UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES
CAMPUS ARAGÓN

"VALIDACIÓN DEL SOFTWARE COMO REQUERIMIENTO DE DESARROLLO DE MODELOS DE CALIDAD"

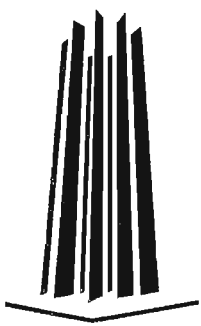
T E S I S

QUE PARA OBTENER EL TITULO DE:

INGENIERO EN COMPUTACIÓN

P R E S E N T A :

RICARDO JIMÉNEZ RÍOS



ASESOR: ING. RODOLFO VÁZQUEZ MORALES

SAN JUAN DE ARAGÓN, ESTADO DE MÉXICO 2005

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES
ARAGÓN

DIRECCIÓN

RICARDO JIMENEZ RIOS
P r e s e n t e

Con fundamento en el punto 6 y siguientes, del Reglamento para Exámenes Profesionales en esta Escuela, y toda vez que la documentación presentada por usted reúne los requisitos que establece el precitado Reglamento; me permito comunicarle que ha sido aprobado su tema de tesis y asesor.

TÍTULO:

"VALIDACIÓN DEL SOFTWARE COMO REQUERIMIENTO
DE DESARROLLO DE MODELOS DE CALIDAD"

ASESOR: Ing. RODOLFO VÁZQUEZ MORALES

Aprovecho la ocasión para reiterarle mi distinguida consideración.

A t e n t a m e n t e
"POR MI RAZA HABLARÁ EL ESPÍRITU"
San Juan de Aragón, México, 22 de junio de 2004.
LA DIRECTORA

ARQ. LILIA TURCOTT GONZÁLEZ

U.N.A.M. ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES ARAGÓN
DIRECCION

C p Secretaria Académica
C p Jefatura de Carrera de Ingenieria en Computación
C p Asesor de Tesis

LTG/AIRA

AGRADECIMIENTOS DEL ASESOR

Universidad Nacional Autónoma de México:
Gracias por recordarme que el crecimiento personal y profesional nunca termina, no lo olvidare.

Ricardo:
Te agradezco infinitamente el hacerme parte de tus logros, sobre todo en este difícil momento, ya que me ayudara a continuar con mi labor docente.

A nuestros revisores:

Ing. Ricardo Gutiérrez Orozco:
Gracias por su apoyo y palabras, amigos como usted nos fortalecen día a día.

Ing. Hugo Portilla Vázquez:
Gracias por su colaboración y apoyo, con mucho cariño y respeto.

Ing. José Antonio Avila García:
Gracias por su disposición y entusiasmo a este trabajo, me halaga saber que le haya causado tan buena impresión.

Ing. María Gabriela González Hernández:
Gracias por las contribuciones y tiempo dedicados a este proyecto.

Y a quien me dio la oportunidad de desarrollo en esta institución, hoy mi compromiso es doble, pero siempre tendré presente que "Por mi raza hablará el espíritu".

Ing. Rodolfo Vázquez Morales.

Capítulo 2
Modelos de calidad para el Software 54

Capítulo 3
Validación del Software 95

INTRODUCCION

En la actualidad la calidad en el desarrollo de software es una problemática a nivel mundial, ya que no se desarrolla conforme a las prácticas generales y mucho menos con procesos o normas de calidad. Existen diversos y muy variados tipos de software, entre ellos están el software que sirve para manejar dispositivos o máquinas, el software que sirve como aplicación, entre otros. Pero al no desarrollarse de acuerdo a procesos establecidos, como consecuencia arrastran problemas desde el principio del desarrollo del software, entre los problemas que se pueden encontrar en el desarrollo del software están:

- Mala organización de objetivos para satisfacer las necesidades de los usuarios.
- Mala planeación de requisitos.
- Error de diseño.
- Error de codificación.
- Error en pruebas de operación.
- Error en el mantenimiento, entre otros.

Estos errores son causas de infinidad de problemas para las industrias que utilizan el software como tal y el software que es un dispositivo para la producción de máquinas, aparatos electrónicos, comida, medicamentos, etc. A continuación se mencionan algunos de los efectos que causan los problemas del software:

- Mala producción de la empresa que utiliza el software.
- En el caso de los que utilizan programas para bases de datos o sistemas de información, toda la información sería afectada por error de software.
- Puede generar perdidas millonarias a las empresas que dependen del software.
- Rechazo o abandono de los sistemas.

Como una alternativa de solución existen diferentes tipos de herramientas para el buen desarrollo o ingeniería del software, pero esto no es suficiente ya que dichas

herramientas son utilizadas con poca formalidad y ejecutadas a conveniencia del programador.

Es por eso que las instituciones gubernamentales y privadas han recurrido a la creación de modelos de procesos de calidad en el software, que tienen como objetivo:

- Contribuir a la identificación, generación, promoción y adopción de estándares y mejores prácticas relacionadas con la calidad en la Ingeniería de Software.
- Implementar disciplina en los procesos de desarrollo.
- Mejorar la calidad en los productos de software utilizando las herramientas que sean necesarias.
- Establecer estándares de medición de calidad y rendimiento.

Existen diversos modelos de calidad para el desarrollo del software dichos modelos cuentan con métodos o lineamientos a seguir para su buen desarrollo, los modelos que cuentan con estándares de calidad o que sugieren normatividad, cuentan entre otros puntos como requisito indispensable **la validación** del proceso, entre los modelos que hay para el desarrollo o proceso de calidad en el diseño o desarrollo del software se encuentran:

CMM por sus siglas en ingles (Capability Maturity Model) Modelo de Capacidad y Madurez. CMM es un modelo de calidad desarrollado en el Instituto de Ingeniería de Software (SEI) de la Universidad de Carnegie Mellon, Estados Unidos. Este modelo permite a organizaciones o áreas vinculadas con la producción de software definir un camino de mejoramiento continuo para sus procesos de desarrollo.

PSP

Por sus siglas en ingles (Personal Software Process) Proceso personal del software. El proceso personal del software ayuda a ingenieros individuales a mejorar su funcionamiento trayendo disciplina en la manera en que desarrollan el software. De acuerdo con las prácticas que se encuentran en el modelo de la madurez de la capacidad (CMM), el PSP puede ser utilizado por los ingenieros como guía a un acercamiento disciplinado y estructurado para el desarrollo del software.

En la evolución del PSP se siguen 4 pasos o etapas.

- Medición: Proceso.

- Planificación: Estimación del tamaño e Informe de Pruebas.
- Calidad: Revisión y **Validación** del código y diseño.
- Ciclo de desarrollo.

TSP

Por sus siglas en ingles (Team Software Process) Proceso de software en equipo. Este método de proceso trata muchos de los problemas actuales de desarrollar productos orientados al software y demuestra a equipos y a su gerencia explícitamente cómo dirigirse a ellos. Este proceso de desarrollo tiene las siguientes fases.

- Lanzamiento.
- Estrategia.
- Planeación: es aquí donde se propone el plan de calidad a seguir en el diseño del programa, es aquí donde se planea la **verificación y validación**.
- Requerimientos.
- Diseño.
- Implementación.
- Pruebas.

SPICE

Por sus siglas en ingles (Software Process Improvement and Capability) Mejora y Capacidad en el Proceso del Software, este es un estándar internacional que proporciona un marco para la evaluación de los proyectos del software.
La evaluación de un proceso se determina en 4 etapas.

- **Parte 1 (informativo).**
- **Parte 2 (normativa.**
- **Parte 3 (normativa):** este estándar internacional define un marco para conducir la evaluación y precisar los rangos o índices de la capacidad del proceso. La evaluación incluye por los menos una instancia del proceso de cada proceso identificado en el alcance de la evaluación. Los componentes del grado o nivel.

En esta parte es en donde se determina y **valida cada paso del proceso**

- **Parte 4 (informativa)**

MOPROSOFT

Modelo de Procesos para la Industria de Software (MoProSoft)

El propósito del Modelo de Procesos para la Industria de Software es fomentar la estandarización de su operación a través de la incorporación de las mejores prácticas en gestión e ingeniería de software. La adopción del modelo permitirá elevar la capacidad de las organizaciones para ofrecer servicios con calidad y alcanzar niveles internacionales de competitividad.

Los objetivos a cumplir para asegurar el cumplimiento del propósito del proceso son:

- Lograr que los productos de salida sean consistentes con los productos de entrada en cada fase de un ciclo de desarrollo mediante las actividades de **verificación, validación** o prueba.
- Sustentar la realización de ciclos posteriores o proyectos de mantenimiento futuros mediante la integración de la *Configuración de Software* del ciclo actual.
- Llevar a cabo las actividades de las fases de un ciclo mediante el cumplimiento del *Plan de Desarrollo* actual.

Comparando los cuatro modelos anteriores, se puede comprobar que la **validación** es un requisito fundamental y obligatorio en los procesos de cada modelo. Todos los modelos requieren de diferentes tareas pero los cinco coinciden en una que es fundamental para el aseguramiento de la calidad del software y este es la validación.

Como se vio la **validación en el proceso del software** es de suma importancia, ya que asegura la calidad de este en su ejecución, además si se quiere obtener una certificación de la norma ISO este es un requisito básico y obligatorio para poder implementar un sistema de calidad siguiendo los pasos de la norma y obteniendo así una certificación, por lo tanto, un aseguramiento de la calidad tanto en el sistema como en el software, y se puede concluir, que la **validación del software** es fundamental para asegurar la calidad desde este, ya que todos los modelos de

aseguramiento de calidad del software que se mencionaron anteriormente piden como requisito la validación del software.

CAPITULO 1

ENTORNO DE DESARROLLO DEL SOFTWARE

Dentro de los componentes de una computadora el hardware es la primera mitad y el SOFTWARE es su complemento, este ultimo es responsable de las necesidades de crecimiento y de capacidad que han surgido para hacer realidad toda la creatividad ingenio y desempeño humano, claro esta que el software depende totalmente del hardware y viceversa.

El Software son todas las instrucciones y datos que corren en mayor o menor medida dentro de la computadora, es decir, la información misma, la razón del ser del hardware. En nuestros tiempos, a medida que la magia de la electrónica pone al alcance todas estas maquinas que son verdaderas prótesis mentales, mediante el abaratamiento de la tecnología y por tanto de los costos, en dirección completamente opuesta aumenta la inversión de los servicios y programas necesarios para optimizar y hacer más eficiente dichos equipos.

En sus orígenes la programación de las computadoras era hecha sólo, para y por los mismos científicos que las construían para propósitos muy específicos. El cálculo de la trayectoria de los proyectiles usados en la II Guerra Mundial, y posteriormente usos muy parecidos, hasta que mucho después fue utilizada en el Censo de los Estados Unidos, fue reconociéndose su valor en el campo administrativo donde estuvo hasta hace 2 décadas, cuando gracias a la Computadora Personal pasaron al dominio público donde con tantas necesidades fueron surgiendo las aplicaciones diversas para cada oficio.[1]

Al software se le puede clasificar de la siguiente manera:

- **Lenguajes de programación:** mediante los programas se indica a la computadora que tarea debe realizar y como efectuarla, pero para ello es preciso introducir estas ordenes en un lenguaje que el sistema pueda entender. La computadora solo entiende las instrucciones en código máquina, sin embargo, a partir de estos se elaboran los llamados lenguajes de alto y bajo nivel.

[1] Ralph M. Stair, George W. Reynolds "Principios de los Sistemas de Información" 2000 Thomson Editores

- **Software de uso general**: este ofrece la estructura para un gran número de aplicaciones empresariales, científicas y personales. La mayoría de software para uso general se vende como paquete; es decir, con software y documentación orientada al usuario (manual de referencia, plantillas de teclado y demás).
- **Software de aplicación:** este esta diseñado y escrito para realizar tareas específicas personales, empresariales o específicas, como el procesamiento de nóminas, administración de recursos humanos o el control de inventarios. Todas estas aplicaciones procesan datos para el usuario.[2]

A diferencia de algunos años atrás, hoy existe una infinidad de aplicaciones para satisfacer desde diversiones o entretenimiento de niños hasta sofisticados programas de investigación científica como los simuladores o multiagentes, pasando por programas de administración; sin embargo, para las necesidades de la mayoría de las personas que trabajan en instituciones o empresas y aún para los particulares existe un número preciso de aplicaciones, que como herramientas no deben faltar en ninguna computadora de uso personal.

1.1 Sistemas de Información (Una aplicación del Software)

Como ya se menciono existen software para cada necesidad que el usuario requiera, a continuación mencionaremos el uso del software en los sistemas de información, ya que es de suma importancia para este, debido a que la función del software en el sistema de información es de procesar los datos que se den para que este lo transforme en información y pueda ayudar en la toma de decisiones. A continuación se dará una descripción de que son y para que sirven los sistemas de información.

1.1.1 Definición de Sistemas de Información.

Un sistema de información (SI) es un conjunto de componentes interrelacionados para recolectar, manipular y diseminar datos e información y para disponer de un mecanismo de retroalimentación útil en el cumplimiento de un objetivo.[3]

[2] Ralph M. Stair, George W. Reynolds "Principios de los Sistemas de Información" 2000 Thomson Editores
[3] Kenneth C Laudon, Jane P Laudon "Administración de los Sistemas de Información" Editorial Person Education

Todos interactuamos en forma cotidiana con sistemas de información, para fines tanto personales como profesionales; utilizamos cajeros automáticos, los empleados de las tiendas registran nuestras compras sirviéndose de códigos de barra y escáneres u obtenemos información en módulos equipados con pantallas sensibles al tacto.

1.1.2 Problemas en los Sistemas de Información

Existen varios problemas fundamentales que se encuentran en los sistemas de información y en su desarrollo como se mencionan a continuación:

- *Los sistemas no están hechos a la medida y a las necesidades de la empresa.*
- *Los sistemas no se desarrollan de acuerdo a una metodología específica.*
- *Mala planeación en el desarrollo de los sistemas.*
- *Los sistemas se van parchando con funciones que no resuelven del todo los problemas.*
- *Mal conocimientos de los usuarios y por lo tanto un mal uso en el sistema de información.*
- *Mala planeación de los costos de software y hardware.*

Estas deficiencias traen consigo diferentes problemas a las empresas que los utilizan, tales como, pérdida de ganancias y de clientes, así como un nulo desarrollo de la empresa.

Otro problema que se encuentra cuando no están bien realizados los sistemas de información, por ejemplo, es el de una inscripción escolar, que es donde se manejan miles de nombres y datos correspondientes de los alumnos, esta información es manipulada constantemente y requerida en cualquier momento que se necesite, y si no esta bien realizado un sistema este puede acarrear varias consecuencias, como un intercambio de nombres, una baja del alumno, un mal estado de sus calificaciones, una mala inscripción, etc.

Estos son unos ejemplos claros de la consecuencia que pueden acarrear los problemas que se mencionaron, y lo principal en el desarrollo del Sistema de

Información es el manejo y desarrollo del software y por supuesto el hardware, ya que estos son fundamentales para el buen uso de los Sistemas.

1.1.3 Usos de los Sistemas de Información

Los usos de los sistemas de información son extensos y muy variados, desde una simple compra en un súper mercado, hasta grandes comunicaciones y transacciones de bancos y de diferentes áreas; tales como aérea, de computación, de paquetería, internet, etc. Prácticamente los Sistemas de Información se utilizan en todos los campos en que se maneja la información. A continuación se mencionan algunas áreas de donde se utilizan los Sistemas de Información:

- *Cajeros automáticos.*
- *Registros de compras de cualquier índole (tiendas de autoservicio, tiendas de ventas).*
- *Transacciones bancarias.*
- *Paquetería.*
- *Comercio electrónico.*
- *En administración de las empresas.*
- *Registro escolar (inscripción, cambios, bajas).*
- *Compras por teléfono (tiketmaster).*
- *Administración aérea.*
- *CRM y ERP*

Por si esto no fuera poco las principales compañías de la lista Fortune 500 gastan en la actualidad más de mil millones de dólares al año en la tecnología de la información como ejemplo se tiene a:

- Federal Express (FedEx) Compañía de paquetería y almacenamiento que gasta alrededor de 1000 mil millones de dólares en tecnología de información.
- UPS compañía de paquetería que desde 1986 a invertido 9000 millones de dólares en la tecnología de la información.[4]

[4] Ralph M. Stair, George W. Reynolds "Principios de los Sistemas de Información" 2000 Thomson Editores

Las causas de los problemas en los sistemas de información como ya se mencionaron son muchas y variadas pero las podemos organizar en 4 partes:

- **Usuarios:** Si el usuario no fue comprometido en el desarrollo del sistema de información por los desarrolladores y este no fue acompañándoles en el desarrollo, puede que este lo sienta ajeno, y no conozca a fondo el manejo del sistema o por otra parte si el usuario no era parte de la empresa cuando se desarrollo el sistema este tiene la responsabilidad de estudiar a fondo el sistema de información.
- **Grupo de desarrollo**: Se necesitan personas que tengan conocimientos exactos de la forma en que se desarrollan los sistemas, estos deben seguir las metodologías correspondientes para se desarrollo y deben involucrar a las personas en la empresa, tales como: los más importantes los usuarios, los administradores y la alta dirección.
- **Metodología:** Las metodologías son amplias y variadas pero siempre siguen los siguientes pasos: Investigación, Análisis, Diseño, Puesta en Operación y Mantenimiento, cada paso de la metodología debe ser seguido con suma importancia ya que cada paso depende de su anterior, el mantenimiento es tan importante como la investigación ya que en el mantenimiento se encarga de verificar, cambiar y mejorar el sistema de manera que sea más útil para sus usuarios, en pocas palabras es la administración del sistema.
- **El uso del hardware y desarrollo del software:** estos son de suma importancia, ya que el software es el que da la posibilidad de un manejo apropiado del sistema de información. El desarrollo del software es el que lleva la pauta para que un Sistema de Información de los resultados adecuados para una buena toma de decisiones.

A continuación se muestra el diagrama de las causas principales que hacen un mal Sistema de Información:

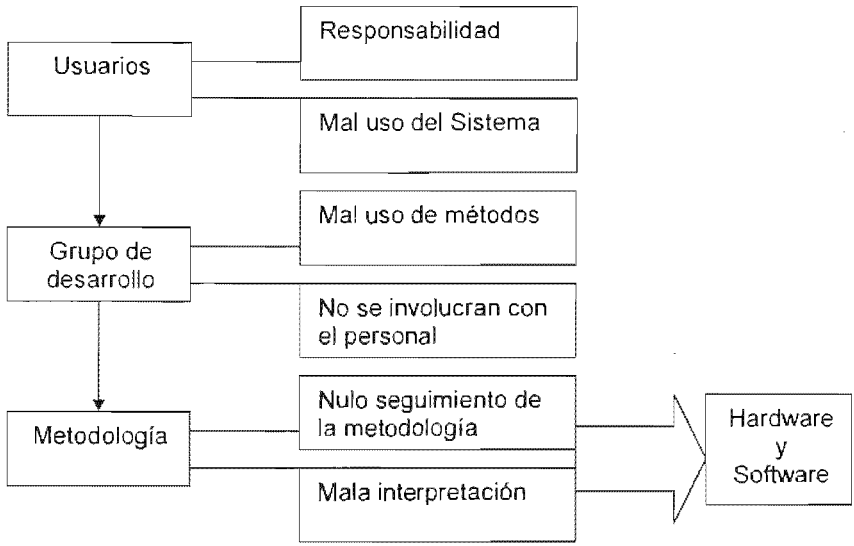


Figura 1.0

Todo esto hace que un sistema de información tenga problemas en su desarrollo o en su funcionamiento pero existen métodos que verifican si estos siguen o no los procedimientos y si estos procedimientos están bien realizados.

1.1.4 Proceso del Sistema de Información

El proceso del Sistema de Información se muestra en la siguiente figura:

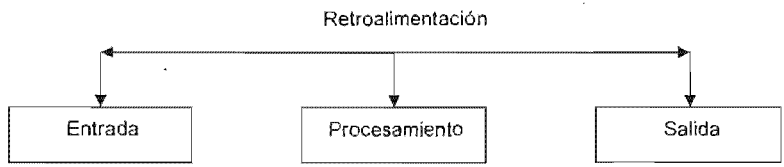


Figura 1.1

Entrada, procesamiento, salida y retroalimentación.

- **Entrada:** en los Sistemas de Información la entrada es la actividad que consiste en recopilar y capturar datos primarios. Por ejemplo, cuando se elaboran cheques de pago, antes de proceder a su cálculo e impresión debe

recolectarse información sobre el número de horas trabajadas por cada empleado. En un sistema universitario de calificaciones, los profesores deben proporcionar las calificaciones de sus alumnos para que sea posible reunirlas en un reporte semestral o trimestral destinado a los estudiantes. La entrada puede adoptar muchas formas, en un sistema de información diseñado para la producción de cheques de pago, la tarjeta de registro de llegada y salida de cada empleado podría ser la entrada inicial. En un sistema de teléfono de emergencia, toda llamada recibida sería una entrada. Las entradas de un sistema de mercadotecnia pueden contener las respuestas de clientes a encuestas, mas allá del sistema de que se trate, el tipo de entrada esta determinada por la salida que se desea obtener del sistema. La entrada puede ser un proceso manual o automatizado, el escáner para leer códigos de barra e introducir el precio y la información para identificar el producto en las cajas registradoras computarizadas de un supermercado es un ejemplo de un tipo de proceso de entrada automatizado. Pero independientemente del método de entrada que se utilice, la exactitud de la entrada es decisiva para obtener la salida deseada.

- **El procesamiento:** supone la conversión o transformación de datos en salidas útiles. Esto puede implicar ejecutar cálculos, realizar comparaciones y adoptar acciones alternas y el almacenamiento de datos para su uso posterior. El procesamiento puede llevarse a cabo de manera manual o con la asistencia de computadoras. En el caso de la aplicación en el pago de nómina, el número de horas trabajadas por cada empleado debe convertirse en un pago neto. El procesamiento requerido puede implicar primero que nada, multiplicar el número de horas trabajadas por el índice salarial por hora del empleado, con lo que se obtendría la cifra correspondiente al pago bruto. Si en una semana determinada el empleado trabajó más de 40 horas, también tendría que considerarse el pago de horas extras, por último se resta al pago bruto las deducciones que procedan, lo cual da la cifra del pago neto.
- **La salida:** implica producir información útil, por lo general en forma de documentos y/o reportes. Entre las salidas pueden encontrarse los cheques de pago de los empleados, reportes dirigidos a los administradores y la información que debe suministrarse a los accionistas, bancos, organismos

gubernamentales y otros grupos. En algunos casos la salida de un sistema, bien podría ser la entrada de otro. Por ejemplo, la salida de un sistema para el procesamiento de pedidos de venta, podría servir de entrada a un sistema para elaborar las facturas de los clientes. A menudo es común que la salida de un sistema sirva como entrada para el control de otros sistemas o dispositivos. La salida puede producirse por diversos medios, en lo referente a las computadoras, entre los dispositivos de salida más comunes están impresoras y pantallas. Sin embargo, la salida también puede ser un proceso manual, pues a menudo supone informes y documentos manuscritos.

- La **retroalimentación**: es la salida que se utiliza para efectuar cambios en actividades de entrada o procesamiento. Por ejemplo, la presencia de errores o problemas, podría imponer la necesidad de corregir datos de entrada o modificar un proceso, supongamos que en cuanto al número de horas trabajadas por un empleado, se introdujo a una computadora la cantidad de 400 en vez de 40, afortunadamente la mayoría de los sistemas de información disponen de recursos para comprobar que los datos son congruentes con escalas predeterminadas. La escala del número de horas trabajadas podría ir de 0 a 100. Es improbable que un empleado trabaje más de 100 horas a la semana. En este ejemplo el sistema de información determinara que la cifra de 400 horas rebasa la escala, tras de lo cual proporcionaría retroalimentación al respecto, en forma de un mensaje de error, por ejemplo. Gracias a esta retroalimentación, se revisará y corregirá la entrada a fin de fijar en 40 el número de horas trabajadas. La retroalimentación también es de gran importancia para administradores y tomadores de decisiones. La salida de un sistema de información podría indicar, por ejemplo, que los niveles de inventarios de ciertos artículos son cada vez más bajos, un administrador podría utilizar esta retroalimentación para decidir el pedido de más artículos, los nuevos pedidos para el reabastecimiento del inventario se convertirían entonces en entradas del sistema. Además de este método reactivo, un sistema de computación también puede adoptar un método proactivo y prever la futura ocurrencia de determinados hechos con el propósito de evitar problemas, este concepto llamado **pronóstico**, puede ser útil para estimar

ventas futuras y realizar pedidos de inventario antes de que este sea insuficiente.[5]

Conocer el potencial de estos sistemas y poseer la capacidad para aplicarlos puede resultar en una exitosa trayectoria profesional en el cumplimiento de las metas de las organizaciones y en una mayor calidad de vida para la sociedad.

Computadoras y sistemas de información no cesan de producir cambios en la manera de trabajar de las organizaciones, vivimos inmersos en una economía de información. La propia información posee valor y el comercio implica a menudo el intercambio de información más que de bienes tangibles. Los sistemas basados en computadoras son de uso creciente como medios para la creación, almacenamiento y transferencia de información. Los inversionistas se sirven de sistemas de información basados en computadoras para tomar decisiones en las que están en juego miles de millones de dólares; las instituciones financiaras los emplean para transferir por medios electrónicos enormes cantidades de dinero en todo el mundo; las compañías manufactureras por su parte, los utilizan para hacer pedidos de suministros y distribuir bienes con mayor rapidez que nunca antes.

Los sistemas de información son como cualquier otro sistema dentro de una empresa, refiriéndose a que tienen propósitos que interactúan con otros componentes de la compañía. La tarea de los sistemas de información consiste en procesar la entrada, mantener archivos de datos en relación con la empresa y producir información, informes y otras salidas.

Los sistemas de información están integrados por subsistemas que incluyen el hardware, software y almacenamiento de datos para los archivos y base de datos.

1.1.5 Tipos de Sistemas de Información

En las empresas desarrollan dos tipos diferentes de sistemas de información:

- **Los sistemas de procesamiento de transacciones**: estos mejoran las actividades diarias de las cuales dependen la compañía. Las aplicaciones normales incluyen el proceso de datos contables, preparación de nóminas y manejo de pedidos de venta, los sistemas de procesamiento de transacciones

[5] Ralph M. Stair, George W. Reynolds "Principios de los Sistemas de Información" 2000 Thomson Editores

normales sustituye al procesamiento computarizado por medio de procedimientos manuales. Los sistemas de procesamiento de transacciones automatizadas postulan como su objetivo básico la eficiencia, velocidad y exactitud en el procesamiento de grandes cantidades de datos y los sistemas de decisiones administrativas que se utilizan para dar apoyo directo a los gerentes responsables de la toma de decisiones dentro de la empresa. Aunque este tipo de sistemas no le dice al gerente que decisión tomar, le ayuda proporcionándole información importante que servirá de entrada al proceso de decisión.[6]

- **Los sistemas de informe para la gerencia, también se llaman *Sistemas de Información Gerencial (SIG):*** son una clase de sistemas de decisiones administrativas que proporcionan información en forma periódica para ayudar a los gerentes con las decisiones que surjan y que puedan anticiparse, por lo tanto, son altamente estructurados. Los sistemas de apoyo para la toma de decisiones son otro tipo de sistemas de decisión administrativa, en contraste, apoyan la toma de decisiones que esta menos estructurada y que no es rutinaria, esto es, aquellas decisiones donde parte del proceso de decisión consiste en definir que información es la necesaria así como la manera de utilizarla.[7]

1.1.6 Esquema general de Sistemas de Información

A continuación mencionaremos una descripción general de lo que es un Sistema de Información.

1.1.6.1 Concepto de Información

Como ya se vio un Sistema de Información recolecta y manipula la información ya que esta es sumamente valiosa por que de ella depende la organización total de las empresas; a continuación mencionaremos las definiciones de datos e información.

[6] Ralph M. Stair, George W. Reynolds "Principios de los Sistemas de Información" 2000 Thomson Editores
[7] John G. Burch Jr. Felix R. Strater Jr. "Sistemas de Información, Teoria y Práctica" Editorial Limusa México

1.1.6.2 Diferencia entre datos e información.

Los **datos** son realidades concretas en su estado primario, como el nombre de un empleado y la cantidad de horas trabajadas por el en la semana, los números de parte de un inventario o los pedidos de ventas.[8] Para representar estas realidades es posible usar varios tipos de datos como se muestra en la tabla siguiente:

Datos	Representados por
Alfanuméricos	Números, letras y otros caracteres
De imágenes	Imágenes gráficas
De audio	Sonido, ruido o tonos
De video	Imágenes en movimiento

Tabla 1.0

Cuando dichas realidades son organizadas o dispuestas en forma significativa, se convierte en información.

La información es un conjunto de datos organizados de tal modo que adquieren un valor adicional más allá del propio. Por ejemplo un administrador podría considerar más acorde con su propósito (es decir más valioso) conocer las ventas mensuales totales que la cantidad de venta de cada vendedor individual. Los datos representan hechos reales, si bien no pasan de ser realidades concretas en su estado primario, poseen escaso valor más allá del de su sola existencia. Hay que pensar en los datos, por ejemplo, como piezas de madera, en cuya condición tienen escaso valor más allá de el que inherentemente poseen como objetos específicos. En cambio si se define algún tipo de relación entre las piezas de madera, estas adquirirán mayor valor: apiladas de cierta manera, se les puede emplear como gradería, etc. Por su parte, la información es muy similar, pueden establecerse reglas y relaciones para organizar datos a fin de que provean útil y valiosa información. El tipo de información creada depende de las relaciones definidas entre los datos existentes, la adición de datos nuevos o diferentes significa la posibilidad de redefinir las relaciones y de crear nueva información.[9] El proceso de transformación de datos a información es un **proceso** o serie de tareas lógicamente relacionadas entre sí y ejecutadas con el fin de producir un resultado definido, esto es lo que hace un sistema de información y para que este tenga un resultado adecuado y óptimo necesita de un hardware y software de calidad.

[8] Kenneth C Laudon, Jane P Laudon "Administración de los Sistemas de Información" Editorial Person Education
[9] Kenneth C Laudon, Jane P Laudon "Administración de los Sistemas de Información" Editorial Person Education

Los datos se organizan o procesan en forma mental o manual, y para ello se utilizan computadoras. Este proceso de transformación se muestra en la figura siguiente:

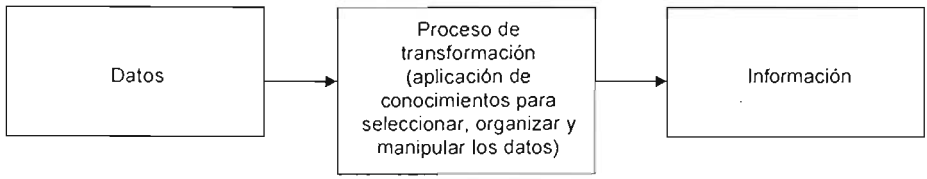


Figura 1.2

El valor de la información. El valor de la información está directamente relacionado con la utilidad que represente para los responsables de decisiones en el cumplimiento de las metas de la organización, puede medirse por ejemplo, con base en el tiempo requerido para tomar una decisión o en el aumento de las utilidades de la compañía. Considérese el caso de un pronóstico de mercado de acuerdo con el cual la demanda de un nuevo producto será alta, si la información de este pronóstico de mercado se toma en cuenta en el desarrollo de el nuevo producto y gracias a ello la compañía obtiene utilidades adicionales por 10000 dólares, el valor de esa información para la compañía equivaldría a esa misma cantidad menos el costo de la información. La información valiosa también puede ser de utilidad para los administradores en su decisión de invertir o no en sistemas y tecnología de información adicionales.

1.1.6.3 Concepto de Sistemas y Modelado

¿Que es un sistema? Componentes y conceptos

Un sistema es simplemente un conjunto de componentes que interactúan para alcanzar algún objetivo y cuya misión es servir a los propósitos de un todo.[10] De hecho todo lo que rodea al ser humano es un sistema. Los propios elementos y las relaciones entre ellos determinan el funcionamiento del sistema. **Los sistemas poseen entradas, procesamientos, mecanismos, salidas y retroalimentación.** Piensese, por ejemplo, en el lavado automático de automóviles, por supuesto las "entradas tangibles" de este proceso son un auto sucio, agua y los diversos ingredientes de la limpieza en uso; tiempo, energía, habilidad y conocimiento

[10] L. Von Bertalanffy, W. Ross Ashy "Tendencias de la Teoria General de Sistemas" Editorial Alianza

también son indispensables como entradas de este sistema, tiempo y energía son necesarios para que el sistema opere; la habilidad sería la capacidad para operar exitosamente el rociador de líquido, el cepillo espumante y los dispositivos de secado con aire, y el conocimiento interviene para definir los pasos a seguir en la operación de lavado de auto y de el orden en que estos deben ejecutarse. **Los mecanismos de procesamiento** consisten en seleccionar primero que nada, la opción de limpieza deseada (solo lavado, lavado y encerado, etc.) y hacérsela saber al operador del servicio de lavado. Obsérvese además la existencia de un **mecanismo de retroalimentación** (la opción del cliente acerca del grado de limpieza de su automóvil. Los rociadores expulsan agua, jabón líquido o cera dependiendo en que paso del proceso esté el automóvil y de las opciones seleccionadas. **La salida** es un auto limpio. Es importante señalar que para obtener buenos resultados es preciso que los elementos o componentes independientes de el sistema (rociador de líquido, cepillo y secador) interactúen entre si.

El límite de un sistema define al sistema y lo distingue de todo lo demás (el entorno). La forma en que están organizados o dispuestos los elementos del sistema se llama configuración. De modo muy similar a como ocurre con los datos, las relaciones entre los elementos de un sistema se definen por medio del conocimiento, en la mayoría de los casos conocer el propósito o resultado que se desea obtener de un sistema es el primer paso en la definición de la manera en que se configurarán sus elementos. Por ejemplo, el resultado deseado de nuestro sistema es un auto limpio, por experiencia sabemos que sería ilógico disponer las cosas en tal forma que el elemento del rociador de líquido precediera al elemento del cepillo, pues los pasos del proceso estarían invertidos (enjuagar y luego enjabonar), con lo cual el automóvil no quedaría precisamente limpio. Tal como se deduce de este ejemplo, el conocimiento es necesario tanto para definir las relaciones entre las entradas de un sistema (el auto sucio y las instrucciones al operador) como para organizar los elementos del sistema que se utilizan para procesar las entradas (el cepillo debe preceder al rociador del líquido).

1.1.6.4 Tipos de sistemas.

Los sistemas pueden clasificarse de acuerdo con numerosas dimensiones; pueden ser **simples o complejos, abiertos o cerrados, estables o dinámicos, adaptables o no adaptables, permanentes o temporales.**[11] A continuación se definen sus características:

- *Simple:* posee pocos componentes, y cuya relación o interacción entre ellos es sencilla y directa.
- *Complejo:* posee muchos elementos estrechamente relacionados e interconectados.
- *Abierto:* interactúan con su entorno.
- *Cerrado:* no interactúa con el entorno.
- *Estable:* sufre escasos cambios al paso del tiempo.
- *Dinámico:* sufre rápidos y constantes cambios al paso del tiempo.
- *Adaptable:* es capaza de modificarse en respuesta a cambios en el entorno.
- *No adaptable:* es incapaz de modificarse en respuesta a cambios en el entorno.
- *Permanente:* está diseñado para existir durante un periodo relativamente largo.
- *Temporal:* está diseñado para existir durante un periodo relativamente corto.

¿Que tipo de sistemas requieren las compañías? Por ejemplo, una compañía de servicios de limpieza dedicada al aseo de oficinas fuera de horas hábiles representa sin duda un sistema simple y estable, por que la necesidad de sus servicios es constante y sumamente regular. En cambio, una exitosa compañía fabricante de computadoras sería compleja y dinámica, pues opera en un entorno sujeto a cambios. Si una compañía no es adaptable, será imposible que sobreviva. Muchas de las compañías de computadoras surgidas en los inicios de esta industria como Osborne Computer, fabricante de una de las primeras computadoras portátiles, y VisiCorp, que desarrollo el primer programa de hoja de calculo, fueron incapaces de cambiar con la rapidez que exigía el mercado y no pudieron evitar su desaparición.

[11] Ralph M. Stair, George W. Reynolds "Principios de los Sistemas de Información" 2000 Thomson Editores

1.1.6.5 Desempeño y estándares de sistemas.

El desempeño de un sistema puede medirse de varias maneras. La **eficiencia** es una medida de lo que se produce dividido entre lo que se consume; puede ir de el 0 al 100%. Por ejemplo, la eficiencia de un motor es la energía producida (en términos de trabajo realizado) dividida entre la energia consumida (en términos de electricidad o combustible).[12] La eficiencia de algunos motores es de 50% o menos, debido a la pérdida de energía por causa de fricción y generación de calor. La eficiencia es un término relativo empleado para comparar sistemas. Un motor de gasolina, por ejemplo, es más eficiente que un motor de vapor, pues con un monto equivalente de insumo de energía (gasolina o carbón) el motor de gasolina produce más energía. El índice de eficiencia de energía (el insumo o entrada de energía dividida entre la producción o salida de energía) de los motores de gasolina es alto en comparación con el de los motores de vapor, otro ejemplo claro es el de las computadoras de décadas anteriores comparadas con las computadoras actuales, debido a la micro tecnología ahora son mas eficientes y pequeñas que antes ya que sus procesadores son mucho mas rápidos y eficaces. La **eficacia** es una medida del grado en el que un sistema cumple sus metas. Se le puede calcular al dividir las metas alcanzadas en realidad entre el total de las metas establecidas.[13] Por ejemplo, una compañía podría fijarse como meta reducir sus partes defectuosas en 100 unidades. En beneficio del cumplimiento de esta meta, podría instalar un nuevo sistema de control, sin embargo, supongamos que la reducción real de partes defectuosas equivale a solo 85 unidades; la eficacia del nuevo sistema de control sería entonces de 85 por ciento. Lo mismo que la eficiencia, la eficacia también es un término relativo que sirve para comparar sistemas. Eficacia y eficiencia son objetivos de desempeño fijados en relación con un sistema general. El cumplimiento de estos objetivos supone considerar no solo la eficiencia y la eficacia deseadas; sino también el costo, complejidad y nivel de control que desean del sistema. El **costo** comprende tanto los gastos iniciales de un sistema como la totalidad de sus gastos directos permanentes, la **complejidad** tiene que ver qué tan complicada es la relación entre los elementos del sistema. El **control** es la capacidad de un sistema para funcionar dentro del marco de normas predefinidas, tales como políticas,

[12] Ralph M Stair, George W. Reynolds "Principios de los Sistemas de Información" 2000 Thomson Editores
[13] Ralph M. Stair, George W. Reynolds "Principios de los Sistemas de Información" 2000 Thomson Editores

procedimientos y presupuestos, así como el esfuerzo administrativo requerido para mantener dentro de esos límites el funcionamiento del sistema. El cumplimiento de objetivos definidos de eficiencia y eficacia puede implicar disyuntivas en términos de costo, control y complejidad. La evaluación del desempeño de un sistema demanda también el empleo de estándares de desempeño. Un estándar de desempeño de sistemas es un objetivo específico del sistema, por ejemplo, un estándar de desempeño de sistemas de una campaña de mercadeo podría ser la de conseguir que cada representante de ventas venda una cantidad de cierto tipo de productos equivalente a $1, 000,000 cada año.

Un estándar de desempeño de sistemas de un proceso de manufactura podría ser la producción de no más de un 1 por ciento de partes defectuosas.

Una vez establecidos los estándares, se mide el desempeño del sistema y se le compara con el estándar. Las variaciones respecto al estándar son determinantes del desempeño del sistema. El cumplimiento de estándares de desempeño de sistemas también puede imponer disyuntivas en términos de costo, control y complejidad.

1.1.6.6 Variables y parámetros de sistemas.

Ciertas partes de un sistema son susceptibles de control administrativo directo, mientras que otras no.

Una **variable** de sistemas es una cantidad o unidad que puede ser controlada por el tomador de decisiones, el precio que una compañía fija a su producto es una variable de sistemas, porque puede controlarse.[14]

Un **parámetro** de sistemas es un valor o cantidad imposible de controlar como el costo de una materia prima.[15] La cantidad de gramos de un producto químico que habrá de utilizarse en la fabricación de cierto tipo de plástico es otro ejemplo de cantidad o valor no controlable por los administradores (ya que en este caso está sujeto al control de las leyes de la química).

