UNIVERSIDAD AUTONOMA DE NUEVO LEON

FACULTAD DE CIENCIAS FISICO MATEMATICAS

PROGRAMA DE ASEGURANZA DE CALIDAD EN SOFTWARE

T E S I S

QUE EN OPCION AL TITULO DE
LICENCIADO EN CIENCIAS COMPUTACIONALES

P R E S E N T A

Norma Alicia Rodríguez Pérez

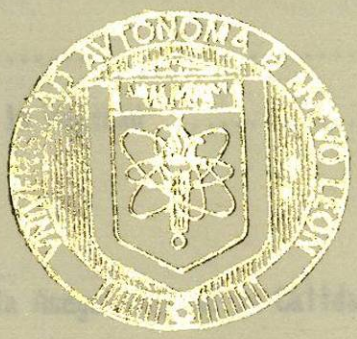
SAN NICOLAS DE LOS GARZA, N. L.

SEPTIEMBRE DE 1989


1080171521

UNIVERSIDAD AUTONOMA DE NUEVO LEON

FACULTAD DE CIENCIAS FISICO MATEMATICAS



PROGRAMA DE ASEGURANZA DE CALIDAD EN SOFTWARE

T E S I S

QUE EN OPCION AL TITULO DE
LICENCIADO EN CIENCIAS COMPUTACIONALES

P R E S E N T A

Norma Alicia Rodriguez Pérez

SAN NICOLAS DE LOS GARZA, N. L.

SEPTIEMBRE DE 1989

I N D I C E

PROLOGO

P R O L O G O

La selección de este tema fue debida a la importancia de la calidad en la creación de cualquier producto.

La calidad es un tema muy importante y más en el ambiente del software.

Generalmente hablar de calidad es algo tan común, que en muchos casos, como en esta área, no se sabe cómo aplicarlo o no se le toma la importancia que tiene este punto dentro del desarrollo de sistemas. Por esto creo que es un tema del cual se puede aprender mucho.

La idea principal para desarrollar la tesis en este tema fue cuando me di cuenta de los problemas tan graves que se cometen por no tener una metodología establecida, ni en el trabajo personal, ni a nivel de empresa, por lo tanto no existe un monitoreo apropiado, porque no se sabe ni cuál es el avance, ni cómo se va en el desarrollo del sistema.

Cuando se realizan las pruebas correspondientes al sistema, al estar terminado, son tantos los errores e incongruencias que se encuentra uno, que en ocasiones más valdría volver a comenzar con un nuevo desarrollo.

El Programa de Aseguranza de Calidad en Software que desarrollo en esta tesis fue basado en el estándar militar 'DOD-STD-2168' que habla específicamente sobre la Aseguranza de Calidad en Software.

Los puntos más importantes en general dentro de esta tesis son la Metodología y el Programa de Aseguranza de Calidad, ambas para el Desarrollo de Software.

INTRODUCCION

1 INTRODUCCION

Actualmente en el área de desarrollo de sistemas se tienen muchas herramientas para el desarrollo mismo, tanto en Hardware como en Software; pero todas estas facilidades si no son usadas de un manera óptima, no se va alcanzar el mejor resultado, por eso debe existir un procedimiento para poder evaluar el trabajo realizado durante el desarrollo.

La Aseguranza de Calidad en el área de Desarrollo de Sistemas es un tema muy poco conocido en México. Esto no quiere decir que no se desarrollan sistemas con calidad, sino que no se tienen procedimientos para verificar la Calidad, esto es, un plan o un formato establecido para supervisar el desarrollo del sistema, que sea algo similar a la Metodología para el desarrollo y que puedan trabajar de una manera conjunta.

Las primeras actividades relacionadas con la Aseguranza de Calidad, de una manera formal, y con el monitoreo de las tareas en el desarrollo fueron realizadas por los Laboratorios Bell en 1916 y a partir de entonces se comenzó a expandir por todo el mundo involucrado en el desarrollo de sistemas.

Durante los inicios solamente se pensaba en tener actividades que orientaran el desarrollo para que se satisfacieran los requerimientos de los clientes, pero no

había una forma general y estructurada de seguir esos pasos en cualquier tipo de desarrollo.

Fue durante 1970 cuando este tema cobró una real importancia dado que el Departamento de Defensa de Estados Unidos comenzó a trabajar con esta idea ya más en concreto y lo primero que definió fue un conjunto de "estándares" que se pudieran aplicar en cualquier desarrollo, y que nos llevara a lograr la Calidad esperada.

Estos estándares marcan el nacimiento del Programa de Aseguranza de Calidad para proyectos de software militar y proporcionan una idea clara de la forma en que se debe desarrollar un programa para sistemas no militares.

1.1 IMPLICACIONES DE LA CALIDAD.

La calidad requiere ineludiblemente de una dedicación, paciencia, esfuerzo, y tiempo. El problema con la calidad no es lo que las personas ignoran, sino lo que creen saber de ella.

Esto pasa por las ideas convencionales sobre la calidad, que la gente ha desarrollado a través del tiempo.

A este respecto, menciona Philis B. Closby que "la calidad tiene mucho en común con el sexo, todo mundo está a su favor, todos piensan entenderlo (aunque no sépan o quieran

explicarlo). Todos piensan que para hacerlo sólo hay que seguir la inclinación natural". Por supuesto, esto es falso ya que calidad es el cumplimiento de requerimientos, por ejemplo, un Caribe es un automóvil de calidad cuando cumple con todos los requisitos de diseño que fueron establecidos para la manufactura de un Caribe.

Conforme se vayan presentando cada uno de los capítulos se irá explicando mejor lo que es la calidad, y se ampliarán los conceptos que en cualquier ambiente donde se quiera aplicar la calidad se deben seguir: éstos son los estándares, las especificaciones y los requerimientos.

También se debe contemplar que la herramienta principal para que todo lo anterior exista es tener una continua supervisión.

La frase más importante o célebre respecto a la calidad es "hacer las cosas bien desde la primra vez", y esto tiene una relación directa con el concepto de la supervisión pues se debe convencer al personal que él es el supervisor número uno de su trabajo y que nadie lo conoce mejor que él por lo tanto nadie lo puede hacer mejor.

GENERALIDADES DE LA ASEGURANZA DE CALIDAD

2 GENERALIDADES DE LA ASEGURANZA DE CALIDAD

El objetivo de este segundo capítulo es proporcionar una idea general de lo que significa tener la calidad en un producto. También es la intención mostrar el significado total de la palabra 'CALIDAD' y de esta manera seguir un camino que nos lleve a estudiar la necesidad de tener calidad en un tipo de producto con unas características muy especiales como lo es el Software, producto intangible pero palpable de una manera subjetiva.

2.1 ALCANCE DE LA ASEGURANZA DE CALIDAD

Generalmente todos suponen entender el significado de calidad y ésta la relacionanos con lujo, alto costo, y otras características que implican dinero en escalas mayores; por otro lado se tiende a creer que es un concepto que no se puede aplicar en cierto tipo de negocios o a cierto tipo de productos. Esto es totalmente falso porque solo muestra que no se conoce el sentido de lo que el término calidad significa. Podrán existir muchas definiciones de varios autores referentes al significado de calidad o a las implicaciones de ella, pero una manera sencilla y en pocas palabras de presentar, es como Philip B. Closby lo menciona calidad es " establecimiento de requerimientos y cumplimiento de los mismos ".

Pero, ¿ qué es el "Establecimiento y Cumplimiento de Requerimientos" ?

La respuesta se muestra sencilla, determinar claramente las características del producto, los productos para elaborarlo, y las herramientas a utilizar. Esto se determina para tener puntos en dónde evaluar y estas evaluaciones son el seguimiento al control y por lo tanto a la verificación del cumplimiento de los requerimientos. De esta manera se puede uno dar cuenta de que la calidad es algo medible, palpable, que no es ningún lujo ya que si queremos satisfacer las necesidades de nuestros clientes es necesario saber cuáles son ellas y establecerlas a la vez. La calidad se emplea en nuestro negocio desde el momento en que el producto es para un consumidor ya que éste siempre buscará lo mejor y al mejor precio. La calidad será tan cara como descuidado sea el trabajo de elaboración del producto, dado que podemos estar checando y corrigiendo en el momento apropiado si es necesario y con la supervisión frecuente, y teniendo bien establecido los estándares de trabajo se puede dar soporte para lograr dichas revisiones y correcciones.

Por lo tanto, ya que calidad es el cumplimiento de requerimientos, hay que dejarlos bien establecidos y bien claros, y entonces es nececesario enfatizar por un lado, que las personas que van a elaborar el producto como encargadas de la calidad, deben estar conscientes de los requerimientos,

y por otro lado, se les debe proveer de los medios necesarios para cumplir con ellos. No obstante, no debe dejar de existir una constante supervisión que sirva también como soporte de ayuda a la persona que está produciendo y que en cierto momento se le puede presentar algún obstáculo.

En resumen, los requerimientos deben ser nuestras reglas para desarrollar un excelente trabajo. Además, se debe tener presente siempre que: 'Hacerlo bien desde la primera vez', implica alta calidad en un sentido total.

2.2 COSTO DE CALIDAD

El costo más alto de la calidad es cuando no hacemos bien las cosas desde la primera vez.

En realidad no existe un método definido para llevar la medición de la calidad, en otras palabras, no existe una fórmula o regla general con lo cual la gerencia por medio de números pueda darse cuenta de cómo están con respecto a la calidad.

Como se mencionó anteriormente, se tiene la idea de que la calidad implica alto costo, cuando en realidad la calidad es GRATIS, esto es, es mas costoso el retrabajo en un producto

o el tener desperdicios, que hacer bien las cosas desde la primera vez.

No obstante lo anterior, una manera de visualizar o medir la calidad, podría ser checando costos de reprocesos rechazos, desperdicios, muestreos, etc.

Estos costos se pueden clasificar en:

a) Costo de Prevención:
   - Capacitación
   - Programas de prevención

b) Costo de Valoración:
   - Supervisión
   - Documentos o planes de evaluación

c) Costo de Falla:
   - Materiales
   - Recursos humanos

A continuación se dará una breve explicación de cada un de estos costos y un ejemplo de cada uno de ellos.

Costo de Prevención:

Este costo es el resultado de visualizar el proceso antes de arrancar e indentificar la oportunidad de error que se pudiera presentar. Aquí se pueden incluir, dependiendo de cada negocio, diferentes actividades de prevención, por ejemplo en un negocio en desarrollo de software, dichas

actividades podrían ser:

- Revisión de especificaciones.
- Entrenamientos al personal.
- Programa de 'cero defectos'.

