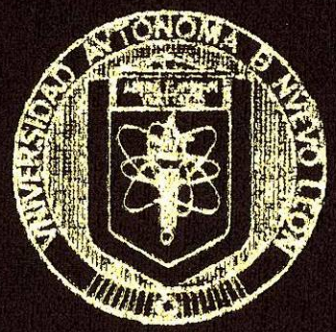
UNIVERSIDAD AUTONOMA DE NUEVO LEON

FACULTAD DE CIENCIAS FISICO MATEMATICAS



ESTIMACION DE PROYECTOS DE SOFTWARE COSTOS Y TIEMPOS

T E S I S

QUE PARA OBTENER EL TITULO DE
LICENCIADO EN CIENCIAS COMPUTACIONALES

P R E S E N T A

AIDE ESPINOSA GARZA

ASESOR: ING. AURELIO RAMIREZ GRANADOS

MONTERREY, N. L. FEBRERO DE 1989

Febrero 14; martes

1989

Pte : Ing Aurelio Ramírez

Secretario : Ing Eduardo Molina

Vocal : Ing J. Jesus De la Garza Ochoa.

Hora de inicio: 7:15 pm

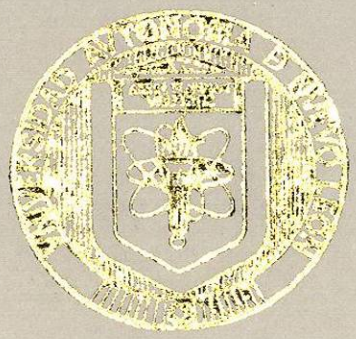
Hora de final: 8:05 pm

Conocimiento
Presentación
Exposición
Tarea

X
X
X

UNIVERSIDAD AUTONOMA DE NUEVO LEON

FACULTAD DE CIENCIAS FISICO MATEMATICAS



ESTIMACION DE PROYECTOS DE SOFTWARE COSTOS Y TIEMPOS

T E S I S

QUE PARA OBTENER EL TITULO DE
LICENCIADO EN CIENCIAS COMPUTACIONALES

P R E S E N T A

AIDE ESPINOSA GARZA

ASESOR: ING. AURELIO RAMIREZ GRANADOS

MONTERREY, N. L. FEBRERO DE 1989

DEDICATORIA

A DIOS.

Porque ha colmado el camino de mi vida de amor y de fé.

A MIS PADRES.

Margarita Garza de E.
Antonio Espinosa Uribe.

Porque me han brindado un hogar lleno de amor y armonía y me han dado el ejemplo día con día.

A MI ABUELA.

Ma. de los Santos Morales.

Porque es el recuerdo que se ha quedado como el ejemplo del camino recto, y de la esperanza de una nueva vida.

AGRADECIMIENTO

A MI ASESOR.

Ing. Aurelio Ramírez Granados.

Quien aceptó mi petición de ayuda y me guió siempre en el desarrollo de mi trabajo, dándome la oportunidad de recibir mi título profesional.

A MI JEFE Y AMIGO.

Pablo Culebro Terán.

Quien siempre me corrigió y me dió mejores ideas y quien ha sido el primer gran maestro en el camino de mi carrera profesional.

A H.T.L. COMPUTACION Y SERVICIOS.

Por brindarme simpre todas las facilidades en el desarrollo de mi trabajo.

INTRODUCCION

Los proyectos de sistemas de información que logran éxito se dirigen de manera adecuada. Por lo contrario un proyecto de sistemas de información que no tenga una base firme en una estimación de tiempo y costo obtenida como resultado de un método organizado, fracasará.

Una estimación insuficiente no cumplirá con los requerimientos establecidos y por lo tanto se tendrán desilusiones por parte de usuarios y clientes entusiastas.

El propósito principal de este trabajo es introducir al lector hacia el análisis de los métodos de estimación, mostrar la manera en que el desarrollo de la estimación de tiempo se formula junto con los programas de trabajo y cómo se utilizan para la evaluación del desempeño.

Los proyectos que se desarrollan a tiempo y por lo tanto dentro del presupuesto tienen características comunes: 1) Una estimación cuidadosamente formulada, 2) La posibilidad de la verificación del progreso por parte de la gerencia, 3) Un proceso de comparación del desempeño real y el planeado y 4) Documentación suficiente para atender los problemas cuando se presentan. Cada una de estas características será presentada en el contenido de este trabajo. Se ofrecerá además descripciones de factores que pueden afectar a las estimaciones y cuál puede ser su impacto en la calendarización proyectada.

C A P I T U L O II

II. ESTIMACION DE COSTOS DEL SOFTWARE

2.1. INTRODUCCION

2.2. FACTORES EN EL COSTO DEL SOFTWARE

2.3. TECNICAS DE ESTIMACION DE COSTO DEL SOFTWARE

2.4. ESTIMACION DE LOS COSTOS DEL MANTENIMIENTO DEL SOFTWARE

2.5. RAZONES DE FALLAS O CAUSAS DE RIESGO

C A P I T U L O III

III. DOCUMENTACION

3.1. INTRODUCCION

3.2. DOCUMENTACION DEL DISEÑO DETALLADO

IV. APENDICE

I. ESTIMACION Y ADMINISTRACION DEL TIEMPO DE DESARROLLO.

1.1. ESTIMACION DE LOS REQUERIMIENTOS DE TIEMPO.

1.1.1. INTRODUCCION.

Las estimaciones son, como su nombre sugiere, aproximaciones de horas, días ó meses de esfuerzo que se necesitan para producir el sistema deseado.

