL. FUNDACIÓN OSDE.

Weitzenfeld, A. (2004). Ingenieria de Software orientada a objetos con UML y Java.

Whitten, B. y. (1996). Análisis y diseño de sistemas de información. En B. y. Whitten. McGraw-Hill Interamericana.