Costo de Valoración:

En este costo se incluyen las inversiones debidas a las inspecciones para determinar si el producto cumple con los requerimientos y estándares determinados. Siguiendo con el mismo ejemplo anterior, algunas inspecciones ilustrativas son:

- 'Walkthrough' (*).
- Plan de prueba.
- Análisis del cumplimiento de especificaciones.

Costo de Falla:

En este costo están contemplados los casos que no cumplen con los requerimientos y que por lo tanto causan una erogación extra, ejemplos:

- Rediseños.
- Costos de acciones correctivas.
- Confiabilidad del producto.

(*) 'Walkthrough' es la revisión del codigo para optimizarlo.

2.3 EL CONCEPTO 'CERO DEFECTOS'

Este concepto debe de existir en cada una de las personas que van a colaborar en la realización de un producto. Esto es, debe estar claramente comprendido que cada una de esas personas son el supervisor más importante de su propio trabajo y por lo tanto en ellas empieza la calidad.

El concepto de 'Cero Defectos' se basa en el hecho de que sólo 2 cosas ocasionan un error en la elaboración de un producto:

1) Falta de conocimiento, y
2) Falta de atención.

La primera, se puede considerar fácil de solucionarla pues capacitando y entrenando a la persona, podemos sacar adelante ese problema.

La segunda, es algo emocional por llamarlo de alguna manera, por lo tanto mas difícil de atacar. Hay que lograr que la persona vea el trabajo como un reto y se sienta apoyado para vencer cualquier distracción que motive esa falta de atención.

Por supuesto que para alcanzar 'Cero Defectos' en un producto, todos los involucrados deben tener bien claro los estándares e irse formando la mentalidad de que el estándar

personal es hacer su trabajo con CERO DEFECTOS.

De esta manera se puede obtener el nivel de calidad de la excelencia porque se produce un trabajo perfecto, o bien, con 'Cero Defectos'.

2.4 FUNDAMENTOS DE LA ADMINISTRACION DE CALIDAD.

Con lo que se ha visto se pueden determinar los fundamentos de la administración de calidad. Esto implica que el apegarse a dichos fundamentos conducirá a un trabajo de calidad. Estos fundamentos son:

1) La calidad significa cumplimiento de requerimientos, y no elegancia.

2) No existe tal cosa como la economía de la calidad; siempre es más económico hacer bien las cosas desde la primera vez.

3) La única medición del desempeño es el costo de calidad.

4) El único estándar de desempeño es cero defectos.

Como se mencionó al principo del capítulo la finalidad de éste era dejar en claro el concepto CALIDAD. Otro punto que durante el capítulo se trató y es muy importante, es que las personas que colaboran en la elaboración del producto se

deben sentir parte de la calidad y sólo de esta manera se obtendrá mayor y mejor calidad.

En el capítulo siguiente se presentará específicamente lo que "calidad en software" significa y la importancia de ésta en el desarrollo del software.

ASEGURANZA DE CALIDAD

EN SOFTWARE

3 ASEGURANZA DE CALIDAD EN SOFTWARE

La finalidad de este capítulo es definir lo que CALIDAD EN SOFTWARE significa. En el capítulo de Generalidades de la Aseguranza de Calidad y en éste, Aseguranza de Calidad en Software, se tiene la misma meta pues ambos hablan de la Calidad, siendo ésta, dar al cliente lo que pide, al menor costo y de primera calidad.

Dado que el Software es un producto como cualquier otro, con sus particularidades, debe seguir la misma línea de cualquier otro producto, esto es, que en el momento en que se tome la desición de hacer un análisis para un nuevo sistema o mantenimiento de uno ya existente, es necesario que el concepto de Calidad empieze a presentarse y forme parte de todo el trabajo que se tenga que realizar.

3.1 CALIDAD EN SOFTWARE

Existen muchas definiciones de Calidad de Software pero se presentará una que sirva a los propósitos del Programa que en el capítulo 5 se propondrá.

La Calidad de Software es la adecuación a requerimientos funcionales, y desempeño establecido explícitamente, estándares

de desarrollo y las características implícitas que todo software desarrollado profesionalmente debe tener.

Tres puntos importantes en la Calidad en Software son:

1) Los requerimientos de software son los fundamentos para que la calidad se mida. Una falla de conformidad con los requerimientos es una falla de calidad.

2) Los estándares definen un conjunto de criterios que guían la manera con la que se desarrollará el software

3) Si el software se somete a estos requerimientos explícitos pero fallan los requerimientos implícitos entonces se dudará de la calidad del software.

3.2 FACTORES DE LA CALIDAD EN SOFTWARE

Los factores son medios por los cuales se puede evaluar el nivel de la calidad del sistema.

Los factores que intervienen en la calidad y que forman parte de una relación que más delante se discutirá, se decribirán a continuación:

1) Exactitud.

La extensión con que el programa satisface las especificaciones y cumple con las nececidades del cliente.

2) Constancia.

El grado con el cual se espera que el programa desempeñe su función con la precisión requerida.

3) Eficiencia.

La cantidad de recursos computacionales y código requerido para que el programa cumpla con su función.

4) Integración.

El grado con que los accessos al software o datos por personas autorizadas son controlados.

5) Facilidad de uso.

El trabajo requerido para entender, operar, preparar la información de entrada y la interpretación de la salida del programa.

6) Flexibilidad.

El esfuerzo requerido para modificar un programa.

7) La facilidad de prueba.

La facilidad con que se pueda probar un programa y que cumpla con las especificaciones, o sea que haga lo que tiene que hacer.

8) Portabilidad.

La flexibilidad del sistema de poderse transferir de un hardware y software donde esté funcionanmdo a otro con el mínimo de cambios.

9) Interoperatividad.

El esfuerzo requerido para acoplar un sistema a otro.

10) Mantenimiento

La facilidad o dificultad de la correción de un error en un programa.

11) Reusabilidad

La facilidad de usar un programa o parte de un programa en otras aplicaciones.

Los factores arriba descritos no podrían ser medidos por sí mismos sino que se requiere de la ayuda de otros elememtos para ser evaluados, dichos elementos serán mencionados a continuación:

1) Auditorías.

Son el checar el desarrollo del sistema por medio de estándares.

2) Precisión.

La precisión del control y computación.

3) Comunicación.

El uso de interfases, protocolos, y bandas de comunicación.

4) Complementación.

El grado con el cual la implantación esté completa bajo las funciones de los requerimientos que se deben llevar acabo.

5) Consistencia.

Checando el uso de diseño uniformado y documentación técnica hasta el final del desarrollo del proyecto en software.

6) Tamaño.

La complementación de los programas en llamadas y líneas de código.

7) Datos comúnes.

El uso de estándares en la estructura de datos y a través de todo el programa.

8) Tolerancia de errores.

El daño que ocurre cuando el programa se encuentra con un error.

9) Eficiencia de ejecución.

El tiempo de corrida del programa.

10) Crecimiento.

El grado con que la arquitectura, procedimientos o datos puedan extenderse o crecer.

11) Generalidades.

La forma en que algún procedimiento de un programa se puede aplicar a otro.

12) Instrumentación.

El grado con el que el programa monitorea sus propias operaciones e identifica errores que ocurren.

13) Modularidad.

Las funciones independientes de los componentes de los programas.

14) Operatividad.

La facilidad de operación de un programa.

15) Seguridad.

La actividad que controla o protege los programas y datos.

16) Autodocumentación.

El grado con que se origina la documentación dentro del código.

17) Simplicidad.

El grado en que se puede entender un programa al existir una dificultad.

18) Independencia del medio ambiente de desarrollo.

El grado con que el programa es independiente de las características no estándares del lenguaje de

programción, del sistema operativo y otras constantes del medio ambiente.

19) Entrenamiento.

El grado con el que el software permite nuevos usuarios a las aplicaciones del sistema.

20) Fácil de seguir.

La facilidad de seguir el diseño de un programa apoyándose en los requerimientos.

En la figura 3.1 se muestra una tabla con la relación entre los factores y las medidas. Esta tabla permite visualizar como se pueden medir cada uno de los factores de calidad por medio de medidas de calidad.

3.3 ASEGURANZA DE CALIDAD EN SOFTWARE

La Aseguranza de Calidad en Software es la actividad que se encarga de hacer auditorías al desarrollo del sistema para que éste se esté produciendo con calidad, esto es, que cumpla con los estándares, requerimientos y especificaciones establecidos al iniciar el desarrollo del sistema (de estos tres conceptos se hablará en el capítulo siguiente).

Actividades de la Aseguranza de Calidad en Software

Las actividades de la Aseguranza de Calidad en Software son las que soportan o respaldan a la calidad, éstas son:

FACTORES Y MEDIDAS DE CALIDAD

MEDIDAS DE CALIDAD EN SOFTWARE \ FACT. DE CALIDAD	1	2	3	4	5	6	7	8	9	10	11
AUDITORIA				X				X			
PRECISION		X									X
COMUNICACION											
COMPLEMENTACION	X										
CONSISTENCIA		X					X	X			
TAMANO	X	X				X	X				
DATOS COMUNES											X
TOLERANCIA DE ERRORES		X									
EFICIENCIA DE EJECUCION			X								
CRECIMIENTO							X				
GENERALIDADES							X		X	X	X
INSTRUMENTACION				X	X			X			
MODULARIDAD		X				X	X	X	X	X	X
OPERATIVIDAD			X		X						
SEGURIDAD				X							
AUTODOCUMENTACION						X	X	X	X	X	
SIMPLICIDAD		X				X	X	X			
INDEP. DEL SISTEMA DE SOFTWARE									X	X	
ENTRETENIMIENTO	X										
FACIL DE SEGUIR											X

1. EXACTITUD
2. CONSTANCIA
3. EFICIENCIA
4. INTEGRACION
5. FACILIDAD DE USO
6. MANTENIMIENTO
7. FLEXIBILIDAD
8. FACILIDAD DE PRUEBA
9. PORTABILIDAD
10. RESUABILIDAD
11. ITEROPERATIVIDAD

FIGURA 3.1

1) Aplicación de Métodos Técnicos.

   Son los que ayudan al analista para llevar acabo una alta calidad en especificaciones y diseños.

2) Conducción de Revisiones Técnicas Formales.

   Es una junta que tiene la gente encargada del desarrollo del sistema con el propósito de encontrar problemas de calidad.

3) Pruebas de software.

   Combinar múltiples estrategias con una serie de casos de pruebas y métodos diseñados que ayuden a asegurar la efectividad de detección de errores.

4) Reforzamiento de Estándares.

   Los estándares generalmente son diseñados por cada compañía, en otros casos son puestos por el cosumidor. La Aseguranza de Calidad es la encargada de darle segumiento a que se estén cumpliendo estos estándares.

5) Control de Cambios.