[14] Kenneth C. Laudon, Jane P Laudon "Administración de los Sistemas de información" Editorial Person Education
[15] Ralph M. Stair, George W. Reynolds "Principios de los Sistemas de Información" 2000 Thomson Editores

1.1.6.7 Modelado de un sistema.

La realidad es compleja y dinámica. Así, cuando se desea someter a prueba diferentes relaciones y sus efectos, se recurre a modelos de sistemas, los cuales son modelos simplificados, no reales. Un **modelo** es una abstracción o aproximación que sirve para representar la realidad.[16] Los modelos nos permiten examinar situaciones reales y obtener una mejor comprensión de ellas. Los seres humanos hemos utilizado modelos desde el inicio mismo de la historia documentada. La descripción por escrito de una batalla, la maqueta de un edificio antiguo y el uso de símbolos en representación de dinero, números y relaciones matemáticas, son todos ellos ejemplo de modelos. En la actualidad, administradores y tomadores de decisiones emplean modelos para entender lo que ocurre en sus organizaciones y adoptar mejores decisiones. Existen varios tipos de modelo; entre los principales se halla el **narrativo, el físico, el esquemático y el matemático,** como se muestra en la siguiente figura:

Físico Esquemático

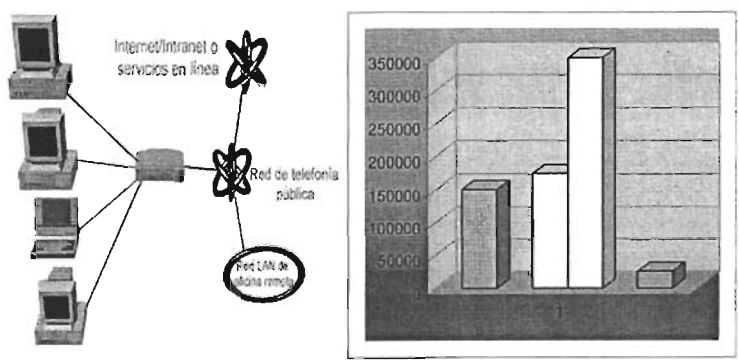


Figura 1.3

Tal y como lo indica su nombre, un modelo **narrativo** se basa en palabras. Las descripciones de la realidad, tanto oral como escrita, se consideran modelos narrativos. Los informes documentos y conversaciones respecto a un sistema son importantes modelos narrativos en las organizaciones; por ejemplo, podrían citarse los siguientes: la descripción oral de productos competidores expuesta por un vendedor a un gerente de ventas, la descripción es un reporte por escrito de la

[16] Ralph M. Stair, George W. Reynolds "Principios de los Sistemas de Información" 2000 Thomson Editores

función de una nueva pieza de equipo de manufactura y un artículo periodístico sobre economía o futuras ventas de exportación, las computadoras pueden servir para crear modelo narrativos, por ejemplo, los programas de procesamiento de texto son útiles para elaborar reportes por escrito, mientras que el software de respuesta oral es capaz de almacenar y reproducir mensajes por vía telefónica tales como saldos bancarios. Un modelo **físico** es una representación tangible de la realidad, y muchos se diseñan o realizan en computadora. Un ingeniero podría elaborar un modelo físico de un reactor químico para obtener importante información sobre el probable desempeño de un reactor de gran escala; un constructor podría crear un modelo a escala de un centro comercial con el cual ofrecería información a posibles inversionistas a cerca del aspecto general de la propuesta; un departamento de investigación de mercado podría desarrollar un prototipo de un nuevo producto o un dentista podría producir un diente de plástico. En todos estos casos, el modelo físico proporciona información. Una empresa de plástico puede utilizar directamente un sistema de computación especializado para producir un modelo físico (un prototipo de plástico) de un nuevo producto. Una vez diseñado un envase de plástico, por ejemplo, el sistema de cómputo controla el equipo con el que se produce el modelo físico, gracias a lo cual el periodo de desarrollo se acorta en varios días y se reducen los costos.

Un modelo **esquemático** es una representación gráfica de la realidad. Gráficas, mapas, figuras, diagramas, ilustraciones e imágenes son todos ellos tipos de modelos esquemáticos. En el desarrollo de programas y sistemas de computación se utilizan ampliamente modelos de esta clase. En un diagrama de flujo se muestra la manera en que se desarrollarán programas de computación. En diagramas de flujo de datos se muestra la dirección que siguen los datos en una organización. Entre alguno modelos esquemáticos utilizados en empresas se cuenta, el plano de un edificio, una gráfica de proyecciones presupuéstales y financieras, diagramas de cableado eléctrico y gráficas que indican el momento en que es preciso haber concluido ciertas tareas o actividades para cumplir un programa. Los programas de gráficos pueden ser de utilidad para crear modelos esquemáticos simples o complejos. Un modelo **matemático** es una representación aritmética de la realidad. Las computadoras sobresalen en la resolución de modelos matemáticos, en todas las áreas de la actividad empresarial se emplean modelos de este tipo, por ejemplo,

para determinar el costo total de un proyecto podría desarrollarse el siguiente modelo matemático:[17] CT = (V) (X) +CFDonde:

CT = costo total

V = costo variable por unidad

X = número de unidades producidas

CF = costo fijo

Para elaborar un modelo es importante buscar la mayor exactitud posible, un modelo inexacto derivará por lo general en la incorrecta solución de un problema. La mayoría de los modelos contienen una variedad de supuestos, de manera que es importante que estos sean los mas realistas posible, también es importante que los posibles usuarios del modelo conozcan los supuestos en los que se fundó el desarrollo de este.

1.1.6.8 Sistemas de información basados en computadoras.

Este tipo de sistemas (SIBC) está compuesto por hardware, software, bases de datos, telecomunicaciones, personas y procedimientos específicamente configurados para recolectar, manipular, almacenar y procesar datos para ser convertidos en información.[18]

A los sistemas de información basados en computadoras también se le conoce como **infraestructura tecnológica** de una compañía, porque constituyen los recursos compartidos del sistema de información.[19]

Hardware. El hardware es el equipo de computación que se utiliza para llevar a cabo las actividades de entrada, procesamiento y salida. Entre los dispositivos de entrada están los teclados, dispositivos de exploración automática y muchos otros. Entre los dispositivos de procesamiento se incluyen la unidad central de procesamiento y la memoria principal. Por ultimo entre los abundantes dispositivos de salida destacan los dispositivos de almacenamiento secundario, las impresoras y las pantallas de los monitores.

[17] Ralph M. Stair, George W. Reynolds "Principios de los Sistemas de Información" 2000 Thomson Editores
[18] John G. Burch Jr. Felix R. Strater Jr. "Sistemas de Información, Teoría y Práctica" Editorial Limusa México
[19] Ralph M. Stair, George W. Reynolds "Principios de los Sistemas de Información" 2000 Thomson Editores

Software. Este está constituido por los programas de computación que dirigen las operaciones de una computadora. Con ellos, una computadora puede procesar la nómina de una compañía, remitir facturas a clientes y dotar a los administradores de información útil para elevar utilidades, reducir costos y ofrecer un mejor servicio a los clientes. Son dos los tipos básicos de software: software del sistema (el cual controla las operaciones fundamentales de una computadora tales como arranque e impresión) y software de aplicaciones (hace posible la ejecución de tareas específicas tales como procesamiento de texto o tabulación de números). Como ejemplo de software de aplicaciones puede citarse el caso de los programas para crear hojas de cálculo (como Excel, Lotus, etc).

Bases de datos. Una base de datos es un conjunto organizados de datos e información. La base de datos de una compañía puede contener datos e información referente a clientes, empleados, inventarios, ventas de los competidores y muchos más. Se cuentan entre los componentes más valiosos e importantes de los sistemas de información basados en computadoras según administradores y ejecutivos que coincide en su mayoría.

Telecomunicaciones, redes e internet. Las **telecomunicaciones** son la transmisión electrónica de señales de comunicación que permiten a las organizaciones conectar entre sí sistemas de computación para integrar redes. Las **redes** sirven para enlazar las computadoras y equipo de computación de un edificio, un país o el mundo entero, con la finalidad de establecer comunicaciones electrónicas. Telecomunicaciones y redes hacen posible que las personas se comuniquen entre sí por medio del correo electrónico y el correo de voz, y facilitan el trabajo en equipo. El **internet** es la red de computación más grande del mundo; consiste en realidad en miles de redes interconectadas, todas las cuales intercambian libremente información. Institutos de investigación, universidades, preparatorias y empresas son apenas unos cuantos ejemplos de organizaciones que utilizan la internet. Cualquier individuo con acceso a internet pude comunicarse con cualquier otro que también disponga de acceso a esta red. La tecnología base para crear el internet se aplica hoy en día en compañías y organizaciones para conformar **intranets;** por medio de estas redes internas los miembros de una organización pueden intercambiar información y trabajar en proyectos comunes. La Word Wide Web es por su parte una red de enlaces con documentos dotados de hipermedia

(texto, gráficos, video, audio). La información respecto a tales documentos y el acceso a ellos se hallan bajo el control y administración de decenas de miles de servidores Web. La Web es uno de los muchos servicios disponibles en internet y permite acceder a literalmente millones de documentos.

Personas. Las personas son el elemento más importante de la mayoría de los sistemas de computación basados en computadoras. El personal de sistemas de información incluye a todos los individuos que administran, operan, programan y mantienen el sistema. Los usuarios son todos aquellos que utilizan sistemas de información para obtener resultados, entre los cuales se encuentran los ejecutivos financieros, los representantes de mercadotecnia, los operadores de manufactura y muchos otros individuos. También el personal de sistemas de información es usuario de computadoras.

Procedimientos. Estos son las estrategias, políticas, métodos y reglas para el uso del SIBC. Los procedimientos describen, por ejemplo, en que momento ejecutar un programa, quien puede tener acceso a información de la base de datos, que debe hacerse en casos de desastres, como incendios, temblores o huracanes, en cuyos casos el SIBC sea inutilizable.

Cuando se comienza a ver lo abundante que son los sistemas, no sorprende darse cuenta que cada sistema de el negocio depende de una o más entidades abstractas llamadas "sistemas de información". Por medio de estos sistemas los datos pasan de persona a persona y de un departamento a otro y puede realizarse cualquier cosa, desde comunicaciones entre oficinas y comunicaciones telefónicas hasta un sistema de computadora que generen informes periódicos para diferentes usuarios. A continuación se mencionara en forma general el desarrollo de un sistema.

1.1.7 Inicio del Desarrollo de Sistemas

1.1.7.1 Actividades

El desarrollo de sistemas empieza cuando un individuo o grupo con la capacidad de iniciar cambios en la organización perciben un posible beneficio de un sistema nuevo o modificado. Lo mas frecuente es que los administradores o gerentes tengan poder en una organización para iniciar el cambio, es común que sean ellos quienes empiecen proyectos de desarrollo de sistemas. Además, el grado de apoyo

administrativo general, en especial de la alta dirección, influye en forma considerable en las probabilidades de éxito o fracaso de un proyecto de desarrollo de sistemas. Las iniciativas planeadas o no, de desarrollo de sistemas surgen en todos los niveles de las organizaciones. La planeación adecuada y la participación de los administradores ayudan a garantizar que estas iniciativas den soporte a los objetivos globales de la organización, los proyectos de desarrollo de sistemas pueden comenzarse por diversas razones como se muestra en la siguiente figura:

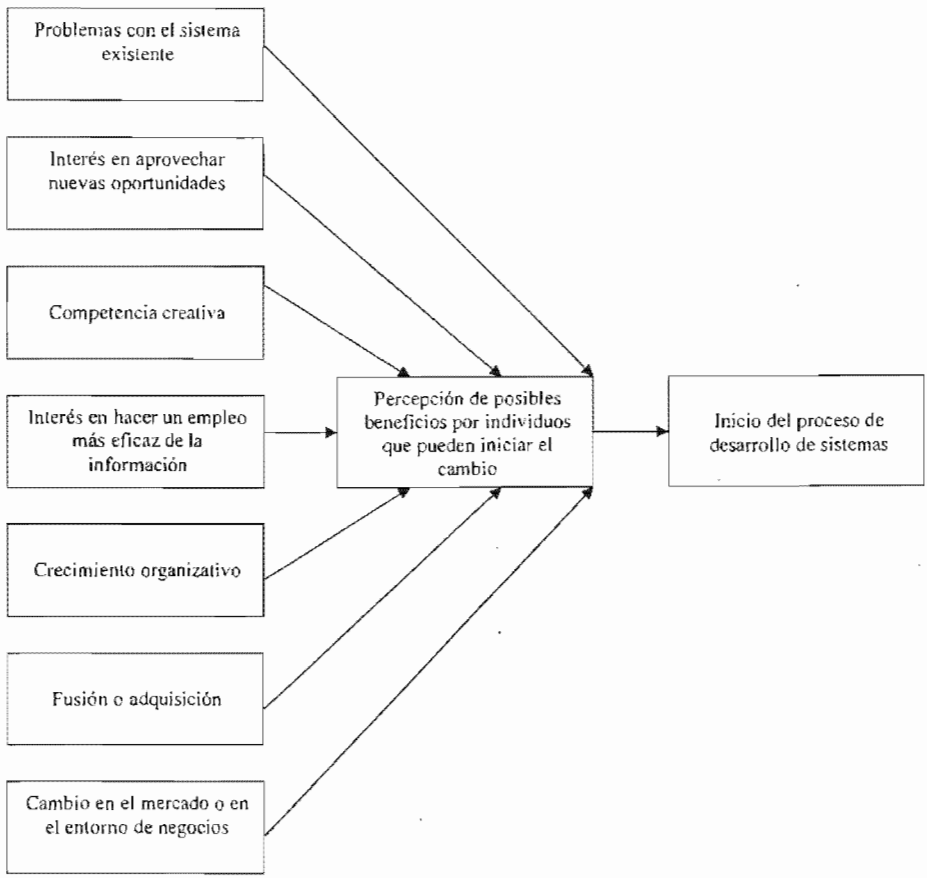


Figura 1.4

1.1.7.2 Requisitos básicos

Se buscan respuestas a las siguientes preguntas:

¿Cuál es el proceso básico?

¿Qué datos se utilizan o se producen durante este proceso?

¿Cales son los limites impuestos por tiempo y cantidad de trabajo?

¿Que controles de rendimiento se utilizan?

1.1.7.3 Entender el proceso

Se empezara con lo básico, se harán aquellas preguntas que proporcionarán información suficiente, las siguientes preguntas ayudarán a adquirir el conocimiento necesario:

¿Cuál es el propósito de esta actividad?

¿Cuáles son los pasos que se realizan?

¿Dónde se realizan?

¿Quién los ejecuta?

¿Cuánto tiempo consumen?

¿Con que frecuencia se realizan?

¿Quien utiliza la información resultante?

1.1.7.4 Identificación de los datos utilizados y la información producida

A continuación se debe encontrar que datos se deben utilizar para realizar cada actividad. La mayor parte de las transacciones de negocios producen también información que sirve a los gerentes cuando evalúan a los empleados, al negocio y al rendimiento de los sistemas y puede ser útil en otro contexto para el gerente y analista.

1.1.7.5 Determinación del tiempo del proceso y la cantidad

La frecuencia de las actividades del negocio varía enormemente, por ejemplo; algunas actividades como el pago de impuestos se presentan unas cuantas veces al año, mientras que pagar a los empleados es una actividad semanal, por lo tanto al analista debe aprender con que frecuencia se repite la actividad. Cuando el analista lo conoce puede usarla de guía para encontrar muchas preguntas adicionales e importantes y determinar la razón de la frecuencia y su efecto en las actividades del

negocio. Muchas veces la forma más fácil de obtener esta información es identificar la razón de la actividad: ¿Qué ocasiona que se desarrolle esta actividad? Los analistas a veces se refieren a la causa directa como la "función de iniciación" (inicia la actividad), los clientes pueden iniciar una actividad mediante pedidos, llamadas telefónicas, así como por sucesos (por ejemplo la terminación de una solicitud para abrir una nueva cuenta bancaria). A menos que los analistas sepan que inicia una actividad no pueden entender la razón de la actividad.

Algunas actividades, como completar una requisición de compra se llevan unos cuantos segundos, y otras se presentan poco pero requieren una gran cantidad de tiempo cuando sucede, el tiempo por si mismos no determina la importancia de una actividad, aunque afecta la manera en que los analistas evalúan ciertos paso al llevar a cabo el desempeño.

1.1.7.6 Participantes en el desarrollo de sistemas

El desarrollo eficaz de sistemas requiere de un esfuerzo de grupo, el equipo suele consistir en los beneficiarios, usuarios, administradores, especialistas en el desarrollo de sistemas y personal de apoyo diverso. Este equipo llamado grupo de desarrollo se encarga de establecer los objetivos del sistema de información, y generar un sistema que satisfaga tales objetivos para la organización. **Los beneficiarios** son aquellos individuos que obtienen alguna ventaja, en última instancia, ya sea en forma directa o a través del área de la organización a la que representan del proyecto, de desarrollo de sistemas. **Los usuarios** son las personas que interactúan con el sistema en forma regular; puede tratarse de empleados, administradores, clientes o proveedores. En proyectos de desarrollo de sistemas de gran escala, en los que la inversión en el sistema y el valor de este son considerables, es común que en el grupo de desarrollo participen miembros de la alta dirección, lo que podría incluir al director de la compañia y vicepresidentes funcionales (finanzas, mercadotecnia y asi sucesivamente).

Según sea la naturaleza del proyecto de sistemas, el grupo de desarrollo también incluirá programadores y analistas de sistemas, entre otros. **Un analista de sistemas** es un profesional especializado en el análisis y diseño de sistemas, los analistas de sistemas desempeñan funciones diversas en su interacción con los beneficiarios, usuarios, administradores, proveedores y vendedores, programadores

de software y otro personal de apoyo. A semejanza de un arquitecto que elabora planos para un edificio, el analista de sistemas elabora planes detallados para el sistema nuevo o modificado. **El programador** se encarga de modificar o desarrollar programas para satisfacer las necesidades de los usuarios, el programador toma los planes del analista de sistemas y crea o modifica el software necesario.

El personal de apoyo adicional del grupo de desarrollo consiste de especialistas técnicos, entre ellos los expertos en bases de datos y telecomunicaciones, ingenieros de hardware y software y representantes de proveedores. Una o más de estas funciones pueden encargarse a consultores externos. Según sea la magnitud del proyecto de desarrollo de sistemas y la cantidad de especialistas en el desarrollo de sistemas del departamento de sistemas de información que participa en el grupo, este también puede incluir uno o mas administradores de sistemas de información. La composición del grupo de desarrollo puede variar con el tiempo y de un proyecto a otro. En empresas pequeñas el grupo podría incluir un analista de sistemas y el propietario del negocio como beneficiario principal. En empresas grandes el personal de sistemas de información puede incluir cientos de personas que participan en diversas actividades de sistemas de información, entre ellas el de desarrollo de sistemas. Cada grupo de desarrollo debe tener un líder, también han de considerarse diferentes tipos de lideres del grupo para proyectos y grupos de desarrollo distintos. La siguiente figura muestra la interacción que hay entre los participantes del desarrollo de un sistema de información.

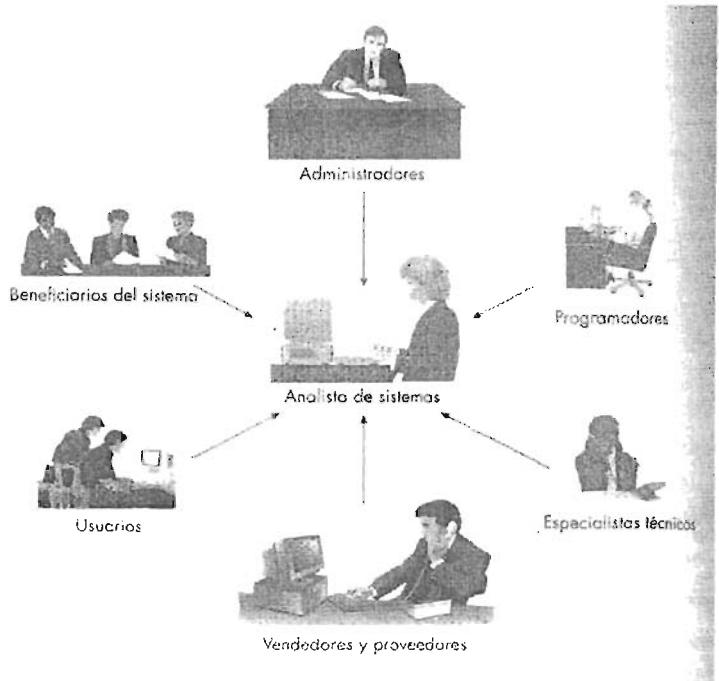


Figura 1.5

1.1.7.7 Planeación de los sistemas de información

El plan estratégico de una organización contiene tanto los objetivos de la organización misma como una delineación general de los pasos necesarios para alcanzarlos, de modo que dicho plan tiene efecto en el tipo de sistema que necesita en una organización. Por ejemplo, un plan estratégico podría identificarse como objetivos de la organización, la duplicación de los ingresos por ventas en cinco años, reducción de los gastos administrativos en 20% a los tres años, adquisición de por lo menos dos empresas competidoras en un año o lograr el liderazgo en una categoría de productos determinada. El compromiso de la organización, con políticas tales como el mejoramiento continúo, también se refleja en el plan estratégico. Tales compromisos y objetivos definen los requisitos generales de rendimiento del sistema. Es frecuente que en una parte del plan estratégico incluya lineamientos acerca de cómo lograr los objetivos de la organización, este plan también incluye directrices para las áreas funcionales de la organización, incluidas las de

mercadotecnia, producción, finanzas, contabilidad y recursos humanos. En lo relativo al departamento de sistemas de información, estas directrices forman parte del plan de sistemas de información.

El término **planeación de los sistemas de información** se refiere a la traducción de los objetivos estratégicos y organizacionales en iniciativas de desarrollo de sistemas. Una parte del plan de sistemas de información para una compañía que vende automóviles de lujo consistiría en crear un nuevo sistema de registro de productos a fin de mejorar el servicio a los clientes.[20] La planeación de un sistema de información adecuado garantiza que los objetivos del desarrollo de sistemas específicos sustentes los objetivos de la organización. Uno de los beneficios principales de la planeación de los sistemas de información es que permiten tener un panorama a largo plazo del uso de la tecnología de información en la organización. Aunque el plan de sistemas de información puede originar iniciativas específicas de desarrollo de sistemas, para lograr el éxito futuro dicho plan también debe incluir un marco de referencia amplio. Es imperativo que el plan de sistemas de información comprendan directrices sobre como deben desarrollarse a través del tiempo la infraestructura de sistemas de información en la organización. Otro beneficio de la planeación de los sistemas de información es que garantiza el mejor uso de los recursos de sistemas de información, lo que incluye fondos, personal de los sistemas de información y tiempo para programar proyectos específicos.

1.1.7.8 Ciclo de vida del desarrollo de sistemas

El proceso de desarrollo de sistemas también se denomina ciclo de vida del desarrollo de sistemas (SDLC; Systems development life cycle), dado que las actividades relacionadas con dicho proceso son continuas. A medida que se crea cada sistema, el proyecto tiene calendarios y fechas límite, hasta que por ultimo se instala y se acepta, la vida del sistema continúa con su mantenimiento y revisión. Se inicia un nuevo proyecto si el sistema requiere mejoras significativas, que van más allá del alcance de su mantenimiento, si es necesario reponerlo a causa de una

[20] Ralph M. Stair, George W. Reynolds "Principios de los Sistemas de Información" 2000 Thomson Editores

nueva generación de tecnología, o las necesidades del sistema de información de la organización cambian en forma importante.[21]

Un aspecto básico en el desarrollo de sistemas es que entre más avanzado este el SDLC en el momento de la detección de un error, será más costosa su corrección como se muestra en la figura.

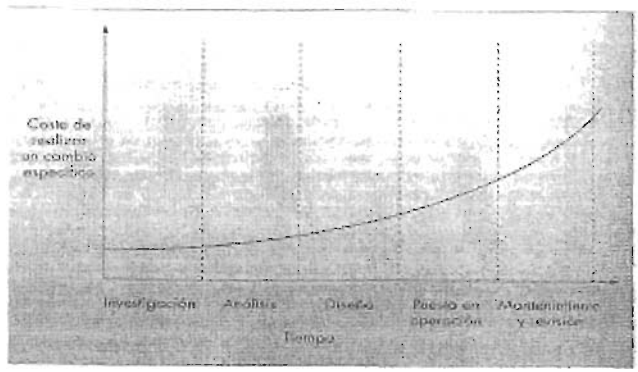


Figura 1.6

Una razón para ello es que la identificación de un error en una fase avanzada del SDLC obliga a modificar, hasta cierto punto, las fases previas. Otra razón sería que los errores detectados en una etapa más avanzada del SDLC tienen efecto en más personas. Por ejemplo un error identificado después de instalar un sistema puede requerir el readiestramiento de los usuarios, toda vez que se cuente con la solución del problema. Así pues los desarrolladores experimentados de sistemas prefieren un método que detecte errores en la etapa temprana del ciclo de vida del proyecto. Son cuatro los tipos de ciclos de vida de desarrollo de sistemas de uso más general: tradicional, de prototipos, de desarrollo rápido de aplicaciones (RAD; rapid application development) y de desarrollo por parte de los usuarios finales.

1. **Ciclo de vida de el desarrollo de sistemas tradicional**: El desarrollo de sistemas se refieren a todas las actividades que entran en la producción de una solución en los sistemas de información, el desarrollo de sistemas es una forma bien estructurada para las posibles soluciones que se tengan en la organización, este puede ir desde un pequeño proyecto como la adquisición de un programa de bajo costo de computadora, hasta un gran proyecto como la instalación de un sistema con costo de millones de dólares. Los pasos del

[21] Kenneht C. Laudon, Jane P. Laudon "Administración de los Sistemas de Información" Editorial Person Education

desarrollo de sistemas tradicional varían de una compañía a otra, existen varias metodologías para el desarrollo de sistemas, que incluyen diferentes fases, algunas definen mayores etapas y algunas lo definen con menos pasos, se ilustran varias metodologías a continuación:[22]

Metodología 1

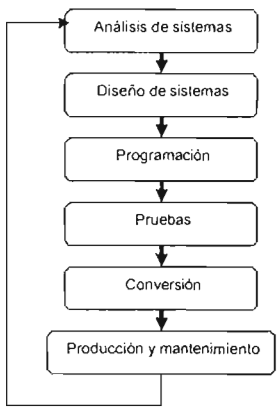


Figura 1.7

Metodología 2

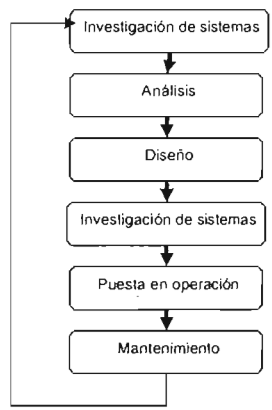


Figura 1.8

[22] Ralph M. Stair, George W. Reynolds "Principios de los Sistemas de Información" 2000 Thomson Editores

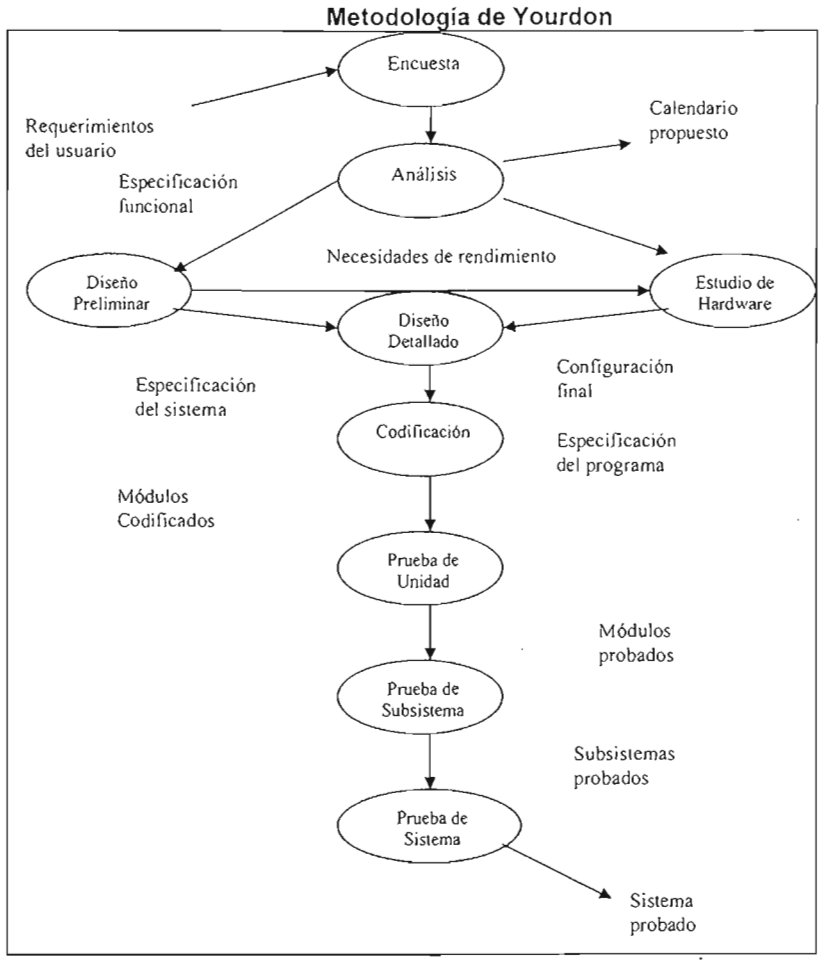


Figura 1.9

Todas las metodologías se enfocan en estas cinco fases primordiales: **investigación, análisis, diseño, puesta en operación y mantenimiento y por ultimo revisión,** como lo muestra la figura:

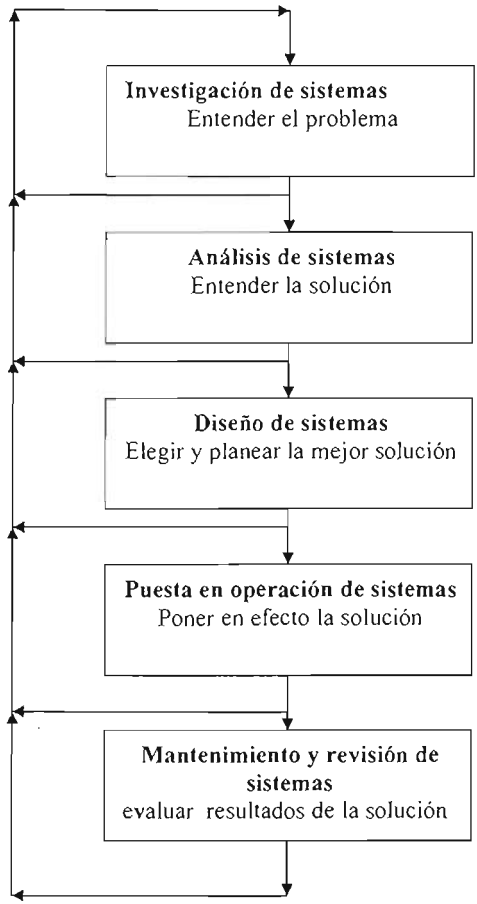


Figura 1.10

En la fase de **investigación de sistemas** se identifican los posibles problemas y oportunidades y se consideran a la luz de los objetivos de la empresa, la investigación de sistemas intenta responder a las preguntas: ¿Cuál es el problema? y ¿Vale la pena resolverlo? El resultado principal de esta fase es un proyecto de sistemas de información definido para el que se redactaron definiciones de problemas y oportunidades de negocios, y al cual se ha comprometido un monto específico de recursos de la administración y para el cual se recomienda el análisis de sistemas. **El análisis de sistemas** debe responder a la pregunta: ¿Qué debe hacer el sistema de información para resolver el problema? Esta fase incluye el estudio de los sistemas y procesos de trabajo existentes para identificar puntos

fuertes y débiles, así como oportunidades de mejoramiento. El resultado principal de este análisis es una lista de requisitos y prioridades. **El diseño de sistemas** busca responder a la pregunta: ¿De que manera el sistema de información hará lo necesario para resolver el problema? esta fase genera principalmente un diseño técnico en la cual se describe el nuevo sistema o las modificaciones que se harán a los sistemas existentes. En el diseño de sistemas se detallan las salidas y entradas del sistema, interfaces del usuario y componentes de hardware, software, base de datos, telecomunicaciones, personal y procedimientos además de mostrar la relación de todos estos componentes. **La puesta de operación de sistemas** consiste en crear o adquirir los diversos componentes del sistema detallados en el diseño de sistemas, conjuntarlos y poner en operación el sistema nuevo o modificado. La puesta en operación del sistema produce un sistema de información operativo que satisface las necesidades de la empresa para la cual se desarrollo. Por ejemplo el propósito del **mantenimiento y revisión de sistemas** es garantizar la operación del sistema y modificarlo de modo que continué cubriendo las necesidades cambiantes de la empresa.

El SDLC tradicional permite un alto grado de control administrativo. Al final de cada fase, se emprende una revisión formal y se toma la decisión de continuar el proyecto, interrumpirlo o quizá repetir algunas tareas de la fase actual. Además usar el SDCL tradicional crea mucha documentación, tales como los diagramas de entidad relación, si se mantiene actualizada esta documentación puede ser útil cuando llegue el momento de modificar el sistema. El SDLC tradicional también garantiza que cada requisito del sistema pueda relacionarse con una necesidad de la empresa, asimismo se pueden revisar los productos resultantes para verificar que satisfagan los requisitos del sistema y se ajusten a las normas de la organización.

Un problema importante con el SDLC tradicional es que el usuario no utiliza la solución hasta que el sistema esta casi terminado. Es frecuente que los usuarios reciban un sistema que no satisface sus necesidades reales, pues su desarrollo se baso en la interpretación que el grupo de desarrollo hizo de tales necesidades, además el método tradicional es inflexible, esto es, que no da cabida a cambios en las necesidades de los usuarios durante el desarrollo. Sin embargo, ello no ha impedido que este método todavía se utilice para sistemas grandes y complejos que

afectan a una compañía entera. Las ventajas y desventajas del SDLC tradicional se presentan en la tabla siguiente.

VENTAJAS	DESVENTAJAS
La revisión formal al final de cada fase permite el control administrativo máximo	Los usuarios reciben un sistema que satisface sus necesidades desde el punto de vista de los desarrolladores; puede ser que no corresponda a lo que en realidad se necesitaba
Este método genera la documentación considerable del sistema	La documentación es costosa y su creación requiere tiempo, también es difícil mantenerla actualizada
La documentación formal garantiza que sea posible vincular los requisitos de sistemas con las necesidades específicas de la empresa	Es frecuente que las necesidades de los usuarios no se expresen o se les interprete en forma correcta
Genera muchos productos intermedios que se pueden revisar con el fin de indagar si satisfacen o no las necesidades de los usuarios y se ajustan a los estándares	Los usuarios no pueden revisar fácilmente los productos intermedios y evaluar si un producto específico(por ejemplo: un diagrama de flujo de datos) satisface sus necesidades

Tabla 1.1

2. **Ciclo de vida de desarrollo de sistemas prototipo.** El uso de **prototipos** consiste en recurrir a un método iterativo en el proceso de desarrollo de sistemas. En cada iteración se identifica y analizan soluciones opcionales del problema, se diseñan nuevas soluciones y se pone en operación una parte del sistema.[23] Luego se instala a los usuarios para que prueben el prototipo y proporcionen retroalimentación como lo muestra la siguiente figura.

[23] Kenneth C. Laudon, Jane P. Laudon "Administración de los Sistemas de Información" Editorial Person Education

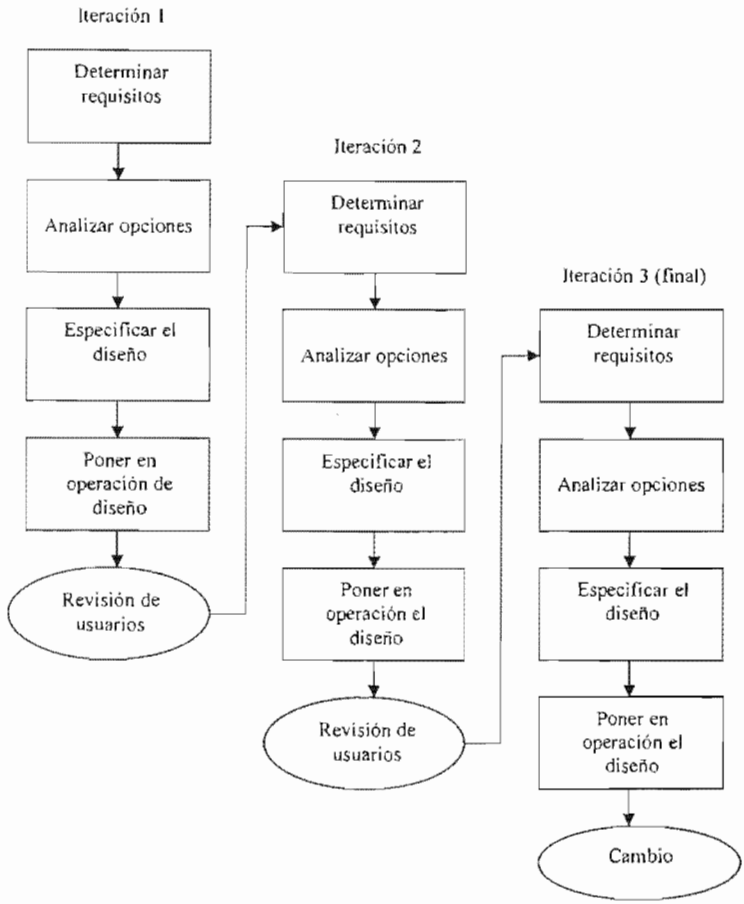


Figura 1.11

El proceso se inicia con la creación de un modelo preliminar de un subsistema principal o una versión escala del sistema completo. Por ejemplo, podría desarrollarse un prototipo que presente formatos de informes de muestra y pantallas de entrada de datos, una vez que se desarrollan y se mejoran los prototipos de informes y de pantallas se utilizan como modelos para el sistema real, que puede desarrollarse por medio de un lenguaje de programación de usuario final, tal como SAS, FOCUS, o VISUAL BASIC. El primer modelo preliminar se mejora para producir modelos de segunda y tercera generación, y así en forma sucesiva hasta que se desarrolla el sistema completo.

Existen dos tipos de prototipos los operativos y los no operativos. **Un prototipo operativo** es el que funciona; es decir, permite el acceso a archivos de datos reales y la edición de los datos de entrada, realiza los cálculos y comparaciones necesarios y produce una salida real.[24] Los informes financieros desarrollados por completo son ejemplo de ello. El prototipo operativo puede tener acceso a archivos reales, quizá sin permitir la edición de la entrada. **Un prototipo no operativo** es un modelo, posiblemente a escala, es característico que incluya los formatos y especificaciones de entrada y salida de datos, las salidas comprenden informes impresos para los administradores y el diseño en pantalla de informes desplegados en las computadoras personales o terminales. La entrada de datos incluye la forma de capturar los datos, los comandos que deben emplear los usuarios y la forma en que el sistema tiene acceso a otros archivos de datos. La ventaja principal de los prototipos no operativos es que se pueden desarrollar con mucho más rapidez que los operativos, este tipo de prototipos se pueden desechar y el sistema operativo completo se desarrolla con base en lo aprendido de ellos.[25]

Etapas en la construcción de prototipos:

Etapa 1. Identificar los requisitos básicos del usuario: El diseñador del sistema trabaja con el usuario solo lo suficiente para obtener sus necesidades básicas de información.

Etapa 2. Desarrollar un prototipo inicial: el diseñador del sistema crea rápidamente un prototipo operativo, muy probablemente usando las herramientas necesarias. El prototipo puede llevar a cabo sólo las funciones más importantes del sistema propuesto o puede ser todo el sistema con un archivo restringido.

Etapa 3. Uso del prototipo. Se estimula al usuario a que trabaje con el sistema con el objetivo de determinar que también satisface sus necesidades y para hacer recomendaciones para mejorarlo.

Etapa 4. Revisión y mejora del prototipo: el diseñador del sistema anota todos los cambios solicitados por el usuario y afina el prototipo de acuerdo con ellos.

[24] Kenneth C. Laudon, Jane P. Laudon "Administración de los Sistemas de Información" Editorial Pearson Education
[25] Ralph M. Stair, George W. Reynolds "Principios de los Sistemas de Información" 2000 Thomson Editores

Luego de que el prototipo ha sido revisado, el ciclo regresa a la etapa 3 y 4, que se repite hasta que el usuario quede satisfecho.