   Cada cambio es un error potencial introducido o crea algún efecto que produzca errores. Los cambios controlados contribuyen a minimizar la probabilidad de que se introduzcan esos errores, porque antes de llevar a cabo dichos cambios, se debe hacer un análisis del impacto que causaría ese cambio.

6) Medidas.

Son las que permiten evaluar la habiliad de la metodología y cambios de procedimientos que permitan mejorar la calidad de software.

7) Reportes.

Estos sirven para tener documentación acerca del proyecto, resultados, auditorías y revisiones.

3.4 REVISIONES DE SOFTWARE

Las revisiones es la actividad principal de la Aseguranza de Calidad en Software, pues es la que va a permitir que se dectecten a tiempo la fallas, y sin las revisiones podría decirse que la calidad es nula.

Las revisiones funcionan como un filtro, porque al estar revisando continuamente se van encontrando errores y se van corrigiendo y esto va a permitir que se depure el software.

Existe un modelo de la IBM81 que permite ver lo importante que son las revisiones periódicas y que deben existir en cada una de las fases del desarrollo de sistemas. El modelo recibe el nombre de Amplificación de Defectos y Traslación que se muestra en la figura 3.2.

Este modelo demuestra como durante todo el desarrollo

MODELO DE AMPLIFICACION DE DEFECTOS Y TRASLACION

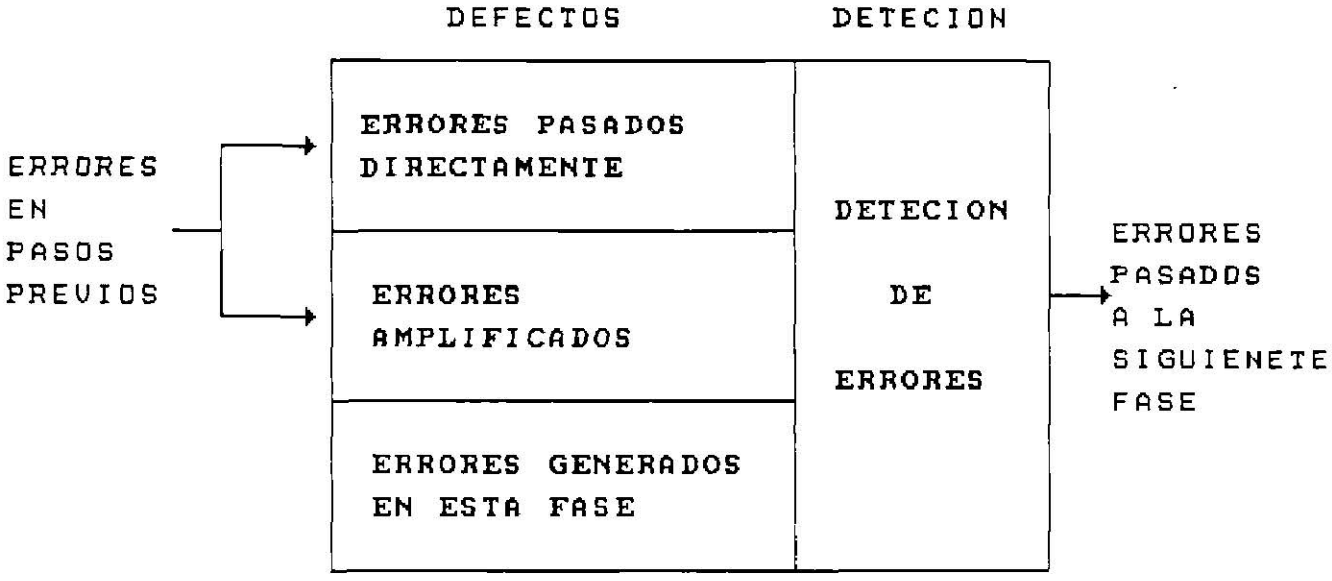


FIGURA 3.2

de sistema se van generando errores, que cuando se empieza el desarrollo son minimos, pero que si no se corrigen se van arrastrando a través de cada fase, hasta salir a flote y el problema ya es mayor, por lo tanto más compicado de manejarse.

A continuación se explican los parámetros del modelo:

Errores pasados directamente : Estos son los errores que se generan en la fase anterior y se quedan en la fase actual.

Errores amplificados: Estos son los errores que se generan en la fase anterior y producen otros en esta fase.

Errores generados en esta fase: Estos son generados propiamente por esta fase.

Detección de errores: Es el porcentaje en que se lograron eliminar los errores.

La generación y detección de errores se encuentra desde el análisis hasta la implantación. A continuación se mostrarán dos ejemplos usando este modelo. Uno de los ejemplos es con cero supervisión, esto es, no existen revisiones periódicas, todo el tiempo se supone que todo está correcto por lo tanto los errores se van arrastrando a través de todas la fases y

al finalizar el desarrollo del sistema, cuando ya sea demasiado tarde, van a salir a flote los errores, esto originará un mayor retrabajo y una muestra de la mala calidad (figura 3.3). El segundo ejemplo mostrará cuando exista contínua supervisión, en este caso pasa lo contrario de la situación anterior, aquí existen contínuas auditorías que permiten detectar, si no todos, la mayoría de los errores, así que la calidad del producto será mejor. Esto está ilustrado en la figura 3.4.

Con estos ejemplos nos podemos dar cuenta que al no existir una supervisión constante va a hacerse mucho mayor el retrabajo, por lo tanto la calidad es mínima o puede ser nula.

Para concluir, la Aseguranza de Calidad en Software es la encargada de las auditorías para que de esta manera todo siempre obtenga el grado más alto de calidad y menor retrabajo siguiendo siempre los tres puntos mas importantes: seguimiento de estándares, supervisión contínua en el apoyo a las especificaciones, y cumplimiento de requerimientos.

En el siguiente capítulo se mostrará la metodología en la cual se basará el programa de Aseguranza de Calidad en Software, en donde además de presentar la forma de trabajo, se practicarán los principios vistos en el presente capítulo.

SUPERVISION CERO

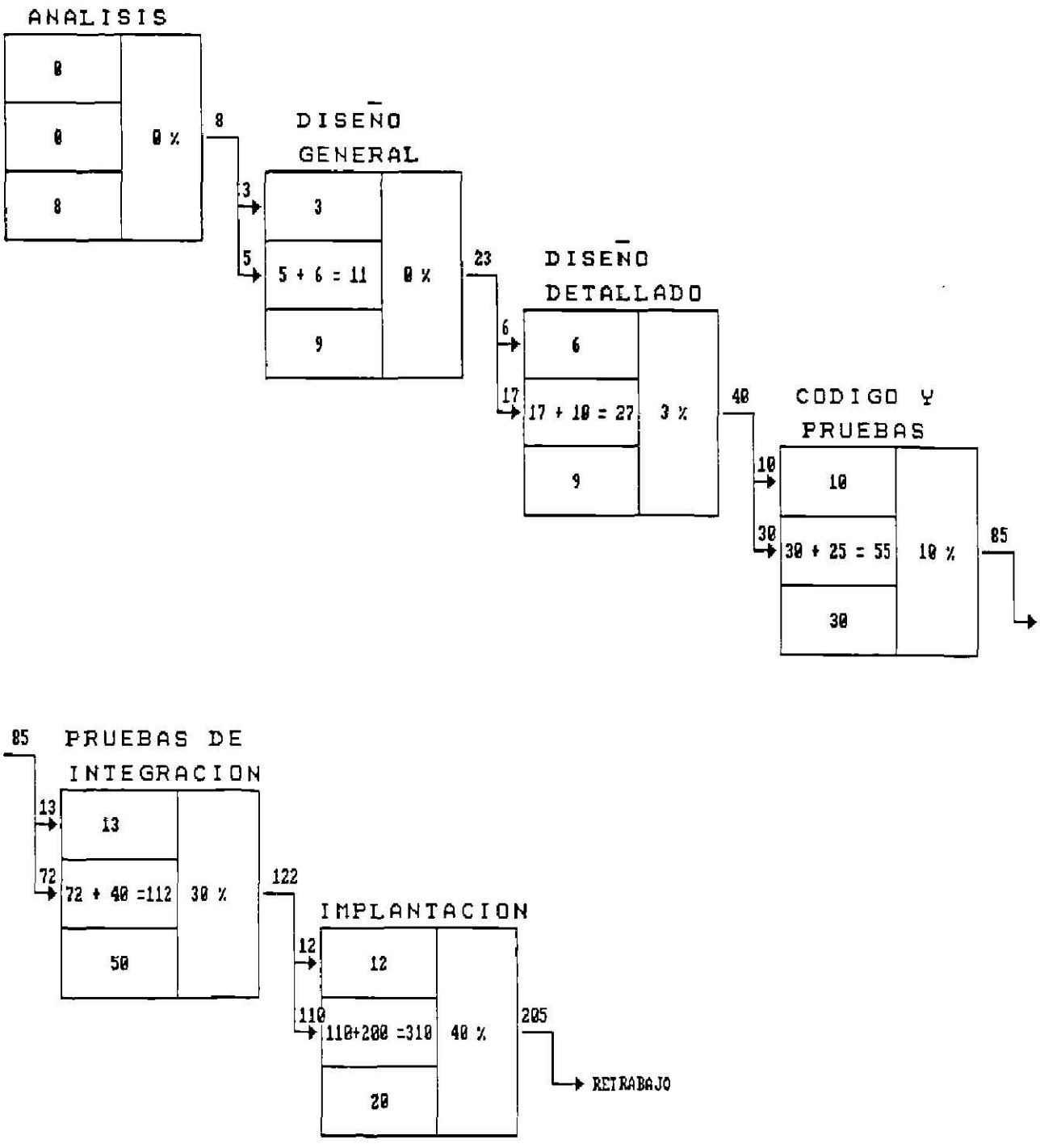


FIGURA 3.3

SUPERVISION CONSTANTE

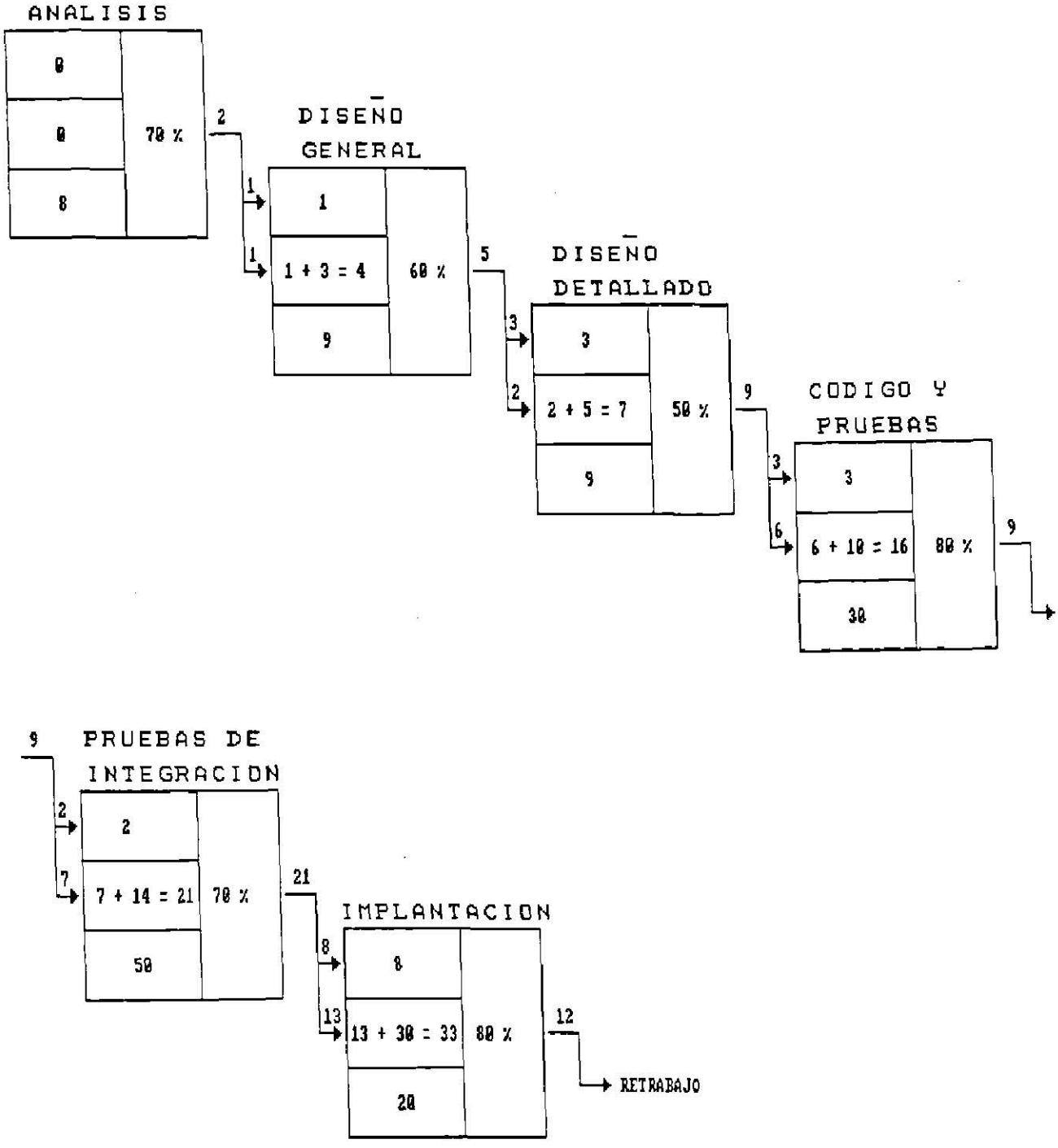


FIGURA 3.5

UNA METODOLOGIA PARA EL DESARROLLO DE SISTEMAS

4 UNA METODOLOGIA PARA EL DESARROLLO DE SISTEMAS

Como se mencionó en el capítulo anterior, la Aseguranza de Calidad se aplica en el desarrollo de cualquier sistema. Ahora bien, para llevar a cabo un buen desarrollo es necesario tener una metodología que ayude a estructurar el sistema en cuestión, por un lado, y por otro tener un avance seguro en el desarrollo.

La intención de este capítulo es presentar una metodología que sirva como base para aplicar el programa de Aseguranza de Calidad, que se verá en el siguiente capítulo. No se trata de discutir qué alternativa es mejor, sino solamente tener como referencia una buena metodología estructurada donde se pueda aplicar el programa ya arriba mencionado.

4.1 OBJETIVOS DE LA METODOLOGIA

El objetivo principal de la metodología es tener control durante todo el desarrollo del sistema, dividiendo este desarrollo en fases, para que de esta manera se pueda hacer una evaluación y así tener un control más preciso. Otro punto también muy importante es que el analista al tener el patrón, o sea, la metodología que le permita al usuario involucrarse en cada una de las fases, podrá verificar el avance más de cerca y el cumplimiento de las necesidades de dicho usuario.

4.2 FASES DE LA METODOLOGIA

La metodología propuesta como base tiene las siguientes fases o etapas:

1) Análisis del Sistema.
2) Diseño General del Sistema.
3) Diseño Detallado del Sistema.
4) Codificación y Prueba de Programas.
5) Integración del Sistema.
6) Documentación del Sistema.
7) Implantación del Sistema.
8) Mantenimiento del Sistema.

A continuación se explicará en términos generales, cada una de las fases que componen la metodología que se tomará como apoyo durante el desarrollo del Programa de Aseguranza de Calidad.

Análisis del sistema.

El análisis generalmente nace de la necesidad de buscar la solución a un problema o simplemente para mejorar un sistema ya existente.

En un principio, las razones o soluciones al problema o necesidades son vagas y están mal definidas, pero conforme el analista se va involucrando con las necesidades del cliente,

las cosas se ven más claras. Esto es, el analista debe tener la habilidad y la visibilidad para saber dónde buscar, con quién o a quién preguntar y así ir formando un juicio correcto sobre el sistema. Las personas son la fuentes más importantes para el analista; quien además de cuidar el no suponer nada que no esté comprobado, irá documentando toda la información que se obtenga, incluyendo las conclusiones que se van obteniendo.

En el análisis también deben considerarse algunos otros puntos como: a) tener completamente clara la necesidad del usuario, b) considerar la economía de la empresa, c) el presupuesto asignado, d) la tecnología disponible, y si no la hay, buscar la mejor solución o la más factible con la cual se pueda solucionar el problema actual teniendo en cuenta también el futuro, y e) verificar la disponibilidad de todos los recursos necesarios. Como se presentó anteriormente se trata de analizar las variables al máximo para correr el mínimo riesgo.

La documentación que se espera de esta fase, es un reporte del análisis, donde se va a plasmar o explicar las conclusiones y se va a demostrar si están totalmente comprendidas la necesidades del usuario y por qué la solución escogida o seleccionada es la mejor. El que tiene la última palabra es el usuario, así que la documentación es un punto bastante importante tanto en esta fase como en las restantes, ya que en ellas se resumirá gran parte del trabajo desarrollado.

En esta documentación los puntos que se deben cubrir son:

- Identificación del problema o situación existente.
- Sistema actual.
- Sistema propuesto.
- Ventajas y desventajas del sistema actual.
- Ventajas y desventajas del sistema propuesto.

Una vez presentado el reporte al usuario, es indispensable esperar a que sea aprobado y así poder seguir con el resto del trabajo. Lo anterior es importante para el analista pues de esta manera respalda mejor sus resultados.

Diseño General del Sistema.

Esta fase se enfoca a la formulación de especificaciones generales para el nuevo sistema o subsistema propuesto.

Lo primero que hay que hacer en esta fase es revisar la información obtenida en la fase anterior y en segundo término, planear las actividades que se deben realizar para llevar a cabo esta parte.

Para el Diseño General debemos tener claro las siguientes situaciones:

a) Los recursos de la organización.

Estos recursos son las máquinas, las personas, los materiales, dinero y métodos de trabajo, el propósito

de conocer esto es utilizar al máximo cada uno de estos recursos y otros que se irán considerando durante el desarrollo del sistema.

b) Las necesidades del usuario.

Esto es, cuando se desarrolla el análisis, se obtiene información global y por lo tanto soluciones globales, pero conforme se va avanzando en el diseño, se puede identificar que son muchas las necesidades de los diferentes usuarios y entonces el analista debe tener la visión necesaria para atacar los de mayor impacto, pero tratando de satisfacer a todos los usuarios y sus necesidades.

c) La interrelación con otros sistemas.

Aquí el analista debe tener la visualización de que no va a ser el único sistema y que es muy probable que a éste se le agreguen algunos otros, por lo tanto debe dejarlo preparado para ello.

d) La operación de datos.

En esta parte se examinan y se estudian los datos que van a alimentar al sistema, por lo tanto es muy importante el comenzar a clasificarlos.

Lo que hay que definir primordialmente dentro del diseño general es:

a) El flujo de información.

Es el camino que seguirá la información a través del

sistema, definiendo cuál es la información de entrada y cuál la de salida.

b) Recursos a utilizar.

Aquí se definirá el equipo a usarse, el personal a emplearse, la seguridad y el lugar donde se instalará el equipo.

c) Estándares.

Aquí se definirá el maquillaje del sistema. Como las pantallas que tengan el mismo formato, los mensajes, los reportes. La estandarización es muy importante ya que va a permitir que el usuario se involucre más rápidamente, esto es, que conforme vaya avanzando a través del sistema va a saber qué respuesta él va a dar ya que cada una de las fases o programas se comportan de la misma manera y responden igual, el sistema va a dar la apariencia de que fue hecho por una sola persona y tendrá una presentación más profesional.

d) Consideraciones para el plan de prueba del sistema.

Las consideraciones son tanto de software como hardware, donde el hardware es el equipo en el que se va a probar, una consideración de software es que los datos que van a servir de prueba estén listos para procesarse, otra es la gente que va a estar encargada de las pruebas.

Cada una de las situaciones mencionadas anteriormente deben estar claramente documentadas, porque éstas servirán de apoyo para las fases siguientes y para la presentación con el usuario, y en base a esta documentacion él podra decidir si se continúa adelante o se rediseña alguna parte.

Diseño Detallado del Sistema.

Lo primera actividad es revisar la información obtenida en la fase anterior, y planear las actividades para esta fase.

En esta fase, como su nombre lo indica, es donde se van a definir los detalles del sistema, por lo tanto todo debe estar completamente claro y especificado, recordando siempre que del análisis detallado depende la funcionalidad del sistema.

En esta fase se establecerán:

a) Sumario del sistema detallado.

   Aquí está contemplada toda la información a detalle del sistema en general y los módulos que lo soportan.

b) Estructura del sistema detallado.

   Aquí se muestra de una forma gráfica el flujo de la información a través de los modulos de sistema y también se muestra la interrelación de programas con bases de datos, esto se ilustra en las figuras 4.1. y 4.2 respectivamente.