Cuando ya no se requieren más interacciones, el prototipo aprobado se transforma en un prototipo operativo que proporciona las especificaciones finales para la aplicación. Algunas veces el prototipo mismo se adopta como la versión definitiva de producción del sistema.[26]

Las ventajas y desventajas del uso de prototipos se describen en la tabla siguiente:

VENTAJAS	DESVENTAJAS
Los usuarios pueden probar el sistema y proporcionar retroalimentación constructiva durante su desarrollo	Cada iteración se basa en la iteración previa y depura de manera adicional la solución. Ello dificulta rechazar la solución inicial por inadecuada y empezar de nuevo. Así pues la solución final solo es incrementalmente mejor que la primera
Puede producirse un prototipo operativo en semanas	No se efectúan revisiones formales al final de cada fase, así que es muy difícil controlar el alcance de el prototipo y el proyecto parecería nunca terminar
Los usuarios adoptan una actitud mas positiva hacia la puesta en operación de el sistema, a medida que observan como surge una solución que satisface sus necesidades	Es frecuente que se carezca de documentación del sistema o que esta sea incompleta, pues el método principal se centra en el desarrollo de los prototipos
El uso de los prototipos permite la detección temprana de errores y omisiones	Pueden omitirse aspectos de respaldo, recuperación, rendimiento y seguridad de el sistema, en la prisa por desarrollar el prototipo

Tabla 1.2

Como se describió, los sistemas de información son de suma importancia para la toma de decisiones y ya que estos transforman simples datos en información, pero para que esto sea posible se tiene que contar con recursos económicos y esto va a derivar en recursos materiales y tecnológicos. Y para que los sistemas de información tengan éxito, deben de seguir metodologías paso a paso, llegando así a la importancia de utilizar del **hardware y software**, este último con la responsabilidad del desempeño general del sistema. El software debe estar basado

[26] Kenneth CV. Laudon, Jane P. Laudon "Administración de los Sistemas de Información" Editorial Pearson Education

en técnicas o procesos para una buena calidad ya que generalmente este se hace en forma "artesanal" es decir, cada ingeniero o persona que desarrolla un software lo hace de acuerdo a su propio conocimiento sin seguir ninguna metodología o normatividad.

A continuación mencionan algunas metodologías generales del software así como los problemas y la necesidad de un desarrollo adecuado.

1.2 Creación del Software

1.2.1 Técnicas para el desarrollo del software

Las técnicas que se usan para el desarrollo del software son muchas y variadas. El desarrollo del software va desde una forma prácticamente artesanal, hasta modelos complejos para el desarrollo del software, intentando llegar a modelos de sistemas de calidad para el software, donde estos modelos según sus acuerdos garantizan la calidad total del software. A continuación mencionaremos los modelos y técnicas tradicionales así como los problemas que puedan tener estos modelos.

Los programadores utilizan diversas herramientas, técnicas y métodos; entre estos se pueden mencionar:

- **Diseño estructurado:** es un método eficaz de conocimiento para designar y desarrollar software de aplicación. El objetivo general del diseño estructurado es desarrollar software más eficiente para reutilizarlo en procedimientos y métodos para resolver una variedad de problemas. Este diseño divide en un problema grande y complejo en otros menores, cada uno de ellos lo suficientemente sencillo para manejarlo y resolverlo en forma independiente. Luego estos módulos se pueden utilizar en otros programas nuevos. El desarrollo y mantenimiento de este software modular o de bloques por lo común es menos costoso, además de que se facilitan tanto su modificación como actualizaciones futuras. Por ejemplo puede desarrollarse en forma independiente un módulo para generar un cierto tipo de informe y luego incluirlo en diversos programas que requieran dicho informe. Una manera de poner en práctica el diseño estructurado es la programación estructurada.
- **Programación estructurada:** en muchos proyectos de programación el mantenimiento y actualización de los programas pueden requerir más tiempo

y esfuerzo que el proceso de desarrollo original. La programación estructurada facilita y acelera el mantenimiento de programas, por lo que esta técnica de programación es importante. Un objetivo fundamental de este tipo de programación es desempeñar y reducir la complejidad de los programas de computación. Los programas estructurados se originan de acuerdo a las operaciones que ellos ejecutan. En esencia el programa se descompone en procedimientos individuales (también conocidas como funciones) cada uno de los cuales se descompone en subprocedimientos hasta llegar al nivel de procedimiento individual sencillo. Como se menciona el concepto básico es mejorar la lógica del flujo de un programa al dividirlo en grupo de instrucciones, estos grupos se forman según las funciones que tienen las instrucciones. Así un grupo leería datos y otro realizaría una cierta tarea de procedimientos. Cuando se usa la programación estructurada las instrucciones de un grupo deben ajustarse a una estructura estándar. Por principio de cuentas solo debe haber un punto de entrada al grupo y otro punto de salida. Aislando los procesos dentro de funciones un programa estructurado minimiza el riesgo de que un procedimiento afecte a otro. Esto facilita también la detección de problemas mediante la localización de errores y hace el programa más claro. Un concepto importante introdujo la programación estructurada "**abstracción**". La abstracción es la capacidad de examinar algo sin preocuparse de los detalles internos. En un programa estructurado es suficiente conocer que un procedimiento dado realiza una tarea específica. El como se realiza una tarea no es importante, mientras que el procedimiento sea fiable se puede utilizar sin saber como funciona su interior. Esto se conoce como una abstracción funcional y esto es núcleo de la programación estructurada.

En la programación estructurada solo se permiten tres tipos de estructuras. **La estructura secuencial,** en este tipo de estructura debe haber puntos de entrada y salida definidos. Después de iniciarse la secuencia las instrucciones del programa se ejecutan una tras otra hasta terminar con todas las instrucciones de la secuencia. Luego el programa termina o continúa con otra secuencia. **La estructura de decisión** permite que la computadora se ramifique si se cumplen ciertas condiciones, lo común es que existan solo dos

ramificaciones posibles. El último tipo de estructura es **la estructura de ciclo,** en realidad existen dos estructuras de uso común con relación a los ciclos. Una es la *do-until (continuar hasta que)* y la estructura *do-while (continuar mientras que).* Ambos llevan a cabo la misma tarea. En el ciclo *do-until* el ciclo continúa hasta que se satisface una condición específica. En la estructura *do-while* el ciclo termina cuando existe una cierta condición. En la figura siguiente se muestran los tipos de estructuras.[27]

- **Herramientas CASE**: Estas herramientas son de uso frecuente en el desarrollo de software para automatizar algunas técnicas. Por ejemplo con ellas se puede generar en forma automática el código fuente. De los tipos de herramientas CASE las de tipo avanzado son las de uso más probable en la programación del software. Permite contar con un ambiente gráfico de programación e incluye compiladores, verificadores de sintaxis, y módulos de software que generan el código de programa real.[28]

Como se vio estas son algunos de los diferentes tipos que un programador puede ocupar para realizar el código de un programa a continuación se mencionan las herramientas para su desarrollo.

1.2.2 Actividades para la creación de software.

Para el desarrollo del software se necesitan de diferentes actividades para organizar el proceso del software, estas se basan de diferentes métodos para el desarrollo del software, pero por lo general estas características son: **Especificación, Diseño, y Evolución.**[29]

Para la administración debe de existir previamente un modelado. Para hacer el modelado tiene un proceso el cual consta de varias etapas:

- **Especificación:** Es que la lleva a cabo el esclarecimiento de los requerimientos del cliente, también establece las restricciones.
- **Diseño:** Este diseña los requerimientos antes de pasarlos a la computadora.

[27] Luis Joyanes Aguilar "C++ A su Alcance Un enfoque Orientado a Objetos" Editorial McGrawHill
[28] Ralph M. Stair, George W. Reynolds "Principios de los Sistemas de Información" 2000 Thomson Editores
[29] Revision Internet, http://www.itsteziutlan.edu.mx/crisis.htm

- **Manufactura:** Es donde se desarrolla el sistema.
- **Prueba:** Asegurarse de que el sistema funcione tratando de tronarlo.
- **Instalación:** Es la entrega del sistema al usuario.
- **Mantenimiento:** Es mantener una constante revisión del sistema para ver si no hay problemas.

1.2.3 Ciclo de Vida del Software.

Un modelo de ciclo de vida define el estado de las fases a través de las cuales se mueve un proyecto de desarrollo de software.

El primer ciclo de vida del software, "Cascada", fue definido por Winston Royce a fines del 70. Desde entonces muchos equipos de desarrollo han seguido este modelo. Sin embargo, ya desde 10 a 15 años atrás, el modelo cascada ha sido sujeto a numerosas críticas, debido a que es restrictivo y rígido, lo cual dificulta el desarrollo de proyectos de software moderno. En su lugar, muchos modelos nuevos de ciclo de vida han sido propuestos, incluyendo modelos que pretenden desarrollar software más rápidamente, o más incrementalmente o de una forma más evolutiva, o precediendo el desarrollo a escala total con algún conjunto de prototipos rápidos.[30]

1.2.3.1 Definición de un modelo de ciclo de vida.

Un modelo de ciclo de vida de software es una vista de las actividades que ocurren durante el desarrollo de software, intenta determinar el orden de las etapas involucradas y los criterios de transición asociadas entre estas etapas.

Un modelo de ciclo de vida del software: [31]

- Describe las fases principales de desarrollo de software.
- Define las fases primarias esperadas de ser ejecutadas durante esas fases.
- Ayuda a administrar el progreso del desarrollo, y
- Provee un espacio de trabajo para la definición de un detallado proceso de desarrollo de software.

Así, los modelos por una parte suministran una guía para los ingenieros de software con el fin de ordenar las diversas actividades técnicas en el proyecto, por otra parte suministran un marco para la administración del desarrollo y el mantenimiento, en el

[30] http://academicos.cualtos.udg.mx/Informatica/Ceneval2003/Ciclo%20de%20Vida%20del%20Software.doc
[31] http://academicos.cualtos.udg.mx/Informatica/Ceneval2003/Ciclo%20de%20Vida%20del%20Software.doc

sentido en que permiten estimar recursos, definir puntos de control intermedios, monitorear el avance, etc.

Existen varios tipos de modelado entre los más conocidos están:

- **Modelo de Cascada.** Este separa los prototipos en fases separadas. Este es el más básico de todos los modelos, y sirve como bloque de construcción para los demás modelos de ciclo de vida. La visión del modelo cascada del desarrollo de software es muy simple; dice que el desarrollo de software puede ser a través de una secuencia simple de fases. Cada fase tiene un conjunto de metas bien definidas, y las actividades dentro de una fase contribuye a la satisfacción de metas de esa fase o quizás a una subsecuencia de metas de la fase. Las flechas muestran el flujo de información entre las fases. La flecha de avance muestra el flujo normal. Las flechas hacia atrás representan la retroalimentación.

  El modelo de ciclo de vida cascada, captura algunos principios básicos:

  - Planear un proyecto antes de embarcarse en él.
  - Definir el comportamiento externo deseado del sistema antes de diseñar su arquitectura interna.
  - Documentar los resultados de cada actividad.
  - Diseñar un sistema antes de codificarlo.
  - Probar un sistema después de construirlo.

- **Desarrollo Evolutivo**. El desarrollo y la evolución están intercalados. Como el modelo de desarrollo incremental, el modelo de desarrollo evolutivo (algunas veces denominado como prototipado evolutivo) construye una serie de grandes versiones sucesivas de un producto. Sin embargo, mientras que la aproximación incremental presupone que el conjunto completo de requerimientos es conocido al comenzar, el modelo evolutivo asume que los requerimientos no son completamente conocidos al inicio del proyecto.

  En el modelo evolutivo, los requerimientos son cuidadosamente examinados, y sólo esos que son bien comprendidos son seleccionados para el primer incremento. Los desarrolladores construyen una implementación parcial del sistema que recibe sólo estos requerimientos.

  El sistema es entonces desarrollado, los usuarios lo usan, y proveen retroalimentación a los desarrolladores. Basada en esta retroalimentación, la

especificación de requerimientos es actualizada, y una segunda versión del producto es desarrollada y desplegada. El proceso se repite indefinidamente.

El desarrollo evolutivo es 100% compatible con el modelo cascada. El desarrollo evolutivo no demanda una forma específica de observar el desarrollo de algún incremento. Así, el modelo cascada puede ser usado para administrar cada esfuerzo de desarrollo. Obviamente, el desarrollo incremental y evolutivo puede ser combinado también.

Todo lo que uno tiene que hacer es construir un subconjunto de requerimientos conocidos (incremental), y comprender al principio que muchos nuevos requerimientos es probable que aparezcan cuando el sistema sea desplegado o desarrollado.

El desarrollo de software en forma evolutiva requiere un especial cuidado en la manipulación de documentos, programas, datos de prueba, etc. desarrollados para distintas versiones del software. Cada paso debe ser registrado, la documentación debe ser recuperada con facilidad, los cambios deben ser efectuados de una manera controlada.

- **Prototipado o de requerimientos.** Este es un prototipo del sistema final. El prototipado de requerimientos es la creación de una implementación parcial de un sistema, para el propósito explícito de aprender sobre los requerimientos del sistema. Un prototipo es construido de una manera rápida tal como sea posible. Esto es dado a los usuarios, clientes o representantes de ellos, posibilitando que ellos experimenten con el prototipo. Estos individuos luego proveen la retroalimentación sobre lo que a ellos les gustó y no les gustó acerca del prototipo proporcionado, quienes capturan en la documentación actual de la especificación de requerimientos la información entregada por los usuarios para el desarrollo del sistema real. El prototipado puede ser usado como parte de la fase de requerimientos (determinar requerimientos) o justo antes de la fase de requerimientos (como predecesor de requerimientos). En otro caso, el prototipado puede servir su papel inmediatamente antes de algún o todo el desarrollo incremental en modelos incremental o evolutivo.

  El Prototipado ha sido usado frecuentemente en los 90, porque la especificación de requerimientos para sistemas complejos tiende a ser

relativamente dificultoso de cursar. Muchos usuarios y clientes encuentran que es mucho más fácil proveer retroalimentación convenientemente basada en la manipulación que leer una especificación de requerimientos potencialmente ambigua y extensa.

Diferente del modelo evolutivo donde los requerimientos mejor entendidos están incorporados, un prototipo generalmente se construye con los requerimientos entendidos más pobremente.

En caso que ustedes construyan requerimientos bien entendidos, el cliente podría responder con "sí, así es", y nada podría ser aprendido de la experiencia.

- **Modelo Espiral.** El modelo espiral de los procesos para el software es un modelo del ciclo de *meta-vida*. En este modelo, el esfuerzo de desarrollo es iterativo. Tan pronto como uno completa un esfuerzo de desarrollo, otro comienza. Además, en cada desarrollo ejecutado, puedes seguir estos cuatros pasos:
  - Determinar qué quieres lograr.
  - Determinar las rutas alternativas que puedes tomar para lograr estas metas. Por cada una, analizar los riesgos y resultados finales, y seleccionar la mejor.
  - Seguir la alternativa seleccionada en el paso 2.
  - Establecer qué tienes terminado.

El modelo espiral captura algunos principios básicos:

- Decidir qué problema se quiere resolver antes de viajar a resolverlo.
- Examinar tus múltiples alternativas de acción y elegir una de las más convenientes.
- Evaluar qué tienes hecho y qué tienes que haber aprendido después de hacer algo.
- No ser tan ingenuo para pensar que el sistema que estás construyendo será "EL" sistema que el cliente necesita.
- Conocer (comprender) los niveles de riesgo, que tendrás que tolerar.

- **Transformación Formal.** Se basa en un modelo matemático del sistema.
- **Desarrollo basado en reutilización.** Sistema como dice se basa en la reutilización de componentes ya existentes.

A continuación se explica el Modelado de Cascada.

Este consta de varias etapas las cuales son secuenciales de ahí su nombre de cascada y el diagrama es de esta manera:

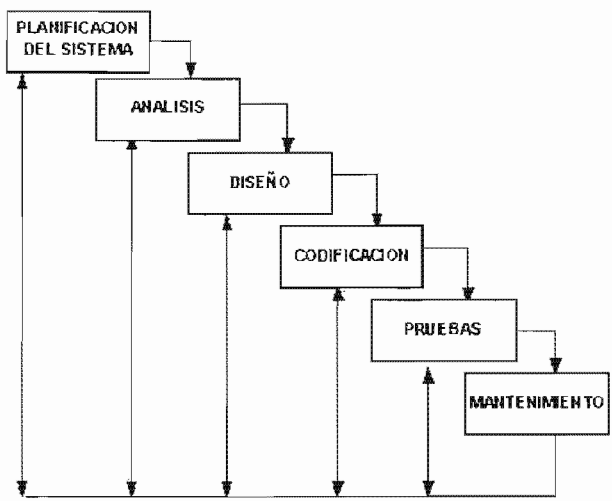


FIGURA 1.12

Las etapas de modelo son las siguientes:

- Análisis de requerimientos.
- Diseño de los requerimientos.
- Implementación y codificación.
- Pruebas de unidades y de sistemas.
- Mantenimiento y operación del sistema.

Este modelo como los demás tiene un gran problema. El principal problema de este es que como es secuencial y solo tiene una fase de pruebas y hasta que se llegue a esta se hace lo mejor por resolver los problemas pero después de esta fase si

existen mas problemas ya nos se pueden arreglar y no se puede hacer cambios entre las etapas.

Este es un ejemplo de modelado así como se menciono existen varios tipos de modelados y son los principales métodos para la ingeniería de software ya que ayuda a la organización para el desarrollo del software.

1.2.3.2 Modelo UML

Existe otro tipo de modelado de sistemas desarrollado para varios tipos de sistemas, el UML, que a continuación se describirá brevemente:

El Lenguaje Unificado de Modelado (UML por sus siglas en ingles) prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan. Mientras que ha habido muchas notaciones y métodos usados para el diseño orientado a objetos, ahora los modeladores sólo tienen que aprender una única notación.

UML se puede usar para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware, y organizaciones del mundo real. UML ofrece nueve diagramas en los cuales modelar sistemas:

- **Diagramas de Casos de Uso:** para modelar los procesos 'business'.
- **Diagramas de Secuencia:** para modelar el paso de mensajes entre objetos.
- **Diagramas de Colaboración**: para modelar interacciones entre objetos.
- **Diagramas de Estado:** para modelar el comportamiento de los objetos en el sistema.
- **Diagramas de Actividad:** para modelar el comportamiento de los Casos de Uso, objetos u operaciones.
- **Diagramas de Clases:** para modelar la estructura estática de las clases en el sistema.
- **Diagramas de Objetos**: para modelar la estructura estática de los objetos en el sistema.
- **Diagramas de Componentes:** para modelar componentes.
- **Diagramas de implementación:** para modelar la distribución del sistema.

UML es una consolidación de muchas de las notaciones y conceptos más usados orientados a objetos. Empezó como una consolidación del trabajo de Grade Booch, James Rumbaugh, e Ivar Jacobson, creadores de tres de las metodologías orientadas a objetos más populares. En 1996, el Object Management Group (OMG), un pilar estándar para la comunidad del diseño orientado a objetos, publicó una petición con propósito de un metamodelo orientado a objetos de semántica y notación estándares. UML, en su versión 1.0, fue propuesto como una respuesta a esta peticíón en enero de 1997. Hubo otras cinco propuestas rivales. Durante el transcurso de 1997, los seis promotores de las propuestas, unieron su trabajo y presentaron al OMG un documento revisado de UML, llamado UML versión 1.1. Este documento fue aprobado por el OMG en Noviembre de 1997. El OMG llama a este documento OMG UML versión 1.1. UML prescribe unas notaciones estándar y semánticas esenciales para el modelado de un sistema orientado a objetos. Previamente, un diseño orientado a objetos podría haber sido modelado con cualquiera de la docena de metodologías populares, causando a los revisores tener que aprender las semánticas y notaciones de la metodología empleada antes que intentar entender el diseño en sí. Ahora con UML, diseñadores diferentes modelando sistemas diferentes pueden sobradamente entender cada uno los diseños de los otros. Aun así, UML no prescribe un proceso o método estándar para desarrollar un sistema. Hay varias metodologías existentes, entre las más comunes se encuentran:

- **Catalysis:** Un método orientado a objetos que fusiona mucho del trabajo reciente en métodos orientados a objetos, y además ofrece técnicas específicas para modelar componentes distribuidos.
- **Objetory:** Un método de Caso de Uso guiado para el desarrollo, creado por Ivar Jacobson.
- **Shlaer/Mellor:** El método para diseñar sistemas de tiempo real, puesto en marcha por Sally Shlaer y Steven Mellor en dos libros de 1991, Ciclos de vida de Objetos, modelando el Mundo en Estados y Ciclos de vida de Objetos, Modelando el mundo en Datos (Prentice Hall). Shlaer/Mellor countinúan actualizando su método continuamente (la actualización más reciente es el OOA96 report), y recientemente publicaron una guía sobre cómo usar la notación UML con Shlaer/Mellor.

- **Fusion**: Desarrollado en Hewlett Packard a mediados de los noventa como primer intento de un método de diseño orientado a objetos estándar. Combina OMT y Booch con tarjetas CRC y métodos formales.
- **OMT:** La Técnica de Modelado de Objetos fue desarrollada por James Rumbaugh y otros, y publicada en el libro de gran influencia "Diseño y Modelado Orientado a Objetos" (Prentice Hall, 1991). Un método que propone análisis y diseño 'iterative', más centrado en el lado del análisis.
- **Booch:** Parecido al OMT, y también muy popular, la primera y segunda edición de "Diseño Orientado a Objetos, con Aplicaciones" (Benjamin Cummings, 1991 y 1994), (Object-Oriented Design, With Applications), detallan un método ofreciendo también diseño y análisis, centrándose en el lado del diseño

Además, muchas organizaciones han desarrollado sus propias metodologías internas, usando diferentes diagramas y técnicas con orígenes varios. Ejemplos son el método Catalyst por Computer Sciences Corporation (CSC) o el Worlwide Solution Design and Delivery Method (WSDDM) por IBM. Estas metodologías difieren, pero generalmente combinan análisis de flujo de trabajo, captura de los requisitos, y modelado de negocio con modelado de datos, con modelado de objetos usando varias notaciones (OMT, Booch, etc), y algunas veces incluyendo técnicas adicionales de modelado de objetos como Casos de Uso y tarjetas CRC. La mayoría de estas organizaciones están adoptando e incorporando el UML como la notación orientada a objetos de sus metodologías. Algunos modeladores usarán un subconjunto de UML para modelar, por ejemplo simplemente el diagrama de clases, o solo los diagramas de clases y de secuencia con Casos de Uso. Otros usarán una suite más completa, incluyendo los diagramas de estado y actividad para modelar sistemas de tiempo real, y el diagrama de implementación para modelar sistemas distribuidos. Aun así, otros no estarán satisfechos con los diagramas ofrecidos por UML y necesitaran extender UML con otros diagramas como modelos relacionados de datos.[32]

[32] G. Booch, J. Rumbagh y I. Jacobson "El lenguaje Unificado de Modelado" Addison Wesley

Como se vio existen diferentes tipos de modelados con ventajas y desventajas. Algunos programadores prefieren modelos mas complejos como el UML ya que este puede funcionar para varios tipos de modelaje que se quiera hacer y otros prefieren modelos prácticos. Sea cual fuere su decisión en el desarrollo del software siempre si tienen problemas; ya sea por su diseño, por no seguir correctamente las metodologías o por factores que están fuera del alcance del programador. A continuación se mencionan los problemas en el software.

1.2.4 Problemáticas en el desarrollo del software

En si son muchos los problemas que se tienen al desarrollar un software. La complejidad de los sistemas informáticos hace a veces necesario el desarrollo de proyectos de software de decenas de miles de líneas de código. Esto no puede ser abordado directamente empezando a programar. Es necesario analizar que es lo que se tiene que hacer, como se va a hacer, como se van a coordinar todas las personas que van a intervenir en el proyecto y como vamos a controlar el desarrollo del mismo, de forma que al final se obtengan los resultados esperados. Las metodologías convencionales de ingeniería del software tienen mecanismos robustos para hacer un análisis de necesidades y diseño de los sistemas, poco han evolucionado con la tecnología relacionado con el diseño computacional. Además cuando los problemas del mundo real se desean resolver con los modelos de sistemas computacionales, esto trae consigo una infinidad de requisitos que compiten entre si y algunos a veces se contradicen. Dar funcionalidad a un sistema es difícil e incluso comprender los requerimientos como: facilidad de uso, rendimiento, costo, capacidad de supervivencia, fiabilidad, entre otros son parte de la complejidad externa que infiere determinantemente en la complejidad interna del sistema. Bajo este contexto nace la importancia de la relación entre desarrolladores y usuarios del sistema. Habitualmente los usuarios suelen tener dificultades en expresar sus necesidades e ideas, esto se da en ocasiones por falta de conocimiento del ámbito de cada uno de los grupos, los usuarios y los desarrolladores tienen perspectivas diferentes sobre la naturaleza de los problemas. Contar con herramientas y metodologías adecuadas que permitan plasmar estos requisitos en forma clara para ambos grupos sería un punto adecuado para resolver los problemas de desarrollo del software.

Por otra parte un programa grande implica la generación de gran cantidad de código. Lo cual trae como consecuencia cierta dificultad de gestionar el desarrollo del software, debido a que gran cantidad de código por lo general requiere ser divididos en módulos (subprogramas), los cuales a la vez requieren de un equipo de desarrolladores, el cual de no ser pequeño tiende a producir problemas de comunicación y coordinación (más si se encuentran dispersos geográficamente).
Algunos de los síntomas de los problemas que se presentan en el desarrollo del software y que deben ser tratados de manera prioritaria para llegar a un producto satisfactorio son:

- Entendimiento inadecuado de las necesidades del usuario final.
- Incapacidad de manejar requerimientos cambiantes.
- Módulos que no encajan.
- Software que es difícil de mantener.
- Descubrimiento tardío de fallas serias en el proyecto.
- Calidad pobre del software.
- Desempeño inaceptable.
- Miembros del equipo bloquean a los demás: incapacidad de reconstruir, quién cambió qué, cuándo, dónde y por qué.
- Administración de requerimientos improvisada.
- Comunicación ambigua e imprecisa.
- Arquitectura de software frágil.
- Inconsistencias no detectadas entre requerimientos, diseños e implementaciones.
- Pruebas insuficientes.
- Evaluación subjetiva del estatus del proyecto.
- Propagación descontrolada de cambios al software.
- Automatización insuficiente.

1.2.5 Desventajas de los modelos tradicionales

Los tipos de modelos que se describieron con anterioridad cuentan con problemas distintos, uno de los principales problemas es que son secuencial y solo tiene una fase de pruebas y hasta que se llegue a esta se hace lo mejor por resolver los

problemas pero después de esta fase si existen mas problemas ya nos se pueden arreglar y no se puede hacer cambios entre las etapas. Las desventajas que se tienen con estos modelos es que la mayoría de las veces solo se siguen algunos pasos si es que se tomo algún tipo de modelo para desarrollar el software. Además estas metodologías se realizan sin el concepto de **calidad en el software,** es decir solo se dirigen al análisis, diseño y codificación y no se enfocan en las herramientas que se pueden tomar de la calidad total.

Por otra parte los modelos de desarrollo del software no son los adecuados para las aplicaciones y necesidades de hoy en día, debido principalmente a los siguientes aspectos:

- Nuevos modelos de mercado se están definiendo ahora y tenderán a multiplicarse (e-business).
- El panorama del mercadeo y las ventas influyen sobre el desarrollo de sistemas.
- Antes el software era un soporte al negocio, ahora el software es "el negocio".

Como se puede observar, los aspectos anteriores impactan directamente en el software dado que el mundo depende cada día más de él y hacer un buen software es difícil si se toma en cuenta que cada vez es más complejo y no se puede tomar mucho tiempo para hacerlo, además, el proceso de desarrollo no está aislado dado que tiene muchas dependencias. Por otra parte, los equipos de desarrollo casi nunca conocen tanto del negocio como del proceso de construcción del software.

1.3 Necesidad de un desarrollo adecuado y práctico

El desarrollo del software con metodologías adecuadas son necesarias para un buen desempeño de este, ya que en una conferencia patrocinada por la OTAN, con el título de *Ingeniería del software y crisis del software*, se hizo ver que en el desarrollo del software se posee una crisis, en donde, el software es caro, poco fiable y escaso. Las metodologías y técnicas estructurales (de los 70's y 80's) no eliminaron tal crisis, es más está continua hoy en día pese al diseño descendente (este se debe a que la complejidad también ha crecido). Para darse una idea de esta crisis se tiene que los pasos de análisis, diseños, implementación, depuración y mantenimiento, cuentan con los siguientes costos respectivamente 6%, 5%, 7%, 15% y 67%, en

donde se puede notar que la fase de mantenimiento (cambios realizados en la evolución de un programa) resulta lo que posee mayor costo, esto se debe a que esto es el punto débil de los métodos tradicionales. Cuando no se toman las metodologías ni las herramientas adecuadas los efectos de errores del software van mucho más allá del costo de dedicar tiempo a los programadores. Es necesario utilizar las metodologías adecuadas para un desarrollo óptico y de calidad ya que los errores de software pueden causar perdidas de ingresos y de oportunidades de mercado, así como procesamiento incorrecto de la información, todo esto reduce las utilidades de una empresa.

Por ejemplo en un sistema de información, este debe de procesar los datos en información, una nómina, el personal que trabaja tiempo extra si este no fue registrado si el software no hizo el proceso adecuado de convertir los datos de horas trabajadas, simplemente el usuario no tendrá el pago correspondiente a las horas de tiempo extra, o viceversa si el trabajador no hizo tiempo extra el sistema puede asumir que si lo trabajo y así pagar un tiempo que nunca fue trabajado. En la mayoría de los países que desarrollan software o aplicaciones de software y principalmente Estados Unidos, el problema radica en que las compañías tienen tanto éxito como innovadoras de software que no dan importancia a la **disciplina del proceso** adecuada, que incluye prueba de los programas y por supuesto la administración completa del desarrollo. Esto puede producir un programa que no siempre funcione con forme a lo que se anuncie de él y proyectos desbocados cuya administración requiere personal adicional. Las organizaciones y programadores deben de utilizar procesos de desarrollo y metodologías de sistemas estandarizados y medir estos procesos para identificar los proyectos en los que pueden surgir problemas. El mejorar las prácticas de desarrollo de software es una solución a largo plazo que requiere de inversión de tiempo y dinero y solo así es posible que el software tenga un respaldo de producción o desarrollo.

Con esto, se puede concluir que las necesidades de un desarrollo adecuado del software es de suma importancia, ya que este el soporte de cualquier empresa, no importa a que se dedique ya que este se puede utilizar como software mismo, como software que maneje algún dispositivo (máquina, dispositivos médicos, etc.), software para desarrollar aplicaciones, y si no tienen un buen desarrollo esto puede producir colapsos en estas empresas. Existen como ya se vio un ciclo de vida del

desarrollo del software así como una variedad de metodologías, todas ellas con beneficios para un buen desarrollo así como sus desventajas, estas metodologías no se enfocan en una calidad del software es decir, que no tomas herramientas de calidad para un buen desarrollo del software, a continuación se mencionara una herramienta que es de suma importancia para la calidad en el software.

1.4 Validación como herramienta para la calidad del software

Para un buen desarrollo y una buena calidad en el software, se necesita de diferentes actividades en los modelos de calidad para el software, entre ellas una de las más importantes es la validación ya que esta es la que prácticamente califica el proceso de desarrollo y la que pone parámetros a la administración.

El proceso de validación es una clave importante para asegurar que ese aseguramiento de calidad llegue a su meta.

El proceso de **validación** es establecer evidencia documentada la cual proporcione un alto grado de seguridad de que un proceso especifico producirá consistentemente un producto con las especificaciones predeterminadas y los atributos de calidad apropiada. La validación es una parte importante del programa de aseguramiento de calidad y es fundamental para una eficiente operación de producción. La validación es un estudio científico de un proceso que sirve para:

- Para demostrar que el proceso este haciendo consistentemente lo que se supone que tiene que hacer.
- Para determinar las variables del proceso y los límites aceptables para esas variables.

En resumen hay tres razones de el porque se tienen que validar los procesos. La principal razón es que la validación es esencial para el aseguramiento de la calidad; la segunda razón es la reducción de costos y por ultimo es un regulador de requisitos.

La validación de procesos requiere una calificación de cada elemento importante. La importancia relativa de cada elemento puede variar de proceso a proceso.

La garantía de la calidad del producto se deriva de la cuidadosa atención de un número de factores que incluyen la selección de partes de calidad, productos adecuados y diseño de procesos, control de los procesos y prueba de estos. Los

principios básicos del aseguramiento de la calidad tienen como su meta la producción de resultados o artículos que están hechos para su uso intencionado. Estos principios pueden ser declarados como sigue: (1) calidad, seguridad y efectividad deben estar diseñados y construidos dentro del producto; (2) la calidad no puede ser inspeccionada o probada dentro de la finalización del producto; y (3) cada paso del proceso de manufactura debe ser controlado para maximizar la probabilidad de que el producto finalizado conozca todas las especificaciones de calidad y diseño. Por lo tanto el proceso de validación es un elemento clave para asegurar la calidad.

CAPITULO 2

MODELOS DE CALIDAD PARA EL SOFTWARE

Como se vio en el capítulo anterior la programación o codificación de los programas se hacen con herramientas o estándares que existen, muchos de ellos en ocasiones son obsoletos e inadecuados para un buen desarrollo de software, ya que estos no se enfocan en las herramientas de calidad adecuadas el desarrollo del software, ya que estas pueden ayudar en gran cantidad a realizar un proyecto de software fuera de errores o imponer reglas o lineamientos para un seguimiento claro de las metodologías o modelos de calidad para el software. Este último ya es un proyecto completo de calidad, debido a que cuenta con las herramientas necesarias de calidad para su realización completa y en algunos casos estos modelos pueden ayudar a una certificación, que en estos tiempos de globalización es muy requerida ya que en transacciones con países de primer mundo es necesario seguir normas que certifiquen.

2.1 Definición y conceptos de calidad

Calidad es un término difícil de definir, principalmente porque se ha mantenido en constante evolución, por lo que cada definición que se presente debe insertarse en el contexto de la época.

En general, se puede decir que calidad abarca todas las cualidades con las que cuenta un producto o un servicio para ser de utilidad a quien se sirve de él.[33]

Esto es, un producto o servicio es de calidad cuando sus características tangibles e intangibles, satisfacen las necesidades de sus usuarios. Funciones operativas, el precio y la economía de uso, durabilidad, seguridad, facilidad adecuación de uso, que sea simple de manufacturar y de mantener en condiciones operativas, que sea fácil de desechar, todo esto le otorga a un producto **LA CALIDAD AL CONSUMIDOR.**[34]

[33] Humberto Cantú Delgado "Desarrollo de una Cultura de Calidad" Editorial McGrawHill
[34] Humberto Cantú Delgado "Desarrollo de una Cultura de Calidad" Editorial McGrawHill

Así por ejemplo:

- La NORMA JISZ8101 define al control de calidad como un sistema que permite que las características de un producto o servicio satisfagan en forma económica los requerimientos del consumidor.
- La NORMA ANSIZI.7-1971 dice que son las técnicas operacionales y actividades que sustentan la calidad de un producto o servicio para satisfacer ciertas necesidades.
- La NORMA ISO9000 interpretan la calidad como la integración de las características que determinan en que grado un producto satisface las necesidades del consumidor.

2.1.1 La administración de calidad total (TQM)

Es aquí donde se hace hincapié en el mercado y las necesidades del consumidor, reconociendo el efecto estratégico de la calidad en el proceso de competitividad. Por sus siglas en ingles es un concepto que hace de la calidad una responsabilidad total a ser compartidas por todas las personas dentro de una institución, con el alcance del control de calidad considerado como un fin en sí mismo. Se espera que todos contribuyan en la mejora general de la calidad: el ingeniero que evita las fallas de diseño, el trabajador de producción que detecta los errores, el representante de ventas que presenta el producto adecuadamente y aun la secretaria que evita las faltas al escribir. La administración de la calidad total abarca todas las funciones de la administración. La administración de la calidad total fue popularizada por los japoneses, quienes cedieron la responsabilidad de la consistencia en la calidad a los trabajadores, que de hecho fabrican el producto o hacen el servicio, en oposición al departamento de control de calidad. La administración japonesa adopto la meta de cero defectos enfocándose en la mejora de sus productos y servicios antes de la entrega, en vez de corregirlos luego. Los estudios han demostrado repetidamente que mientras mas temprano en el ciclo de los negocios se detecte un problema menos cuesta a la empresa su eliminación. Entonces el enfoque hacia la calidad de los japoneses no solo trajo un cambio en el enfoque hacia los trabajadores y un respeto creciente hacia la calidad del producto y el servicio, sino que abatió los costos, este movimiento hacia la calidad se ha extendido a Europa y los Estados Unidos y hace algunos años atrás se introdujo a México con el fin de elevar la

calidad en los productos y en los servicios para poder competir económicamente con países de primer mundo.

2.1.2 Modelos de calidad

La elección de las herramientas adecuadas para el desarrollo de proyectos de calidad en una organización es fundamental para el éxito de dichas iniciativas.
Generalmente, las organizaciones de clase mundial acuden a una combinación de modelos y metodologías para lograr resultados tangibles en su sistema de calidad, debido a que cada esquema provee elementos y enfoques diferentes que son complementarios entre sí. Los beneficios de los modelos se pueden potenciar con la aplicación de metodologías o normatividad, y viceversa. Casarse con una sola herramienta de calidad puede limitar los logros de las organizaciones en materia de calidad y llevarlas a dejar pasar oportunidades importantes, por lo que un enfoque multidisciplinario ayuda a una mayor rapidez en la maduración de los sistemas de calidad, así como a hacer más amplio y efectivo el panorama de opciones para el logro de resultados en beneficio de los clientes y de la organización.

2.2 Necesidades del aseguramiento de calidad en el software

"La calidad del software es el grado con el que un sistema, componente o proceso cumple con los requerimientos especificados y las necesidades o expectativas del cliente o usuario (IEEE, Std 610-1990)"

"Concordancia del software producido con los requerimientos explícitamente establecidos con los estándares de desarrollo prefijados y con los requerimientos implícitos no establecidos formalmente que desea el usuario (Presuman 1998)".

La producción de software de alta calidad es crítico para la mayoría de las grandes instituciones a causa de la función central que tiene en tantos departamentos: nóminas, cuentas por cobrar, manufactura, ventas, investigación, administración, transacciones, etc. Un error oculto en el software de crédito de una empresa o en el de control de procesos puede resultar en una perdida de millones de dólares. Para más y más empresas el software ha llegado a ser una parte integrante de los productos que se venden. El software de cómputo es parte integrante de los sistemas de consumo de combustible del automóvil, computo, los controles de videos y dvd, etc. Hace algunos años un problema oculto de software en los

sistemas de larga distancia de AT&T hizo que se cayera el sistema, deteniendo todas las actividades financieras con cede en Nueva York e interfiriendo con miles de millones de dólares de negocios en todo el país durante horas. Los vehículos modernos de pasajeros y comerciales dependen cada vez más de programas de cómputo para las funciones críticas. Un defecto oculto en el software de sistemas de frenos pudiera tener como consecuencia pérdida de vidas. Como otros tipos de producción la producción de software es única y presenta su propio conjunto de problemas, una característica especial del desarrollo de software es que su meta normal es construir solo un ejemplar del producto final (excepto las empresas que construyen software para ventas al público). Con el software los problemas de calidad deben resolverse desde la primera vez; el diseño debe ser de la más alta calidad a la primera.[35]

El cumplir con las necesidades del usuario puede ser difícil en un proceso en donde el usuario final se compromete con el producto antes de que este se haya construido. En efecto, el sistema final es "comprobado" anticipadamente, vendido "sin verlo". Definir las necesidades del usuario y juzgar la calidad del sistema terminado han demostrado ser los retos principales. La mayor parte de los proyectos de desarrollo de sistemas se inician en la definición de los requerimientos de la información del usuario y en las especificaciones en la forma de análisis de sistemas y documentos de diseño. El problema es que cumplir con las especificaciones no necesariamente garantiza la calidad. El sistema terminado puede de hecho satisfacer las especificaciones, pero no las necesidades del usuario. Esto ocurre a causa de especificaciones poco precisas, incompletas o mal detalladas, omisión de funciones a en las especificaciones o cambio de las necesidades del usuario durante el periodo de desarrollo. A menudo la calidad no se cuantifica en las especificaciones, de manera que los juicios acerca de la calidad de los sistemas se vuelven subjetivos. Este tipo de situación puede deteriorarse fácilmente entre los sistemas de información y la comunidad de los usuarios, cada una de ellas atacando a la otra sin alcanzar la calidad de los sistemas. Las especificaciones a menudo fallan en la consideración del sistema desde el punto de vista de los usuarios. Mientras que los diseñadores se concentrarán en la funcionalidad, con frecuencia soslayan la facilidad de aprendizaje y uso, precisión y confiabilidad incuestionables o

[35] Kenneth C. Laudon, Jane P Laudon "Administración de los Sistemas de Información" Editorial Person Education

la velocidad de respuesta. Todos estos factores son importantes para el éxito de un sistema. El tiempo de respuesta de un sistema es un ejemplo común de especificaciones detalladas que se omiten o definen inadecuadamente. Un representante de servicios al cliente, quien es el usuario final en una función de servicio al cliente, puede necesitar un tiempo de respuesta a una consulta de no más de cinco segundos, pero las especificaciones pueden no contener referencia alguna con respecto al tiempo de respuesta. Si a la entrega el tiempo de respuesta fuera de diez segundos, el sistema cumpliría con las especificaciones pero no cubriría las necesites de los usuarios. Probablemente un tiempo de respuesta de cinco segundos se hubiera especificado y se alcanzaría durante las pruebas con uno o dos usuarios simultáneos, pero ¿satisfaría el sistema esta especificación si el tiempo de respuesta se deteriora a veinte segundos cuando 100 usuarios están en el sistema simultáneamente? ¿Sirve tal sistema a las necesidades de los usuarios?. Si el sistema no responde en cinco segundos el 80% del tiempo cuando 100 personas están en el sistema simultáneamente, ¿ha cumplido el sistema con las especificaciones cuando no cumple con esta especificación el 20% del tiempo? ¿Puede esperarse que un sistema cumpla con tal requerimiento 100% de el tiempo?. La falta de normas acordadas y medibles en las especificaciones conduce a usuarios insatisfechos y a sistemas que no cumplen con las necesidades de los usuarios. Las expectaciones de los usuarios tampoco pueden ser equiparadas con la calidad, los usuarios crearan su propio modelo mental de un sistema "perfecto". Estos tipos de expectativas no especificadas afectarán la manera como un sistema se ve y se juzga. Una sencilla imagen puede aclarar el problema: dos personas que compran la misma camisa pueden tener expectativas muy diferentes. Si uno de los compradores ve la camisa como cara puede esperar que conserve su color a lo largo de muchas lavadas, que dure mucho tiempo y que no necesite de ser planchada. La otra persona puede ver el mismo bien como de precio medio o bajo, y por tanto tener expectativas menores. Ambos compradores adquieren la camisa al mismo precio, pero es muy probable que el primer comprador se decepcione de la camisa luego de un tiempo. Los sistema también son juzgados de muchas maneras subjetivas, haciendo difícil de terminar si el sistema en realidad cumple con las necesidades de los usuarios.[36]

[36] Kenneth C. Laudon, Jane P Laudon "Administración de los Sistemas de Información" Editorial Person Education

2.3 Modelos de calidad para el software

Al día de hoy, ha aumentado la complejidad para lo que se utiliza el software, por lo que resulta difícil generar productos que cumplan cabalmente con las expectativas del cliente.