ESTRUCTURA DEL SISTEMA DETALLADO

(MODULO)

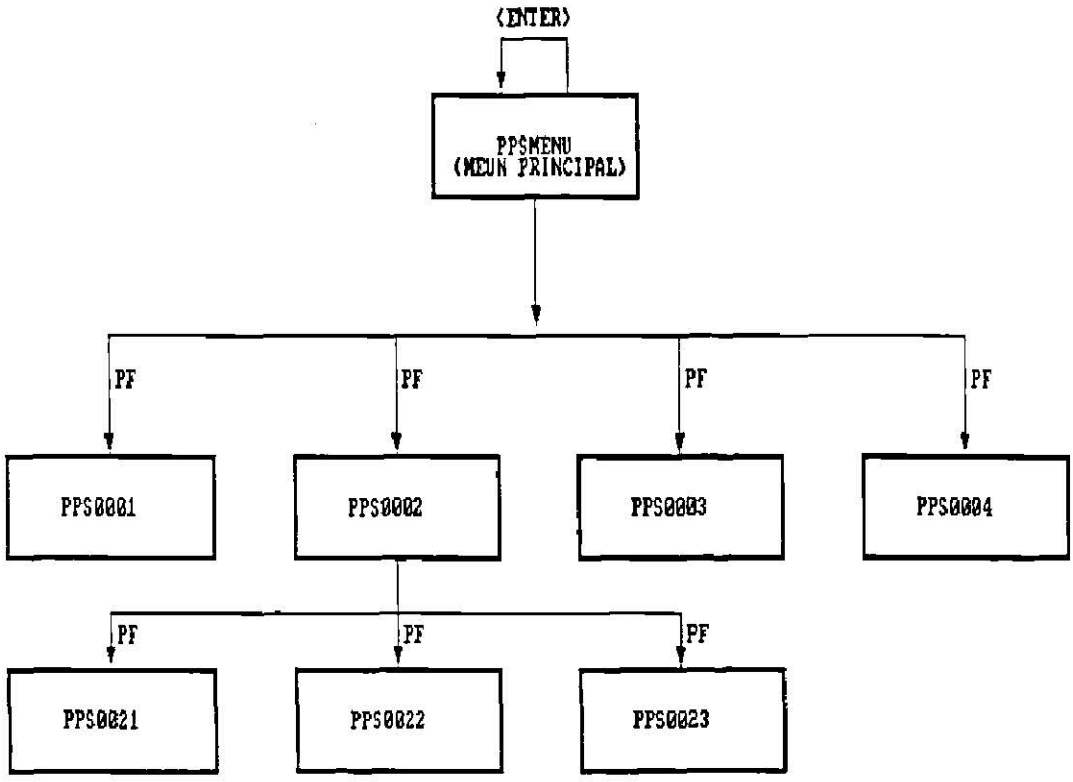


FIGURA 4.1

ESTRUCTURA DEL SISTEMA DETALLADO

(PROGRAMAS)

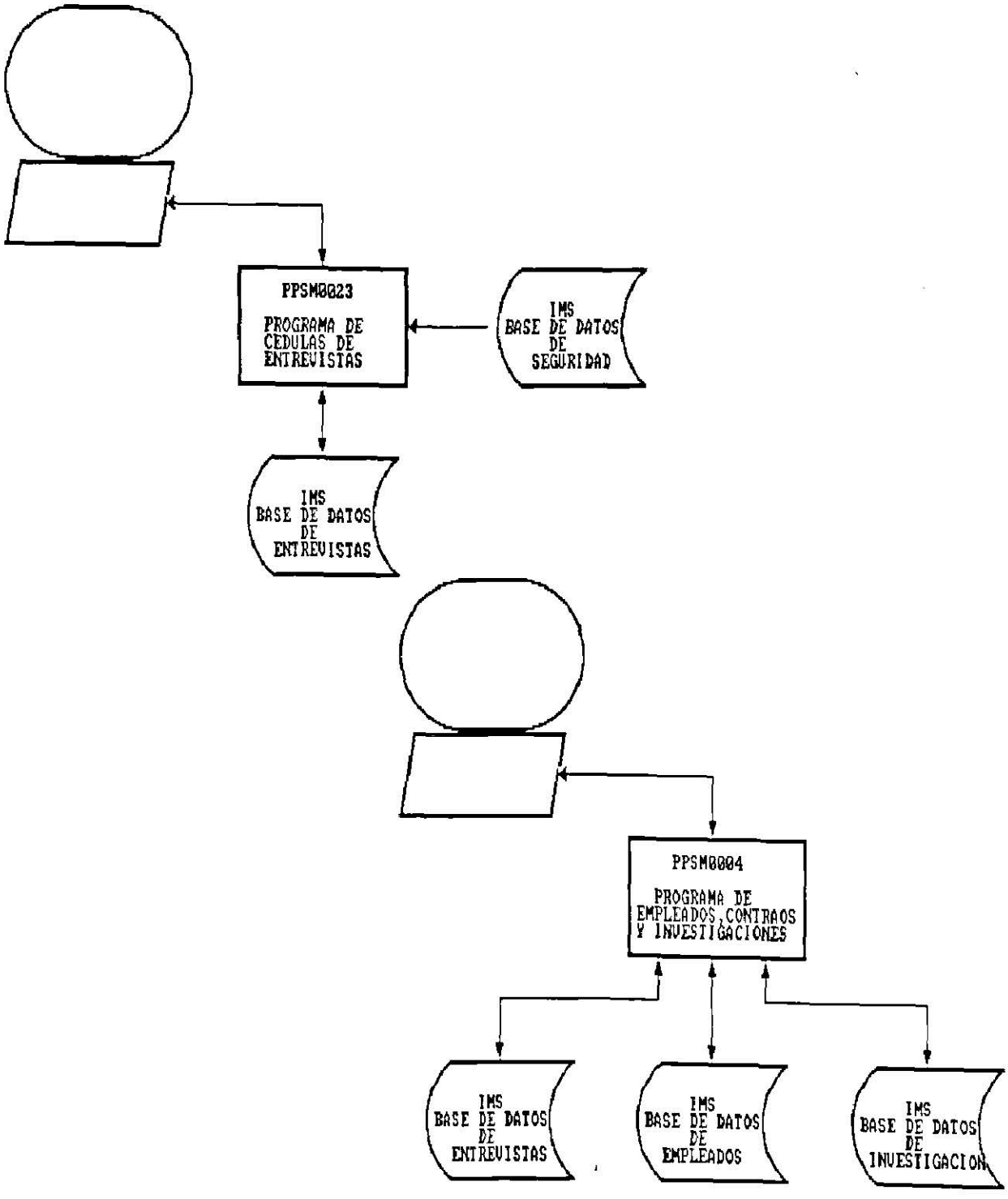


FIGURA 4.2

c) Los programas que integran el sistema.

Esto es, la lista de programas que lo compondrán, qué tipo de programas van a ser y dentro de qué módulo quedarán.

d) Los archivos.

Aquí es donde se van a definir qué campos van a componer cada archivo, qué longitud y cuáles serán las llaves de acceso, cuidando de no tener información redundante.

e) Diagrama jerárquico.

Esta es un forma bastante visible para conocer todo el sistema, tanto para el analista que va a desarrollar, como para la gente que va a trabajar con él, porque se puede ver dónde se está, y cómo afecta un programa a otros, o sea, es más fácil para cualquier persona ver el flujo de información con este diagrama (fig. 4.3).

f) Diccionario de datos.

Este documento contiene toda la información de los campos utilizados en las pantallas, reportes, y archivos, con el significado de cada uno de ellos.

g) Matriz de referencias.

Aquí se tienen matrices que contienen información referente a programas, archivos y campos, y la relación que existe entre ellos.

De esta manera es más fácil visualizar qué campos utilizamos, en qué programas y de qué archivos, reflejándose un panorama más claro, para el momento en que se quiera hacer una modificación se pueda ver a quién afectará y a quién no.

Las matrices propuestas son:

1) Programa/Programa
2) Programa/Campos
3) Programa/Archivos
4) Archivos/Archivos

Un ejemplo de una matriz de referencia se muestra en la figura 4.4.

h) Lista de mensajes.

Dentro de esta lista se definirán todos los mensajes a utilizar dentro de los procedimientos del sistema. De esta manera todo el sistema estará estandarizado al mismo tipo de mensajes.

i) Rutinas comúnes.

Son las rutinas usadas a través de todo el sistema y que se encuentran fuera del código del programa que las requiere, estas rutinas son las comúnes al nivel del todo el sistema y por lo tanto los usan varios programas.

DIAGRAMA JERARQUICO

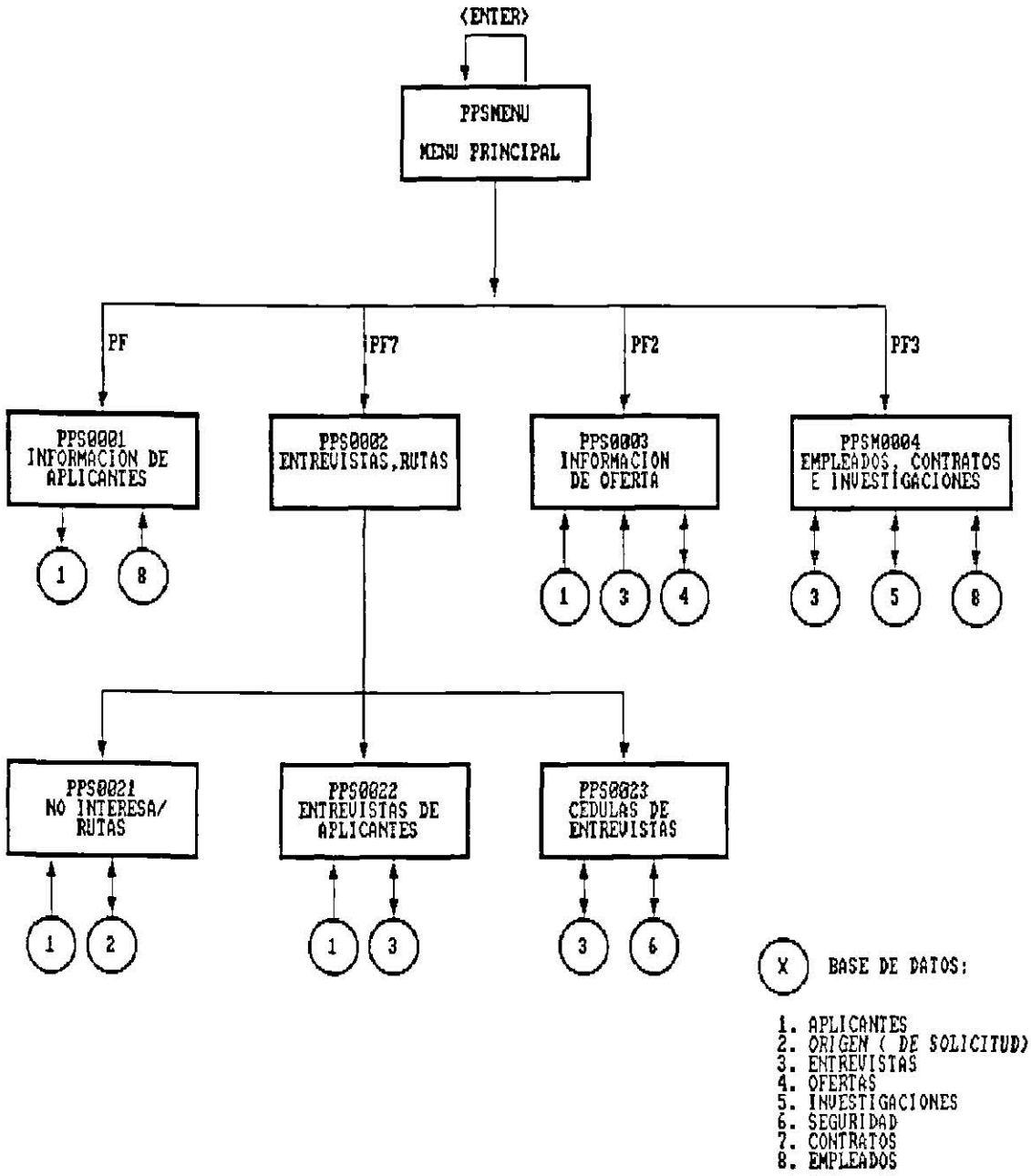


FIGURA 4.3

MATRIZ DE REFERENCIA CRUZADA

PROGRAMAS VS. BASE DE DATOS

PROGRAMAS \ B.D.	1	2	3	4	5	6	7	8
PPSMENU								
PPS0001	ADCI	ADCI						
PPSM002								
PPS0021	I		ADCI		C			
PPS0022				ADCI		I	ADCI	
PPS0003					ADCI			I
PPS0004		I				ADCI		ADCI

BASE DE DATOS

1. Aplicantes
2. Origen
3. Entrevista
4. *Investigaciones*
5. Seguridad
6. Contratos
7. Empleados
8. *Ofertas*

PROCESOS

A = Insertar

D = Borrar

C = Cambios

I = Consultar

FIGURA 4.4

j) Especificaciones de los programas.

Estos documentos es el enlance con la siguiente fase ya que de éstos se procederá a la programación del sistema, por lo tanto, las especificaciones son los documentos que definirán exactamente qué va a hacer el sistema y de qué forma. Por estas razones, el contenido de las especificaciones debe ser amplio y sin ambiguedades dicho contenido incluye:

1) Narrativa de un programa, es como se debe de hacer el programa. Mensajes que se desplegarán durante la ejecución del programa.
2) Diagrama de flujo a grandes rasgos.
3) Rutinas comúnes son rutinas que usa el programa pero están externas al código de ese programa, por lo tanto es necesario listarlas y anexarlas a las especificaciones (estas rutinas son a nivel programa).
4) Versión y revisión.
5) Los archivos y campos a utilizar.
6) Las pantallas a utilizar y/o reportes.
7) Cálculos, esto es necesario mostrarlo cuando se hacen operaciones complicadas dentro del programa, esto es, cuando por ejemplo se efectúa un cálculo de un costo, es necesario definir qué campos se utilizan y en base a qué fórmula se está realizando el cálculo.

8) Directrices para la prueba del programa, éstas contienen los puntos más importantes que servirán como base para las pruebas del programa.

Codificación y Prueba de Programas.

Lo primero a realizar en esta fase es el distribuir cada programa a las personas que lo van a desarrollar.

Respecto a la programación individual, hay una serie de actividades que se tienen que seguir y que están orientadas a estructurar el trabajo en fases.

Dichas actividades son:

a) El comprender al 100% los requerimientos del programa y su relación con el resto de programas.

b) Desarrollar el diagrama de flujo.

c) Desarrollar el plan de prueba.

d) Codificación.

e) Pruebas.

f) Resultados de pruebas.

Es necesario que en esta fase exista un análisis profundo sobre cada especificación por parte del programador, independientemente de la realizada por el analista, para evitar o considerar cualquier tipo de error. Esto último es más sencillo de realizar con toda la documentación ya obtenida anteriormente, como son las matrices de referencias, el diagrama jeráquico, y la demás información del Diseño Detallado.

Las especificaciones son muy importante ya que estas son el enlace entre el Diseño Detallado y la codificación. En lo que se refiere a las pruebas, éstas, se realizarán de una manera más efectiva con el plan de prueba ya realizado, antes de haber empezado la codificación. Por lo tanto la posibilidad de que el programa en particular falle es mínima puesto que el plan de prueba ya fue aceptado y verificado así que el programa en teoría hace lo que tiene que hacer.

Estas actividades también debe estar documentadas en cada uno de los puntos mencionados arriba y aprobados por el analista, y deben de tomarse esta actividades como reglas, cumpliendo el orden en que se mencionaron.

Integración del Sistema.

Lo primero a realizar en esta fase es el revisar las especificaciones de cada programa ya desarrollado, de esta manera se previene algún cambio ocurrido durante el desarrollo de los programas.

En esta fase es donde están contempladas las pruebas a todo el sistema, esto es, checar cómo afecta un movimiento de la información a través de todo el flujo de los programas.

Es necesario determinar al personal que hará estas

pruebas y el medio ambiente en que se harán, de esta manera el plan de trabajo debe de contemplar estos dos puntos y además cómo se va a probar, o sea, la información que va a entrar y la que se debe obtener.

Todo debe quedar documentado claramente para revisiones posteriores, principlamente los resultados de las pruebas.

Estos deben de incluir:

1) Propósito de la prueba (Específico).
2) La información de entrada.
3) Resultados obtenidos.

Documentación del sistema.

Esta fase ya está casi completa, pues en cada fase anterior se estuvo documentado todo, y esta información en su mayoría compone el Manual Técnico así que sólo queda completar este manual y preparar los manuales restantes, los manuales serían:

1) Manuel Técnico
2) Manual de Operaciones
3) Manual del Usuario

El Manual Técnico es el que contiene la información que se genera durante el desarrollo del sistema, esta documentacion es obtenida principalmente en la fase de diseño

detallado, como las pantallas del sistema, bases de datos o archivos y campos de cada una de ellas, lista de errores de mensajes, especificasiones de cada uno de los programas que contiene el sistema.

El Manual de Operaciones es el que contempla la operación del sistema desde el punto de vista de Hardware/ Software, esto es, el equipo en que debe instalarse, qué hacer cuando exista un error, como realizar respaldos, etc.

El Manual del Usuario es el que muestra la documentación para que trabaje el usuario con el sistema, esto es, cómo puede el usuario moverse a través del sistema.

Implantación del sistema.

La implantación es finalmente la puesta en marcha del sistema y la fase que pone en relación directa al usuario y al sistema mismo.
Claro que para ser posible esto y que se opere de la manera correcta es necesario la capacitación del personal que va a operar el sistema, de tener todo claramente documentado y esto debe incluir:

1) Manera de implantarlo,
2) Cómo arrancar el sistema,
3) Operación del equipo, y
4) Capacitación del usuario.

Mantenimiento del sistema.

El mantenimiento es la fase del desarrollo de sistemas que se refiere a las actividades que se deben realizar cuando una modificación o cambio al sistema o parte de él se presente, esto es, actualizar lo ya implantado con los nuevos requerimientos teniendo bastante cuidado en esos cambios, recordando que existe una documentación previa del sistema y que cuando se desarrolló el sistema se hizo con una cierta estructura para poder dar mantenimiento posterior.

Cuando estos cambios se realicen, por cada uno de ellos es necesario que se sigan todas las fases ya mencionadas teniendo presente que posteriormente puedan hacerse necesarios otros cambios.

De este mantenimiento dependerá la vida del sistema e igualmente que en las fases anteriores, hay que documentar estos cambios.

4.3 VENTAJAS DE LA METODOLOGIA

Las ventajas de la metodología son que ésta sirve como patrón, como guía que va a permitir visualizar el avance del proyecto tanto al analista como al usuario y principalmente al analista le permite evaluar más de cerca al proyecto ya que teniendo el desarrollo en fases la

evaluación es más exacta, y por lo mismo si existiera algún error se puede corregir en ese momento y esto permite que no se detecte estos errores cuando el proyecto está terminado sino en cada una de las fases.

4.4 ROL DE LOS REQUERIMIENTOS, LAS ESPECIFICACIONES Y LOS ESTANDARES.

Después de haber mostrado la metodología que servirá de apoyo para el Programa de Aseguranza de Calidad, en este punto se mostrará el papel tan importante que juegan, dentro de la Calidad en Software, estos tres conceptos los requerimientos, las especificaciones y los estándares.

1) Rol de los Requerimientos:

   Los requerimientos son la definición o establecimientos de todas las necesidades o peticiones del cliente, por lo tanto estos son la base para iniciar y guiar el desarrollo del sistema, es por esto que es uno de los puntos esenciales del analista, pues es el punto de partida para su planeación y la meta, que es satisfacer las necesidades del usuario con el desarrollo del sistema.

2) Rol de las Especificaciones:

   Las especificaciones son las definiciones escritas y establecidas claramente de lo que hay que hacer.

Es ya específicamente las funciones a realizarse durante cada fase del sistema a desarrollarse, esto es, establecer cada una de las actividades que deben realizarze de una manera bastante calra y apegándose 100 % a los requerimientos. Se debe de tener presente que la especificaciones son la base para el desarrollo del sistema.

3) Rol de los Estándares:

Los estándares son los patrones establecidos para uniformizar todas las actividades dentro del desarrollo del sistema. Esto es el establecimiento de de cómo, bajo que condiciones se deben desarrollar cada una de las actividades. Para que exista una estandarización en los programas debe de existir un formato de codificación, como la numeración, nombres, variables, comentarios. Otro ejemplo es la estandarización de mensajes de error que para decir una misma cosa no existan diferentes mensajes de de errores.

Concluyendo este capítulo, se mostró la metodología donde cada fase tiene una actividad específica de tal manera que permitirá tener mayor control sobre cada actividad especifica que se encuentra dentro de esa fase y por lo tanto mayor control sobre lo que se hace, no se hace y lo que se debe hacer.

Esta metodologia será la base para el Programa de Aseguranza de Calidad en Software donde la calidad se va aplicar especificamente en cada fase y cada actividad de cada una las fases.

PROGRAMA DE ASEGURANZA

DE CALIDAD EN SOFTWAE

5 PROGRAMA DE ASEGURANZA DE CALIDAD EN SOFTWARE

En este capítulo se definirá lo que es y debe de ser un Programa de Aseguranza de Calidad que brinde soporte en el Desarrollo de Software.

El Software es un producto como cualquier otro el cual está compuesto por los programas y la documentación de respaldo, y a la cual debe de asociarse una característica del concepto de calidad. Esta es la razón principal por la que se propone un programa de Aseguranza de Calidad.