Para responder a esta situación, han surgido una serie de herramientas, técnicas y modelos que facilitan a las organizaciones, encargadas de las tecnologías de la información, generar productos que cumplan las expectativas del cliente e incluso las rebasen, herramientas que prometen ser la solución a los problemas de calidad, costo y tiempos de desarrollo; de éstas podemos mencionar a los "modelos de calidad" como la norma ISO 9000-2000, la ISO/IEC TR 15504, el modelo MOPROSOFT y el modelo CMM (Capability Maturity Model del Software Engineerig Institute SEI).

Esquemas de los modelos de calidad

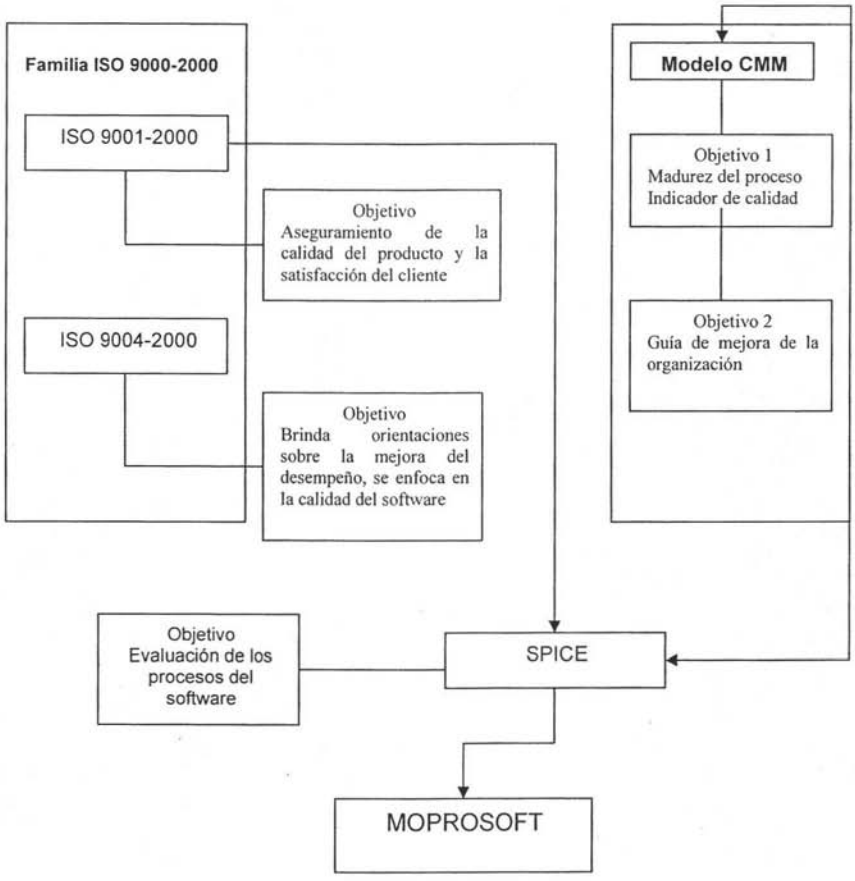


Figura 2.0

2.3.1 CMM: Modelo de Madurez de la Capacidad del Proceso del Software[37]

A principios de los años 80 el Departamento de Defensa de los Estados Unidos enfocó sus tareas a la revisión de los problemas del software y a su mejoramiento. Para contribuir a este programa se creó el Instituto de Ingeniería de Software (SEI) a finales de 1984.

El CMM por sus siglas en ingles es un modelo de calidad desarrollado en el Instituto de Ingeniería de Software (SEI) de la Universidad de Carnegie Mellon, Estados

[37] http://www.asiconsultores.com/serv_cons_calidad.htm

Unidos. Como parte de su trabajo, el Instituto se dio a la tarea de desarrollar el Modelo de Madurez del Proceso de Software y para 1986 se comenzó el Proyecto de Evaluación de la Capacidad del Software. Después de varios años de realizar cuestionarios, evaluaciones, consultas e investigaciones, junto a otras organizaciones, en 1991 SEI produce el Modelo de Madurez de la Capacidad del Proceso de Software.

Este modelo permite a organizaciones o áreas vinculadas en la producción de software definir un camino de mejoramiento continuo para sus procesos de desarrollo, para producir de manera consistente y predecible productos de calidad superior. Tales mejoras se traducen directamente en beneficios como: aumento de la productividad, reducción de costos, incremento en la calidad de los productos, entre otros. El modelo brinda guías para seleccionar estrategias de mejoramiento del proceso mediante la determinación de las capacidades actuales del proceso y la identificación de los puntos críticos para mejorar el proceso y la calidad del software.

Pero a que se refiere con **capacidad y madurez** del proceso del software. A continuación se describe brevemente:

- La *capacidad del proceso:* es la habilidad inherente de un proceso para producir los resultados planeados. El principal objetivo de un proceso de software maduro es el de producir productos de calidad que cumplan los requisitos del usuario. La capacidad del proceso de software describe el rango de resultados esperados que se obtienen siguiendo un proceso de software, mientras que el desempeño del proceso de software representa los resultados reales obtenidos.
- Cuando se habla de *madurez del proceso:* se entiende como el crecimiento alcanzado en la capacidad del proceso de software y que se considera como una actividad a largo plazo. La madurez del proceso de software está dada cuando un proceso en específico es explícitamente definido, administrado, medido, controlado y es efectivo.

El CMM es un esquema que describe los elementos necesarios para obtener un proceso de software efectivo. El modelo de calidad CMM describe un camino de mejoramiento continuo que parte de **organizaciones inmaduras**, con procesos

informales y poco definidos, hasta llegar a **organizaciones maduras**, cuyos procesos son disciplinados y continuamente mejorados. El CMM abarca las prácticas de ingeniería y de gestión asociadas al desarrollo y mantenimiento de software. Las prácticas que describe este modelo posibilitan que las organizaciones cumplan con las metas de costo, programación, funcionalidad y calidad establecidas para sus productos. El esquema de mejoramiento que propone CMM a través de sus cinco niveles de madurez se ilustra en el siguiente esquema:

Figura 2.1

Cada uno de estos niveles de madurez está compuesto por áreas clave de proceso. Estas áreas de concentración para mejorar el proceso son denominadas "clave" porque se asume que existen otros procesos en la organización que no lo son. La distinción principal que realiza el modelo CMM es dividir a las organizaciones en:

- **Inmaduras:** caracterizadas por la existencia de recursos denominados "héroes", quienes invierten gran cantidad de horas de trabajo por no poseer las herramientas y la metodología adecuada para desempeñar eficientemente sus actividades. En este tipo de organizaciones, los procesos son inexistentes, el proceso de software es generalmente improvisado, no existen planes rigurosos, sus actividades se enfocan en resolver las crisis que se presentan y los recursos carecen de la capacidad necesaria para estimar los tiempos, costos y calidad asociada a los productos que desarrollan.

- **Maduras:** cuando la organización alcanza cierto grado de madurez posee una gran habilidad para administrar el proceso de desarrollo y mantenimiento del software, se hacen pruebas y análisis de costo-beneficio para mejorar el proceso, el administrador monitorea la calidad del producto y la satisfacción del cliente, se llevan registros y todos los integrantes están involucrados en el proceso de desarrollo. caracterizadas por el cumplimiento de los compromisos vinculados con los proyectos de software y la utilización de información histórica para predecir el comportamiento futuro. El desarrollo de prácticas de CMM permite que las organizaciones incrementen su nivel de madurez a medida que van obteniendo distintos niveles de certificación.

2.3.1.1 Niveles.

Como ya se menciono el modelo CMM consta de 5 niveles, diseñados de manera que los niveles inferiores provean las bases para que de forma progresiva se alcancen los superiores. Estas 5 etapas de desarrollo son referidas como niveles de madurez y en cada uno la organización alcanza una capacidad superior del proceso. En la figura anterior se muestra la estructura de los niveles del modelo. A continuación se describen:

1. **Inicial:** El proceso de software es un proceso improvisado y caótico. Pocos procesos están definidos y el éxito que se pueda obtener depende de las habilidades, conocimientos y motivaciones del personal. No existen calendarios ni estimados de costos y la funcionalidad y calidad del producto es impredecible. No existe un ambiente estable para el desarrollo y mantenimiento del software. El proceso del software es impredecible por el continuo cambio o modificación a medida que avanza el trabajo.
2. **Repetido:** Se establecen procedimientos de administración del proceso que son básicos para determinar costos, calendarios y funcionalidad. Se establecen las políticas para la administración del proceso y los procedimientos de implantación. El proceso se basa en repetir éxitos anteriores en proyectos de similares características, por lo que los mayores

riesgos se presentan cuando se enfrentan a nuevos proyectos. Existen problemas de calidad y no hay una mejoraría.

Las áreas clave de nivel 2 son:

- Gestión de Requerimientos.
- Planificación de Proyectos de Software.
- Seguimiento y Supervisión del Proyecto.
- Gestión de Subcontratación.
- Aseguramiento de la Calidad del Software.
- Gestión de Configuración de Software.

3. **Definido:** El proceso de software para las actividades administrativas y técnicas está documentado, estandarizado e integrado en un proceso de software estándar dentro de la organización que ayudará a obtener un desempeño más efectivo. El grupo que trabaja en el proceso enfoca y guía sus esfuerzos al mejoramiento del proceso, facilita la introducción de técnicas y métodos e informa a la administración del estado del proceso. La capacidad del proceso está basada en una amplia comprensión común dentro de la organización de las actividades, roles y responsabilidades definidas en el proceso de software.

Las áreas clave de nivel 3 son:

- Foco en el Proceso de la Organización.
- Definición del Proceso de la Organización.
- Programa de Capacitación.
- Coordinación Intergrupal.
- Revisión por Pares.

4. **Administrado:** Se recopilan métricas detalladas del proceso de software y de la calidad del producto. Ambos son cuantitativamente entendidos y controlados. El ciclo de Shewhart es constantemente utilizado para planear, implementar y registrar las mejoras al proceso. Este nivel de capacidad del

proceso permite a la organización predecir las tendencias en el proceso y en la calidad del producto dentro de los límites establecidos y además tomar las acciones necesarias en caso que sean excedidos. Los productos son predeciblemente de alta calidad.

5. **Optimizado:** El mejoramiento continuo del proceso es garantizado por la retroalimentación cuantitativa desde el proceso y desde las pruebas de técnicas y herramientas innovadoras. La organización tiene los medios para identificar los puntos débiles del proceso y conocer cómo fortalecerlos. Su actividad clave es el análisis de las causas de defectos y su modo de prevención.

Cada nivel sirve de base para que los siguientes establezcan una implantación del proceso eficiente y efectivo. La organización puede, sin embargo, de forma provechosa usar procesos descritos en otros niveles. Saltar niveles es contraproducente debido a que cada uno es básico para obtener el siguiente y la capacidad de poder implementar procesos superiores de madurez no implica que se pueda saltar un nivel.

2.3.1.2 Estructura de los niveles.

Cada nivel de madurez está compuesto de varias áreas claves del proceso. Cada una es organizada en 5 secciones definidas como características comunes. Éstas especifican las prácticas claves para el cumplimiento de las metas en el área correspondiente. Las características comunes son:

- **Compromiso.** Describe las acciones que la organización debe tomar para establecer el proceso y que pueda ser soportado. Esta característica está asociada con el establecimiento de políticas y con la responsabilidad de la alta dirección.
- **Habilidades.** Describe las precondiciones que deben existir en el proyecto u organización para implementar un proceso de software de manera competente. Involucra los recursos, estructura de la organización y capacitación requerida

- **Actividades.** Describe los roles y procedimientos necesarios para implementar las metas de un área clave del proceso. Considera los planes, procedimientos, actividades, revisiones y acciones correctivas que se requieren.
- **Mediciones.** Describe las necesidades de medir el proceso y analizar los resultados.
- **Verificación de la implantación.** Describe los pasos para asegurar que las actividades se desarrollan de acuerdo con lo establecido en el proceso. Generalmente abarca las revisiones y auditorias de la dirección y de los aseguradores de la calidad.

Las prácticas claves describen la infraestructura y actividades que más contribuyen a la efectiva implantación e *institucionalización* del proceso.

2.3.1.3 Aplicación del modelo.

El modelo describe los principios y prácticas relacionadas con la madurez del proceso de software y propone ayudar a las organizaciones dedicadas al desarrollo del software a alcanzar la madurez de su proceso de software en términos del tránsito evolutivo desde un proceso improvisado y caótico a uno maduro con una adecuada disciplina y mayor capacidad.

CMM es un modelo descriptivo en el sentido que describe los atributos esenciales que se esperan caractericen una organización dentro de un nivel de madurez en particular. Es un modelo normativo ya que las prácticas detalladas caracterizan el tipo normal de comportamiento que se espera de una organización que realiza proyectos a gran escala. No es prescriptivo ya que no dice a la organización cómo mejorar. Excepto para el nivel 1, cada nivel de madurez es divido en varias áreas claves que indican el área en la organización hacia la cual debe enfocarse el mejoramiento del proceso de software, identifican las políticas que se deben seguir para obtener un nivel de madurez y describen como la organización puede madurar. Cada área identifica un grupo de actividades relacionadas que, cuando se desarrollan de forma colectiva, permiten lograr una serie de objetivos considerados importantes para ampliar la capacidad del proceso. Cuando las metas que propone el área son cumplidas, la organización puede afirmar que se ha institucionalizado la

capacidad del proceso caracterizada por ésta. Las metas indican el alcance, las fronteras y la intención para cada una.

2.3.1.4 *Ventajas*

- Específico para el desarrollo y mantenimiento de software.
- Definido como un conjunto de áreas clave de procesos.
- Tiene un modelo de evaluación.
- Desde 1998 empezó a popularizarse en México.
- Existen organizaciones evaluadas.

Desventajas

- Es un modelo extranjero, no internacional.
- No es fácil de entender (inglés, 220 páginas).
- No es fácil de aplicar (pensado para organizaciones grandes).
- La mejora no está enfocada directamente a los objetivos de negocio.
- La evaluación es costosa y no tiene periodo de vigencia.
- Se está abandonando a favor de CMM-I.

2.3.2 Modelo PSP[38]

Por sus siglas en ingles (Personal Software Process) Proceso Personal del software. El Proceso Personal del Software ayuda a ingenieros individuales a mejorar su funcionamiento trayendo disciplina en la manera en que desarrollan el software. De acuerdo con las prácticas que se encuentran en el modelo de la madurez de la capacidad (CMM), el PSP puede ser utilizado por los ingenieros como guía a un acercamiento disciplinado y estructurado para el desarrollo del software.

En la mayoría de las profesiones, el trabajo competente requiere el uso disciplinado de prácticas establecidas. No es una cuestión de creatividad contra disciplina, sino de traer disciplina al trabajo de modo que la creatividad pueda suceder. El uso de planes y de procedimientos trae orden y eficacia a cualquier trabajo y permite a

[38] Alma Alejandra Mejia Marcial, Judith Eduwiges Penagos Trejo "Modelos de Calidad para el Desarrollo del Software

trabajadores producir un producto superior. Un esfuerzo disciplinado quita la basura, el error, y la ineficacia, liberando los recursos financieros para aplicaciones mejores. Debido a que los ingenieros del software generalmente no planean sus procesos, estos no tienen un camino trazado de trabajo y la calidad del software es raramente medida

El PSP demuestra a ingenieros cómo manejar la calidad de sus productos y cómo hacer comisiones que pueden resolver. También provee de ellas los datos para justificar sus planes. El PSP se puede aplicar a muchas partes del proceso del desarrollo del software, incluyendo el desarrollo del pequeño-programa, la definición del requisito, la escritura del documento, pruebas del sistema, y mantenimiento y el realce de los sistemas grandes de software.

En la evolución del PSP se siguen 4 pasos o etapas.

- Medición: Proceso
- Planificación: Estimación del tamaño e Informe de Pruebas
- Calidad: Revisión y **Validación** del código y diseño.
- Ciclo de desarrollo

2.3.3 ModeloTSP[39]

Por sus siglas en ingles el TSP (Team Software Process) **Proceso del Software en Equipo.** Define como crear y manejar equipos en el desarrollo del software siguiendo una guía en la aplicación de buenas prácticas en la ingeniería del software. El TSP proporciona calidad al trabajo de quipo a los ingenieros del software, ya que estos siguen un proceso definido.

El objetivo principal del TSP es completar con éxito a través de varios ciclos de desarrollo un proyecto de software de calidad.

El proceso de desarrollo TSP contiene las siguientes fases que se describen a continuación.

- Lanzamiento. En esta fase se realiza una revisión de los objetivos del TSP, de lo que es su estructura general, se realizan grupo de ingenieros y a cada uno

39 Alma Alejandra Mejia Marcial, Judith Eduwiges Penagos Trejo "Modelos de Calidad para el Desarrollo del Software

se le da un rol diferente, finalmente se describen las necesidades de los clientes.

- Estrategia. Aquí se crea un diseño conceptual del producto se establece la estrategia de desarrollo decidiendo que se producirá en cada ciclo. Se realizan estimaciones iniciales acerca del tamaño y el esfuerzo requerido y se identifican los riesgos.
- Planeación. Se identifican todas las tareas a ser realizadas y se asignan a cada miembro del equipo. Se propone además un plan de calidad que fijen parámetros a ser alcanzados.
- Requerimientos. Se realiza un análisis de necesidades, se efectúan entrevistas con el cliente y se especifican y examinan los requisitos. Finalmente se desarrolla un plan para realizar las pruebas del sistema, el proceso de identificación de requerimientos consiste principalmente en hacer preguntas al cliente, una vez que se ha entendido lo que el usuario requiere se describen los requerimientos en nuestras propias palabras y se verifica lo escrito con el cliente. La especificación de los requerimientos permiten entre otras cosas definir el ambiente del sistema operativo.
- Diseño. Este el proceso creativo de decidir como construir el producto, se debe construir una especificación completa y precisa de cómo será nuestro producto. Un diseño completo define las partes principales, describen como interactúan dichas partes y especifica como se unen para producir el producto final. Esta parte inicia con la creación de un diseño de alto nivel. Posteriormente se realiza un plan de prueba para la integración. El diseño de alto nivel permite a los ingenieros ver en forma global la interacción de las partes del sistema.
- Implementación. En esta fase cada parte se traduce en código, se revisa se compila y prueba. Los pasos necesarios para el proceso de implementación parten desde la generación del diseño detallado, inspección del diseño detallado, codificación en el lenguaje de programación definido, pruebas unitarias, revisión de la calidad del componente y liberación del mismo. Es importante antes de la revisión tener el diseño detallado completo. Dentro del trabajo en equipo se deben administrar y fijar los estándares de nombres, interfaces, mensajes, rutinas y el glosario de nombres.

- Pruebas. Aquí se verifica el sistema, no la corrección del mismo. En forma simultánea se genera la documentación de usuario. La estrategia para construir e integrar el sistema tiene como propósito asegurarse de que están todas las partes necesarias para ensamblar el sistema y permitir integrarlo y probarlo.
- Postmorten. En esta última fase se realiza un análisis del producto, se escribe un reporte de todo el ciclo, se generan todas las evaluaciones acerca del equipo y finalmente se realiza una presentación del proyecto, la importancia de esta fase radica en la retroalimentación de cada uno de los miembros del equipo. Aquí es cuando podemos identificar problemas que se presentaron, determinar las causas y proponer medidas para evitarlos.

El TSP aporta diferentes aspectos, no solo en la calidad del producto sino que también en la organización entre grupos de ingenieros. También se puede decir que es aplicable a cualquier producto de software.

2.3.4 ISO/IEC 15504 (SPICE)[40]

SPICE por sus siglas en ingles (Software Process Improvement and Capability) Mejora y Capacidad de los Procesos del Software, es un estándar internacional que proporciona una evaluación a los procesos del software. Intenta armonizar las diferentes aproximaciones para la evaluación de los procesos del software y proporciona una aproximación en nombre de una organización con los siguientes objetivos:

- Entender el estado de sus propios procesos.
- Determinar lo correcto de sus propios procesos para un grupo en particular de requisitos.
- Determinar lo correcto de otros procesos de la organización para contrato en particular.

ISO/IEC 15504 consta de nueve partes.

[40] http://www.sqi.gu.edu.au/spice/suite

1. Conceptos guía de introducción.
2. Un modelo de referencia para los procesos y capacidad de procesos.
3. Desarrollo de evaluación.
4. Guía para el desarrollo de la evaluación.
5. Un modelo de evaluación y una guía indicatoria.
6. Una guía para el entrenamiento de asesores.
7. Guía para el uso en la mejora de los procesos.
8. Guía para el uso en la determinación de la capacidad de procesos del proveedor.
9. Vocabulario.

A continuación se muestra la figura de los componentes de SPICE.

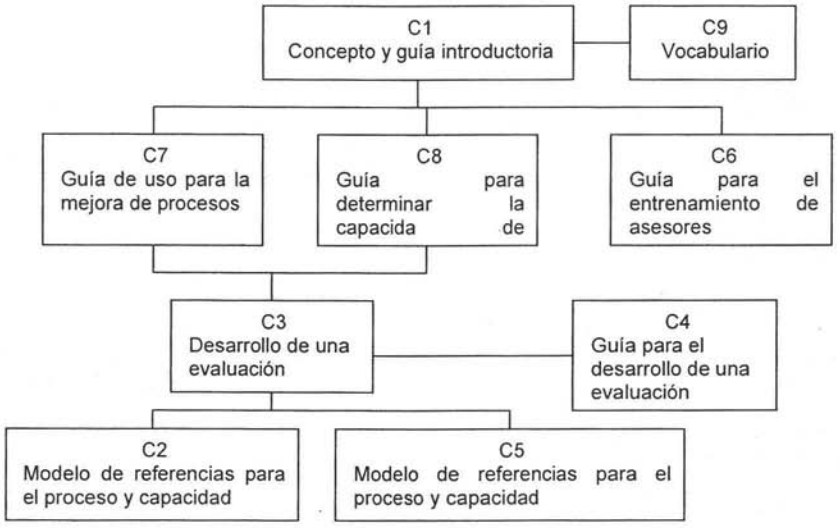


Figura 2.2

- **Parte 1 (informativo).** Describe como las partes interactúan en conjunto
- **Parte 2 (normativa).** En esta parte el estándar internacional se define en un alto nivel.

- **Parte 3 (normativa):** este estándar internacional define un marco para conducir la evaluación y precisar los rangos o índices de la capacidad del proceso. La evaluación incluye por los menos una instancia del proceso de cada proceso identificado en el alcance de la evaluación. Los componentes del grado o nivel. En esta parte es en donde se determina y **valida cada paso del proceso.**
- **Parte 4 (informativa).** Provee la guía de la conducción del equipo basado en la evolución del proceso de software.
- **Parte 5 (normativa).** Define al marco de elementos requeridos para construir un instrumento para asistir al asesor en la realización de una evaluación.
- **Parte 6 (informativa).** Describe la capacidad, la educación, el entrenamiento y la experiencia de los asesores que es relevante para conducir la evaluación de procesos.
- **Parte 7 (informativa).** Define las entradas a utilizar y los resultados de la evaluación para los propósitos de la mejora del proceso.
- **Parte 8 (informativa).** Define las entradas y describe como usar los resultados de la evaluación con el fin de la determinación de la capacidad.
- **Parte 9 (informativa).** Es un vocabulario

2.3.4.1 Ventajas

- Específico para el desarrollo y mantenimiento de software.
- Fácil de entender (24 procesos, 16 páginas).
- Definido como un conjunto de procesos.
- Orientado a mejorar los procesos para contribuir a los objetivos del negocio.

Desventajas

- No es práctico ni fácil de aplicar.
- Tiene solamente lineamientos para un mecanismo de evaluación.
- Todavía no es norma internacional.

2.3.5 PROSOFT[41]

Es un Modelo de Procesos para la Industria de Software que fomenta la estandarización de su operación, a través de la incorporación de las mejores prácticas en gestión e ingeniería de software. La adopción del modelo permite elevar la capacidad de las organizaciones para ofrecer servicios con calidad y alcanzar niveles internacionales de competitividad.

Aunque en el pasado se reconocía la necesidad de crear software de calidad, no se había hecho un esfuerzo serio para que nuestra industria generara productos que nos dieran la oportunidad de competir en el mercado internacional, con calidad equiparable o superior a la de países como la India o Irlanda. Afortunadamente, dicha situación ha cambiado; nuestro gobierno en conjunto con la industria, ha iniciado un esfuerzo serio para impulsar la industria del software a través del Programa para el Desarrollo de la Industria del Software (PROSOFT).

PROSOFT reconoce el estado incipiente de la industria mexicana de software, así como la necesidad de invertir cantidades crecientes de recursos en capital de tecnologías de información con objeto de contribuir de manera sostenible al crecimiento de la economía y la generación de empleos bien remunerados.

Con el programa, se pretende establecer una industria de software competitiva internacionalmente y asegurar su crecimiento a largo plazo, lo que situaría a México como líder de esta industria en Latinoamérica en 2012, además de convertirlo en líder desarrollador de soluciones de tecnologías de información de alta calidad y uso de software en Latinoamérica.

Este programa tiene siete estrategias de donde emergen varios proyectos que ayudarán a que se alcancen las metas previstas en éste.

1. Promover las exportaciones y la atracción de inversiones.
2. Educar y formar personal competente en el desarrollo de software, en cantidad y calidad convenientes.
3. Contar con un marco legal promotor de la industria.
4. Desarrollar el mercado interno.
5. Fortalecer a la industria local.

[41] http://www.avantare.com/articulos/anteriores/MoProSoftV2.0

6. Alcanzar niveles internacionales en capacidad de procesos.
7. Promover acciones conjuntas con los gobiernos estatales y construir infraestructura.

Para el caso de la estrategia 6, la Asociación Mexicana para la Calidad en Ingeniería de Software (AMCIS), con el auspicio de la Secretaría de Economía propone un modelo concebido, diseñado y desarrollado por mentes mexicanas, adecuado para las necesidades específicas de México y con ventajas respecto de otros. El nuevo modelo, denominado MoProSoft, ofrece características que los otros no tienen de manera independiente; para su concepción, se tomaron las mejores prácticas de los otros modelos y se integraron y mejoraron otras.

2.3.6 MoProSoft: Modelo de Procesos para la Industria de Software

El modelo de procesos para la industria de software mexicana MoProSoft fue desarrollado en el año 2002 como consecuencia de los acuerdos de la mesa 6 del Programa para el Desarrollo de la Industria de Software dirigido por la Secretaría de Economía.

La Secretaria de Economía, por medio del Programa para el Desarrollo de la Industria de Software (ProSoft), desea fortalecer a la industria de software en México. Este programa está conformado por siete estrategias:

1. Promover las exportaciones y la atracción de inversiones.
2. Educación y formación de personal competente en el desarrollo de software en cantidad y calidad convenientes.
3. Contar con un marco legal promotor de la industria.
4. Desarrollar el mercado interno.
5. Fortalecer a la industria local.
6. Alcanzar niveles internacionales en capacidad de procesos.
7. Promover la construcción de infraestructura física y de telecomunicaciones.

Se organizaron mesas de trabajo para definir la forma de abordar cada una de las estrategias mencionadas. Para el número seis se reunió un equipo constituido por representantes de organizaciones o áreas de desarrollo de software mexicanas.

Inicialmente, para lograr los objetivos de la estrategia 6 este equipo propuso las siguientes líneas de trabajo:

6.1 Definición de modelos de procesos y de evaluación apropiados para la industria de software mexicana.

6.2 Formación de instituciones de capacitación y asesoría en mejora de procesos.

6.3 Apoyo financiero para la capacitación y la evaluación de capacidad de procesos.

Una de las primeras tareas del grupo encargado de la estrategia 6 fue identificar las necesidades de la industria mexicana de software con respecto a un modelo de proceso. Las principales necesidades identificadas fueron:

- Específico para el desarrollo y mantenimiento de software.
- Fácil de entender (comprensible).
- Definido como un conjunto de procesos.
- Práctico y fácil de aplicar, sobre todo en organizaciones pequeñas.
- Orientado a mejorar los procesos para contribuir a los objetivos del negocio y no simplemente ser un marco de referencia de certificación.
- Debe de tener un mecanismo de evaluación o certificación, que indique un estado real de una organización durante un periodo de vigencia específico.
- Aplicable como norma mexicana.

Estas necesidades se tomaron en cuenta para realizar un análisis de los modelos existentes, tales como ISO 9001:2000, SW-CMM ÆÊ, ISO/IEC TR:15504 entre otros, para evaluar si estos modelos podrían ser aplicables como norma mexicana.
El punto más débil de estos modelos es cumplir con el requisito de ser prácticos y fáciles de aplicar, sobre todo en organizaciones pequeñas. Esto motivó la propuesta de realizar un proyecto cuyo propósito era generar el documento base para la Norma Mexicana para la Industria de Desarrollo y Mantenimiento de Software, que diera respuesta a las necesidades planteadas y, en particular, a este punto.

Para la realización del proyecto, la Secretaría de Economía estableció un convenio con la Facultad de Ciencias de la Universidad Nacional Autónoma de México bajo la responsabilidad de la Dra. Hanna Oktaba, quien convocó a personas con experiencia y conocimiento en los modelos de procesos y calidad de software para conformar al grupo editor. Este grupo estuvo constituido por:

- M. en C. Claudia Alquicira Esquivel
- M. en C. Angélica Su Ramos
- M. en C. Alfonso Martínez Martínez
- M. en C. Gloria Quintanilla Osorio
- Mara Ruvalcaba López
- Francisco López Lira Hinojo
- Maria Elena Rivera López
- Mat. Maria Julia Orozco Mendoza
- Dra. Yolanda Fernández Ordóñez
- Miguel Ángel Flores Lemus

El proyecto se realizó en el periodo de septiembre a diciembre del 2002. Como resultado se generó el documento titulado Modelo de Procesos para la Industria de Software (MoProSoft).
La elaboración del Anexo "Relación de MoProSoft con ISO 9001:2000, CMM v1.1 e ISO/IEC TR 15504-2:1998", estuvo a cargo de Øyvind Mo, alumno de la Maestría en Ciencia e Ingeniería de la Computación (UNAM).

2.3.6.1 Propósito del Modelo de Procesos para la Industria de Software (MoProSoft)

El propósito del Modelo de Procesos para la Industria de Software es fomentar la estandarización de su operación a través de la incorporación de las mejores prácticas en gestión e ingeniería de software. La adopción del modelo permitirá elevar la capacidad de las organizaciones para ofrecer servicios con calidad y alcanzar niveles internacionales de competitividad.

Con base en el conocimiento de los modelos de referencia del grupo editor, se establecieron un conjunto de criterios que conformaron la base para la estructura del modelo, así como de la elaboración del patrón de proceso que se utilizó para la documentación de los procesos. Los criterios fueron los siguientes:

- Generar una estructura de los procesos que esté acorde con la estructura de las organizaciones de la industria de software (Alta Dirección, Gestión y Operación).
- Destacar el papel de la Alta Dirección en la planeación estratégica, su revisión y mejora continua como el promotor del buen funcionamiento de la organización.
- Considerar a la Gestión como proveedor de recursos, procesos y proyectos, así como responsable de vigilar el cumplimiento de los objetivos estratégicos de la organización.
- Considerar a la Operación como ejecutor de los proyectos de desarrollo y mantenimiento de software.
- Integrar de manera clara y consistente los elementos indispensables para la definición de procesos y relaciones entre ellos.
- Integrar los elementos para la administración de proyectos en un sólo proceso.
- Integrar los elementos para la ingeniería de productos de software en un solo marco que incluya actividades o prácticas de soporte, tales como **verificación, validación**, documentación, administración de configuración y mediciones.
- Destacar la importancia de la gestión de recursos, en particular los que componen la base de conocimiento de la organización tales como: productos generados por proyectos, datos de los proyectos, incluyendo las mediciones, documentación de procesos y los datos recaudados a partir de su uso y lecciones aprendidas.
- Basar el modelo de procesos en ISO9000:2000 [1] y nivel 2 y 3 de CMM V.1.1. Usar como marco general ISO/IEC 15504 - Software Process Assesment e incorporar las mejores prácticas de otros modelos de referencia tales como PMBO, SWEBOK y otros más especializados.

El desarrollo y mantenimiento de software se lleva a cabo a través de una serie de actividades realizadas por equipos de trabajo. La Ingeniería de Software se ha dedicado a identificar las mejores prácticas para realizar estas actividades recopilando las experiencias exitosas de la industria de software a nivel mundial. Estas prácticas se han organizado por áreas de aplicación, y se han dado a conocer como áreas clave de procesos, en caso de CMM, o como procesos de software en ISO/IEC 15504.

El modelo que se propone está enfocado en procesos y considera los tres niveles básicos de la estructura de una organización que son: la Alta Dirección, Gestión y Operación. El modelo pretende apoyar a las organizaciones en la estandarización de sus prácticas, en la evaluación de su efectividad y en la integración de la mejora continua.

2.3.6.2 Estructura de MoProSoft

MoProSoft tiene tres categorías de procesos: Alta Dirección, Gestión y Operación que reflejan la estructura de una organización. Como lo muestra la siguiente figura:

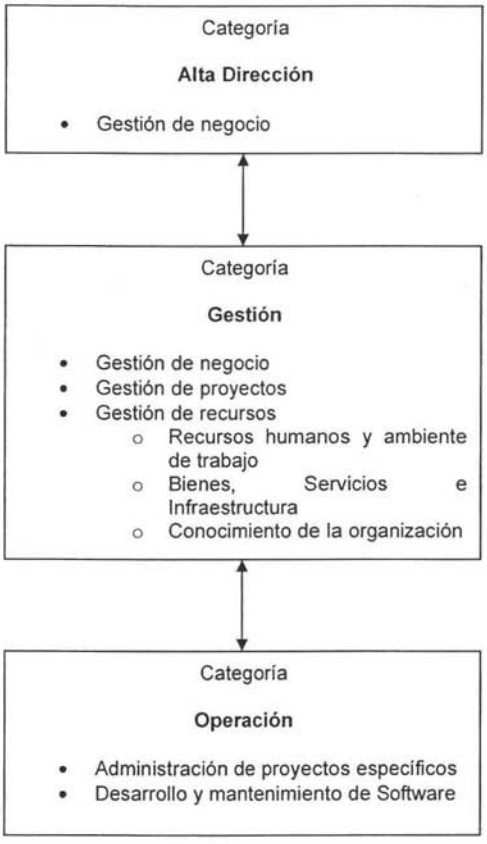


Figura 2.3 Estructura del modelo de procesos

La categoría de Alta Dirección (DIR) aborda las prácticas relacionadas con la gestión del negocio. Proporciona los lineamientos a los procesos de la Categoría de Gestión y se retroalimenta con la información generada por ellos. Esta categoría contiene el proceso Gestión de Negocio (DIR 1).

La categoría de Gestión (GES) aborda las prácticas de gestión de procesos, proyectos y recursos en función de los lineamientos establecidos en la Categoría de Alta Dirección. Proporciona los elementos para el funcionamiento de los procesos de la Categoría de Operación, recibe y evalúa la información generada por éstos y comunica los resultados a la Categoría de Alta Dirección. Esta categoría se integra por los procesos de Gestión de Procesos (GES 1), Gestión de Proyectos (GES 2) y

Gestión de Recursos (GES 3). Éste último está constituido por los subprocesos de Recursos Humanos y Ambiente de Trabajo (GES 3.1), Bienes, Servicios e Infraestructura (GES 3.2) y Conocimiento de la Organización (GES 3.3). La categoría de Operación (OPE) aborda las prácticas de los proyectos de desarrollo y mantenimiento de software. Esta categoría realiza las actividades de acuerdo a los elementos proporcionados por la Categoría de Gestión y entrega a ésta la información y productos generados. Esta Categoría se integra por los procesos de Administración de Proyectos Específicos (OPE 1)y de Desarrollo y Mantenimiento de Software (OPE 2).

En cada proceso están definidos los roles responsables por la ejecución de las prácticas. Los roles se asignan al personal de la organización de acuerdo a sus habilidades y capacitación para desempeñarlos.

En MoProSoft se clasifican los roles en Grupo Directivo, Responsable de Proceso y otros roles involucrados. Además se considera al Cliente y al Usuario como roles externos a la organización.

2.3.6.3 Procesos de MoProSoft

En esta sección se presentan los propósitos, los objetivos y las salidas de cada uno de los procesos de MoProSoft.

DIR.1 Gestión de Negocio. El propósito de Gestión de Negocio es establecer la razón de ser de la organización, sus objetivos y las condiciones para lograrlos, para lo cual es necesario considerar las necesidades de los clientes, así como evaluar los resultados para poder proponer cambios que permitan la mejora continua.

Adicionalmente habilita a la organización para responder a un ambiente de cambio y a sus miembros para trabajar en función de los objetivos establecidos.

Los objetivos a cumplir para asegurar el cumplimiento del propósito del proceso son:

- Lograr una planeación estratégica exitosa mediante el cumplimiento del *Plan Estratégico*.
- Lograr que la organización trabaje en función del *Plan Estratégico* mediante la correcta comunicación e implantación del mismo.
- Mejorar el *Plan Estratégico* mediante la implementación de la *Propuesta de Mejoras*.

Como productos de salida del proceso de Gestión de Negocio son: *El Plan Estratégico*, *Plan de Comunicación e Implantación*, *Reporte de Mediciones y Sugerencias de Mejora*, *Plan de Adquisiciones y Capacitación*, y *Lecciones Aprendidas*.

GES.1 Gestión de Procesos. El propósito de Gestión de Procesos es establecer los procesos de la organización, en función de los procesos requeridos identificados en el plan estratégico. Así como definir, planear, e implantar las actividades de mejora en los mismos. Los objetivos a cumplir para asegurar el cumplimiento del propósito del proceso son:

- Planear las actividades de definición, implantación y mejora de los procesos en función del *Plan Estratégico*.
- Dar seguimiento a las actividades de definición, implantación y mejora de los procesos mediante el cumplimiento del *Plan de Procesos*.
- Mejorar el desempeño de los procesos mediante el cumplimiento del *Plan de Mejora*.
- Mantener informado a Gestión de Negocio sobre el desempeño de los procesos mediante el *Reporte Cuantitativo y Cualitativo*.

Como productos de salida del proceso de Gestión de Procesos son: *Plan de Procesos*, *Documentación de Procesos*, *Reporte Cuantitativo y Cualitativo*, y *Lecciones Aprendidas*.

GES.2 Gestión de Proyectos. El propósito de la Gestión de Proyectos es asegurar que los proyectos contribuyan al cumplimiento de los objetivos y estrategias de la organización. Los objetivos a cumplir para asegurar el cumplimiento del propósito del proceso son:

- Cumplir con el *Plan Estratégico* de la organización mediante la generación e instrumentación de proyectos.