Un Programa de Aseguranza de Calidad en Software es el que le va a permitir al departamento o área de calidad, auditar de una manera más efectiva que todo se esté desarrollando dentro de lo ya establecido, ya sea por la empresa propia o por el usuario que en ocasiones exige ciertas condiciones. El Programa de Aseguranza de Calidad debe de tener la particularidad de adaptarse a cualquier metodología usada para el desarrollo del sistema. Para el caso del presente trabajo, el Programa de Aseguranza de Calidad que se va a desarrollar, se adaptará a la metodología mencionada ya antes en el capítulo 4.

Anteriormente se mencionó que la Aseguranza de Calidad la forman procedimientos y que así como en el Desarrollo de Sistemas existen fases que permiten llevar un control más

exacto y más detallado de lo que se está haciendo, los procedimientos dentro de la Aseguranza de la Calidad permiten llevar a cabo las actividades encaminadas a lograr la calidad esperada, y por lo tanto es lo equivalente a las fases del Desarrollo.

Ahora la pregunta es cómo aplicar los procedimientos a las fases de tal manera que se pueda asegurar de una manera confiable la calidad del sistema. Dado que una de las actividades principales en el desarrollo es controlar lo que se va haciendo, es claro que debe existir un Plan para el Desarrollo del Sistema (Software), donde se pueda ir checando el avance así como posibles desviaciones en el proceso; y también las fechas de terminación de las actividades, y productos como los programas y la documentación. A este Plan es al que se le adaptará el Programa de Aseguranza de Calidad mediante un documento llamado Plan de Evaluación de la Calidad.

Este Plan es la herramienta que ayudará a controlar cada uno de los proyectos o sistemas que se encuentren en desarrollo como se verá más delante.

Ya que las actividades más comúnes dentro de la Aseguranza de Calidad son las auditorías y revisiones, también existen otros documentos de apoyo llamados Registros de Control, que son para ir verificando a detalle lo que se revisa o audita, y a la vez que se vaya quedando constancia de los resultados.

5.1 PROCEDIMIENTOS DEL PROGRAMA DE ASEGURANZA DE CALIDAD

El Programa de Aseguranza de Calidad está compuesto de procedimientos que son los que permiten la evaluación del software y enseguida se presentará cada uno de ellos para luego definirlos a detalle y observar las implicaciones de cada uno de los procedimientos, tomando en cuenta que ningún procedimientao es más importante que otro, sino que todos forman parte de una estructura que ayudará a formar el Plan de Evaluación de la Calidad, que ya se comentó antes.

Los procedimientos que forman parte del Programa de Aseguranza de Calidad tienen que ver con las actividades del Desarrollo de Software, por lo que existe una relación entre los procedimientos y las actividades.

El Programa propuesto tiene los siguientes procedimientos para apoyarse en la implantación de las actividades encaminadas al logro de la Calidad en el desarrollo de software:

1) Diseño de Software.
2) Codificación y Prueba de programas.
3) Integración del sistema.
4) Documentación de Software.
5) Preparación para la liberación del Software.
6) Revisiones y Auditorías al Software.

7) Control de la Configuración del Software.

8) Acciones Correctivas en el Desarrollo del Software.

9) Certificación del Software.

5.2 DEFINICION DE PROCEDIMIENTOS

Es importante recalcar que los procedimientos del Programa son los que guían las acciones al estar llevando a cabo las funciones de Calidad, por lo que es en ellos donde se encuentran las directrices de lo que se debe hacer y el momento de hacerlo.

Procedimiento 1: DISEÑO DE SOFTWARE.

Este procedimiento discute las medidas necesarias para el desarrollo y liberación del diseño del software. Dentro del diseño de software está tanto el Diseño Detallado como el Diseño General. Estos procedimientos están estrechamente relacionadas con las fases mencionadas anteriormente en la Metodología, ya que en ésta se describe lo que se debe hacer y con qué se debe cumplir, por lo tanto en el Plan de Evaluación se define al encargado de la revisión de las actividades de las fases del diseño.

Dentro del Diseño General hay que asegurar que esté el flujo de información, definición de estándares y que cada una de las actividades de esta fase se estén llevando de la forma corecta y bajo las reglas o métodos propuestos. Dentro del

Diseño Detallado se debe controlar la creación de archivos, diagramas jerárquicos y todas las actividades que se deben de realizar dentro de esta fase.

La calidad del software crece cuando el software se diseña para seguir consistentemente las medidas de calidad: fácil mantenimiento, eficiencia y funcionalidad.

Procedimiento 2: CODIFICACION Y PRUEBA DE PROGRAMAS.

Este procedimiento es el que discute la manera de evaluar cada una de las actividades que se realizen dentro de la fase de codificación y prueba de programas y que todas y cada una de ellas se realizen dentro de estándares y requerimientos ya establecidos.

Estas actividades son con base en la Metodología ya propuesta anteriormente en el capitulo 4, donde se describió lo que hay que hacer y como se debe de hacer.
Este procedimiento es el encargado de auditar lo que se está haciendo y que se esté cumpliendo con el tiempo y formatos establecidos. Las actividades principales a auditar son:

a) Desarrollo del diagrama de flujo.
b) Desarrollo de Plan de Pruebas.
c) Codificación del programa.
d) Pruebas al programa terminado.
e) Resultados de las Pruebas.
f) Revisión de las Carpetas de los programas.

Este procedimiento es primordial porque asegura la "funcionalidad coherente", respecto al manejo de la información en archivos y/o bases de datos, de los programas que más tarde integrarán un sistema.

Procedimiento 3: *INTEGRACION DEL SISTEMA.*

Este procedimiento es pequeño y sólo evalúa que las pruebas de todo el sistema estén realizadas de acuerdo a los parámetros establecidos y principalmente basándose en la fase de Integracién de la metodología ya descrita.

Un punto muy importante a considerar aquí es que el haber probado cada programa individualmente puede garantizar hasta cierta parte la funcionalidad del sistema pero la consistencia general y paso de un módulo a otro también es importante probar.

Procedimiento 4: DOCUMENTACION DE SOFTWARE.

Este procedimiento marca las actividades encaminadas a evaluar la realización de la Documentación de todo el sistema en cada una de las fases.

Las medidas para asegurar la calidad de la documentación están contenidas en dos categorías:

a) La calidad interna de la documentación misma, y

b) La uniformidad de la documentación establecida por los estándares.

La documentacién es revisada para checar consistencias e integridad de la información presentada; por lo tanto se deben de asegurar los estándares también.

Una parte de la documentación, muy importante, es la llamada documentación interna y ésta corresponde a la que se encuentra dentro de cada código de cada programa, y que proporciona información sobre el proceso mismo del programa.

Respecto a las categorías mencionadas, éstas se refieren prácticamente a: la forma profesional con la que se elabore; y el seguimiento de los estándares y/o patrones marcados.

La mejor muestra de una documentación correcta se podrá encontrar en los tres manuales definidos en la Metodología: Técnico, de Operación, y del Usuario.

Procedimiento 5: PREPARACION PARA LA ENTREGA DEL SOFTWARE.

Las actividades que marca este procedimiento son relativamente sencillas dado que sólo se tiene que checar el producto terminado como tal, después de todas las revisiones y auditorías pasadas.

Los puntos a considerar son:

a) Entrega de programas en cinta o diskette, con sus debidos respaldos,

b) Manuales terminados e identificados, y

c) Especificaciones para el manejo del producto.

Procedimiento 6: REVISIONES Y AUDITORIAS AL SOFTWARE.

Este procedimiento es clave porque aquí se establece cuál va a ser la manera de auditar y revisar ya sea formal o informalmente.

Con lo anterior se deduce que existen dos tipos de auditoría y dos tipos de revisión, que orientan las actividades principales del Programa de Aseguranza de Calidad. Dichos tipos son:

Auditorías Formales e Informales, y Revisiones Formales e Informales.

Para llevar a cabo las revisiones y/o auditorías formales, se toma como herramienta el Plan de Evaluación del que se hablará mas delante con una mayor explicación.

Las revisiones y/o auditorías informales son las que se hacen sin un plan a detalle y su principal objetivo es saber cómo se está desempeñando el trabajo, si se está haciendo dentro de lo establecido, cuál es el avance, qué cambios podría haber, etc.

Toda esta información sirve para ver si el proyecto va dentro de lo planeado y se va a terminar cuando se estimó.

Las revisiones deben de hacerse con base en los requerimientos, especificaciones, y estándares establecidos.

Las personas encargadas de hacer esas revisiones tienen que estar bastante involucradas con estos puntos para que hagan las revisiones y auditorías de una forma mas precisa.

Procedimiento 7: CONFIGURACION DEL SOFTWARE.

Existen dos conceptos para manejar lo relacionado con la situación de la Configuración, hablando específicamente en términos de Software, ellos son: la Versión, y la Revisión.

Definiendo en forma sencilla y clara, se tiene que:

1) Versión: Es el número de veces que un programa se cambia o modifica cuando ya ha sido entregado al cliente y esté en un ambiente de producción. Los cambios o modificaciones se pueden deber a fallas en el programa o bien porque el usuario necesita algo diferente a lo presentado.

2) Revisión: Es el número de veces que un programa se corrige o cambia antes de ser entregado al usuario. Esto es muy importante debido a que las correcciones o cambios pueden reflejar las inspecciones de calidad o las peticiones de cambio por parte del cliente, previas a la entrega final.

Con respecto al procedimiento, resta decir que es el encargado de llevar las versiones y revisiones de los productos que se entregarán al cliente.

Esto es muy importante porque si no se lleva un control así, por ejemplo en un ambiente donde existen 300 programas y todos estén en producción, si al usuario al estar trabajando con el sistema le surge otra necesidad y pide una serie de cambios a varios programas y lo solicita a las personas que desarrollaron el sistema, éstos hacen el cambio y los envian sin llevar este control y al poco tiempo el usuario tiene otra necesidad y vuelve a solicitarlo y el encargado de hacer el cambio tiene dos versiones, no va a saber cuál es la última versión y se corre el riesgo de mandar el cambio pero con el la versión anterior y lo que funcionaba anteriormente ya no va a funcionar, pero si se lleva el control se puede evitar esto.

Este ejemplo refleja de una manera simple la situación crítica que en un momento se podría presentar al perder el control del mantenimiento.

Procedimiento 8: ACCIONES CORRECTIVAS EN SOFTWARE.

Este es el procedimiento que da seguimiento a las situaciones que se presentan cuando no se están llevando los pasos dentro de una fase, en la forma correcta, y se tiene que corregir bajo patrones ya establecidos o reglas ya establecidas. Además como apoyo se debe levantar un documento para llamar la atencién de la persona que no está haciendo su trabajo como debe de ser y esto sirve como un aviso de que

realmente se está supervisando constantemente conforme lo determina el Plan de Evaluación.