- Mantener bajo control las actividades Gestión de Proyectos mediante el cumplimiento del *Plan de Gestión de Proyectos*.
- Proveer la información del desempeño de los proyectos a Gestión de Negocio mediante la generación del *Reporte Cuantitativo y Cualitativo*.
- Atender los Comentarios y Quejas del Cliente mediante la definición y ejecución de *Acciones Correctivas o Preventivas.*

Como productos de salida del proceso de Gestión de Recursos son: *El Reporte Cuantitativo y Cualitativo, Reporte de Acciones Correctivas o Preventivas Relacionadas con Clientes, Reporte de Mediciones y Sugerencias de Mejora, Plan de Adquisiciones y Capacitación, Contrato, Registro de Proyecto, Lecciones Aprendidas, Responsable de Administración del Proyecto Específico, Descripción del Proyecto, Metas Cuantitativas para el Proyecto y Acciones Correctivas o Preventivas.*

GES.3 Gestión de Recursos. El propósito de Gestión de Recursos es conseguir y dotar a la organización de los recursos humanos, infraestructura, ambiente de trabajo y proveedores, así como crear y mantener la base de conocimiento de la organización. La finalidad es apoyar el cumplimiento de los objetivos del plan estratégico de la organización. Los objetivos a cumplir para asegurar el cumplimiento del propósito del proceso son:

- Lograr los objetivos del *Plan Estratégico* mediante la provisión de los recursos suficientes y calificados a la organización.
- Proveer a los miembros de la organización de los medios y mecanismos adecuados para el uso y resguardo de la información mediante la *Base de Conocimiento*.
- Mantener a la organización informada oportunamente sobre las tendencias tecnológicas mediante la elaboración de *Propuestas Tecnológicas*.

Como productos de salida del proceso de Gestión de Recursos son: *El Reporte Cuantitativo y Cualitativo, Propuestas Tecnológicas, Reporte de Mediciones y*

Sugerencias de Mejora, Plan Operativo de Recursos Humanos y Ambiente de Trabajo, Plan Operativo de Bienes, Servicios e Infraestructura, Plan Operativo de Conocimiento de la Organización, Acciones Correctivas y Lecciones Aprendidas.

GES.3.1 Recursos Humanos y Ambiente de Trabajo. El propósito de Recursos Humanos y Ambiente de Trabajo es proporcionar los recursos humanos adecuados para cumplir las responsabilidades asignadas a los roles dentro de la organización, así como la evaluación del ambiente de trabajo. Los objetivos a cumplir para asegurar el cumplimiento del propósito del proceso son:

- Proveer a la organización de recursos humanos calificados mediante la selección y capacitación adecuada a los roles que se les asignen.
- Evaluar el ambiente de trabajo de la organización mediante la *Encuesta sobre el Ambiente de Trabajo*.

Como productos de salida del subproceso de Recursos Humanos y Ambiente de Trabajo son la *Asignación de Recursos*, *Reporte de Mediciones y Sugerencias de Mejora*, *Reporte de Recursos Humanos Disponibles, Capacitación y Ambiente de Trabajo*, *Plan de Capacitación* y *Lecciones Aprendidas*.

GES.3.2 Bienes, Servicios e Infraestructura

El propósito de Bienes, Servicios e Infraestructura es proporcionar proveedores de bienes, servicios e infraestructura que satisfagan los requisitos de adquisición de los procesos y proyectos. Los objetivos a cumplir para asegurar el cumplimiento del propósito del proceso son:

- Proporcionar a la organización los bienes y servicios requeridos por los procesos y los proyectos mediante la selección y evaluación de los proveedores.
- Mantener la infraestructura de la organización mediante el cumplimiento del *Plan de Mantenimiento*.

Como productos de salida del subproceso de Bienes, Servicios e Infraestructura son el *Reporte de Bienes, Servicios e Infraestructura*, *Reporte de Mediciones y Sugerencias de Mejora* y *Lecciones Aprendidas*.

GES.3.3 Conocimiento de la Organización. El propósito de Conocimiento de la Organización es mantener disponible y administrar la base de conocimiento que contiene la información y los productos generados por la organización. El objetivo a cumplir para asegurar el cumplimiento del propósito del proceso es:

- Proporcionar a la organización la *Base de Conocimiento* de forma confiable, oportuna y segura mediante el cumplimiento del *Plan de Administración de la Base de Conocimiento*.

Como productos de salida del subproceso de Conocimiento de la Organización son la *Base de Conocimiento*, *Reporte del Estado de la Base de Conocimiento*, *Reporte de Mediciones y Sugerencias de Mejora* y *Lecciones Aprendidas*.

OPE.1 Administración de Proyectos Específicos. El propósito de la Administración de Proyectos Específicos es establecer y llevar a cabo sistemáticamente las actividades que permitan cumplir con los objetivos de un proyecto en tiempo y costo esperados. Los objetivos a cumplir para asegurar el cumplimiento del propósito del proceso son:

- Lograr los *Objetivos* del proyecto en tiempo y costo mediante la coordinación y el manejo de los recursos del mismo.
- Mantener informado al Cliente mediante la realización de reuniones de avance del proyecto.
- Atender las *Solicitudes de Cambio* del cliente mediante la recepción y análisis de las mismas.

Como productos de salida del proceso de Administración de Proyectos Específicos son el *Reporte de Mediciones y Sugerencias de Mejora*, *Plan del Proyecto*, *Reporte*

de Seguimiento, *Documento de Aceptación*, *Plan del Proyecto*, *Plan de Adquisiciones y Capacitación*, *Lecciones Aprendidas* y *Plan de Desarrollo*.

OPE.2 Desarrollo y Mantenimiento de Software. El propósito de Desarrollo y Mantenimiento de Software es la realización sistemática de las actividades de análisis, diseño, construcción, integración y pruebas de productos de software nuevos o modificados cumpliendo con los requerimientos especificados. Los objetivos a cumplir para asegurar el cumplimiento del propósito del proceso son:

- Lograr que los productos de salida sean consistentes con los productos de entrada en cada fase de un ciclo de desarrollo mediante las actividades de **verificación, validación** o prueba.
- Sustentar la realización de ciclos posteriores o proyectos de mantenimiento futuros mediante la integración de la *Configuración de Software* del ciclo actual.
- Llevar a cabo las actividades de las fases de un ciclo mediante el cumplimiento del *Plan de Desarrollo* actual.

Como productos de salida del proceso de Desarrollo y Mantenimiento de Software son: *La Especificación de Requerimientos, Análisis y Diseño, Componente, Software, Configuración de Software, Manual de Usuario, Manual de Operación, Manual de Mantenimiento, Reporte de Actividades, Lecciones Aprendidas, Reporte de Mediciones y Sugerencias de Mejora, Registro de Rastreo, Plan de Pruebas de Sistema, Reporte de Pruebas de Sistema, Plan de Pruebas de Integración y Reporte de Pruebas de Integración.*

2.3.6.4 Patrón de procesos

El patrón de procesos es un esquema de elementos que sirvió para la documentación de los procesos. Está constituido por tres partes:

- Definición general del proceso.
- Prácticas y

- Guías de ajuste.

En la Definición general del proceso se identifica su nombre, categoría a la que pertenece, propósito, descripción general de sus actividades, objetivos, indicadores, metas cuantitativas, responsabilidad y autoridad, subprocesos en caso de tenerlos, procesos relacionados, entradas, salidas, productos internos y referencias bibliográficas. En las Prácticas se identifican los roles involucrados en el proceso y la capacitación requerida, se describen las actividades en detalle, asociándolas a los objetivos del proceso, se presenta un diagrama de flujo de trabajo, se describen las verificaciones y validaciones requeridas, se listan los productos que se incorporan a la base de conocimiento, se identifican los recursos de infraestructura necesarios para apoyar las actividades, se establecen las mediciones del proceso, así como las prácticas para la capacitación, manejo de situaciones excepcionales y uso de lecciones aprendidas. En las Guías de ajuste se sugieren modificaciones al proceso que no deben afectar los objetivos del mismo.

2.3.6.5 Aplicación de MoProSoft

El modelo de procesos MoProSoft está dirigido a las empresas o áreas internas dedicadas al desarrollo y/o mantenimiento de software. Las organizaciones, que no cuenten con procesos establecidos, pueden usar el modelo ajustándolo de acuerdo a sus necesidades. Mientras que las organizaciones, que ya tienen procesos establecidos, pueden usarlo como punto de referencia para identificar los elementos que les hace falta cubrir.

Entonces podemos decir que Moprosoft es un Modelo de Procesos para la Industria de Software que fomenta la estandarización de su operación, a través de la incorporación de las mejores prácticas en gestión e ingeniería de software. La adopción del modelo permite elevar la capacidad de las organizaciones para ofrecer servicios con calidad y alcanzar niveles internacionales de competitividad.

2.3.6.6 Ventajas

- Fácil de entender.
- Fácil de aplicar.

- No es costoso en su adopción.
- Sirve de base para alcanzar evaluaciones exitosas con otros modelos o normas, tales como ISO 9000:2000 [1] o CMM.1 V1.1[2].

A decir de sus creadores, el modelo está orientado a pequeñas y medianas empresas, hecho favorable si se considera que aproximadamente el 80% de las empresas desarrolladoras de software del país caen en esta categoría. Su principal fortaleza es que integra varias de las prácticas propuestas por los otros modelos y corrige algunas de sus desventajas, como son el hecho de que no ha sido liberado por completo o al menos falta el modelo de evaluación; además, está en proceso de convertirse en norma compitiendo con el proyecto de norma ISO/IEC TR 15504 y aunque no ha sido probado, se planea realizar pilotos en algunas organizaciones para evaluar qué tan fácil resulta su implantación determinando los recursos necesarios.

2.3.7 Norma ISO 9000-2000[42]

La Organización Internacional para la Normalización tiene sus orígenes en la Federación Internacional de Asociaciones Nacionales de Normalización (1926–1939). De 1943 a 1946, el Comité Coordinador de las naciones Unidas para la Normalización (UNSCC) actuó como organización interina. En octubre de 1946, en Londres, se acordó por representantes de veinticinco países el nombre de Organización Internacional para la Normalización. La organización conocida como ISO (International Organization for Standarization), celebró su primera reunión en junio de 1947 en Zurich, Alemania, su sede se encuentra ubicada en Ginebra, Suiza. Su finalidad principal es la de promover el desarrollo de estándares internacionales y actividades relacionadas incluyendo la conformidad de los estatutos para facilitar el intercambio de bienes y servicios en todo el mundo.

Los sistemas de calidad basados en reglamentos y procedimientos estandarizados según normas internacionales de aceptación mundial representan, desde hace algunos años, la mejor opción para las empresas de todos tipos y tamaños que se desenvuelven en diferentes industrias, empresas comprometidas a involucrar procedimientos adecuados y eficientes que reflejen un alto grado de calidad y

[42] Norma Mexicana IMNC ISO 9004-2000 (Sistemas de gestión de calidad)
Norma Mexicana IMNC ISO 9000-2000 (Sistemas de gestión de calidad)

mejora continua. A diferencia de muchos programas de mejora continua de la calidad, la implantación de estándares, como las normas ISO 9000, no caduca, sino que se renuevan en forma dinámica logrando mantener niveles máximos de calidad en forma permanente. La certificación ISO 9000, para una empresa determinada, no significa la eliminación total de fallas en sus procesos internos, pero ofrece métodos y procedimientos eficaces sistematizados para determinar las causas de los problemas para luego corregirlos y evitar que estos se repitan nuevamente.

Pero que entendemos por **norma.** Una norma es por definición un "documento establecido por consenso y aprobado por un organismo reconocido, que provee, para el uso común y repetitivo, reglas, directrices o características para actividades o, sus resultados dirigido a alcanzar el nivel optimo de orden en un concepto dado" [ISO/IEC Guía 2:1996]

Las normas fueron creadas, en un principio, como respuesta a la necesidad de documentar procedimientos eficaces de procesos tecnológicos, luego se comercializaron para utilizarlas en procedimientos administrativos; su desarrolló se generó a través del campo de la ingeniería. Las tecnologías desarrolladas por el ser humano a lo largo de la historia fueron utilizadas, en un principio, a niveles regionales; cuando éstas comenzaron a ser exportadas de su lugar de origen no lograban compatibilidad con las tecnologías existentes en otros países; es por eso que se crearon organizaciones nacionales, regionales y luego internacionales, formando una jerarquía bien definida, estas organizaciones determinan las características concretas que deben poseer los equipos para que puedan ser utilizados en cualquier parte del mundo asegurando su máximo desempeño.

Los cambios en las normas ISO 9000:2000, fueron muy representativos en cuanto a los principios básicos de la Gestión de la Calidad. Una vez que surge la idea de llevar a cabo todo un proceso de trabajo que conllevara a la certificación internacional, es necesario enfocarse primeramente en los principios que rigen la norma ISO 9001, ya que son considerados como la base de todo un proceso de cambios. Los requisitos de la norma ISO 9000:2000 son flexibles y algunos de ellos se pueden omitir dependiendo de las necesidades o características de cada organización. Dentro de este trabajo se ha buscado una forma clara de dar a conocer todo un proceso que va desde una idea hasta el reconocimiento internacional para una empresa, organización, institución etc. Por ello, en este

capitulo se introducen los principios de la gestión de la calidad como requisitos, aclarando por supuesto que son solo los principios de la Gestión de la Calidad.
La experiencia acumulada por la implementación de las normas ISO 9000 en cientos de miles de organizaciones en todo el mundo indican la necesidad de mejorarlas, hacerlas más amigables sobre todo para la pequeña y mediana empresa. Dicha experiencia ha mostrado que los resultados deseados se alcancen más eficientemente cuando las actividades y los recursos relacionados se gestionan como un proceso. En consecuencia uno de los caminos para lograr la mejora fue adoptar un sistema de gestión con un enfoque de procesos para lo cual se requirió desarrollar un modelo.
Cuando alguna institución desea adoptar un sistema de gestión de calidad en sus procesos que emplean, los productos que proporcionan son porque desean beneficios directos y una importante gestión de costos y riesgos.
En este caso, el modelo de calidad ISO 9000-2000 tiene impacto sobre:

- La fidelidad del cliente.
- La reiteración de negocios y referencia o recomendación de la empresa.
- Los resultados operativos, tales como los ingresos y participación de mercado.
- Las respuestas rápidas y flexibles a las oportunidades del mercado.
- Los costos y tiempos de ciclos mediante el uso eficaz y eficiente de los recursos, entre otros.

Siguiendo los pasos de la norma para un sistema de calidad y posteriormente una certificación se deben seguir fases o etapas:

- Sistema de Gestión de la calidad.
- Responsabilidad de la dirección.
- Gestión de los recursos.
- Realización del producto.
- Medición análisis y mejora.

En forma general estas son las etapas que se deben seguir en ISO 9000-2000, la etapa de realización del producto tiene sus actividades a realizar, aquí la alta dirección debe asegurarse de la operación eficaz de los procesos de realización y de apoyo, esta etapa tiene subetapas, para poder desarrollarla en la subetapa **Revisión del diseño y desarrollo,** la alta dirección debe asegurarse que se designa al personal apropiado para gestionar y conducir las revisiones sistemáticas para determinar el logro de los objetivos del diseño y desarrollo. Los siguientes puntos son a considerar en dichas revisiones.

- Adecuación de los elementos de entrada para llevar a cabo las tareas de diseño y desarrollo.
- Progreso del proceso planificado de diseño y desarrollo.
- Logro de metas de **verificación y validación.**
- Ciclo de vida del producto, entre otras.

La **validación** de los resultados de los procesos de diseño y desarrollo es importante para la exitosa recepción y utilización por parte de los clientes, proveedores, personal de la organización y otras partes interesadas.
La participación de las partes afectadas permite a los usuarios actuales evaluar los resultados mediante medios como:

- La **validación** de los diseño de ingeniería previamente a la construcción, instalación, o aplicación.
- **La validación del software resultante previamente a la instalación o el uso.**
- La validación de los servicios previamente a su introducción generalizada, entre otros.

Como ya se vio la **validación del software** es de suma importancia y un requisito básico y obligatorio para poder implementar un sistema de calidad siguiendo los pasos de la norma ISO, y se puede concluir, que la **validación del software** es fundamental para asegurar la calidad de este ya que todos los modelos de

aseguramiento de calidad del software que se mencionaron anteriormente piden como requisito la validación del software.

La norma ISO 9000-2000 consta de lo siguiente.

- ISO 9000: Sistemas de Gestión de la Calidad - Conceptos y Vocabulario.
- ISO 9001: Sistemas de Gestión de la Calidad – Requisitos.
- ISO 9004: Sistemas de Gestión de la Calidad – Directrices.
- ISO 10011: Directrices para Auditar Sistemas de la Calidad.

El prologo de la norma internacional ISO 9004-2000 indica que la 9001 y la 9004 forman un par coherente de normas sobre la gestión de la calidad. La norma ISO 9001 esta orientada al aseguramiento de la calidad del producto y aumentar la satisfacción del cliente, mientras que la norma ISO 9004 tiene una perspectiva más amplia sobre la gestión de la calidad brindando orientaciones sobre la mejora del desempeño.

A continuación se mencionara en forma general los requisitos así como la determinación de estos de la norma ISO 9001-2000.

Requisitos de la Norma ISO 9001:2000

- Sistema de Gestión de la Calidad.

1. Requisitos Generales.

   1.1.2 Generalidades.

   1.1.3 Manual de Calidad.

   1.1.4 Control de Documentos.

   1.1.5 Control de los Registros.

2. Responsabilidad de la Dirección.

   2.1 Compromiso de la Dirección.

2.1.2 Enfoque al cliente.

2.1.3 Política de Calidad.

3. Planificación.
    1. Objetivos de la Calidad.
    2. Planeación del Sistema de Gestión de la Calidad.

4. Determinación de los requisitos.

Relacionados con el producto

4.1.2 Revisión de los requisitos relacionados con el producto.
4.1.3 Comunicación con el cliente.
4.1.4 Diseño y desarrollo, planificación del diseño y desarrollo.
4.2 Elementos de entrada para el diseño y desarrollo.
4.2.1 Resultado del diseño y desarrollo.
4.2.2 Revisión del diseño y desarrollo.
42.3 Verificación del diseño y desarrollo.
4.2.4 Validación del diseño y desarrollo.
4.2.5 Control de los cambios del diseño y desarrollo.

Estos son solo algunos puntos específicos para la importancia de este trabajo, ya que la norma ISO 9001-2000 es muy compleja y larga. La norma 9004 -2000 en el punto numero (7) Realización del producto y en subpunto 7.3.3 Revisión del diseño y desarrollo pide entre otros puntos la validación del software resultante previamente a la instalación o uso.

Entonces podemos entender que la norma ISO 9000-2000, es una norma internacional destinada a evaluar la capacidad de la organización para cumplir los requisitos del cliente, los reglamentarios y los propios de la organización, este tipo de normas cuenta con desventajas y ventajas, y a continuación se mencionan algunas.

2.3.7.1 *Ventajas*

- Tiene un mecanismo de certificación bien establecido.
- Está disponible y es conocida.

Desventajas

- No es específica para la industria de software.
- No es fácil de entender.
- No está definida como un conjunto de procesos.
- No es fácil de aplicar.

2.4 Cuadro comparativo

MODELOS DE CALIDAD		
	VENTAJAS	DESVENTAJAS
ISO 9000-2000	• Factor competitivo para las empresas. • Ahorro de tiempo y dinero al evitar demostrar la calidad una y otra vez. • Adoptado en mas de 90 países e implementado en todo tipo de organizaciones. • Garantía de que las cosas se hacen bien.	• Estático, de escaso valor y caro. • Es cuestión de tiempo que deje de ser un factor competitivo. • Adoptado en muchos casos por obligación y para "cubrir el expediente". • Diferencias en cuanto a la interpretación de las cláusulas del estándar.
CMM	• Específico para el desarrollo y mantenimiento de software. • Definido como un conjunto de áreas clave de procesos. • Tiene un modelo de evaluación. • Existen organizaciones evaluadas.	• Modelo extranjero. • No es fácil de entender. • No es fácil de explicar. • No es norma certificadora. • Se desarrolla muy poco en México.
ISO/IEC TR 15504	• Específico para el desarrollo y mantenimiento de software. • Fácil de entender • Definido como un conjunto de procesos. • Orientado a mejorar los procesos para contribuir a los objetivos del negocio.	• No es práctico ni fácil de aplicar. • Tiene solamente lineamientos para un mecanismo de evaluación. • Todavía no es norma internacional.

MOPROSOFT	• Es un modelo especificamente Mexicano. • Su metodologia es fácil de entender. • Es comprensible. • Es nuevo. • Toma lo mejor de los modelos anteriores.	• Todavia no es una norma.

Tabla 2.0

Las deficiencias en el desarrollo del software trae consigo trabajo adicional y problemáticas para los desarrolladores del software, estos pierden recursos económicos como de tiempo en el proceso de mantenimiento o en tratar de arreglar esos errores, llegando así a perdidas que en ocasiones son millonarias y por supuesto perdida de oportunidades en el mercado nacional e internacional. Es por eso que se trata de desarrollar modelos de calidad (en el caso de México MoProSoft) para el desarrollo software, para que desarrolladores e ingenieros de software se basen en procesos estudiados y establecidos con lineamientos o normas para un buen proceso y desarrollo obteniendo así un software de calidad, estos modelos siguen puntos o reglas muy importantes entre esos puntos uno de los que se considera más importantes es el de la **validación.**

Así, tenemos que la **validación** es muy importante para un sistema de calidad cuando se desarrolla a partir de una norma o un modelo como Moprosoft o las normas ISO ya que para que se pueda cumplir con una certificación y cumplir con los lineamientos de estas normas es necesaria la validación en este caso la validación del software.

La validación es una herramienta del proceso de calidad que ayuda al aseguramiento de calidad del proceso o producto, validar un producto, servicio o proceso no es una tarea fácil, ya que la validación es muy completa para asegurar la calidad.

CAPITULO 3

VALIDACION DEL SOFTWARE

El nivel de competencia que existe hoy en día entre las empresas, les obliga a tomar decisiones rápidas y acertadas, es necesario para ello el funcionamiento adecuado del software en cualquier tipo sistema, esto mediante la incorporación de nuevas tecnologías, herramientas y modelos y su continua actualización. De esta forma, combinando esas tecnologías con una adecuada organización y una gestión eficiente, las empresas podrán alcanzar sus objetivos de manera satisfactoria.

El aseguramiento de un producto de calidad se deriva de la cuidadosa atención de un número de factores que incluyen la selección de partes y materiales de calidad, adecuados diseño de procesos, control de procesos y prueba de estos. Los principios básicos del aseguramiento de la calidad tienen su meta en la producción de artículos o dispositivos que son convenientes para su uso intencionado, estos principios son:

- La calidad, seguridad y efectividad deben ser diseñadas y construidas dentro del diseño y proceso del producto.
- La calidad no puede ser inspeccionada o probado en la finalización del producto.
- Cada paso del proceso de desarrollo debe estar controlado para maximizar la probabilidad de que el producto final reúne todas las especificaciones del producto.

El proceso de validación es una clave importante para asegurar que ese aseguramiento de calidad llegue a su meta.

3.1 Validación

3.1.1 Definiciones:

El proceso de ***validación*** *es establecer evidencia documentada la cual proporcione un alto grado de seguridad de que un proceso específico producirá*

consistentemente un producto con las especificaciones predeterminadas y los atributos de calidad apropiada.[43]

Validación *es una actividad que toma lugar mientras el proceso de medición esta siendo desarrollado, es decir, que esta evaluando cada paso involucrado en un determinado proceso para encontrar la falla o desviación, si es que existe.*[44]

La validación es una parte importante del programa de aseguramiento de calidad y es fundamental para una eficiente operación de producción. La validación es un estudio científico de un proceso que sirve para:

- Para demostrar que el proceso este haciendo consistentemente lo que se supone que tiene que hacer.
- Para determinar las variables de el proceso y los limites aceptables para esas variables

En resumen hay tres razones de el porque se tienen que validar los procesos. La principal razón es que la validación es esencial para el aseguramiento de la calidad; la segunda razón es la reducción de costos y por ultimo es un regulador de requisitos.

La validación de procesos requiere una calificación de cada elemento importante. La importancia relativa de cada elemento puede variar de proceso a proceso. Los elementos importantes de validación que definen su alcance serán discutidos brevemente a continuación

3.1.2 Importancia de la validación

La garantía de la calidad del producto se deriva de la cuidadosa atención de un número de factores que incluyen la selección de partes de calidad, productos adecuados y diseño de procesos, control de los procesos y prueba de estos. Los principios básicos del aseguramiento de la calidad tienen como su meta la producción de resultados o artículos que están hechos para su uso intencionado. Estos principios pueden ser declarados como sigue: (1) calidad, seguridad y efectividad deben estar diseñados y construidos dentro de el producto; (2) la calidad

[43] Robert G. Kieffer, Joseph D. Nally "Validation of Pharmaceutical Processes"
[44] QFB. Marco Antonio Hernández Vargas "Apuntes" "Validación de Métodos Estadisticos"

no puede ser inspeccionada o probada dentro de la finalización de el producto; y (3) cada paso de el proceso de manufactura debe ser controlado para maximizar la probabilidad de que el producto finalizado conozca todas las especificaciones de calidad y diseño el proceso de validación es un elemento clave para asegurar la calidad. A continuación se muestra un diagrama que describe en forma general el proceso de validación.

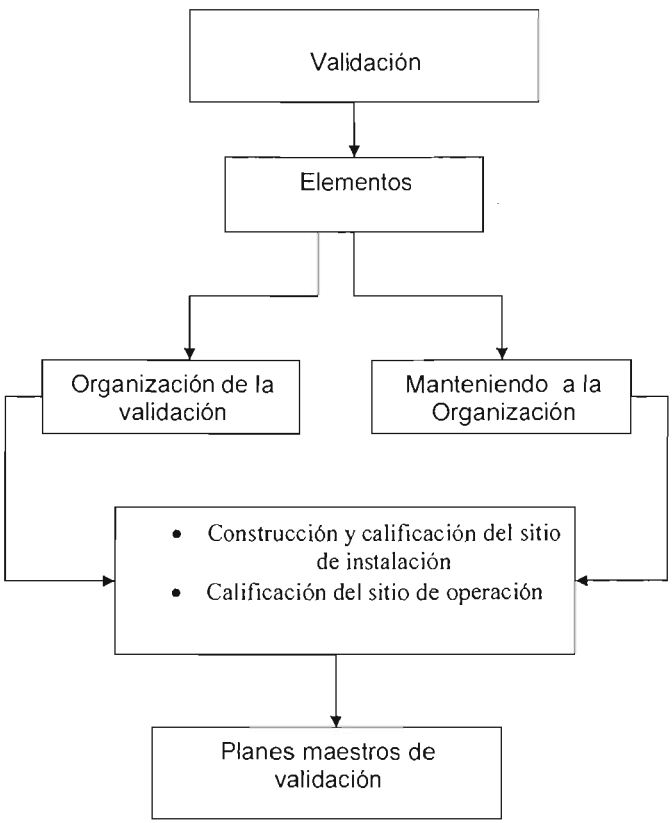


Figura 3.0

3.1.3 Elementos que intervienen en la validación

La validación requiere calificación de cada elemento importante. La importancia relativa de un elemento puede variar de proceso a proceso.

A. Entradas de Proceso.

1. Diseño de el producto

El diseño de el producto consiste el la formulación, contenido y el cierre de el sistema, los procedimientos básicos de manufactura, las especificaciones de control de calidad y la prueba de métodos. Cronológicamente el diseño del producto es el primer elemento de validación a ser estudiado. Aunque el diseño de el producto normalmente es la responsabilidad de las funciones de "investigación y el desarrollo (R&D research and development). Es sabio involucrar al personal de planta, debido a su experiencia y su conocimiento de las capacidades de la planta.

La investigación y el desarrollo deben proporcionar las operaciones de planta con lo siguiente:

- Un diseño de producto robusto para que los componentes, la formulación, los procedimientos de manufactura y las especificaciones del producto sean validados.
- Identificación de las variables críticas en los productos y procesos.
- Limites tentativos para esas variables. Los limites tiene que ser modificados como resultado del proceso de validación hecho por la planta, debido a que muchos componentes del proceso serán diferentes de aquellos usados dentro de el estudio del R&D.
- Métodos para medir, monitorear y controlar las variables criticas.

2. Instalaciones

La calificación de un dispositivo incluye:

- Diseño.

- Construcción verificación (Calificación de la Instalación = IQ, Calificación Operacional = OQ, Calificación de la actuación = PQ).
- Mantenimiento continuo y monitoreo.

En el diseño o la fase de planeación; el objetivo del dispositivo, los productos que son manufacturados y los requisitos, así como el costo deben ser considerados. La construcción de la fase requiere cuidadosa supervisión para asegurarse que todas las especificaciones de diseño estén reunidas. El proceso de verificar es que la construcción del dispositivo conozca todos los requisitos establecidos, empieza cuando la construcción comienza y finaliza con la instalación (IQ) y calificación (OQ, PQ) del equipo y los sistemas críticos. La fase de "verificación" debe ser documentada y los diseños de las especificaciones e ingeniería modificados si es necesario. La última fase de calificar el dispositivo consiste en establecer un mantenimiento preventivo continuo, control de cambio y monitoreos de procedimientos apropiados.

3. Sistemas de apoyos críticos

El sistema de apoyo es cualquier sistema general que las plantas usan en las operaciones diarias. Esto incluye sistemas de aire, red eléctrica y otros. La calificación de un sistema de apoyo crítico consiste en cuatro fases:

- Diseño.
- Instalación.
- Reto y verificación.
- Mantenimiento continuo y monitoreo.

El diseño de sistemas o definición de sistemas existentes es la primera fase, la segunda fase se involucra asegurándose que el sistema instalado está como fue diseñado. La tercera fase se asegura que para las entradas normales, la salida del sistema es aceptable. Finalmente el sistema debe ser monitoreado en intervalos para asegurar que continúa funcionando apropiadamente y está manteniendo las recomendaciones de los productores.

4. Producción de equipo

La calificación del equipo tiene las mismas cuatro fases que el punto anterior. Este comienza con el diseño o selección de procesos, seguido por la instalación (IQ) y verificación (OQ, PQ) de que el equipo funcione como se desea. La calificación del equipo también requiere el desarrollo de procedimientos escritos que describan la operación del equipo, el desarrollo de un programa de mantenimiento preventivo y el entrenamiento del personal.

5. Controles computarizados

"Procesos de Manufactura" las computadoras se usan con gran frecuencia, como equipos de control de procesos. La calificación de las computadoras se califica similarmente a otros equipos. Normalmente el vendedor del equipo proporcionara programas para verificar los sistemas.

B. Actividades de Proceso Industrial

1. Etapas de proceso.
2. Prueba de métodos y equipo.
3. Almacenamiento.

C. Rendimiento de proceso industrial

1. Datos de actuación del producto

El desarrollo del producto en el mercado y en las manos del usuario es de vital importancia para el producto o diseñador. Hay muchos fuentes de información de producto que confirmarán la calidad del producto y consistencia de los procesos validados o muestre que las mejoras o las acciones correctivas se necesiten.

D. Proceso de apoyo

1. Personal: Entrenamiento y Calificación

El operador usualmente es el elemento más importante dentro de un proceso. Así, la calificación del operador por entrenamiento y experiencia es absolutamente esencial en todo el programa de validación. Un operador no entrenado puede que niegue el trabajo hecho en la calificación de otro elemento del proceso. Es importante enfatizar la necesidad de no hacer cambios en un proceso validado sin considerar las consecuencias del cambio.

2. Control de cambios

Un cambio en cualquiera de los elementos puede tener efecto en los procesos y en la calidad o consistencia del producto. Una evaluación de cambio y el control de sistemas identifican un cambio, evalúan el efecto y comienzan la recalificación cuando se necesita. Esto es esencial en el mantenimiento del estado de validación y control.

3. Revalidación

La revalidación es parte del proceso de validación global y frecuentemente un subconjunto de control de cambios. La revalidación es necesaria para:

- Confirmar periódicamente que los parámetros de operación y los resultados de desarrollo del equipo critico o procesos están todavia controlados.
- Evaluar un cambio significativo dentro del equipo, instalaciones, materiales o pasos de los procesos y ambiente que pueden afectar al producto.
- Recalificar un proceso o producto que ha estado inactivo por algún tiempo.
- Ayudar a determinar la causa de una inexplicable falla de producto.

3.1.4 Beneficios de la validación

3.1.4.1 Aseguramiento de la calidad.

La validación es una extensión del concepto del aseguramiento de la calidad, es necesario asegurar la calidad del producto y esto no es posible de controlar apropiadamente sin un minucioso conocimiento de la capacidad del proceso. En el pasado, el control de calidad consistía principalmente en pruebas que se hacían en la producción final.

Las pruebas en la producción final y la inspección tienen deficiencias relacionadas al aseguramiento de la calidad. La validación y los procesos de control son métodos mucho más superiores del aseguramiento de la calidad. Los datos derivados de la validación de los procesos de manufactura y los controles de los procesos a veces pueden proporcionar un gran aseguramiento.

La inspección de procedimientos caen dentro de dos categorías: el 100% de inspección o una inspección de una muestra estadística. Obviamente la inspección en una muestra estadística no da confianza absoluta de que cada unidad del proceso conocerá las especificaciones. Para un nivel especificado de defectos y un plan de muestra especificada, la probabilidad de un artículo defectuoso no descubierto puede ser calculada.

De la misma manera, la prueba final del producto y la falta de validación dan poca seguridad de calidad por varias razones, entre ellas están:

- El número limitado de pruebas hechas.
- La sensibilidad limitada de la prueba.

3.1.4.2 Optimización de procesos

Cuando un proceso es estudiado a conciencia, es inevitable de establecer alguna forma de optimización. La optimización de un proceso para una máxima eficiencia es consecuencia de la validación. El diccionario define "optimizar" como: hacerlo tan efectivo, perfecto o útil como sea posible y aquí se puede agregar; "al menor precio". La optimización del dispositivo, equipo, sistemas y procesos da como resultado un producto que conoce los requisitos de calidad en los costos más bajos. El entrenamiento y personas calificadas son una llave elemental en cualquier proceso; por lo tanto ambas tienen la más grande influencia sobre la eficiencia y la productividad. Algunas áreas en cual la experiencia muestra que el ahorro del costo de optimización como resultado del estudio de la validación es posible son las siguientes:

- Reducción del tiempo de equipo debido al programa de mantenimiento preventivo que esta basado en un entendimiento profundo del equipo y del proceso.

- Reducción de los tiempos de los procesos de manufactura.
- Pruebas reducidas o más rápidas y procedimientos de pruebas analíticas más precisas.
- Desarrollo de estándares para el proceso, estándares para el trabajo, equipo que resultan en un mejor plan de producción y asignación de recursos.
- Un mejor producto o componente debido al reto de las especificaciones.
- Reducción de los costos de energía.
- El ahorro del capital del incremento del equipo.

3.1.4.3 Reducción de los costos de calidad

Tradicionalmente los costos de calidad están divididos dentro de la: prevención, evaluación, fallas internas y costos de fallas externas. Estos costos están definidos en la tabla (3.0).

Aunque estos costos mensurables sean altos, los costos escondidos pueden ser mayores. Hay que considerar el costo de llamadas y quejas. Una llamada puede afectar a un producto o una compañía, en lo peor o en lo mejor, empañando la reputación del producto o de la compañía, resultando en un decremento de ventas y beneficios. Los problemas de fallas persistentes en una planta pueden afectar la moral y la creatividad entre los departamentos y entre la dirección y los trabajadores. Es obvio que un proceso validado y controlado resultara en menos fallas internas: menos rechazos, volver a hacer el trabajo, volver a realizar las pruebas, volver a inspeccionar y menos desperdicio. La validación hace posible hacer el trabajo bien desde la primera vez. También un proceso científicamente estudiado y controlado hace improbable que los productos con defecto sean enviados al consumidor; así no hay llamadas ni quejas. Teóricamente para un proceso validado con absoluto control de todas las variables, no debe haber necesidad de hacer cualquier inspección o prueba de producto final.

1. *Costos de prevención:* son costos incurridos para prevenir fallas o reducir evaluación de costos.

   - Planeación de la calidad.
   - Entrenamiento.
   - Documentación.
   - Mantenimiento preventivo.
   - Calibración.
   - Validación de procesos.
   - Auditoria en el aseguramiento de la calidad e inspección.
   - Revisión anual de datos o tendencia de análisis.

2. *Valoración de costos:* son costos de inspección, prueba y valoración de la calidad, algunos ejemplos de costos de valoración son:

   - Inspección y prueba de materias primas.
   - Inspección y prueba de materiales dentro del proceso.
   - Inspección y prueba de productos terminados.
   - Prueba de estabilidad.

3. *Costos de fallas internas*: son costos asociados con la no conformidad del material, del material que no se conoce los estándares de calidad, algunos ejemplos son:

   - Rechazos.
   - Hacer el trabajo de nuevo.
   - Reinspecciones.
   - Volver a realizar pruebas.
   - Desperdicio.

4. *Costos de fallas externas:* son costos asociados con la no conformidad después de que el producto haya dejado la propiedad de la compañía.

   - Llamadas.
   - Quejas.
   - Devoluciones debido a los problemas relacionados de calidad.

TABLA 3.0

3.1.5 Limitaciones

No hay limitaciones en el concepto de validación y su propia habilidad de asegurar la calidad y reducir costos, pero en la práctica, la validación no es la cura absoluta para todo. Alguna de las limitaciones en la práctica es la gente, la disponibilidad de las instalaciones y equipo, costo y tecnología.

Un proceso de validación para su funcionamiento adecuado, requiere que las personas en la empresa sigan constantemente los procedimientos y sin error y no modificar el sistema. Esto requiere el nivel correcto de entrenamiento, una motivación en el ambiente del trabajador y el equipo apropiado y los sistemas de soporte.

Esto lleva a la consideración de costos: el costo del proceso de validación. La dirección debe asignar recursos al programa de validación y debido a que los recursos generalmente son limitados, esto necesariamente lleva a algo de compromiso en la validación. Se puede gastar más dinero en equipamiento, desarrollo de sistemas, entrenamiento, controles de proceso, y estudios de validación. La calidad estándar debe ser examinada con relación al uso del producto y la exigencia del consumidor y los requisitos del producto. El aseguramiento total de la calidad es lejano, pero un alto nivel de confianza es posible.

La validación puede proporcionar un alto aseguramiento de la calidad y puede asistir en la reducción de costos de manufactura, pero puede existir algún riesgo inherente en la calidad del producto.

3.1.6 Organización para la validación

Los aspectos formales asociados con la organización para la validación son muchos y variados:

- Un interés en la aplicación de la administración total de la calidad (TQM) por sus siglas en ingles.
- El movimiento en las corporaciones para bajar su tamaño o para fusionarse.

Cuando estos factores son adheridos a la evolución de algunas áreas tradicionales de validación. Tal como la validación de los sistemas computarizados, las influencias sobre la organización del esfuerzo de la validación han sido significativas.
En cualquier discusión de validación, la semántica juega un rol importante en el entendimiento de los términos. Por lo tanto, una aproximación común de los términos de validación se menciona a continuación:

- Procesos de validación: Establecimiento de evidencia documentada que un proceso hace lo que tiene por objeto realizar.
- Calificación de la instalación: Verificación documentada de que todos los aspectos de la instalación cumplan con códigos apropiados y diseño aprobado y las recomendaciones de la manufactura estén consideradas apropiadamente.
- Calificación operacional: verificación documentada de que el sistema o subsistema se ejecuten como fue pensado durante los rangos de operación anticipada.
- Certificación ISO: certificación de una serie de sistemas de calidad desarrollados por la ISO (Organización Internacional de Estandarización).
- Ciclo de vida de la validación: es el ciclo que describe las fases del proceso de validación. Es la recaudación de conceptos a través del diseño, construcción, calificación y mantenimiento.

3.1.6.1 Establecimiento de la misión

La formulación de una misión departamental es esencial para asegurar la definición de un rol adecuado de un departamento en la organización. Esto es necesario, tanto

que no solo los miembros del staff del proceso de validación tienen que entender la amplitud de su trabajo si no que otros grupos corporativos con quien haya interacción también entiendan.

En algunas organizaciones el staff superior de miembros representando al proceso de validación, búsqueda y desarrollo, la seguridad de la calidad, producción y grupos de ingeniería convocan para formar comités de asesoramiento para el programa de validación. Estos comités pueden proporcionar un valor importante para el programa de validación, para definir la misión, para hacer decisiones sobre temas específicos que conciernen. Sin embargo para asegurar que esas decisiones están hechas con suficiente información, es necesario que los profesionales de la validación proporcionen suficiente información técnica al comité. Una vez desarrollada, la documentación, entonces permite todo lo complejo en el programa para ser conciente de las prioridades tan bien como el marco general de los esfuerzos de la validación.

La calificación de la instalación (IQ), la calificación operacional (OQ) y la realización de la calificación (PQ) son elementos fundamentales de un programa de validación. Un exitoso esfuerzo de proceso de validación construye calidad dentro del proceso, tanto que confía en la prueba de producto final y esta es minimizada. Esto, en efecto, proporciona mucho más confianza estadística acerca de los procesos que pueden ser probados al final. Esto también es un rasgo de la filosofía de la calidad total. Lo cual habla de una mejoría continua.

Hay ya numerosos ejemplos de diferentes grupos que trabajan en la validación de un proceso:

- Los ingenieros deben conocer las bases para la prueba de IQ y OQ para que se conozcan las instalaciones de los equipos.
- La complejidad de los sistemas de cómputo, el profesional de los sistemas computarizados con el usuario y el grupo de validación en el esfuerzo total de la validación.

En estos ejemplos cada departamento es responsable de un pequeño esfuerzo de la validación. Ellos tienen la responsabilidad de demostrar que el paso del proceso que ellos están evaluando haga lo que se espera, una y otra vez.

3.1.6.2 Tareas del grupo de staff

Cuando un grupo de validación, la misión y la organización ejercen algo de influencia, principalmente en la historia académica de los miembros, una considerable variedad de fondos académicos son usualmente encontrados entre los profesionales de la validación, con miembros teniendo grados de química, estadística, ciencias de la computación y una variedad amplia de ingenierías.

Algo más importante que el área actual del fondo académico son estas tres habilidades: capacidad de resolver los problemas, habilidad interpersonal y habilidades orales y escritas. El talento técnico que reconoce y resuelve problemas es fundamental para la validación. Habilidades interpersonales fuertes permiten una efectividad máxima. Finalmente a no ser que los objetivos de la validación estén efectivamente expresadas tanto oral como escrita el esfuerzo de la validación puede ser en vano.

Una posición que también puede ser usada efectivamente y proporcionar beneficios substanciales es la Validación Técnica. Esta en particular son experiencias de los operadores quienes han sido ascendidos para la clasificación de su próximo trabajo. Claramente la validación técnica proporciona técnicamente una oportunidad de contribuir a la solución del problema. Por lo contrario la validación técnica proporciona al departamento una fuerza de trabajo de gente competente quien proporciona estabilidad, mientras otros en el departamento pueden estar en otros caminos dinámicos.

Existe una diversidad de procesos de manufactura, instalaciones, sistemas computarizados y métodos analíticos que requieren una validación y es muy raro que un individuo tenga habilidad en todo esto.

3.1.6.3 Interacciones departamentales

Una vez que las misiones de los departamentos han sido formalizadas y la operación de la validación organizada el retos es implementar el plan. Para conseguir sus objetivos, la organización de la validación necesita interactuar con muchos grupos. Dentro de la compañía esos grupos diferentes incluyen lo siguiente:

- ***Investigación de operaciones:*** complejo con nuevos productos desarrollados y nuevos procesos mejorados.

- *Ingeniería:* complejo con nuevos o equipos modificados.
- *Producción:* que procesos requieren validación.
- *Mantenimiento:* concerniente al control de cambios.
- *Control de calidad:* complejo con pruebas de laboratorio.
- *Aseguramiento de la calidad*

a. Investigación de Operaciones

La organización de la investigación es compleja con la introducción de nuevos productos y a menudo existen procesos mejorados. La clave del departamento de validación debe ser objetiva en esta área para asegurar la aceptabilidad de nuevos productos o la mejora de procesos en el área de manufactura. Ciertamente todas las empresas tienen productos viejos o procesos que se desarrollaron en un nivel óptimo menor. Una realización exitosa de los objetivos de la validación asegura que los nuevos productos de procesos no reciben el mismo destino.

b. Ingeniería

La relación que a lo largo con la investigación y el desarrollo (R&D; por sus siglas en ingles) poseen el potencial para aceptar los beneficios de la validación es con el grupo de ingeniería. En las etapas iniciales del proyecto, existe la oportunidad ideal para asegurar la aceptabilidad del proceso. La preocupación de validación debe estar construida en la etapa de diseño y contìnuada a través de la construcción. Para el esfuerzo de la validación es necesario que se incluyan actividades como la documentación de calidad combinada y distribución de la verificación.

Una vez que la construcción está completa, la calificación de la fase puede empezar. La calificación del diseño definido de protocolos y los criterios de operación necesitan ser desarrollados y firmados con todas las partes involucradas.

c. Producción

Las interacciones con el personal de producción deben enfatizar los beneficios del programa de validación. Si los beneficios son realmente comprendidos, el

personal de producción apoyara los esfuerzos. Después de todo un proceso validado, es el único en el cual hay confianza, y esa confianza tiene implicaciones en la salvación de costos. Lo que resulta de un proceso validado es su alta calidad de producción, que se produce más eficientemente. Estos efectos positivos de una buena validación justifica el esfuerzo de las razones económicas, mejor que el cumplimiento regulatorio. Después de la terminación del estudio de la validación, los resultados son presentados en un reporte escrito el cual después es aprobado por todo el protocolo de firmas. Los reportes después deben de ser distribuidos a todas las operaciones y los parámetros de los procesos que difieren del reporte de validación puedan ser incorporados dentro de los estándares de procedimientos de producción. Para proteger contra futuros cambios en el proceso que no son objeto de validación, la validación debe revisar todos los cambios sobre un producto o proceso. Su aprobación asegura que los temas de validación están dirigidos para cada cambio.

d. Mantenimiento

Sin el soporte y la cooperación de la organización del mantenimiento, el mejor estudio de validación diseñado e implementado será inútil. Esto ocurrirá en el instante que un cambio indocumentado es hecho para validar una parte del equipo. Como resultado de un programa de educación es esencial hacer que el personal de mantenimiento entienda el efecto de sus actividades de mantenimiento. Una vez que es entendido, la documentación de cualquier cambio hecho a un sistema debe ser comunicado para validarlo y que una valoración del efecto pueda ser hecha.

e. Control de calidad

Debido a que la mayoria de las organizaciones de la validación depende de los laboratorios de control de calidad para el apoyo de la prueba, la comunicación efectiva es extremadamente importante. Ciertamente los protocolos de validación que requieren soporte de laboratorio deben requerir dirección del mismo. Esto asegura que el personal del laboratorio conozca no solo el número y los tipos de prueba requerido para el estudio, si no que también como la prueba se ajuste dentro del programa total de validación. Esto permite al personal la oportunidad

de entender como los datos serán usados y evitar situaciones en la cual las pruebas del personal del laboratorio invaliden el intento de validar.

f. Aseguramiento de la calidad

Interacciones significantes también ocurren con el aseguramiento de la calidad. Estas interacciones están diseñadas para asegurar una confianza de la empresa. El punto clave es comunicar y que el cumplimiento regulatorio de la validación se encuentre.

g. Asociación profesional

Este grupo de personas son extremadamente beneficiosas, no solo debido al conocimiento impartido durante la presentación estructurada, si no también debido a las oportunidades disponibles para una informal discusión de problemas y asuntos.

h. Comunicación

Hasta este punto el rol formal que los departamentos de la validación tienen en cualquier organización han sido discutidos. Estos son los roles definidos por la misión del departamento y como se ha visto, estos también son afectados por la estructura organizacional y las interacciones departamentales que existen. Las sesiones de comunicación o educación para cualquiera de los grupos sociales fomenta el espíritu cooperativo, que es esencial para el esfuerzo de la validación. La administración superior también debe apoyar a la validación y a su rol en toda la organización. La educación es el corazón de un programa de validación exitoso, la validación necesita de la ayuda de los grupos sociales de la empresa.

3.1.7 Manteniendo a la organización

3.1.7.1 Educación continua

Para una continua realización de los objetivos de la validación, la calidad del staff debe ser mantenida. Un programa de educación continua es crítico para conseguir esto. Esto es necesario para que la organización proporcione a los miembros del staff oportunidades de tomar cursos que puedan ayudarlos en tecnología de actualidad. Al mismo tiempo es responsabilidad de los empleados, aprovechar las

oportunidades que están disponibles. Cursos y seminarios son frecuentemente patrocinados por varias asociaciones y son a menudo ofrecidas en conjunción con encuentros diseñados para mantener su membresía técnica en su campo relacionado. Universidades y seminarios profesionales también se añaden al complemento de la educación técnica de cursos disponibles.

En general, los cursos deben ser escogidos para brindarles a los miembros del staff un nivel básico de entendimiento de la destreza necesitada para hacer el trabajo. La experiencia más efectiva de aprendizaje, es la encontrada en el trabajo.

Más allá que la educación continua es necesaria para el trabajo de cualquier persona, esta debe ser tratada como tal, y usada para suplementar el aprendizaje en el trabajo. Las oportunidades son variadas y han sido realzadas por las tecnologías ahora disponibles; sistemas de computo, centros de video aprendizaje, y también cursos en video, estos ahora traen estas oportunidades de aprendizaje al lugar del trabajo, en donde las habilidades necesarias para el éxito están disponibles para todos.

3.1.7.2 Transferencia organizacional

Otro camino de construcción de fortaleza en la organización entera es a través de la transferencia interdepartamental del personal. Los profesionales de la validación están concientes de los procedimientos de manufactura de calidad y pueden aplicar estos conceptos en una Producción o en un Aseguramiento de la Calidad. Las áreas técnicas dentro de la ingeniería o la R&D también pueden encontrar útil talento de validación en el principio, debido a que la validación interactúa muy cerca con estas otras áreas.

3.1.8 El enfoque regulatorio

La discusión del concepto de el ciclo de vida de la validación confirma que; "La validación es un proceso, no un evento, y debe ocurrir durante la vida de un proyecto. De esta manera la validación por si misma es un proceso que empieza en la etapa de conceptualización, recaudación a través del diseño, construcción, calificación y mantenimiento.

¿Como es este esfuerzo, para estar organizado y conseguir este enfoque regulatorio?. La respuesta depende probablemente del tamaño de la empresa.

Múltiples departamentos tal vez cargaron con las responsabilidades de la validación, con cada enfoque sobre alguna facilidad especializada o proceso. Esta tiene sentido, debido a la diversidad de productos y procesos.

3.1.9 Construcción y calificación del sitio de instalación

Esto normalmente comienza a través del diseño detallado. Las actividades normales involucradas dentro de las precomisiones de ingeniería comienzan tan pronto como la construcción del personal estén disponibles para completar el trabajo sobre los sistemas enteros. Las fases de ingeniería en esta etapa involucran a las precomisiones seguidas por las comisiones. Todas las actividades asociadas con esto están abarcadas en la ejecución de actividades de IQ y OQ. La documentación de la IQ debe estar completa para el principio de esta fase de construcción. El grupo de construcción ofrecerá sistemas completados a la comisión de ingeniería para una revisión. El ingeniero revisará el sistema en contra del diseño y la construcción del esquema y proporcionar un puñado de listas. Las listas identificaran donde están las anomalías que requieren rectificación. La importancia de control de cambios en esta fase es evidente. La revisión debe ser completada enfrente de las especificaciones aprobadas durante el diseño detallado. Los cambios deben ser evaluados para revisar si existe cualquier implicación que choque con la calidad del producto. Una vez que los cambios son acordados entonces estos pueden ser aprobados. Este proceso es normalmente controlado usando un procedimiento de control de cambios aprobado.

La entrega e instalación del equipo en el lugar es otra vez parte de el IQ y debe tener todos sus preliminares de documentación completados y adelantados. La finalización de una prueba de aceptación de la empresa (FAT por sus siglas en ingles) antes de que se envíe simplificará a menudo el IQ y a algunas actividades de el OQ. Frecuentemente la documentación final no está disponible para los vendedores hasta que la FAT este completa. Esto tiende a retrasar la preparación de la documentación del OQ. Sin embargo la necesidad de finalizar tanto antes esta etapa es esencial.

3.1.10 Calificación del sitio de operación

La fase calificación del sitio de operación, tiene dos objetivos claves:

- Asegurar que el sistema o el subsistema trabajen y se desarrollen como fueron planeados.
- Asegurar que las operaciones del personal reciban el entrenamiento y la experiencia relevante.

Los protocolos y los procedimientos preparados ahora son usados en la ejecución de esta fase. Esta ejecución será un conjunto de ejercicios, conducidos por los ingenieros, personal de operación y colegas de control de calidad. Los sistemas serán probados usando los protocolos aprobados. Es esencial que el sistema y los procedimientos para esta área estén en lugar, y que estén documentados completamente. Esto asegurará que esas operaciones del personal, quien conducirá los procesos calificados, están por ellos mismos calificados para su ejecución. Sin esto, sería razonable preguntar si la calificación del proceso subsiguiente fue en sí misma validada o conducida por personal no capacitado.

3.1.11 Planes maestros de validación

La validación ha sido definida muchas veces y los ejemplos típicos son:

- "Acción de demostrar que cualquier equipo trabaja correctamente, y liderea los resultados esperados. El mundo de la validación a veces es muy amplio para incorporar el concepto de calificación.
- El proceso de validación es establecer evidencia documentada la cual proporcione un alto grado de aseguramiento, que un proceso especificado producirá consistentemente un producto que conoce sus especificaciones predeterminadas y sus atributos de calidad.

Cada una de estas definiciones enfatiza la necesidad de demostrar que un sistema hace lo que tiene por objeto a realizar. Para poder ejecutar esto, es esencial un plan. Para un dispositivo completo requerimos de un PROTOCOLO MAESTRO o como se le llama ahora un PLAN MAESTRO DE VALIDACIÓN. Este plan cubre toda la habilidad y todos los aspectos del proceso de validación. Esto puede ser limitado por referirse solo a esas áreas pertinentes a la ingeniería, diseño, construcción y

precomisión. En tales términos esto puede ser mas apropiadamente llamado "un plan de calificación"

A continuación se describe los controles de un PLAN MAESTRO DE VALIDACIÓN TÍPICO:

Contenido

Los contenidos típicos de un plan maestro de validación son los siguientes:

- Introducción.
- Metodología.
- Calificación.
    1. calificación de la instalación.
    2. calificación operacional.
    3. calificación de procesos.
- Personal.
- Horario.
- Mantenimiento preventivo.
- Control de cambios.
- Procedimientos.
- Documentación.
- Apéndices.

La lista anterior puede variar dependiendo la fase del proyecto. En la etapa conceptual, esto será muy preliminar, mientras que en la fase detallada de ingeniería necesita tener un detalle substancial y dirigir todos los aspectos de la calificación. A continuación se describen los contenidos mencionados anteriormente.

1. Introducción

Esta sección primeramente debe estar escrita en primer lugar como una introducción al proceso de validación. La conciencia de la audiencia potencial es muy importante, debido a que el plan puede ser usado para varios propósitos (por ejemplo: como un documento corporativo o una introducción a la validación para la inspección y regulación). Esto debe incluir una descripción del dispositivo,

su establecimiento y equipo y su propósito. La intención y el alcance de la validación deberían estar depositados. En esta sección otros planes y políticas relevantes deben estar referidos y como este plan particular relaciona eso, esto probablemente incluirá declaración de políticas corporativas.

2. Metodología

El plan necesita estar desarrollado en esos estándares que deben ser conocidos, para conocer los requisitos predeterminados. Esta sección del plan debe dirigir estos requisitos por identificación de estándares que están para ser aplicados al dispositivo. Esto será subsecuentemente usado en el desarrollo del criterio de aceptación que son usados para calcular la validación.

Los estándares comprenderán normalmente tres elementos:

- Regulación y orientación de documentos.
- Estándares nacionales.
- Estándares de la compañía.

Esto es normal durante la planeación de la fase de cualquier proyecto para colocar estándares que se aplicarán al el diseño de ingeniería; estos usualmente se refieren como a "los datos básicos de diseño de ingeniería". Esta fase del plan maestro de validación identifica esos estándares que serán críticos en la validación e implementación.

Es importante desarrollar políticas claras para la documentación estándar que será aplicada. Esto puede ser: planeación y ejecución de documentos, protocolos, archivos, reportes y otros. Muchas organizaciones tienen que desarrollar procedimientos y estándares para estos tipos de documentos. Esta sección debe referirse a esos procedimientos. Esos procedimientos deben incluir lo siguiente:

- Planeación para cada tipo de documento.
- Procedimientos autorizados para revisar y aprobar.

3. *Calificación*

Las fases de calificación incluyen: por ejemplo; diseño, instalación, operación, ejecución y proceso. Los elementos de la calificación así como, su alcance se definen como sigue:

- Calificación de Diseño (DQ): se define como: "demostración de evidencia documentada de que el diseño del dispositivo y equipo conocen los requisitos de las especificaciones del usuario".
- Calificación de la Instalación (IQ): se define como: "demostración de verificación documentada de que todos los aspectos claves del diseño, obtención e instalación se adhieran al diseño aprobado y que todas las recomendaciones industriales han sido apropiadamente consideradas".
- Calificación Operacional (OQ): se define como: "demostración de verificación documentada de que el sistema y el subsistema se desempeñan como fueron pensados durante todos los rangos de operación anticipados.
- Calificación de Procesos (PQ) (una vez que se conoce el proceso de validación): se define como: "demostración de verificación documentada de que el proceso hace lo que tiene como propósito hacer".

Las definiciones anteriores se escriben así para que abarquen todos los aspectos del diseño, procedimientos, instalación y comisiones de proceso. Claramente, una flexible pero formalizada propuesta se requiere, y esto puede ser apropiado para adaptar la propuesta de las necesidades específicas del proyecto. El tema importante es asegurar las definiciones dentro de la organización y para un proyecto específico son consistentes y cubren todos los aspectos del proceso de validación, y que la estructura y organización de validación este clara para cualquier inspección de la autoridad.

"Calificación de Diseño": la calificación de diseño cubre todos los aspectos del diseño y el logro del equipo y dispositivos. Esto esta hecho para abarcar todas esas actividades que pueden tomar lugar en la fase de diseño, detalle y desarrollo incluyendo actividades asociadas con procedimientos de equipo y revisión en los trabajos. "La calificación de la instalación" cubre todos los

aspectos de los subcontratistas para situar las actividades y la fase de construcción de un proyecto. Dentro de los términos del proyecto de ingeniería estos límites son esas actividades que un ingeniero debería cubrir en la misma empresa, en el campo, y la finalización mecánica. Es por eso que el IQ incluye diseño, procedimientos, prueba de aceptación de fábrica de equipo antes del envío e instalación. "La calificación operacional" entonces cubre la parte de la precomisión, la fase de OQ es esa parte para que los sistemas o subsistemas se revisen para demostrar que estos actualmente operan como el diseñador lo realizo.

"Calificación de Instalación"; esta calificación del plan maestro de validación debe definir claramente esas áreas de los sistemas de equipo que están para ser calificadas. La lista debe variar dependiendo de la naturaleza del dispositivo. Después de identificar los sistemas que serán calificados, la próxima etapa es desarrollar un plan de calificación.

"Calificación Operacional": el modelo establecido para el IQ es el seguimiento de OQ. La propuesta clave dentro del OQ es identificar y definir esos sistemas que están para ser calificados. Los dispositivos deben estar divididos dentro de los sistemas, con clara definición. Esto debe cubrir en todo el dispositivo, que esta usualmente apropiado para ser constantes con esos desarrollos para el IQ. Los tipos de sistemas identificados serán dependientes de la naturaleza del dispositivo.

Una vez que los sistemas han sido definidos, los protocolos específicos para cada una pueden estar preparados. La información para el IQ y el OQ son frecuentemente presentadas en una forma matricial que identifican esos sistemas a ser calificados.

"Calificación del Proceso": esta es la fase durante cual el proceso de manufactura y procedimientos son calificados. Esto normalmente no sería un área por cual la organización de ingeniería tanto externa como interna debe ser compleja. Esto es responsabilidad de los departamentos de Producción y Control de Calidad.

4. Personal

La gente es la clave para el éxito de cualquier validación. Si el proceso se ejecuta con inapropiada calificación y entrenamiento de personal, entonces la validación

puede ser invalidada. El plan maestro de validación debe depositar los principios para los requisitos del personal. Esto debe dirigir esos aspectos para cada fase del proceso de validación. El personal cambiara durante el diseño de ingenieria, construcción y programación.

El entrenamiento se requiere para aumentar experiencia y calificación y esto se puede proporcionar en la misma empresa o puede ser externo. Los cursos que se toman deben ser dados por especialistas o por proveedores del equipo. Una combinación de ambos es probablemente más atractivo.

5. Horario

Programar un trabajo es esencial y debe ser preparado en una fase temprana. Esto expone el hilo del proceso de validación y lo incorpora dentro del horario del proyecto total. Esto normalmente será en la forma de diagramas y caminos de redes, y esto necesita estar planeado con la misma profundidad con la que esta hecha el proyecto entero.

Un buen plan contendrá todos los rasgos necesarios para identificar cuando las actividades están listas para su ejecución y demostrar en el exterior que el proyecto está bajo control. Esto permite a los recursos estar asignados en un tiempo apropiado para conseguir la actividad. Un típico ejemplo debe ser la finalización de las especificaciones del proceso para permitir colocar el recuento, con la subsiguiente entrega de documentación del vendedor para permitir la preparación del diseño y el protocolo para avanzar.

6. Mantenimiento preventivo

Este elemento es la responsabilidad del departamento del Sitio de Mantenimiento y Operaciones y a menudo se da una baja prioridad en todo el equipo del diseño de ingeniería. Hay un requisito claro para, mantener un dispositivo en un estado de calificación. Un programa de mantenimiento preventivo es un componente esencial de un trabajo de horario para conseguir este objetivo. El plan maestro de validación necesita identificar las necesidades de este programa y, por consiguiente, disminuir su importancia a los diseñadores. El rol del vendedor y los proveedores es muy importante en esta área. Los manuales de operación y mantenimiento deberían estar considerados como una parte clave de la

especificación del programa. Esta actividad debe ser conducida durante la fase de diseño y la documentación requerida debe estar incluida en la requisición. La ejecución de un programa de mantenimiento preventivo puede tomar una gran relevancia en toda la fase de precomisión y comisión.

7. Control de cambios

Una pregunta frecuente es "¿cuándo la validación está completa?. El proceso nunca se finaliza, esto es un proceso continuo, los dispositivos, sus servicios, el equipo y los procesos deben estar siempre en un estado de validación para cumplir con los requisitos regulatorios. El control de cambio se aplica no solo en el proceso de manufactura actual, también en todo el proyecto. El control de cambios debe dirigir todos los aspectos del dispositivo, su diseño, construcción, operación y debe estar dirigido dentro del plan maestro de validación.

Esta sección del plan maestro de validación debe depositar los requisitos para un conjunto de procedimientos para el control de cambios que cubren:

- El proyecto a través del diseño, construcción y comisión.
- El cambio actual inevitablemente ocurrirá en los procesos y equipo y en los aspectos de ingeniería.

8. Procedimientos

Los procedimientos son una parte esencial de cualquier sistema de validación. Esto cubre estándares de ingeniería usados en el diseño del proyecto, a través de fases comisionadas y los procedimientos de operación estándares del dispositivo. Normalmente el plan maestro de validación, identificara el compromiso para escribir procedimientos e identificar un procedimiento para formatos, preparación y autorización de estos procedimientos.

9. Documentación

La sección de documentación del plan maestro de validación normalmente se usa para identificar la documentación que será producida. Dependiendo de la etapa en un proyecto cuando el plan es producido el detalle variará. Un plan

preliminar puede identificar solo las áreas generales de documentos que serán producidas por ejemplo:

- Esquemas de ingeniería.
- Documentos y equipo.
- Aceptación de documentos de fábrica.
- Calificación e instalación de documentos.
- Calificación operacional de documentos.
- Proceso de calificación de documentos.

10. Apéndices

Muchos de los resultados de la ejecución serán documentos escritos. El apéndice se usa comúnmente en los planes maestros más detallados para sostener ejemplos de los tipos de documentos y formatos que serán usados en la ejecución de la etapa.

3.1.12 Resumen

La validación es un elemento clave que debe estar incorporado dentro del diseño y la construcción de dispositivos, servicios y productos. Esto se debe considerar en la fase temprana (ejemplo: estudio conceptual) y debe ser un rasgo importante del proyecto.

El punto de la validación o calificación de requisitos variará con la fase del proyecto. La responsabilidad para su ejecución debe estar claramente definida y designada a la adecuada función de ingeniería, producción y aseguramiento de la calidad. Su ejecución se consigue mejor teniendo un alcance bien definido que se incorpora dentro del plan del proyecto o programa. Esto se consigue mejor con un plan maestro de validación, el cual puede hacer un documento e identificar los requisitos de validación en cada fase del proyecto.

3.2 Validación del Software[45]

La FDA es un organismo con cede en Estados Unidos que se dedica a administrar el correcto funcionamiento de las empresas que se dedican a elaborar como su nombre lo indica alimentos y medicamentos para consumo humano., esto es la FDA impone como requisito la validación de procesos en primera instancia farmaceuticos para obtener las especificaciones predeterminadas y los atributos de calidad que requiera el producto, por una satisfacción del cliente. Pero la FDA no solo verifica el proceso en si del desarrollo de medicamentos, sino también se encarga de todo lo relacionado a su entorno, es decir se encarga de revisar desde la sustancia, hasta los dispositivos o máquinas que se encargan de desarrollar el producto, pasando por el diseño del software, así pues se tiene el siguiente resultado: los análisis de la FDA (Food and Drugs Administration) demuestran que de 3140 llamadas de los dispositivos médicos entre 1992 y 1998 revelan que 242 llamadas (7.7%) están atribuidos a los defectos del software. De esas llamadas 192 (79%) fueron causadas por los defectos del software que fueron introducidos cuando los cambios fueron hechos al software después de su producción y distribución inicial. La validación del software es un requerimiento de la regulación de cualquier Sistema de Calidad, esto fue publicado en el Registro Federal el 6 de Octubre de 1996 y tomo efecto el 1 de Junio de 1997. Los requerimientos de la validación son aplicables para el software usado como: componente en aparatos médicos, para el software que es en sí un aparato médico, para el software en si y para el software que se usa en la producción de maquinas o aparatos o en la implementación de sistemas de calidad.

A no ser que especificamente exente una clasificación de regulación, cualquier software aplicado a un aparato medico desarrollado después del 1 de Junio de 1997 que no interese la clase de aparato que es, esta sujeto para aplicar el diseño de control. Este requerimiento incluye; la terminación del desarrollo actual de los proyectos, todo el nuevo desarrollo de los proyectos, y todos los cambios hechos a los aparatos médicos que incluyan software.

Cualquier software y software usado para automatizar cualquier parte del proceso de producción o cualquier parte del sistema de calidad debe ser validado para su uso. Estos requerimientos se aplican a cualquier tipo de software usado en la automatización de diseño, pruebas, componentes de aceptación, manufactura,

[45] : http://www.fda.gov/cdrh/comp/guidance/938

distribución, o en la automatización de otro aspecto del sistema de calidad. También los sistemas computarizados usados en la creación, modificación, en la obtención y mantenimiento de información y para manejar firmas electrónicas también son sujetos de requerimientos de validación.

El software para aplicaciones más complejas desarrollado en la misma empresa o bajo contrato, es frecuentemente adquirido pensado para un uso en particular. Todo el software para la producción y para el sistema de calidad debe estar documentado en sus requerimientos que definan completamente su uso, y la información contra cualquier resultado de prueba y otras evidencias pueden ser comparadas para mostrar que el software es validado para su uso pensado.

El uso del software en los aparatos médicos automatizados, en la fabricación automatizada y en la operación del sistema de calidad se esta incrementando. El software puede tener muchas capacidades y solo algunas de esas capacidades son necesitadas para la fabricación de los aparatos médicos. La fabricación de estos aparatos es responsabilidad del uso adecuado del software y sus herramientas y el uso para producir proyectos. Cuándo se adquiere el software para la fabricación se debe estar seguro que se va a llevar a cabo hacia lo que fue pensado en cada aplicación.

3.2.1 Contexto para la validación del software.

Mucha gente se ha preguntado y se extraña sobre la validación en sistemas de calidad donde se aplica cualquier tipo de software pero esto se viene haciendo en las industrias de competitividad desde hace más de 20 años. Debido a la gran variedad de dispositivos que necesitan software, software en si, procesos, sistemas y facilidades de fabricación no es posible quedarse en un documento para todas las especificaciones de los elementos que son aplicables en la validación. Estos extensos conceptos proveen una aceptable tarea para construir una comprensiva aproximación para la validación del software.

A continuación se muestra un diagrama en forma general de la validación del software

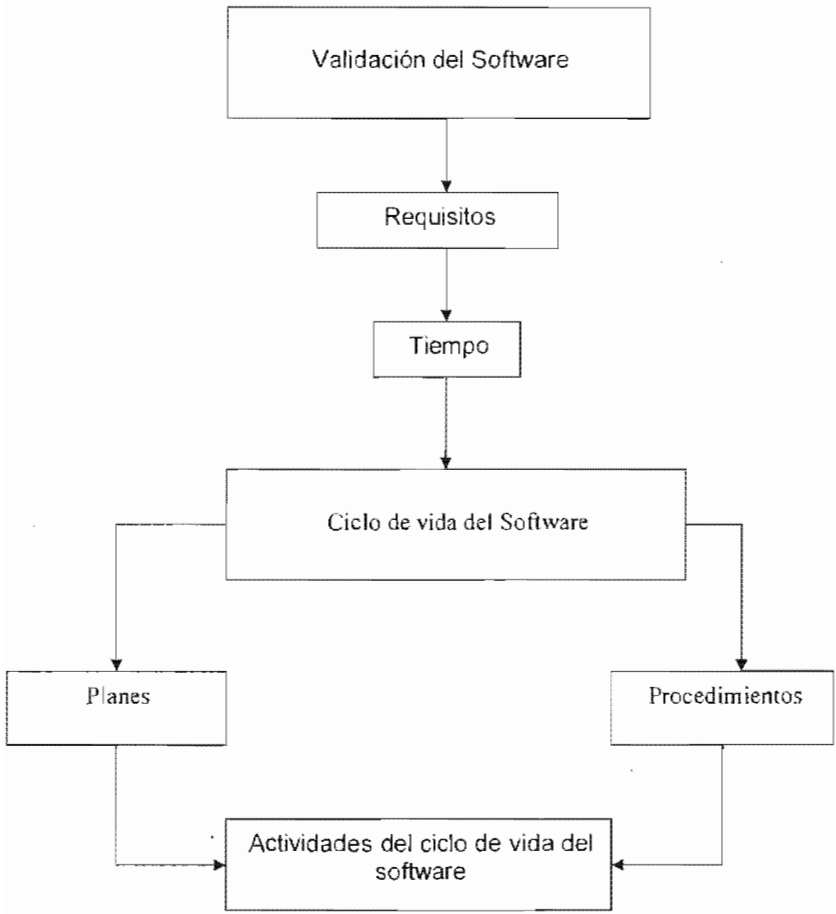


Figura 3.1

3.2.1.1 Requisitos y especificaciones

Mientras que los estados de la regulación del sistema de calidad que diseñan requisitos de entradas deben ser documentados y las especificaciones requeridas deben ser verificadas, las regulaciones no clarifican la distinción entre los términos "requisito" y "especificación". Un **requisito** puede ser cualquier necesidad o

expectación por un sistema o por un software. Los requisitos reflejan lo indicado o la necesidad implícita del cliente. Puede haber diferentes tipos de requisitos (diseño, funcionalidad, implementación, desarrollo o requisitos físicos). Los requisitos del software están típicamente derivados de los requisitos del sistema para esos aspectos de funcionalidad del sistema que han sido distribuidos para el software. Los requisitos del software están típicamente indicados en términos funcionales y están definidos y actualizados como un desarrollo del progreso del proyecto. El éxito en la exactitud de la documentación del los requisitos del software es un factor crucial en el éxito del resultado de la validación del software.

Una **especificación** se define como "un documento que afirma requisitos" Esto puede incluir también dibujos u otros documentos relevantes. Hay diferentes tipos de especificaciones; escritas, especificaciones de los requisitos del sistema, especificaciones de los requisitos del software, etc. Todos esos documentos establecen "requisitos especificados" y diseñan salidas para diversas formas de verificación.

3.2.1.2 Verificación y validación.

La regulación del sistema de calidad esta armonizada con la norma ISO 8402:1994, la cual trata la verificación y la validación como términos separados y distintos. Por otro lado muchos artículos de revistas en la ingeniería del software y libros de texto usan los términos **validación y verificación** como intercambiables, o en algunos casos se refieren a la "verificación, validación, y prueba del software como si este fuera un solo concepto sin ninguna distinción entre los tres términos.

- **La Verificación del software** proporciona evidencia objetiva de que el diseño de la producción de una fase en particular del desarrollo del ciclo vida del software reúne todos los requisitos especificados para esa fase. La verificación del software contempla compatibilidad, totalidad y exactitud del software y su documentación de soporte. La prueba del software es una de muchas actividades de verificación para confirmar que la capacidad de producción del desarrollo del software conoce sus salidas. Otras actividades de verificación incluye análisis dinámico, inspección de código y documentación, y otras técnicas.

- **La Validación del software** es una parte de validación para un dispositivo finalizado. Pero no se define separadamente dentro de la regulación del Sistema de Calidad. La FDA para este aspecto define a la validación del software como: "**es la confirmación por medio de una examinación y estipulación de una evidencia objetiva de que las especificaciones del software conforman las necesidades de los usuarios y sus usos para el que fue desarrollado y que los requisitos particulares implementados a través del software puedan cumplirse en forma consistente**". Desde que el software es parte del sistema del hardware, la validación típica del software incluye evidencia de que los requisitos de este han sido implementados correcta y completamente y es identificable a los requisitos del sistema. Una conclusión de la validación del software es la alta dependencia a la prueba, inspección, análisis y otras tareas de verificación o comprobación a cada fase del desarrollo del ciclo de vida del software. La prueba de la funcionalidad del dispositivo como software en un ambiente simulado y el sitio de prueba de el usuario están típicamente incluidos como componentes de un programa único de diseño de validación para automatizar el dispositivo del software.

La verificación y validación del software son difíciles ya que el diseñador no puede estar probándolo por siempre y es difícil saber cuanta evidencia es suficiente. En gran medida la validación del software es cuestión de desarrollar un "nivel de confianza" de que en el dispositivo se encuentren todos los requisitos y expectativas del usuario para las funciones automatizadas del software y características del dispositivo.

Las medidas tales como defectos encontrados en las especificaciones de los documentos, estimación de defectos permanentes, pruebas de cubrimiento y otras técnicas, todas estas son usadas para desarrollar un nivel aceptable de confianza para enviar el producto. El nivel de confianza y por consiguiente el nivel de validación, verificación y las pruebas de esfuerzo necesitadas cambiaran dependiendo del riesgo de seguridad propuestas por las funciones automatizadas del dispositivo.

3.2.1.3 IQ/OQ/PQ

Durante muchos años la FDA y la industria reguladora han intentado encontrar una definición para la validación del software, dentro del contexto de la terminología del proceso de validación. Por ejemplo los documentos de la industria y otras guías de validación de la FDA a veces describen a la validación del software en términos de requisitos de instalación (Installation Qualification = IQ), requisitos operacionales (Operational Qualification = OQ) y requisitos de desempeño (Performance Qualification = PQ).

Mientras que las terminologías anteriormente mencionadas han servido para su propósito, estas son una de muchas maneras legítimas de organizar las tareas de la validación del software, estas terminologías no pueden ser bien entendidas entre muchos profesionales del software. Sin embargo tanto la FDA como los fabricantes del software necesitan ser consientes de las diferencias entre estas terminologías.

3.2.1.4 Desarrollo del software como parte del diseño del sistema.

La decisión para implementar la funcionalidad del sistema usando un software es una de esas decisiones típicas que se hacen durante el diseño del sistema. Los requisitos del software se derivan típicamente de los requisitos del sistema global, y el diseño para esos aspectos dentro del sistema se llevaran a cabo usando el software. Hay necesidades de usuarios y usos intencionales para finalizar el software, pero los usuarios no especifican si esos requisitos son encontrados por el software, hardware o una combinación de ambos. Por lo tanto la validación del software debe ser considerada dentro del contexto del diseño total de validación para el sistema. Una especificación documentada de requisitos representa las necesidades del usuario y las intenciones de uso del producto desarrollado. **Una meta principal de la validación del software es "demostrar que todos los complementos del software cumplen con la documentación de este y sus requisitos de el sistema**. La exactitud y la integridad de los requisitos del sistema y los requisitos del software deben ser dirigidas como parte del diseño del proceso de validación para el dispositivo. La validación del software incluye la confirmación de la conformidad de todas las especificaciones del software y la confirmación de que todos los requisitos de este son identificables a las especificaciones del sistema. La confirmación es una parte importante de todo el diseño de validación, ya que esta

asegura todos los aspectos de los dispositivos donde se utiliza el software que conforman las necesidades del usuario y sus usos intencionales.

3.2.1.5 Diferencia del software al hardware.

El software comparte muchas de las mismas tareas de ingeniería que el hardware pero este tiene algunas y muy importantes diferencias. Por ejemplo:

- La inmensa mayoría de los problemas del software son atribuidos en los errores hechos durante el diseño y el proceso de desarrollo. Mientras que la calidad de un producto de hardware depende del diseño, desarrollo, y fabricación; la calidad de un producto de software, primeramente depende del diseño y desarrollo con una mínima inclusión en la fabricación de este. La fabricación del software consiste en una reproducción que pueda ser fácilmente verificada. No es difícil fabricar miles de copias que funcionen exactamente como el original; la dificultad viene en la adquisición del programa original y conocer todas sus especificaciones.
- Una de las más significativas características del software es la habilidad de ejecutar alternativamente series de comandos, basadas en distintas entradas. Esta característica es un factor de mayor contribución a otra característica del software, "su complejidad". Incluso los programas cortos pueden ser muy complejos y difíciles de entender por completo.
- Típicamente la prueba en si no puede confirmar enteramente de que el software esta completo y correcto. Además de la prueba otras técnicas de verificación y procesos de desarrollo estructurados y documentados deben ser combinados para asegurar una comprensiva validación.
- La diferencia del hardware al software no es la entidad física. De hecho el software puede mejorar con el uso y el tiempo, cuando se descubren los defectos ocultos y se quitan. Sin embargo como el software es constantemente actualizado y cambiado, tales mejoras a veces son contrarrestadas por los nuevos defectos introducidos dentro del tiempo del cambio o mejora del software.
- A diferencia de algunos errores del hardware, los fracasos del software ocurren sin ninguna advertencia. La división del software le permite seguir

difiriendo el camino durante la ejecución. Pueden esconderse algunos defectos latentes hasta mucho después que un software ha sido introducido en el mercado.

- Otra característica relacionada con el software es la velocidad y la facilidad con la que puede ser cambiado. Este factor puede causar que los profesionales y los no profesionales del software crean que los problemas del software pueden ser corregidos fácilmente. Combinado con una falta de entendimiento del software, esto puede llevar a los gerentes a creer que la ingeniería estrechamente controlada no es tan necesaria para el software como para el hardware. De hecho lo opuesto es la verdad, debido a su complejidad el proceso de desarrollo del software debería ser más controlado que para el hardware, para prevenir problemas que no pueden ser fácilmente detectados en el desarrollo del proceso.
- Los cambios aparentemente insignificantes en el código del software pueden crear problemas inexplicables muy significantes en otra parte del software. El desarrollo del software debe ser suficientemente bien planeado, controlado y documentado para detectar y corregir los resultados inesperados de los cambios del software.
- Dado que la demanda alta de los profesionales del software y la mano de obra móvil, el personal del software que hace los cambios de mantenimiento pueden no haber estado involucrados en el desarrollo original del software. Por consiguiente la documentación precisa y completa es esencial.
- Históricamente los componentes del software no han sido frecuentemente estandarizados e intercambiables como los componentes del hardware. Sin embargo los diseñadores del software están empezando a usar técnicas y herramientas de desarrollo. La metodología basada en Objeto-Orientado, entre otras y el uso de componentes del software sostienen la promesa de un desarrollo de software más rápido y menos caro. Sin embargo los componentes requieren de una atención muy cuidadosa durante la integración. Antes de la integración el tiempo es necesario para definir el desarrollo del código del software y para entender completamente el comportamiento de los componentes.

Por esta y por otras razones la ingeniería del software necesita un mayor nivel de control y dirección minuciosa que la ingeniería del software.

3.2.1.6 Beneficios de la validación del software.

La validación es una herramienta crucial usada para asegurar la calidad de los dispositivos del software y para el software de las operaciones automatizadas. La validación del software puede incrementar la utilidad y la fiabilidad del dispositivo, resultando en un decremento de la taza de fallas, menos llamadas (quejas o soporte) y acciones correctivas, menos riesgos de pacientes y usuarios (en el caso de dispositivos médicos), y reducción de responsabilidad para los fabricantes de un dispositivo. La validación del software también puede reducir los términos costosos haciéndolo fácil y menos caro las modificaciones y los cambios de este. El mantenimiento del software puede representar un largo porcentaje del total del costo de este sobre su ciclo de vida entero. Un proceso de validación establecido y detallado ayuda a reducir los costos a largo plazo del software reduciendo el costo de la validación por cada información subsiguiente de este.

3.2.1.7 Revisión del diseño.

Las revisiones del diseño son documentadas, detalladas y examinadas para evaluar los requisitos adecuados del diseño y su capacidad del software para poder conocer sus requisitos e identificar los problemas. Mientras pueda haber muchas revisiones técnicas informales que ocurren dentro del equipo de desarrollo durante un proyecto de software, una revisión formal del diseño es más estructurada e incluye participación de otras personas afuera del equipo de desarrollo. Las revisiones formales del diseño pueden estar referenciadas o pueden incluir la participación de otros ingenieros que no participen en el equipo de desarrollo. Las revisiones del diseño pueden estar conducidas separadamente para el software, después el software se integra al hardware dentro del sistema. La revisión del diseño debe incluir una examinación de los planes de desarrollo, especificaciones de los requisitos, especificaciones de diseño, prueba de planes y procedimientos, otros documentos y actividades asociados con el proyecto, verificación de resultados de cada fase del ciclo de vida y los resultados de la validación de todo el dispositivo.