Procedimiento 9: CERTIFICACION DE SOFTWARE.

Es obvio mencionar que este procedimiento ayudará al momento de dar por terminado el producto y antes de embarcarlo y liberarlo al cliente.

La certificacién de calidad es una herramienta para asegurar que todo lo que se desarrolló esté basado en los requerimientos. La responsabilidad de la certificación es entonces la aprobación de las descripciones, autorizaciones, y el cumplimento de la terminacióm del trabajo bajo los requerimientos iniciales.

En este documento se describe lo que se va a certificar, por quién se revisará, a qué hora, en qué fecha, cuál fue la versión y revisión que se checó, etc. y todos estos puntos son importante ya que el responable al firmar estará asegurando que el producto tiene la calidad esperada.

Teniendo presente que al autorizar la salida del producto se está asegurando que la calidad es excelente, entonces si al llegar al cliente se encuentra alguna falla, el departamento de calidad es el principal responsable.

5.3 REGISTROS DE CONTROL.

En cada uno de estos procedimientos es necesario que se lleve un control lo mas preciso posible y para esto existen los registros de control que no son más que una ayuda para que no se pase ningún punto. Un registro de control es el que se conoce como "checklist" y estos son creados de acuerdo a la metodología en la que se esté basando. Estos Registros de Control son documentos donde se detalla todo lo que hay que checar. A continuación se mostrarán algunos ejemplos de registros de control:

REGISTRO DE CONTROL DEL DISEÑO DETALLADO

Actividades	Si	No	Comentarios
Existe resumen del sistema ?			
Existe la estructura del sistema detallado ?			
Están definidos los archivos ?			
Están definidos los programas que integrarán el sistema ?			
Existe diagrama jerárquico ?			
Existe diccionario de datos ?			
Existe matriz de referencia ?			
Están señaladas las rutinas comúnes ?			
Existe lista de mensajes ?			

REGISTRO DE CONTROL DEL PROCEDIMIENTO DE CODIFICACION Y PRUEBA

Actividades	Si	No	Comentarios
Están entendidas las especificaciones el 100% ?			
Está el diagrama de flujo ?			
Está el plan de prueba ?			
El código está de acuerdo al diagrama de flujo ?			
Está el código documentado ?			
Está el código estructurado ?			
Pasó el programa el plan de prueba ?			
Existe el plan de resultados ?			

5.4 PLAN DE EVALUACION DE LA CALIDAD DEL SOFTWARE

PLAN DE DESARROLLO DEL SISTEMA

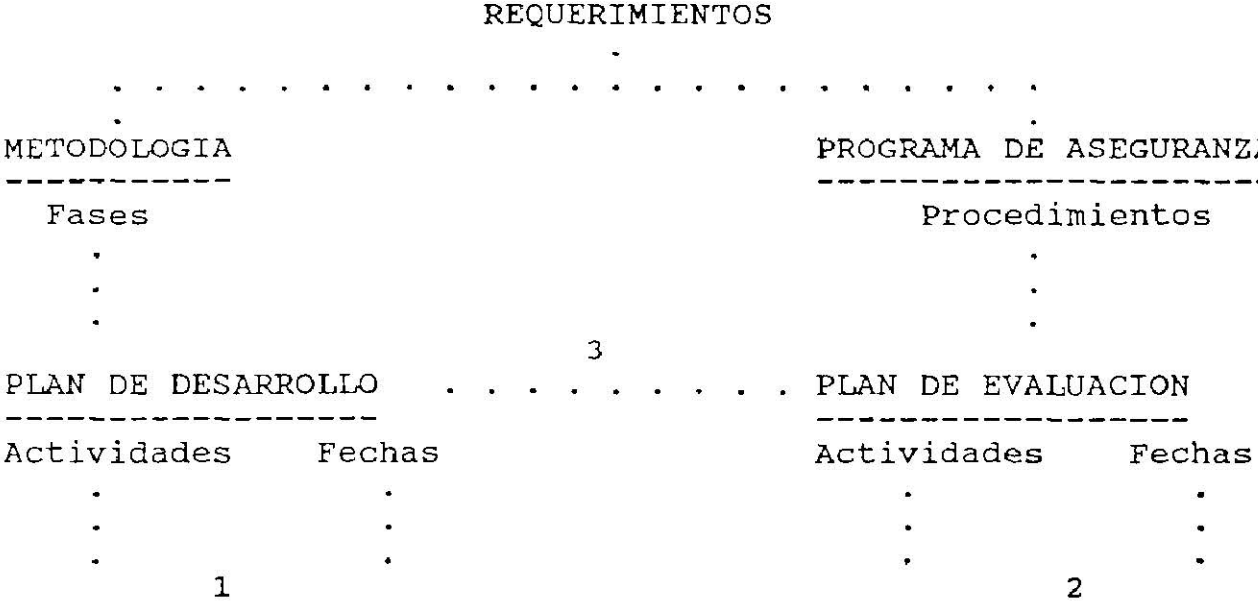
Este plan debe de ser desarrollado desde el momento que es autorizado el desarrollo. El plan es el que marca las fechas en que se termina cada actividad de cada fase, esto es un documento donde se escribe la actividad y fecha en que se terminará y este es el que va rigiendo el desarrollo del sistema.

PLAN DE EVALUACION

Este plan es el que desarrolla el departamento de calidad para evaluar las actividades de cada fase, por lo tanto este es elaborado con base en las fechas de terminación de las actividades, para realizar este plan y ver cuándo se checa formalmente.

5.5 APLICACION DEL PLAN A UN CASO PRACTICO.

La aplicación en general de la calidad dentro del área de software es una estructura como se mustra a continuación.



Los Requerimientos son nesesarios tanto para la Metodología como para el Programa de Aseguranza de Calidad.

1. Primero se debe de diseñar el Plan de Desarrollo.
2. Después el Plan de Evaluación.
3. Por último el Plan de Evaluación se aplicará al Plan de Desarrollo.

La aplicación se va a hacer sobre un Sistema de Reclutamiento de Personal que está compuesto de 3 módulos, el de mantenimiento, el de utilerías, y el módulo de reportes, pero en la aplicación sólo se utilizará el módulo de mantenimiento que consta de 20 programas, en este proyecto solo pidieron el desarrollo de la codificación y prueba, y con base en esto se va ha realizar la aplicación.

PLAN DE DESARROLLO

Programas	Actividad	Responsable	Fecha de terminacién
PPSMEN1	Entendimiento de especificaciones	L. Vásquez	12-06-89
"	Diagrama de flujo	"	12-06-89
"	Plan de prueba	"	13-06-89
"	Código	"	15-06-89
"	Pruebas	"	16-06-89
"	Plan de resultados	"	17-06-89
"	Documentación	"	19-06-89

PLAN DE EVALUACION

El registro de control que se va a utilizar es el de Codificacion y Prueba.

Programas	Actividad	Responsable	Terminación
PPSMEN1	Entendimiento de especificaciones	A. Avila	13-06-89
"	Diagrama de flujo	"	13-06-89
"	Plan de prueba	"	14-06-89
"	Código	"	16-06-89
"	Pruebas	"	19-06-89
"	Plan de resultados	"	19-06-89
"	Documentacién	"	20-06-89

REGISTRO DE CONTROL DEL PROCEDIMIENTO DE CODIFICACION Y PRUEBA

Actividades	Si	No	Comentarios
Están entendidas las especificaciones el 100% ?	X		
Está el diagrama de flujo ?	X		
Está el plan de prueba ?	X		
El código está de acuerdo al diagrama de flujo ?	X		
Está el código documentado ?	X		los comentarios deben ser más explícitos
Está el código estructurado ?	X		
Pasó el programa el plan de prueba ?	X		
Existe el plan de resultados ?	X		

Para concluir este último capitulo se mencionará que en el Programa de Aseguranza de Calidad en Software está basado en una Metodología, por lo tanto si no existe una metodología no podrá existir el Programa de Aseguranza de Calidad en Software. Ya que el Programa de Aseguranza no es más que un seguimiento planeado del las actividades que se tienen que realizar durante el desarrollo del sistema.

CONCLUSIONES

6 CONCLUSIONES.

Considero que para tener éxito en el desarrollo de un sistema es estrictamente necesario tener una Metodología de apoyo que indique las actividades necesarias a realizar para conseguir el objetivo final: un sistema que cumpla con los requerimientos del cliente; además un Programa de Aseguranza de Calidad en Software que nos garantice la confiabilidad y funcionalidad que debe tener cualquier producto de software.

Respecto a la realidad del país, se ha visto que muy pocas empresas Mexicanas que se dedican a desarrollar software, o que tienen departamentos de sistemas, donde se desarrollan los propios sistemas, no tienen definidas sus propias Metodologías de Desarrollo, y mucho menos el Programa de Aseguranza de Calidad en Software.

Para terminar, mi conclusión final es que si una empresa, de cualquier giro tiene su propia metodología o una adaptada, y se le enseña a su personal, por una parte, y por otra parte establece su Programa de Aseguranza para llevar el control de las actividades realizadas, entonces los resultados del desarrollo serán como se esperaba, y el manejo de los sistemas se verá beneficiado ya que estará estándar y fácil de entender; y en cuanto al mantenimiento posterior del sistema, es obvio asumir que se llevará de una manera simple y organizada puesto que seguirá teniendo los mismos patrones que inicialmente marcaron el desarrollo del sistema.

La opción de tener entonces un Programa de Aseguranza de Calidad en el Desarrollo de Software, si bien costará en un principio, a largo plazo beneficiará por los aspectos antes mencionados.

BIBLIOGRAFIA

Software Quality Program Organization.
Emanuel R. Baker, Ph. D.
Matthew J. Fisher, Ph. D.
Defense Systems Managment College, 1982.

Handbook of Software Quality Assurance.
G. Gordon Schulmeyer and James I. McManus.
Van Nostrand Reinhold, 1987.

Soeftware Quality Assurance
IEEE Computer Societ y Press, 1985
Chow T.S.

Sistemas de Informacion
Jhon-G. Buch Jr.
Felix R. Strater Jr.
Editorial Limusa, S.A. de C.V., 1985

La Calidad es Gratis
Philip B. Crosby
SIC

Computerized Business System
Irvine Forkner and Raymod McLeo, Jr.
Jhon Wiley & Sons, Inc., 1973

Analisis y Diseno de Sistemas de Informacion
James A. Senn
McGraw-Hill, 1987

The Systmes Development Process
Charles L. Viggs, Evan Birks, and William Atkins
Touche Ross & Co. (Prentice Hall), 1980