La revisión del diseño es una herramienta primaria para manejar y evaluar el desarrollo de los proyectos. Por ejemplo la revisión formal del diseño permite a la dirección confirmar que todas las metas definidas en el plan de validación del software se han conseguido. La regulación del sistema de calidad requiere por lo menos una revisión formal del diseño durante el proceso de diseño del dispositivo. Sin embargo se recomienda que se hagan múltiples revisiones. La revisión formal del diseño es especialmente importante en la actividad de los requisitos, antes de que los recursos principales hayan sido comprometidos a las soluciones especificas del diseño. Los problemas encontrados en este punto pueden ser resueltos mas fácilmente, ahorrar tiempo y dinero y reducir la probabilidad de tener un problema critico.

Las respuestas a algunas preguntas claves deben ser documentadas durante la revisión formal del diseño. Estas incluyen:

- ¿Se tienen las tareas adecuadas, los resultados esperados, rendimientos o productos establecidos para cada ciclo de vida del software?
- Las tareas y los resultados esperados, rendimientos o productos de cada actividad de el ciclo de vida:
    - ¿Cumplen con los requisitos de otra actividad del ciclo de vida del software en términos de exactitud, integridad, consistencia y precisión?
    - ¿Satisface los estándares, prácticas y convenciones de esa actividad?
    - ¿Establece una base apropiada para comenzar las tareas para la próxima actividad del ciclo de vida del software?

3.3 Principios de la Validación del Software[46]

3.3.1 Requisitos

Una especificación de requisitos del software bien documentada mantiene una linea de base para la validación y la verificación. El proceso de validación del software no

[46] http://www.fda.gov/cber/guidelines.htm

puede ser completado sin una especificación establecida de los requisitos del software.

3.3.2 La prevención del defecto

La garantía de la calidad del software se necesita enfocar en la prevención de la introducción de defectos en el proceso de desarrollo del software y no en tratar de "probar la calidad dentro" del código del software después de que fue escrito. La prueba del software es muy limitada en su habilidad de que aparezcan todos los defectos latentes en el código del software. La prueba del software es una actividad necesaria. Sin embargo en la mayoría de los casos la prueba del software no es suficiente para establecer la confianza de que el software es adecuado para su uso. Para establecer esa confianza, los diseñadores del software deben usar una mezcla de métodos y técnicas para prevenir los errores y detectar los errores que ocurren dentro del software. La mejor mezcla de métodos depende de muchos factores incluso el entorno en donde se desarrolla, aplicación, lugar del proyecto, lenguaje y riesgo.

3.3.3 Tiempo y esfuerzo

Construir un caso de validación de software requiere tiempo y esfuerzo. La preparación de la validación del software empieza durante la planeación del diseño y desarrollo de este. La conclusión final de que el software esta validado debe estar basado en la evidencia coleccionada de los esfuerzos planeados conducidos a través del ciclo de vida.

3.3.4 Ciclo de vida del software

La validación del software toma lugar dentro del ambiente de un ciclo de vida de software establecido. El ciclo de vida del software contiene tareas de ingeniería necesarias para soportar el esfuerzo de la validación. Además el ciclo de vida del software contiene verificación especifica y tareas de validación que son apropiadas para el uso del software.

3.3.5 Planes

El proceso de validación del software esta definido y controlado a través del uso de un plan. El plan de validación del software define eso que será logrado a través del esfuerzo de validación del software. Los planes de validación del software son una significante herramienta en el sistema de calidad. Los planes de validación especifican las áreas como: el alcance, acercamiento, recursos horarios y magnitud de actividades, tareas y artículos de trabajo.

3.3.6 *Procedimientos*

El proceso de validación del software es ejecutado a través del uso de procedimientos. Estos procedimientos establecen como dirigir el esfuerzo de validación del software. Los procedimientos deben identificar y especificar acciones o secuencia de acciones que deben ser tomadas para completar las actividades individuales de validación, tareas y artículos de trabajo.

3.3.7 La validación del software después de un cambio

Debido a la complejidad del software, un cambio local aparentemente pequeño puede tener un impacto significativo en el sistema. Cuando cualquier cambio (aun un cambio pequeño) se le hace al software, la posición de la validación necesita ser reestablecida. Siempre que el software sufra cambios, el análisis de validación debe ser conducido no solo para la validación del cambio que se hizo, si no también debe de determinar el impacto del cambio en todo el sistema del software. Basado en estos análisis el diseñador del software debe dirigir un nivel apropiado de regresión para mostrar que las porciones incambiables pero vulnerables del sistema no han sido adversamente afectadas. Los controles del diseño y las pruebas apropiadas de regresión proporcionan la confianza de que el software esta validado después de un cambio.

3.3.8 La cobertura de la validación

La cobertura de la validación debe estar basada en su complejidad y en el riesgo seguro, no en el tamaño de la empresa ni en el limite de los recursos. La selección de actividades de validación, tareas y artículos de trabajo deberán ser correspondientes al diseño del software y el riesgo asociarlo con el uso del software

para su uso. Para los riesgos más bajos del dispositivo solo las actividades básicas de la validación pueden ser dirigidas. Como el incremento adicional del riesgo, las actividades de validación deben ser incrementadas para cubrir el riesgo adicional. La validación documentada debe ser suficiente para demostrar que los planes de validación del software y los procedimientos han sido completamente exitosos.

3.3.9 Revisión independiente

Las actividades de validación deben ser dirigidas usando la garantía básica de la calidad de "la independencia de la revisión" la validación por si misma es extremadamente difícil. Cuando es posible una evaluación independiente siempre es mejor, especialmente por los altos riesgos de las aplicaciones. Algunas empresas contratan una tercera validación y verificación independiente, pero esta solución no siempre es fiable. Otro acercamiento es asignar miembros internos del staff que no estén involucrados en un diseño en particular o sus implementos, pero quien tiene el conocimiento necesario para evaluar el proyecto y llevar las actividades de la verificación y la validación. Pequeñas firmas de software necesitan ser creativas en como las tareas están organizadas y asignadas para mantener una revisión independiente.

3.3.10 Flexibilidad y responsabilidad

La implementación específica de los principios de validación del software puede ser diferente de una aplicación a otra. La fabricación del dispositivo tiene flexibilidad y escoge como aplicar los principios de la validación, pero tiene la última responsabilidad para demostrar que el software ha sido validado.

El software esta diseñado, desarrollado, validado y regulado en un ancho espectro de ambientes, y para cada variedad de dispositivos con varios niveles de riesgo. La FDA regulo que las aplicaciones de los dispositivos médicos incluyan software que:

- Sea un componente, una parte o un accesorio de un dispositivo médico.
- Sea un dispositivo médico.
- Se use en manufactura, diseño y desarrollo o en otras partes del sistema de calidad.

En cada ambiente, los componentes del software de muchas fuentes, pueden ser usados para crear una aplicación. Además los componentes del software vienen en muchas formas diferentes por ejemplo: aplicación del software, sistemas operativos, compiladores, y muchas más. La validación del software en estos ambientes puede ser una tarea compleja; por consiguiente es apropiado que todos los principios de validación sean considerados cuando se diseña el proceso de validación del software. El resultado del proceso de validación debe estar acorde con el riesgo de seguridad asociado con el sistema, dispositivo o proceso.

Las actividades y tareas de la validación del software pueden estar dispersas, ocurriendo en diferentes locaciones y siendo conducidas por diferentes organizaciones. Sin embargo, sin tener en cuenta la distribución de tareas, relaciones contractuales, fuente de componentes o el ambiente de desarrollo el fabricante del dispositivo, o el diseñador de la especificación retiene la ultima responsabilidad de asegurar de que el software esta validado.

3.3.11 Tareas y actividades

La validación del software es cumplida a través de una serie de actividades y tareas que son planeadas y ejecutadas en varias fases del desarrollo de ciclo de vida del software. Estas tareas pueden ser acontecidas solo una vez, o pueden ser repetidas muchas veces de pendiendo de el modelo de ciclo de vida usado y el alcance de los cambios hechos al progreso de el software.

3.4 Actividades del ciclo de vida del software

Los diseñadores del software deben establecer el modelo de ciclo de vida que sea apropiado para su producto y para su organización. El modelo del ciclo de vida del software incluye lo siguiente:

- El plan de calidad.
- Definición de los requisitos del sistema.
- Detalle de la especificación de requisitos del software.
- Especificación del diseño del software.
- Código.

- Pruebas.
- Instalación.
- Operación y soporte.
- Mantenimiento.
- Retiro.

La verificación, pruebas y otras tareas que soportan a la validación del software ocurren durante cada actividad antes mencionadas. Un modelo de ciclo de vida organiza estas actividades de desarrollo de software de varias maneras y mantiene una estructura de monitoreo y control al proyecto de desarrollo del software.

3.4.1 Tareas típicas de la validación de soporte

Para cada actividad de el ciclo de vida del software, hay ciertas tareas típicas que mantienen la conclusión de que el software esta validado. Sin embargo las tareas específicas que son presentadas, su orden de presentación, y la repetición y el ritmo de la presentación deberían ser dictadas por el modelo de ciclo de vida especifico del software que es seleccionado. Para las aplicaciones de muy bajo nivel, ciertas tareas no necesariamente serán usadas del todo. Sin embargo, el diseñador debe por lo menos considerar cada una de esas tareas y debe definir y documentar si esas tareas son o no apropiadas para su aplicación específica.

3.4.2 Planeación de la calidad

La planeación del desarrollo y el diseño deben culminar en un plan que identifique: las tareas necesarias, procedimientos, recursos necesarios y requisitos de la revisión de la gestión administrativa, incluyendo las revisiones formales de diseño. Un modelo de ciclo de vida de software y las actividades asociadas deberían ser identificadas tan bien como esas tareas necesarias para cada actividad del ciclo de vida del software. El plan debe incluir:

- Las tareas especificas para cada actividad de ciclo de vida del software.
- La enumeración de factores importantes de calidad (por ejemplo: la fiabilidad, mantenimiento y el uso.).

- Métodos y procedimientos para cada tarea.
- El criterio de aceptación de las tareas.
- El criterio para definir y documentar los rendimientos en condiciones que permitirán la evaluación de la conformancia para las entradas de los requisitos.
- Entradas para cada tarea.
- Rendimientos de cada tarea.
- Los roles, recursos y responsabilidad de cada tarea.
- Riesgos y suposiciones.
- Documentación de las necesidades de los usuarios.

La dirección debe identificar y proporcionar el apropiado ambiente para el desarrollo del software y los recursos de este. Típicamente cada tarea requiere el personal así como los recursos físicos. El plan debe identificar al personal, la facilidad y el equipamiento de recursos para cada tarea y el rol de riesgo que tomara la dirección. Una configuración del plan de la dirección debe ser desarrollada, eso guiara y controlara múltiples actividades paralelas de desarrollo y asegurara la documentación y comunicación apropiada. Los controles son necesarios para asegurar la relación positiva entre todas las versiones aceptadas de los documentos especificados, código fuente, código objeto, y pruebas de acompañamiento que comprendan al sistema del software. Los controles también deben asegurar la identificación exacta y a los accesos a las versiones actuales aceptadas.

Los procedimientos deben ser creados para informar y resolver las anomalías del software encontradas a través de la validación u otras actividades. La dirección debe identificar los reportes y especificar los contenidos, formatos y los elementos orgánicos responsables para cada reporte. Los procedimientos también son necesarios para la revisión y aprobación de los resultados del desarrollo del software, incluyendo los elementos orgánicos responsables para cada revisión y aprobación.

Tareas típicas de la planeación de la calidad:

- Riesgo del plan de dirección.

- Configuración del plan de dirección.
- La garantía del plan de la calidad del software.
    1. El plan de validación y verificación del software
        - ✓ Tareas de validación y verificación y criterio de aceptación.
        - ✓ Horarios y asignación de recursos.
        - ✓ Reporte de requisitos.
    2. Diseño formal de la revisión de requisitos.
    3. Otras técnicas de revisión de requisitos.
- Reporte de problemas y resolución de procedimientos.
- Otras actividades de soporte.

3.4.3 Requisitos

Los requisitos desarrollados incluyen la identificación, análisis y documentación de información acerca del dispositivo y su uso intencionado. Áreas de especial importancia incluyen asignación de funciones del sistema para el hardware y software, condiciones de operación, características de el usuario, el peligro potencial y tareas anticipadas. Además los requisitos deben declarar el uso intencional del software.

El documento de los requisitos especificados del software debe contener una definición escrita de las funciones del software. No es posible validar el software sin unos requisitos documentados y predeterminados del software. Los requisitos típicos del software especifican lo siguiente:

- Todas las contribuciones de los sistemas del software.
- Todos los rendimientos del sistema del software.
- Todas las funciones que el sistema del software va a desarrollar.
- Todos los requisitos que el software encontrara.
- La definición de todo lo exterior y la interfaz del usuario así como cualquier sistema interno de interfaz.
- Como es que los usuarios van a interactuar con el sistema.
- Que constituye un error y como los errores deben ser tratados.
- Los tiempos de respuesta necesarios.

- El ambiente de funcionamiento para el software, si este es una restricción de diseño (ejemplo: plataforma del hardware, sistema operativo.).
- Todos los rangos, límites, lo predeterminado y los valores específicos que el software aceptara.
- La seguridad de los requisitos relacionados, especificaciones, características, o funciones que serán implementadas en el software.

La seguridad de los requisitos del software está derivada de un riesgo técnico de dirección de proceso que esta integrado con el proceso de desarrollo de los requisitos del sistema. La especificación de los requisitos del software debe identificar claramente los riesgos potenciales que pueden resultar de un fracaso del software en el sistema así como cualquier requisito de seguridad que esta implementado en el software. Las consecuencias del fracaso del software deben ser evaluadas, junto con los medios para mitigar dicho fracaso. De este análisis sería posible identificar las medidas apropiadas para prevenir los daños.
La regulación del sistema de calidad requiere un mecanismo para dirigir requisitos incompletos, ambiguos o conflictivos. Cada requisito (por ejemplo: hardware, software, usuario y seguridad) identificado en la especificación de requisitos del software debe ser avaluados para la precisión, consistencia, corregibilidad y claridad. Por ejemplo los requisitos del software deben ser evaluados para verificar que:

- No haya inconsistencias internas entre los requisitos.
- Que todo el desarrollo de los requisitos para el sistema han sido deletreados.
- La tolerancia de defectos, seguridad y la seguridad de los requisitos estén completas y correctas.
- La asignación de las funciones del software este precisa.
- Que los requisitos del software sean apropiados para los riesgos del sistema.
- Que todos los requisitos estén expresados en términos mensurables u objetivamente confirmables.

Un requisito trazable del análisis del software debe ser dirigido para trazar los requisitos del software a los requisitos del sistema y arriesgar los resultados del análisis. Además para cualquier otro análisis y documentación usada para verificar los requisitos del software, una revisión del diseño formal se recomienda para confirmar que los requisitos están totalmente especificados y apropiados antes que empiece el esfuerzo del diseño del software. Los requisitos pueden ser aprobados o incrementados pero debe tenerse en cuenta que las interacciones e interfaces entre los requisitos del software y hardware estén apropiadamente revisadas, analizadas y controladas.

Tareas tipicas de los requisitos

- Análisis preliminares de los riesgos.
- El análisis de trazabilidad.
    1. Requisitos del software hacia los requisitos del sistema.
    2. Requisitos del software hacia los riesgos del análisis.
- Descripción de las características de los usuarios.
- Lista de características y limitaciones de la memoria primaria y secundaria.
- Evaluación de los requisitos del software.
- Análisis de la interfase de requisitos del usuario del software.
- Prueba de el plan de generación de sistemas.
- La aceptación del plan.
- Revisión ambigua o análisis.

3.4.4 Diseño

En el proceso de diseño, la especificación de requisitos del software es trasladada dentro de una representación lógica y física de la implementación del software. La especificación de diseño del software es una descripción de que el software debe hacer y como lo debe hacer. Debido a la complejidad del proyecto o a la habilidad de las personas con los niveles variantes de responsabilidades técnicas para entender claramente el diseño de la información, la especificación del diseño puede contener un alto resumen de nivel y el diseño y una información detallada del diseño. Una especificación de diseño completa reprime al programador para permanecer dentro

del intento de lo convenido en los requisitos y el diseño. Una especificación completa del diseño del software relevara al programador la necesidad de hacer decisiones de diseño exactas.

El diseño del software necesita dirigir factores humanos. Los factores de ingeniería humana deben ser vistos dentro del diseño entero y el proceso del desarrollo, incluyendo los requisitos del diseño del dispositivo, análisis y pruebas. La seguridad del dispositivo y los problemas de utilidad deben ser considerados en los diagramas de flujo en vías de desarrollo, diagramas de estado, herramientas prototipo y planes de prueba. También las tareas y análisis de la función, los análisis de riesgo, el prototipo pruebas y revisiones y las pruebas de utilidad deben ser desarrolladas.

Las especificaciones de diseño del software deben incluir lo siguiente:

- La especificación de requisitos del software, incluyendo el criterio predeterminado para aceptar al software.
- El análisis de riesgo del software.
- Los procedimientos del desarrollo y guías del código.
- Documentación del sistema, para describir el contexto del sistema cada vez que se intenta correr el programa, incluyendo la relación del hardware con el software y el ambiente físico.
- El hardware usado.
- Parámetros de medición o grabación.
- Estructura lógica (incluyendo control lógico) y pasos de procesamiento lógicos (ejemplo: el algoritmo).
- Estructura de datos y diagramas de datos fluidos.
- Definición de variables (control, y datos) y la descripción de donde son usadas.
- Error, alarma y mensajes de prevención.
- Software de apoyo (ejemplo: sistemas operativos, controladores y otras aplicaciones del software).
- Comunicaciones de links.
- Cualquier restricción adicional en los elementos de arriba.

Los cuatro primeros elementos mencionados arriba usualmente son documentos pre-existentes separados que son incluidos por referencia en la especificación del diseño del software.

Las actividades que ocurren durante el diseño del software tienen varios propósitos. Las evaluaciones del diseño del software son dirigidas para determinar si el diseño esta completo, correcto, consistente, ambiguo, feasible y sostenible. La apropiada consideración de la arquitectura del software durante el diseño puede reducir la magnitud de los esfuerzos de la validación cuando los cambios del software son necesarios. La evaluación del diseño del software puede incluir análisis fluidos de control, datos fluidos, complexión, tamaño, asignación de memoria, análisis críticos y muchos otros aspectos del diseño. Un análisis de trazabilidad debe ser dirigido para verificar que el diseño del software implemente todos los requisitos del software. Como una técnica para identificación cuando los requisitos no son suficientes el análisis de trazabilidad debe también verificar que todos los aspectos del diseño sean identificables a los requisitos del sistema. Un análisis de comunicación debe ser conducido para evaluar el diseño propuesto con respecto al hardware, usuario y los requisitos del software. El análisis del riesgo del software debe ser reexaminado para determinar cualquier peligro adicional que ha sido identificado y si cualquier nuevo peligro ha sido introducido al diseño.

Al final de la actividad de diseño del software, una revisión formal del diseño debe ser dirigida para verificar que el diseño es correcto, consistente, completo y preciso antes de llevar a cabo el plan. Las partes del diseño pueden ser aprobadas y pueden soltarse para la implementación, pero se debe tomar en cuenta que las interacciones y las comunicaciones entre varios elementos están propiamente revisadas, analizadas y controladas.

Más modelos de desarrollo de software serán reiterativos. Es probable que esto produzca varias versiones de la especificación de requisitos del software y la especificación de diseño del software. Todas las versiones aprobadas deben archivarse y controlarse de acuerdo con la dirección de la configuración establecida.

Tareas típicas de diseño:

- Actualización del análisis de riesgo del software.

- La trazabilidad del análisis. La especificación del diseño hacia los requisitos del software y viceversa.
- La evaluación de diseño del software.
- El diseño de la comunicación.
- La prueba del modulo de la generación del plan.
- Integración del plan de prueba.
- Generación del diseño de prueba (modulo, integración y sistema).

3.4.5 Construcción o codificación

El software puede ser construido o codificado. (Es decir programando) o por la conjunción de los componentes del código del software (ejemplo librerías del código) para usar en una nueva aplicación. La codificación es la actividad del software donde la especificación detallada del diseño esta implementada como código fuente. La codificación es el nivel más bajo de abstracción para el proceso de desarrollo. Es la última etapa dentro de la descomposición de los requisitos del software donde se traducen las características del módulo en lenguaje de programación. La codificación usualmente involucra el uso de un lenguaje de alto nivel, pero también puede usar lenguaje ensamblador o micro código para operaciones de tiempo crítico. El código fuente puede ser cualquier compilador o interprete para usarse en una plataforma de hardware. Las decisiones sobre la selección del lenguaje de programación y herramientas de construcción de software deben incluir consideraciones del impacto sobre la calidad subsiguiente de la evaluación de tareas. Algunos compiladores ofrecen niveles opcionales y comandos para checar errores ayudando a poner a punto el código. Los diferentes niveles de chequeo de error pueden ser usados a lo largo del proceso de codificación, advertencias u otros mensajes del compilador pueden o no ser registrados. Sin embargo al final de la codificación el nivel mas riguroso de error es normalmente usado para documentar que errores de compilación sigue permaneciendo en el software. También para la compilación final, debe haber documentación del proceso de compilación y su resultado, incluyendo cualquier advertencia u otro mensaje del compilador y su resolución o justificación para la decisión de dejar los problemas que no estén resueltos.

Las empresas frecuentemente adoptan un código específico que establecen políticas de calidad y procedimientos relacionados al proceso de codificación del software. El código fuente debe ser evaluado para verificar su cumplimiento con la guía de codificación especificada. Cada línea debe incluir las convenciones de la codificación con respecto a la claridad, estilo, complejidad, dirección y comentarios. Los comentarios del código deben proporcionar información útil y descriptiva para el módulo incluyendo entradas esperadas y rendimientos, variables de referencia, los tipos de datos esperados y las operaciones a ser realizadas. El código fuente también debe ser evaluado para verificar su cumplimiento con la especificación de diseño correspondiente. Los módulos listos para la integración y las pruebas deben tener documentación de cumplimiento con la codificación y cualquier otra política de calidad y procedimientos.

Las evaluaciones del código fuente a menudo están implementadas como código de inspección. Tales análisis estáticos proporcionan efectivos medios para detectar errores antes de la ejecución del software. Ellos permiten la examinación de cada error y también ayudan a enfocar la comprobación dinámica del software. Las empresas pueden usar el manual de chequeo con controles apropiados para asegurar la consistencia y la independencia. Las evaluaciones del código fuente deben extenderse para verificar las conexiones internas entre los módulos y las capas (conexiones físicas horizontales y verticales) y el cumplimiento con las especificaciones de su diseño. La documentación de los procedimientos usados y los resultados de las evaluaciones del código fuente deben mantenerse como parte de la verificación del diseño.

Los análisis de trazabilidad del código fuente es una herramienta importante para verificar que todos los códigos están conectados para establecer especificaciones y procedimientos de prueba. Los análisis de trazabilidad del código fuente deben ser dirigidos y documentados para verificar que:

- Cada elemento de la especificación del diseño del software ha sido implementado dentro del código.
- Los módulos y las funciones implementadas en el código pueden ser remontadas a un elemento en las especificaciones de diseño del software y los análisis de riesgo.

- Las pruebas para los módulos y las funciones pueden ser remontadas a un elemento en la especificación de diseño y el análisis de riesgo.
- Las pruebas para los módulos y funciones pueden remontarse al código fuente para los mismos módulos y funciones.

Tareas típicas en la codificación o construcción

- Los análisis de trazabilidad.
    1. El código fuente para diseñar la especificación y viceversa.
    2. El caso de prueba al código fuente y el diseño de la especificación.
- El código fuente y especificación del código fuente.
- El código fuente y el análisis de la interfaz.
- Procedimientos de prueba y generación de pruebas (modulo, integración, sistema y la aceptación).

3.4.6 La prueba por el diseñador del software

Las pruebas del software traen consigo productos seguidos de software bajo condiciones conocidas con entradas definidas y resultados documentados que pueden compararse a sus expectativas predefinidas. Este es un tiempo demandante, difícil y de actividades imperfectas.

Los planes de prueba y los casos de prueba deben ser creados en el proceso de desarrollo del software tan pronto como sea factible. Ellos deben identificar los horarios, ambientes y recursos (personal, herramientas, etc.), metodologías, casos (entradas, procedimientos, resultados esperados), y documentación. La magnitud del esfuerzo a través de la prueba de diseño puede enlazarse a la complejidad, a la gravedad, fiabilidad y la seguridad de los temas. (Ejemplo, funciones de requerimiento, o módulos que producen los resultados críticos para retarlos con pruebas intensas tolerancia en fallas).

Los planes de prueba del software deben identificar las tareas particulares conducidas en cada etapa del desarrollo e incluye la justificación para el nivel de esfuerzo representado por su criterio de realización correspondiente.

La prueba del software tienen limitaciones que deben reconocerse y ser consideradas cuando se están planeando las pruebas de un producto de software.

Excepto para los programas simples, el software no puede ser probado exhaustivamente. Generalmente no es factible probar un producto de software con todas sus entradas posibles, ni es posible probar todos los caminos posibles del procedimiento que pueden ocurrir durante la ejecución del programa. No hay un tipo de prueba o tipo de metodología que pueda asegurar que un producto de software ha sido probado completamente. La prueba de toda la funcionalidad del programa no significa que el programa ha sido probado. Toda la prueba del código del programa no significa que toda la funcionalidad esta presente en el programa. La prueba de toda la funcionalidad del programa y todo el código del programa no significa que el programa es 100% correcto. La prueba del software que no encuentra errores no debe ser interpretada como que los errores no existen en el producto de software, esto puede significar que la prueba es superficial.

Un elemento esencial de un caso de prueba es el resultado esperado. Esta es la llave que permite la evaluación objetiva del resultado actual. Esta información necesaria de prueba se obtiene de la especificación o definición predefinida. Un documento de especificación de software debe identificar: que, cuando, como, porque, etc; será logrado con un nivel de ingeniería (es decir mensurable u objetivamente verificable) de detalle en orden para confirmar a través de la prueba.

Un proceso de prueba del software debe ser basado en los principios que adoptan examinaciones efectivas de un producto de software. Una prueba aplicable del software incluye:

- El resultado de prueba esperado esta predefinido.
- Un buen caso de prueba tiene una probabilidad alta de exponer un error.
- Una prueba exitosa es una que encuentra errores.
- Hay una independencia del código.
- La pericia de la aplicación (usuario) y el software (programación) son empleadas.
- Los verificadores usan diferentes herramientas de los programadores.
- Examinar solo el caso usual es insuficiente.

- La documentación de la prueba permite reutilizar y su confirmación independiente del paso del resultado de la prueba durante una revisión subsiguiente.

Una vez que las tarea previas (por ejemplo: la inspección del código) han sido completadas exitosamente, las pruebas del software empiezan. Empieza con la unidad de comprobación nivelada y concluye con el sistema de comprobación nivelada. Un producto de prueba debe ser desafiado con los casos de prueba basados en su estructura interna y con los casos probados en su especificación externa. Estas pruebas deben proporcionar una minuciosa y rigurosa examinación de la confianza del producto del software con su funcionalidad, actuación y definiciones de interfaz y requisitos.

La prueba del código base es también conocido como prueba estructural o "caja blanca". Define los casos de prueba basados en el conocimiento obtenido del código fuente, la especificación del diseño detallado y otros documentos desarrollados. Este reto de casos de prueba de decisiones de control hechos por el programa y las estructura de datos del programa incluyen tablas de configuración. La prueba estructural puede identificar "al código muerto" que nunca se ejecuto cuando el programa se corrió. La prueba estructural es principalmente cumplida con la unidad de la comprobación nivelada, pero pueden ser extendidas a otros niveles de prueba del software.

El nivel de la prueba estructural puede ser evaluado usando un sistema métrico, que esta diseñado para mostrar que porcentaje de la estructura del software ha sido evaluada durante la prueba estructural. Este sistema métrico esta típicamente referido a cubrir y son una medida de integridad con respecto al criterio de selección de prueba. La cantidad de cubiertas estructurales deben ser acorde con el nivel de riesgo representado por el software. El uso de el termino "cobertura" usualmente significa 100% cubierto. Por ejemplo si un programa de prueba tiene logrado una declaración de cobertura significa que el 100% de las declaraciones en el software han sido ejecutadas por lo menos una vez. Los sistemas métricos de cobertura estructural incluyen:

- **Cobertura de declaraciones**: este criterio requiere suficientes casos de prueba para cada declaración del programa que se ejecutan por lo menos una vez; sin embargo su logro es insuficiente para proporcionar confianza en la conducta del producto de software.
- **Decisión de cobertura:** este criterio requiere suficientes casos de prueba para cada decisión de programa que son ejecutadas para que cada resultado posible ocurra por lo menos una vez. Esto esta considerado para ser un mínimo nivel de cobertura para los productos del software, la decisión de cobertura sola es insuficiente para una aplicación de integridad alta.
- **Condiciones de cobertura:** este criterio requiere suficientes casos de prueba para cada condición en una decisión de programa para tomar todos los posibles resultados por lo menos una vez. Difiere de la decisión de cobertura solo cuando las condiciones múltiples deben ser evaluadas para alcanzar una decisión.
- **Condiciones múltiples de cobertura:** este criterio requiere suficientes casos de prueba para ejercitar todas las combinaciones posibles en una decisión de programa.
- **Cobertura del bucle:** este criterio requiere suficientes casos de prueba para todos los bucles del programa que son ejecutados para cero, una, dos o muchas repeticiones, funcionamiento típico y las condiciones de terminación.
- **La cobertura del camino:** este criterio requiere suficientes casos de prueba para cada camino feasible, el camino de la base, etc., del comienzo al final del segmento definido del programa para que sea ejecutado por lo menos una vez. Debido al número muy grande de caminos posibles a través del programa de software la cobertura del camino generalmente no se logra. La cantidad de coberturas del camino es normalmente establecida basada en el riesgo o en lo crítico del software bajo la prueba.
- **La cobertura del flujo de datos:** este criterio requiere de suficientes casos de prueba para cada cobertura de flujo de datos para que se ejecute por lo menos una vez.

Las pruebas de definiciones basadas o especificaciones basadas también son conocidas como comprobación final o prueba de "caja negra". Esto identifica casos de prueba basados en la definición de que el producto de software se intenta hacer. Este caso de prueba desafía el uso intencional o la funcionalidad del programa y las interfaces internas y externas de los programas internos. La prueba funcional puede ser aplicada en todos los niveles de la prueba del software de la unidad a la prueba del sistema nivelado.

Los siguientes tipos de prueba de funcionalidad del software generalmente incrementan los niveles de esfuerzo:

- **Caso normal:** las pruebas con entradas usuales son necesarias. Sin embargo, las pruebas de un producto de software solo con entradas validas esperadas no prueban completamente al producto de software. Por si mismo el caso normal de prueba no puede proporcionar suficiente confianza en la fiabilidad del software.
- **Fuerza de producción:** escoge las pruebas de entrada para asegurar que la selección de la producción del software son generadas por la prueba.
- **Resistencia:** la prueba del software debe demostrar que un software se comporta correctamente cuando se le dieron entradas inválidas inesperadas. Los métodos para identificar un juego suficiente de tales casos incluye clases equivalentes de partición, valor de análisis e identificación especial del caso. Mientras que lo importante y lo necesario de estas técnicas no aseguran que han sido identificados todos los retos más apropiados para el software.
- **Combinación de entradas:** las pruebas funcionales de métodos identificados arriba enfatizan una individual entrada de prueba. La mayoría del software operan con múltiple entradas bajo sus condiciones de uso. La prueba minuciosa del software debe considerar las combinaciones de entradas de una unidad de software. Los errores supuestos pueden ser extendidos para identificar combinaciones de entradas, pero esta es una técnica ad hoc.

Los casos funcionales y estructurales de la prueba de software identifican técnicas que proporcionan entradas específicas para la prueba, en lugar de las entradas de prueba aleatoria. Una debilidad de estas técnicas es la dificultad dentro del enlace

estructural y funcional del criterio de prueba para la fiabilidad del software. El avance de los métodos de prueba de software, tales como la prueba estadística, pueden ser empleados para proporcionar garantías mejores de que un software es digno de confianza. La prueba estadística usa los datos de prueba generados al azar de las distribuciones definidas basadas sobre un perfil operacional (por ejemplo: el uso esperado, el uso arriesgado, o el uso mal intencionado del software). Grandes cantidades de datos de prueba son generados y pueden ser dirigidos para cubrir áreas particulares o empresariales, proporcionando una posibilidad aumentada de identificación de condiciones de operaciones individuales y múltiples que no fueron anticipados por los diseñadores del software ni por sus probadores. La prueba estadística también proporciona una alta cobertura estructural. Estas pruebas funcionales y estructurales son requisitos para la prueba estadística del software.
Otro aspecto de la prueba del software es la prueba en los cambios de este. Los cambios ocurren frecuentemente durante el desarrollo del software. Estos cambios son el resultado de:

1. Depuración de fallos que encuentran un error y es corregido.
2. Requisitos nuevos o cambiados.
3. Diseños modificados.

Una ves que el software ha sido aprobado, cualquier cambio al software debe tener su propio "mini ciclo de vida", incluyendo la prueba. La prueba de un cambio de software requiere esfuerzos adicionales. No solo debe demostrar que el cambio ha sido implementado correctamente, también debe demostrar que no hizo impactos adversos a otra partes de el software. El análisis de la regresión y prueba son empleadas para proporcionar confianza de que un cambio no ha creado problemas en ningún lugar de el software. El análisis de regresión es la determinación de el impacto de un cambio basado en la revisión de la documentación relevante (ejemplo: la especificación de requisitos de el software, la especificación de diseño de el software, código fuente, los planes de prueba etc.) para identificar las pruebas de regresión necesaria que se corren. La prueba de regresión es el reedirecionamiento de los casos de prueba que un programa ha ejecutado previamente y comparando el resultado actual con el resultado anterior para detectar

efectos imprevistos de un cambio en el software. El análisis de regresión y la prueba de regresión deben también ser empleadas cuando se usan los métodos de la integración para construir un software que asegure que la integración reciente de módulos no son impactos adversos en la operación de los módulos previamente integrados.

Para proporcionar un examinación exhaustiva y rigurosa del software, el desarrollo de pruebas es típicamente organizado dentro de niveles. Como un ejemplo tenemos: la prueba de un producto de software puede ser organizado dentro de una unidad, en la integración y en los niveles del sistema de la prueba.

1. La unidad de la comprobación nivelada se enfoca en la examinación temprana de la funcionalidad del subprograma y asegura que la funcionalidad no visible en el nivel del sistema es examinada por la prueba.
2. La integración de la prueba nivelada se enfoca en la transferencia de datos y control a través de una interfase externa e interna de un programa. Las interfases externas son aquellas con otro software (incluso el sistema operativo del software), el sistema de hardware y los usuarios peden describirse como enlace de las comunicaciones.
3. El sistema de nivel de prueba demuestra que toda la funcionalidad especificada existe y que el producto del software es fidedigno. Esta prueba verifica la construcción de la funcionalidad del programa y el desarrollo con respecto a los requisitos del software como exhibido sobre la plataforma de operación especificada. La prueba del sistema nivelado del software dirige los asuntos funcionales y los elementos siguientes del software que están relacionados con su usos intencionales:
    - Temas desarrollados (ejemplo: tiempos de respuesta, fiabilidad de sistemas).
    - Respuestas a las condiciones acentuadas (ejemplo: comportamiento sobre la carga máxima, uso continuo).
    - Operación de características de seguridad interna y externa.
    - Efectividad en la recuperación de procedimientos incluyendo la recuperación de desastres.
    - La utilidad.

- Compatibilidad con otros productos de software.
- Comportamiento en cada una de las configuraciones del hardware.
- Precisión de documentos.

Las medidas de control (ejemplo: un análisis de trazabilidad) debe ser usado para asegurar que la cobertura intencional sea lograda. El sistema de la prueba nivelada también exhibe el comportamiento del software en el ambiente de operación intencional. La situación de tal prueba depende de la habilidad del programador para producir el ambiente del blanco operacional. Dependiendo de las circunstancias, la simulación o prueba se pueden utilizar en la localidad del cliente. Los planes de prueba deben identificar las necesidades controladas para asegurar que la cobertura intencional se consiguió y que la documentación adecuada este dispuesta cuando la prueba del sistema nivelado se conduzca en sitios no controlados directamente por el diseñador del software.

Los procedimientos de prueba, los datos de prueba, y el resultado de pruebas deben ser documentadas de manera que permitan las decisiones objetivas de pasa no pasa. También deben ser convenientes para la revisión y la decisión objetiva para correr la prueba y deben ser convenientes para el uso de cualquier prueba de regresión posterior. Los errores detectados durante la prueba deben ser anotados, clasificados, revisados y resueltos antes de poner en circulación al software. Los datos de error del software que son reunidos y analizados durante el desarrollo del ciclo de vida pueden ser usados para determinar la conveniencia del software para hacer público la distribución comercial. La prueba de reportes debe cumplir con los requisitos de los planes de prueba correspondientes. Los productos de software que realizan las funciones útiles en los dispositivos médicos o su producción son a menudo complejos. Las herramientas de prueba de software son frecuentemente usadas para asegurar consistencia, minuciosidad y eficiencia en la prueba de tales productos de software y cumplir los requisitos de las actividades planeadas de la prueba. Estas herramientas pueden incluir soporte en la construcción del software para facilitar la prueba del módulo y la prueba posterior de la integración (ejemplo: controladores y resguardos), tan bien como las herramientas comerciales de prueba de software. Tales herramientas deben tener algo de calidad, no menos que el producto de software, estas herramientas se usan para el desarrollo. La apropiada

documentación proporciona evidencia de que la validación de esta herramienta de software para su uso intencional debe ser mantenida.

Tareas típicas – prueba para el diseñador de el software

- Planeación de pruebas.
- La identificación de la causa de la prueba estructural.
- La identificación de la causa de prueba funcional.
- Los análisis de trazabilidad – prueba.
- Ejecución del módulo de prueba.
- Ejecución de la prueba funcional.
- Ejecución de la prueba de sistemas.
- Ejecución de la prueba de aceptación.
- Evaluación de los resultados de prueba.
- Evaluación de el error /resolución.
- Reporte final de la prueba

3.4.7 La prueba del sitio del usuario

La prueba en el sitio del usuario es una parte esencial de la validación del software. La regulación del sistema de calidad requiere instalación e inspección de procedimientos, tan bueno como la prueba y la documentación de la inspección para demostrar la adecuada instalación. Igualmente el equipo de manufactura debe conocer los requisitos especificados y los sistemas automáticos deben validarse para su uso intencional. La terminología con respecto a la prueba del sitio del usuario puede ser confusa. Los términos tales como la prueba beta, validación del sitio, la prueba de aceptación de usuarios, verificación de la instalación y prueba de la instalación han sido usadas para describir la prueba del sitio del usuario. Esta prueba debe tomar lugar en el sitio del usuario con el software y el hardware actual que será parte de la configuración del sistema instalado. La prueba es cumplida a través del uso real o simulado del software que es probado dentro del contexto en que se intente usar.

La prueba del sitio del usuario debe seguir un plan escrito predefinido con un resumen formal de la prueba y un registro de aceptación formal. La evidencia

documentada de los procedimientos de prueba, la entrada de datos de prueba y los resultados de prueba deben retenerse.

Debe haber evidencia de que el software y el hardware están instalados y configurados como dicen las especificaciones. Las medidas deben asegurar que todos los componentes del sistema se ejercen durante la prueba y que las versiones de estos componentes están especificadas. El plan de prueba debe especificar la prueba a lo largo del rango de las condiciones de operación y debe especificar la continuación para un tiempo suficiente para permitir que el sistema encuentre un amplio espectro de condiciones y eventos en un esfuerzo para detectar cualquier fallo latente que no este aparente durante las actividades normales.

Algunas de las evaluaciones que han sido hechas antes por el diseñador del software en el sitio del diseñador deben repetirse en los sitios de uso actual. Este puede incluir pruebas para un alto volumen de datos, cargas pesadas o tensiones, seguridad, pruebas de fallos (descubrimiento, tolerancia y recuperación), mensajes de error, e implementación de los requisitos de seguridad. El diseñador puede ser capaz de suministrar al usuario algunos de los datos de prueba que son usados para este propósito.

Durante la prueba del sitio del usuario, deben mantenerse los archivos de la adecuada realización del sistema y cualquier falla en el sistema que se encuentre. La revisión del sistema compensa fallas detectadas durante esta prueba del sitio del usuario y debe seguir los mismos procedimientos y controles como para cualquier cambio en el software. Los diseñadores del software pueden o no estar involucrados en la prueba de sitio del usuario. Si los diseñadores están involucrados, ellos pueden llevar al sitio del usuario las últimas porciones de la prueba de sistema del nivel diseñado.

Tareas típicas de la prueba del sitio del usuario

- La ejecución de la prueba de aceptación.
- La evaluación de los resultados de la prueba.
- La evaluación de el error / resolución.
- El reporte final de la prueba.

3.4.8 El mantenimiento y los cambios en el software

El término "mantenimiento" aplicado al software no significa lo mismo que para el hardware. El mantenimiento operacional del hardware y el software son diferentes debido a que sus fallas en sus mecanismos son diferentes. El mantenimiento típico del hardware incluye el mantenimiento preventivo del software, reemplazo del componente y los cambios correctivos. El mantenimiento del software incluye: corrección, perfección y mantenimiento adaptado, pero no incluye acciones de mantenimiento preventivo o reemplazo del componente. Los cambios hechos para corregir errores y fallas en el software son el mantenimiento correctivo. Los cambios hechos al software para mejorar el rendimiento, mantenimiento u otros atributos del sistema del software son perfectamente mantenibles. Cuando los cambios son hechos al sistema de software, durante el desarrollo inicial o durante mantenimiento en uso del software, el análisis de regresión y la prueba deben ser conducidas para demostrar que las porciones del software no involucradas en los cambios no fueron impactadas adversamente. El esfuerzo necesario de la validación para cada cambio en el software está determinado por el tipo de cambio, los productos desarrollados afectados, y el impacto de esos productos sobre la operación del software. La documentación completa y cuidadosa de la estructura del diseño y la interrelación de varios módulos, interfases, etc.; puede limitar los esfuerzos necesarios de la validación cuando un cambio esta hecho. El nivel de esfuerzo necesitado para validar un cambio también depende del grado de documentación y de archivo de la validación original hecha al software, por ejemplo la prueba de documentación, los casos de prueba y los resultados de la verificación previa y los resultados de la prueba de validación y verificación necesitan ser archivadas si ellas están disponibles para el desempeño de la subsiguiente prueba de regresión. La falla para archivar esta información para un uso posterior puede incrementar significativamente el nivel de esfuerzo y elevar el costo de la revalidación del software después de haber realizado el cambio.

En suma para la verificación del software y las tasas de validación que son parte del proceso de desarrollo del software estandarizado, la tarea del mantenimiento adicional siguiente deberá ser dirigido por:

- **El plan de revisión de la validación del software:** para el software que fue previamente validado el plan existente de validación del software debe ser revisado para soportar la validación del software revisado.
- **Una mala evaluación:** el software en las organizaciones frecuentemente mantiene la documentación, tal como un reporte de problema de software que describe las anomalías descubiertas en el software y una acción especifica correctiva tomada para arreglar cada una de las anomalías. Sin embargo, a menudo los errores son repetidos por los diseñadores del software que no toman el siguiente paso para determinar la raíz que causa el problema y hace que el proceso y los cambios de procedimiento necesiten evitar la recurrencia del problema. Las anomalías del software deberán ser evaluadas en términos de su gravedad y sus efectos sobre un sistema de operación y seguridad, pero estos deben de ser también tratados como síntomas de deficiencia del proceso en el sistema de calidad. Una raíz causa el análisis de las anomalías que pueden identificar específicamente las deficiencias de un sistema de calidad. Cuando las tendencias de las anomalías son identificadas (ejemplo: la recurrencia de las anomalías similares en el software), las acciones apropiadas para corregir y prevenir deberán estar implementadas y documentadas para evitar recurrencias posteriores en problemas similares de calidad.
- **La identificación del problema y el camino de la solución.** Todos los problemas descubiertos durante el mantenimiento del software deberán ser documentados. La resolución de cada problema deberá ser rastreada para asegurar que esta ha sido arreglada, para una referencia, y para posibles tendencias.
- **El cambio propuesto y evaluado:** todas las modificaciones propuestas, las mejoras, o adiciones deberán ser verificadas para determinar el efecto que cada cambio tendría en el sistema. Esta información debe determinar el alcance para las tareas de verificación y/o validación que necesitan ser iteradas.
- **Tarea de iteración:** para los cambios aprobados en el software, todas las tareas de verificación y validación necesarias deberán ser desempeñadas para asegurar que los cambios planeados son implementados correctamente,

toda la documentación es completa y se actualiza, y no aceptara cambios que hayan ocurrido durante el desempeño del software.

- **La actualización de la documentación:** la documentación deberá estar cuidadosamente revisada para determinar que documentos han sido impactados por un cambio. Todos los documentos aprobados (como son: las especificaciones, evaluación de procedimientos, manuales de usuario, etc) que han sido afectados deberán actualizarse de acuerdo con la configuración de los procedimientos guías. Las especificaciones deberán ser actualizadas antes de que cualquier mantenimiento o cambio en el software sea realizado.

3.4.9 Requisitos definidos del usuario

Una clave importante para la validación del software es un usuario que se ha documentado para requisitar la especificación, la cual define:

- El propósito de uso del software o del equipo automatizado.

El diseñador del dispositivo (ususario) necesita definir el ambiente operacional esperado, incluyendo cualquier configuración requerida del software y hardware, las versiones del software, utilidades, etc. El usuario necesita también:

- Documentar los requisitos para el desempeño del sistema, la calidad, el error manipulable, el encendido de la computadora, el apagado de la computadora, la seguridad, etc.
- Deberá identificar cualquier función o peculiaridad de seguridad, tales como: censores, alarmas, llaves internas, pasos lógicos de procesamiento, o secuencias de comando.
- Definir el criterio objetivo para determinar el desempeño que será el aceptado.

La validación deberá ser conducida en acuerdo a un protocolo documentado, y los resultados de validación deberán estar también documentados. En los casos de prueba deberán estar documentados para que en el ejercicio del sistema se autopruebe mediante su desempeño en contra de su criterio predeterminado, especialmente para los parámetros más críticos. Los casos de prueba deben

direccionar el error y las condiciones de alarma, encendido, apagado y todas las funciones aplicables del usuario y operadores de control, operadores potenciales de error, rangos máximos y mínimos de valores permitidos, y las condiciones aplicables que resaltan para el uso propio del equipo. Estos casos de prueba deberán ser ejecutados y los resultados deberán ser registrados y evaluados para determinar si los resultados apoyan la conclusión de que el software está validado para su propio uso.

El diseñador del dispositivo puede conducir una validación usando su propio personal o puede depender de una tercera parte el cual es el distribuidor de equipo y software o de una consultoría. En cualquiera de los casos, el diseñador del dispositivo tendrá la última responsabilidad para asegura que, la producción y el software en el sistema de calidad:

- Estarán validados en acuerdo a los procedimientos preescritos por el uso propio del particular.
- Se desempeñaran como una aplicación elegida previamente intencionada.

El diseñador del dispositivo deberá tener incluida la documentación:

- Los requisitos definidos del usuario.
- Los protocolos de validación usados.
- El criterio de aceptación.
- Los casos de prueba y resultado.
- Un registro de validación.

Que objetivamente confirme que el software está validado para su propio uso.

3.5 ¿Cuánta evidencia de validación se necesita?

El esfuerzo para un nivel de validación deberá ser conmensurado con el riesgo representado por una operación automatizada, además de otros factores de riesgo tales como, la complejidad de los procesos del software y la degradación para los cuales el creador del dispositivo esta dependiendo de los procesos automatizados para producir un seguro y efectivo dispositivo. Los requisitos documentados y el

análisis de riesgo de los procesos automatizados ayudan a definir el alcance de la evidencia requerida para mostrar que el software es validado para su propio uso. Por ejemplo en una máquina automatizada de tejidos se puede requerir muy poca prueba si el diseñador puede mostrar que la salida de la operación está subsecuentemente bien verificada antes de que sea puesto en venta. De otra forma, una prueba extensiva puede estar requisitada por:

- Un registro amplio en una planta electrónica y un sistema de firma digital.
- Un controlador automatizado para un ciclo de esterilización; o
- Un equipo de prueba automatizado que será usado para la inspección y aceptación del acabado de tarjetas de circuitos y de su periodo de vida por un dispositivo que alargara su funcionamiento.

Numerosas aplicaciones de software pueden ser usadas como una parte del sistema de calidad (ejemplo: una hoja de calculo, un paquete estadístico usado para un sistema de calidad que calcula, un paquete gráfico usado para medir el análisis, o una base de datos comercial usada para guardar dispositivos históricos en registros o para los registros de las quejas gerenciales). El alcance de evidencia de validación requerida por el software depende del creador del dispositivo y de su documentación para su buen uso. Por ejemplo un diseñador de dispositivos elige no usar todas las posibilidades de compra venta del software, solo necesita validar aquellas funciones que serán usadas y para las cuales el diseñador del dispositivo esta echando mano de los resultados del software como una parte de producción o del sistema de calidad. Sin embargo hay aplicaciones de alto riesgo que no deberán ser ejecutadas en el mismo ambiente de operación con las funciones de software no validadas, inclusive si aquellas funciones de software no son usadas. Las técnicas de eliminación de riesgo en las cuales se particiona la memoria o se aproxima a otros recursos de protección podrían estar consideradas cuando las aplicaciones de alto riesgo y las aplicaciones de bajo riesgo serán usadas en el mismo desarrollo operacional. Cuando el software es actualizado o algunos cambios son realizados, el diseñador del dispositivo deberá considerar que esos cambios han afectado las porciones usadas del software, y deberá reconfirmar la validación de esas porciones de software que son usadas.

3.6 Resumen

Con esto podemos entender que la regulación de un sistema de calidad requiere que cuando las computadoras, software y toda la información automatizada de un sistema en procesamiento son usados como parte de la producción o calificación del sistema, estos se validaran para su uso intencionado en acuerdo con un protocolo establecido.

Las herramientas del software son frecuentemente usadas para el diseño, construcción y prueba del software que va a automatizarse en algún dispositivo. Muchas otras aplicaciones comerciales de software, tales como: los procesadores de palabra, hojas de cálculo, bases de datos y el software de diseño de diagrama de flujo son usados para implementar la calidad en el sistema. Todas estas aplicaciones están sujetas a los requisitos de la validación del software, para que la validación aproximada sea usada para cada aplicación y que pueda variar ampliamente.

La validación deberá ser una llave de consideración en la decisión de cómo y por quien este software será desarrollado y por quien será comprado. El diseñador del software define este modelo de ciclo de vida. La validación esta típicamente apoyado por:

- Las verificaciones de las salidas de cada etapa del desarrollo del ciclo de vida del software.
- El apropiado chequeo de la operación del acabado del software en la manufactura del dispositivo y de su intento para ser desarrollado.

CAPITULO 4

CASO PRÁCTICO

Para demostrar el uso de una validación de forma práctica se utilizará el Sistema de Eventos Especiales (SEE), de la Unidad de Extensión Universitaria de la Universidad Continental.

El objetivo del área es contribuir a la formación integral del estudiante a través del fortalecimiento de la extensión y difusión de la cultura interna además de ampliar el servicio que se brinda a la comunidad aledaña, es el programa de extensión y difusión de la cultura, dentro de las actividades artísticas, socioculturales, deportivas, recreativas, así como la publicación de trabajos académicos y de investigación, y el apoyo mediante las impresiones de diversa índole, además de las actividades de vinculación académica con distintas instituciones educativas.

La Unidad de Extensión Universitaria, para realizar todas sus actividades, se divide en diferentes departamentos o áreas, como a continuación se muestran:

- Departamento de actividades culturales.
- Departamento de difusión.
- Departamento de intercambio académico y vinculación.
- Departamento de publicaciones.
- Coordinación de actividades deportivas y recreativas.
- Coordinación de servicios a la comunidad.
- Área de impresiones.
- Área de bolsa de trabajo.

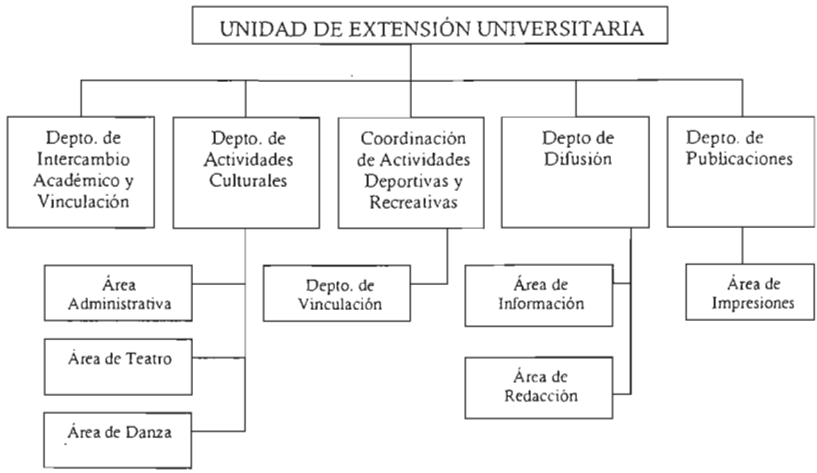


Figura 4.0

Estas ocho áreas son las principales cabeceras de la Unidad de Extensión Universitaria, todas se relacionan entre sí y además se generan otras sub-áreas, no menos importantes sin embargo no realizan tantas actividades como estas ocho principales. A continuación se mencionan las actividades más importantes que realizan las diferentes áreas.

El Departamento de actividades culturales es encargado principalmente de eventos de:

- Teatro
- Danza
- Cine
- Conciertos
- Proyección de películas
- Conferencias
- Talleres
- Recitales
- Entre otros

La mayoría de estos eventos o actividades son realizadas en su auditorio, cada una de éstas se realizan por temporadas, las cuales se dan al menos dos por semestre, y muchas otras son eventuales o en ocasiones especiales que elige la escuela, con todo esto, no existe mes en que al menos no se realiza una actividad en el auditorio, además de los diferentes foros que pueden ocupar las demás actividades dentro de la Institución

El Departamento de difusión, realiza los siguientes eventos:

- Coordinar la difusión de las actividades de todas las áreas.
- Enviar reportes a impresiones y publicación de los eventos.
- Difusión oportuna uno de los eventos internos o externos.

Esta departamento, tiene la cualidad de estar directamente relacionado con el departamento de publicaciones ya que la principal difusión dentro del Campus es escrita o gráfica, sin embargo este departamento también atiende diferentes tareas de la Institución en general y no solo de la Unidad de Extensión Universitaria.

El Departamento de intercambio académico y vinculación se encarga de:

- Mantener actualizados los convenios y acuerdos de la Institución con otras.
- Realizar los intercambios académicos
- Realizar exposiciones en escuelas aledañas para difundir la institución.
- Forman los representantes de la institución en las ferias universitarias.

Este departamento se encarga de renovar constantemente los diferentes convenios académicos de la Institución, así como de difundir los acuerdos escolares con diferentes planteles educativos para el beneficio de toda la población estudiantil.

El Departamento de publicaciones, se encarga de la publicación de:

- Propaganda

- Trípticos
- Volantes
- Libros
- Folletos
- Otros

Publica documentos o información de toda la institución, no solo de la Unidad de Extensión Universitaria, además de publicar mensualmente la **Gaceta** del Campus, que es el diario informativo de más prestigio dentro de la Institución, está área siempre trabaja en conjunto con el departamento de difusión.

Por otra parte el Departamento de Coordinación de actividades deportivas y recreativas, se encarga básicamente de:

- Torneos de fútbol soccer
- Torneo de fútbol rápido
- Básquetbol
- Voleibol
- Ajedrez
- Atletismo
- Físico culturismo
- Gimnasia olímpica
- Lucha olímpica
- Montañismo
- Softbol
- Tae kwon do
- Béisbol

Entre otros deportes, en total maneja más de cincuenta deportes, y se realizan por medio de competiciones, ya sean internas o externas, debido a que la Institución maneja una serie de torneos con instituciones afines ya sea dentro del Campus donde se cuentan con diferentes áreas deportivas o en otras instalaciones

adecuadas. Esto conlleva a que dicho departamento se encuentre siempre ocupado en uno u otro deporte.

El Departamento de Coordinación de servicios a la comunidad, se apoya con el departamento de intercambio académico, para apoyar en diferentes eventos a realizar con la comunidad aledaña.

El Área de impresiones, imprime todo aquello necesario para la institución, auxiliada por el área de publicaciones, como los siguientes documentos:

- Papelería de inscripción
- Folletos
- Gaceta
- Libros
- Propaganda
- Carteles
- Entre otros

Dicha área se coordina con publicaciones para obtener mejores resultados, y aunque no solo atiende a la Unidad de Extensión Universitaria, la mayor parte de trabajo proviene de ella.

El Área de bolsa de trabajo se encarga de tener una relación de diferentes plazas de empleo para los alumnos de la institución, así como los diferentes acuerdos con empresas para la generación de nuevas vacantes para los alumnos.

A continuación se muestra un cuadro que presenta las principales actividades de todas las áreas.

Principales Actividades de las Áreas

Áreas	Culturales	Difusión	Intercambio	Publicaciones	Deportivas	Comunidad	Impresiones	Bolsa
Actividades	Teatro Danza Cine Conciertos Proyección de peliculas	Coordinar difusión de actividades Difusión oportuna de eventos internos o externos.	Actualizados los convenios y acuerdos Intercambios académicos Exposiciones en escuelas aledañas	Propaganda Tripticos Volantes Libros Folletos Gaceta	Torneos de fútbol soccer Básquetbol Voleibol Ajedrez Atletismo	Servicios Comunitarios a los lugares aledaños a la institución	Papeleria de inscripción Folletos Gaceta Libros Propaganda	Bolsa de trabajo ofrecida por instituciones a los alumnos de la institución

	Conferencias Talleres Recitales		Forman representantes en las ferias universitarias.		Fisico culturismo Gimnasia olimpica Lucha olimpica Montañismo Softbol		Carteles	

Tabla4.0

Para que estas ocho áreas sean capaces de realizan tantas actividades controladamente es necesario que laboren alrededor de un proceso, cada una con procesos y planteamientos diferentes pero básicamente son similares en las siguientes características:

- Planificar un evento.
- Ejecutar dicho evento.
- Redactar un reporte de lo sucedido.
- Enviarlo a la dirección de Extensión Universitaria.
- Se adjunta con un oficio para mayor control.
- Se recopila y guarda físicamente en un archivero toda la información.
- Se almacena en la oficina de la dirección para su uso posterior.

Cada una de las áreas maneja diferente información, ya sean torneos, publicaciones o eventos y por lo tanto manejan diferentes procesos para el mejor uso de su información, sin embargo todos tienen un proceso base, el cual se presenta en el siguiente diagrama.

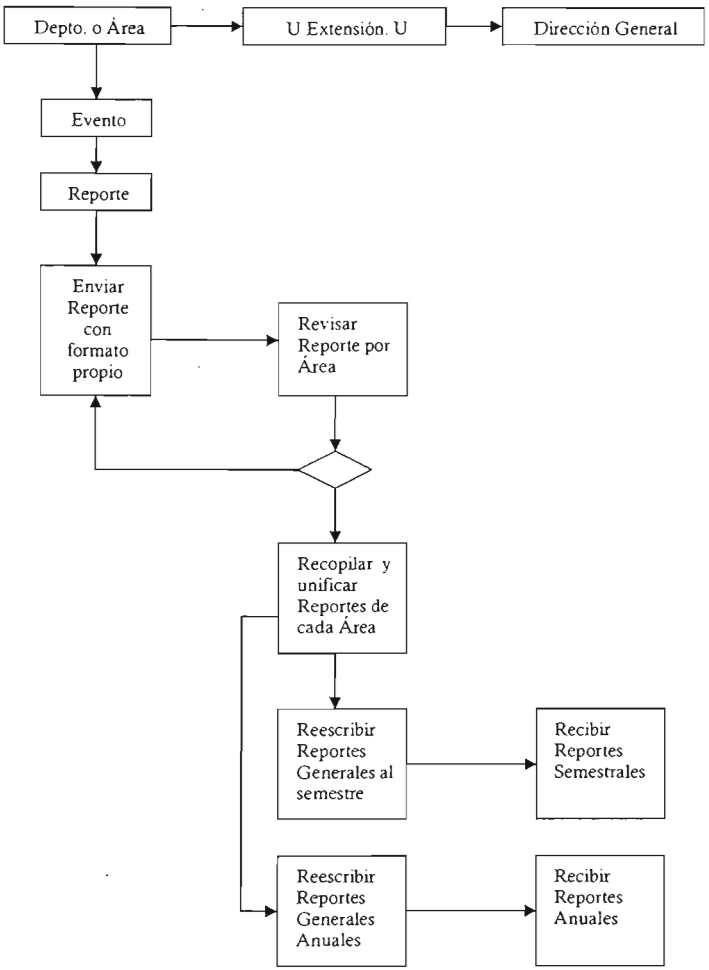


Figura 4.1

El sistema SEE tiene módulos de captura en cada una de las unidades y departamentos, así como formatos establecidos para los reportes de cada una, los cuales, en un ambiente web, esto quiere decir que la afectación de la información, **es en línea en una base de datos.**

Las ventajas que ofrece el sistema SEE son:

- Recibir toda la información mucho mas detallada de los eventos: dado que es en la computadora donde se puede detallar el punto que uno desee.
- Unificación de los procesos.
- Manejo de formatos: todos por medio de eventos y pantallas fáciles para cada área.
- Control por número de oficios.
- Mayor control de dicha información debido a que no hay pérdida y todo queda guardado en la base de datos.
- Sistema de bitácora para verificar quien entra al sistema y en que momento.
- Control de los movimientos realizados al sistema para que no exista un mal uso del mismo.
- Ahorro de tiempo al no tener que reescribir toda la información.
- Evita duplicidad de trabajo, no se repite la descripción de un mismo reporte.
- Optimización de los reportes siendo más rápidos y sencillos.
- Consultas e historial ya que el sistema permite obtener datos de cualquier evento almacenado.
- Reportes finales en cualquier periodo de tiempo.

El sistema esta basado en WEB, todas las áreas usuarias tienen conexión a Internet y se procesan en un servidor central que tiene respaldo de información.
Para tener un buen proceso y como resultado una base de datos bien establecida en el sistema WEB se tiene que manejar un modelo de validación para la programación de este sistema, para demostrar la calidad del mismo, este modelo debe contar con reportes establecidos como se muestra a continuación.

Los reportes de validación varían dependiendo del modelo de validación y el modelo de ciclo de vida del software que se utiliza. El formato que se utilizara para

demostrar la validación dependerá de la empresa, del ingeniero o del formato establecido por un grupo de validación como se vio en el capitulo 3.
A continuación se presentara un modelo de validación con un formato siguiendo el modelo de ciclo de vida del software que se vio en el capitulo 3, y se describirán los puntos más importantes para el resultado de la validación.

En la primera sección de formatos están los objetivos y el alcance del software, es decir una descripción general del producto de software.

Objetivos y Alcance de la Validación

Esta sección describe el software en términos generales. Incluye objetivos y alcances de la aplicación o software, y si son necesarios los requisitos totales que se necesitara, tales como estándares y regulaciones.

Objetivo y el alcance de la validación	
Descripción general	Aquí se hace una descripción general de lo que puede realizar el software.
Alcance de la aplicación	Aquí son los alcances o parámetros que tiene el software, es decir, de donde a donde puede abarcar la potencia del software.
Información del producto	Lo que hace el software.
Requisitos totales	Requisitos de cómo, cuando y donde se maneja la validación, así como los documentos a usar y su resguardo.

Todas las personas involucradas en el proceso de validación, y que estén autorizadas para firmar partes del reporte, deben de estar listadas en este punto

Responsables del diseño	Nombre	Iniciales
Propietario del sistema		
Administrador del sistema		
Administrador de la aplicación		
Usuario del sistema		
Responsable de calidad		
Equipo que desarrolla la validación		

Equipo que revisa		
Equipo que prueba		

El tipo de software que se valida se menciona para determinar la extensión de la validación y su prueba (se escoge un tipo de software)

Tipo de software	
Software adquirido	*Software de desarrollo propio*
Paquete de software que se configura Software comercial Herramienta para asistir en el desarrollo del software Software de desarrollo Código fuente disponible Comentarios:	Programa compilable ejecutable (ejemplo, C/C++, VB) Hoja de calculo Herramienta que asiste en el desarrollo y prueba Software subcontratado para validación Comentarios:

Una vez que se describe el producto del software en forma general; es decir, que tipo de software se esta utilizando, las personas que están involucradas en el proceso del software y de las especificaciones del software, entonces se puede pasar a la siguiente sección.

La sección siguiente es la de las actividades del ciclo de vida del software, según sea el caso de que tipo de ciclo de vida de software se utilizo. En este caso se desarrollaran los formatos siguiendo el ciclo de vida de software mencionado en el capitulo 3 así como sus respectivas actividades.

Como ya se menciono con anterioridad, el modelo de ciclo de vida del software puede variar dependiendo de cual sea el apropiado para el producto o para la organización al realizar la validación.

A continuación se mostraran los formatos con las actividades del modelo de ciclo de vida del software, y se describirán los puntos más importantes en cada actividad del modelo. (*Es importante mencionar que no se esta llevando una validación de un software en específico, solo se esta mostrando un escenario general de una*

validación, es decir, se están proponiendo los diferentes tipos de formatos que se pueden utilizar en una validación de software)

En esta sección están los formatos o tablas que especifican la información que es primordial y relevante para la validación, es decir aquí se encuentran las actividades de todo el ciclo de vida del software. Los formatos son llenados con información acerca de las tareas que son desarrolladas, métodos que se usan, criterios de aceptación, documentación requerida y cualquier información que sea relevante para el proceso de validación.

En las tablas se muestran todas las actividades, y se especificaran las más importantes.

ACTIVIDADES DEL CICLO DE VIDA DEL SOFTWARE

Planeación de la calidad

En esta primera actividad se revisan cada una de las actividades que requiere el plan de calidad de acuerdo a la guía de ciclo de vida del software, este plan se enfoca a las actividades que tiene el software en el proceso general de desarrollo, es decir en su ciclo de vida. Revisa cada tarea de acuerdo al ciclo de vida, para poder demostrar que se siguen las pausas y metodología de desarrollo del software. También revisa los datos de entrada y salida de cada tarea o actividad.

PLANEACION DE LA CALIDAD				
Descripción	Criterio	Resultado	Observaciones	Dia de Inicio
Riesgos en la dirección del plan	Los puntos críticos en la formación , dirección y administración del plan de calidad	Si ☐ No ☐		
Configuración del plan de calidad	Aquí se demuestra en que consiste su plan de calidad del software	Si ☐ No ☐		
Plan de aseguramiento de calidad del software	Como es que se determina que el software es de calidad; es decir, que metodologías de calidad o normas etc, se utilizaron para el aseguramiento de la	Si ☐ No ☐		

	calidad			
Plan de verificación y validación del software	Aquí se muestra el plan de validación que se llevara para poder tener la validación	Si ☐ No ☐		
Diseño formal de la revisión de requisitos	Como es que se revisaran los requisitos de debe tener el software	Si ☐ No ☐		
Otras técnicas de revisión de requisitos		Si ☐ No ☐		
Reporte de problemas y resolución de procedimientos		Si ☐ No ☐		
Otras actividades de soporte		Si ☐ No ☐		

En esta actividad lo que se intenta conocer y debe incluirse en la actividad es:

- Tareas para cada actividad del ciclo de vida del software.
- Factores de calidad.
- Métodos y procedimientos para cada tarea.
- Criterio de aceptación de tareas.
- Aportaciones de cada tarea.
- Rendimientos de cada tarea.
- Roles y responsabilidad de cada tarea.
- Documentación de las necesidades de los usuarios.

Requisitos

Este formato que indica la actividad en el proceso de validación, se enfoca en lo que el sistema nos puede dar como usuarios, en las funciones que realiza el software, así como los requisitos que se establecieron principalmente para su uso intencionado. Se enfoca en todo lo relacionado entre el usuario y el software desarrollado, es decir la interacción que debe haber entre el software y el usuario para un buen proceso y buenos resultados al utilizar el software ya desarrollado.

REQUISITOS				
Descripción	Criterio	Resultado	Observaciones	
Análisis de los riesgos	Los riesgos o limitaciones que en determinado caso pueda tener el software	Si ☐ No ☐		
Análisis de trazabilidad		Si ☐ No ☐		
Requisitos del software para los requisitos del sistema y viceversa		Si ☐ No ☐		
Requisitos del software		Si ☐ No ☐		
Descripción de las características de los usuarios		Si ☐ No ☐		
Listado de las características de la memoria primaria y secundaria		Si ☐ No ☐		

Evaluación de los requisitos del software		Si ☐ No ☐		
Análisis de los requisitos de la interfase del usuario con el software		Si ☐ No ☐		
Generación del plan de prueba del sistema		Si ☐ No ☐		

En esta actividad se requiere lo siguiente:

- Contribuciones del software.
- Rendimiento del software.
- Funciones que el software desarrollará.
- Requisitos del software.
- Definición de todo lo exterior e interfaz del usuario.
- Como el usuario interactúa con el sistema.
- Como se debe tratar el error en el software.
- Como son los tiempos de respuesta.
- Ambiente de funcionamiento para el software.
- Rangos, límites y valores que acepta el software.
- Especificaciones y características del software.

Diseño

El diseño y la implementación de procesos son fundamentales, ya que aquí es donde se demuestra el tipo de procesos que se utilizan para el desarrollo del software, es la representación física y lógica del software, esto es demostrar de que el software muestre lo que hace y como lo hace.

DISEÑO				
Descripción	Criterio	Resultado	Observaciones	
Actualización del análisis de riesgo del software	Limitaciones que se tiene en la codificación para poder determinar posibles problemas al programar	Si ☐ No ☐		
Análisis de trazabilidad (Especificación de diseño para los requisitos del software		Si ☐ No ☐		
Evaluación del diseño del software	Prueba de que pasos consiste el diseño del software (metodologías, herramientas, etc.)	Si ☐ No ☐		
Análisis del diseño de comunicación entre el software y el ambiente	Con que interactúa el software. (usuarios, hardware ambiente de trabajo, y todo lo que esta ligado al software)	Si ☐ No ☐		
Generación del plan de prueba del modulo		Si ☐ No ☐		
Generación del plan de la prueba de integración del software	Como se integra el software al ambiente para el que fue diseñado	Si ☐ No ☐		
Generación de la prueba de diseño		Si ☐ No ☐		

En esta actividad se necesita demostrar lo siguiente:

- Especificación de los requisitos del software.
- El análisis de riesgo del software.
- Procedimientos de desarrollo y guías del código.
- Documentación del sistema (relación del hardware con el software).
- El hardware que se usa.
- Parámetros de medición o grabación.
- Estructura lógica y pasos de procedimientos.
- Descripción de estructura de datos y diagramas de flujo.
- Definición de variables.
- Error, alarma y mensajes de prevención.
- Software de apoyo.

Construcción y codificación

En este actividad se revisan los procedimientos que se llevaron a cabo para la realización o desarrollo del código fuente del programa o software, se revisa que tipo de lenguaje de programación se esta utilizando. El código debe ser evaluado para verificar que cumple con la guía de codificación especificada.

CONSTRUCCION Y CODIFICACION				
Descripción		Criterio	Resultado	Observaciones
Los análisis de trazabilidad	• El código fuente para diseñar la especificación y viceversa • El caso de prueba al código fuente y el diseño de la especificación	Se prueba y verifica cada función del código fuente	Si ☐ No ☐ Si ☐ No ☐	
El código fuente y su documentación		Todo lo referente al código fuente, así como su posible interpretación de cada línea de este	Si ☐ No ☐	
El análisis de la interfaz del código fuente			Si ☐ No ☐	
Procedimientos de prueba y generación de pruebas		Aquí se demuestran las pruebas que se hacen al software para lo que fue diseñado (integración, sistema y aceptación entre otras)	Si ☐ No ☐	

Lo que se quiere demostrar en esta actividad es lo siguiente:

- Cada elemento de la especificación del diseño del software debe ser implementado en el código fuente.
- Los módulos y funciones implementados en el código deben ser localizados en la especificación de diseño del software.
- Localizar los módulos y funciones del código fuente.

Prueba por el diseñador del software

Aquí las pruebas del diseñador son importantes ya que verifica que se realizo como esta establecido el desarrollo del software, un buen uso de caso de prueba puede resaltar algún error que exista en el desarrollo general del software, así que es importante realizar la prueba con responsabilidad. Las pruebas dan una minuciosa examinación de la confianza del software.

PRUEBA POR EL DISEÑADOR DEL SOFTWARE				
Descripción	Criterio	Resultado	Observaciones	
Planeación de pruebas. (objetivos)	Que tipo de pruebas se hacen para retar o probar al software	Si ☐ No ☐		
Identificación de la prueba estructural	Se muestra la estructura general del software	Si ☐ No ☐		
Identificación de la prueba funcional	La funcionalidad que tiene el software, es decir lo que desarrolla o hace en el sistema	Si ☐ No ☐		
Análisis de trazabilidad		Si ☐ No ☐		
Ejecución del modulo de prueba	Pruebas al software para lo que esta diseñado	Si ☐ No ☐		
Ejecución de la prueba de integración		Si ☐ No ☐		
Ejecución de la prueba funcional	Se determina la funcionalidad del software	Si ☐ No ☐		

Ejecución de la prueba de sistemas		Si ☐ No ☐		
Ejecución de la prueba de aceptación		Si ☐ No ☐		
Evaluación de los resultados de prueba		Si ☐ No ☐		
Evaluación del error y solución		Si ☐ No ☐		
Reporte final de la prueba		Si ☐ No ☐		

Aquí lo que se desea demostrar es lo siguiente:

- El ambiente en donde se desarrolló el software.
- Las fuentes, es decir quienes y con que herramientas desarrollaron el software.
- Metodologías.
- Aportaciones y producción del software.
- Procedimientos.
- Resultados.
- Documentación.

Prueba del sitio del usuario

Es aquí donde se califica la adecuada instalación del producto de hardware y software. Ya que depende de un buen sitio de trabajo para que el usuario desarrolle sus actividades adecuadamente. Aquí se prueba y reta al software con las actividades que tiene que realizar para demostrar que este un lugar adecuado para su desempeño.

PRUEBA DEL SITIO DEL USUARIO				
Descripción	Criterio	Resultado	Observaciones	
La ejecución de la prueba de aceptación	Se genera una prueba total del software con el hardware y todo su ambiente para demostrar las cualidades del software	Si ☐ No ☐		
La evaluación de los resultados de la prueba de aceptación de los usuarios		Si ☐ No ☐		
Evaluación y resulución de errores		Si ☐ No ☐		
Reporte final de la prueba		Si ☐ No ☐		

En esta actividad se demostrara:

- Prueba del ambiente en donde se trabajara, es decir probar el sitio de usuario con el hardware y el software actual que serán parte de la instalación en el sistema.

Mantenimiento y cambios de software

Esta fase de la validación, el software esta en uso y sujeto a los requisitos de servicio, mantenimiento, desarrollo y soporte. Es aquí donde se revisan los cambios y actualizaciones que se le pueden hacer al software.

MANTENIMIENTO Y CAMBIOS DE SOFTWARE				
Descripción	Criterio	Resultado	Observaciones	
Problema / Solución		Si ☐ No ☐		
Mantenimiento funcional		Si ☐ No ☐		
Mejora del desarrollo		Si ☐ No ☐		

En esta fase el software esta en uso y sujeto a los requisitos de servicio, mejoras, mantenimiento, desarrollo y soporte.
En esta fase es donde todas las actividades del desarrollo residen y donde se toman las decisiones acerca de los cambios actualizaciones y revalidaciones son echas.

CONCLUSIONES	
	Nombre
Responsable de la validación	
Comentarios	
Fecha	

REALIZADO_____ DICTAMEN_____

CONCLUSIONES

Hoy en día, la calidad en cualquier producto o servicio representa un signo muy importante en el desempeño, crecimiento y mantenimiento de cualquier empresa. Es por esto que las empresas nacionales como internacionales se han preocupado por implementar sistemas de calidad tanto en sus productos como en la dirección y se han preocupado por tratar de convencer a todo el personal de que todas las cosas dentro de la empresa se tienen que realizar con calidad.

Existen ya cientos de normas nacionales e internacionales para infinidad de productos y servicios algunas son obligatorias y otras no. Las normas internacionales entre ellas la ISO son de gran importancia para el desarrollo de cualquier empresa ya que estas promueven la estandarización y actividades relacionadas con la conformidad de estatutos finalizando así en un proceso total de calidad en cualquier producto y servicio. Además es esta la norma que abre muchas puertas de ventas y aceptación en todo el mundo, ya que está abalada por la mayoría de los países del mundo. De este tipo de normas aseguramiento o sistemas de calidad se desprenden gran cantidad de diferentes normas y modelos, entre estos tipos de sistemas de calidad están las de Aseguramiento de Calidad en el Software.

Estas se realizan debido a la necesidad que hay para que el software desarrollado se produzca con la calidad necesaria, ya que en nuestros días, la computadora y sus componentes principales como son el hardware y el software son de vital importancia en todo el entorno y el propio desarrollo de cualquier empresa.

Del software dependen gran cantidad de actividades de las empresas, como los sistemas de información, bases de datos, programas especiales, etc. En las empresas que se dedican a producir y desarrollar medicamentos el software es de suma importancia ya que los aparatos médicos son manipulados por software desarrollados exclusivamente para su uso intencional y de estos depende en gran parte el producto final y su calidad de este, en si todo aparato o dispositivo médico requiere ya ahora de un software específico. Así pues se tiene que la calidad en el software es muy importante y primordial para el buen desempeño que este pueda ofrecer, y para que esto sea posible existen metodologías de calidad en el software. Estas metodologías y normas existentes tienen como finalidad la calidad en del

desarrollo del software siguiendo una variedad de etapas y pasos para el desarrollo general de este.

Los modelos de calidad para el desarrollo del software coinciden en una o varias etapas y pueden tener o no pasos similares, pero todas coinciden en la **validación,** ya que es la que verifica, califica y prueba todo el desarrollo de proceso del software, desde la planeación hasta la codificación y pruebas, además de que cuenta con muchas ventajas que se encuentran en su desarrollo.

La metodología de la validación no es fácil ni ligera ya que se deben de cubrir una variedad de puntos y requiere de un seguimiento correcto, pero una vez que se realiza la validación como se debe hacer esta por su propio mérito puede asegurar la calidad en el producto en este caso la del software.

BIBLIOGRAFIA

Administración de los Sistemas de Información (tercera edición)
Organización y Tecnología 1996
Kenneth C. Laudon
Jane P. Laudon
Editorial Pearson Education

Traducción
Ing. Jorge Rodríguez Rodríguez
Facultad de ingeniería Universidad Nacional Autónoma de México

Revisión Técnica
Lic. en Sistemas Computarizados e informática universidad Iberoamericana

Auditoria Informática Un enfoque práctico (Segunda edición) 2001
Mario G. Piattini
Emilio del Peso
Editorial Alfaomega

"Apuntes" "Validación de Métodos Estadísticos" 2003
QFB. Marco Antonio Hernández Vargas

C++ A su Alcance Un enfoque orientado a objetos 1994
Luis Joyanes Aguilar
Editorial McGrawHill

Desarrollo de una Cultura de Calidad (Segunda edición) 2001
Humberto Cantú Delgado
Instituto Tecnológico de Estudios Superiores de Monterrey Campus Monterrey
Revisor Técnico: Jesús Cantú Rodríguez
Editorial MCGrawHill

El lenguaje Unificado de Modelado" 2002
G. Booch, J. Rumbagh y I. Jacobson
Addison Wesley

Modelos de Calidad para el Desarrollo del Software 2004
Alma Alejandra Mejia Marcial, Judith Eduwiges Penagos Trejo

Norma Mexicana IMNC ISO 9000-2000
Sistemas de gestión de la calidad
Emitida por el Instituto Mexicano de Normalización y Certificación A.C
Publicada 02 de Enero 2001

Norma ISO 9004-2000
Sistemas de gestión de la calidad
Emitida por el Instituto Mexicano de Normalización y Certificación A.C
Publicada 02 de Enero 2001

Principios de Sistemas de Informacion (cuarta edicion) 2000
Un enfoque administrativo
Ralph M. Satair
George W Reynolds
Editorial International Thomson

Revision Internet,
http://www.itsteziutlan.edu.mx/crisis.htm
http://academicos.cualtos.udg.mx/Informatica/Ceneval2003/Ciclo%20de%20Vida%20del%20Software.doc
http://www.asiconsultores.com/serv_cons_calidad.htm
http://www.sqi.gu.edu.au/spice/suite
http://www.avantare.com/articulos/anteriores/MoProSoftV2.0
http://www.monografias.com/trabajos11/reno/reno.shtml#top
http://www.fda.gov/cdrh/comp/guidance/938
http://www.fda.gov/cber/guidelines.htm

Sistemas de Información Teoria y Práctica 1986
John G. Burch, Jr Universidad de Massachussets
Felix R. Strater, Jr Visaron Corporation
Editorial Limusa

Introducción a la Teoría General de Sistemas" 1994
Bertoglio J. Oscar
Editorial Limusa Noriega

Validation of Pharmaceutical Processes
Robert G. Kieffer, Joseph D. Nally
"