Su precisión depende de:

* Habilidad
* Conocimiento
* Experiencia del personal que prepara las estimaciones.

Sus factores determinantes son:

* Habilidad individual de analista y programadores
* Complejidad de los sistemas
* Interrupciones que no se relacionan directamente con el proyecto y que no se encuentran bajo el control del gerente del proyecto

1.1.2. METODOS DE ESTIMACION DE TIEMPOS

Existen tres métodos generales para la estimación de tiempos de desarrollo de proyectos.

* METODO HISTORICO
* METODO INTUITIVO
* FORMULA ESTANDAR

METODO HISTORICO

Está basado en registros elaborados cuidadosamente que se han mantenido de esfuerzos de desarrollos previos. Estos registros indican las características del programa o proyecto, la asignación de tareas, los requerimientos de tiempo del personal y cualquier problema o acontecimiento poco común.

Cuando se proponen nuevos proyectos, se comparan con los registros que se llevan en el archivo y se iguala lo planeado con otro trabajos anteriores para estimar el tiempo de desarrollo esperado. El mantenimiento de registros constituye un proceso que consume tiempo, por lo que muchas empresas prefieren evitarlo. Este método es tan efectivo como los registros que se tengan, e incluso es inútil sólo si el proyecto propuesto se adopta a las características de un desarrollo previo.

METODO INTUITIVO

Se dice que la experiencia es el mejor maestro. El método intuitivo se basa en la experiencia del personal más antiguo que estima, basándose en sus experiencias personales, el tiempo de desarrollo que se espera. Este método se describe en ocasiones como de predicción aprendida, siendo la parte que corresponde al aprendizaje las vivencias anteriores del individuo que elabora predicción. La diferencia del método intuitivo al método histórico, es que, los casos documentados y los registros detallados no se emplean.

FORMULA ESTANDAR

Este método proporciona un enfoque más concreto para la estimación. Los factores individuales que afectan el tiempo de desarrollo de manera más notable se identifican y se cuantifican.

FACTORES

* Características personales
* Detalles de sistemas
* Complejidad del proyecto

Una fórmula aritmética especifica cómo relacionar los elementos individuales para producir una estimación del tiempo de desarrollo en horas, días o semanas.

PARA QUE SON NECESARIOS LOS ESTIMADOS DE TIEMPO ?

* Informar a la gerencia cuándo se termina el proyecto, y fecha de puesta en producción.

* Programación del personal en las tareas.

* Ajustes gerenciales.

1.1.3. TIPOS DE SEGUIMIENTO INCLUIDOS EN LOS ESTIMADOS DE TIEMPOS

* Requerimientos de horas del proyecto
* Requerimientos de tiempo calendario

REQUERIMIENTOS DE HORAS DEL PROYECTO

Es el tiempo necesario para conducir:

1) Una investigación de sistemas

2) Formular diseño lógico

3) Codificar software

4) Preparar archivos

5) Desarrollar datos de prueba

6) Probar software

7) Hacer un pedido e instalar el equipo

REQUERIMIENTOS DE TIEMPO CALENDARIO

Es el tiempo adicional en las actividades del proyecto.

1) Reuniones de la gerencia

2) Revisiones del proyecto

3) Educación y Capacitación

4) Interacción con los usuarios

5) Tiempo de incapacidad por enfermedad

6) Vacaciones

7) Días de fiesta

1.1.4. ESTIMACION DE TIEMPOS DE ACTIVIDAD DEL SISTEMA

Estimación de una investigación de sistemas.

Se determina con:

* Número de personas a entrevistar
* Tiempo total para cuestionarios (desarrollar, circular,recibir y analizar)
* Tiempo para conducir observaciones e inspeccionar registros

Los diseños lógicos implican la creatividad del analista y requiere también del desarrollo de muchos detalles de sistemas, como definiciones de informes, organización de archivos, validación de entrada de datos y métodos de control así como procedimientos de corridas. Estos son más difíciles de estimar que la actividad de análisis.

Sin embargo, en la mayor parte de los sistemas la dificultad mayor para formular requerimientos generales del tiempo radica en la estimación para programación de software y su prueba.

El desarrollo de la lógica del programa requiere aproximadamente 35% del tiempo total de programación, asi como la prueba de datos requiere 25% del tiempo. La documentación 5% del tiempo total concedido al proyecto.

La estimación depende de tres elementos.

1) Nivel de experiencia del programador

2) Nivel de complejidad del programa

3) Nivel de comprensión del programador respecto al programa específico

La experiencia y conocimiento del programador varía mucho de persona a persona. Los proyectos de grandes dimensiones incluirán entrenadores de programación, personal experto y a veteranos con amplios antecedentes. El gerente del proyecto debe considerar el nivel de experiencia cuando asigna a las personas tareas específicas y debe estimar el tiempo que cada persona necesitará para una tarea determinada.

1.1.5. IDENTIFICACION DE VARIABLES DE DESARROLLO

Algunos gerentes de proyecto emplean un sistemas de puntuación para evaluar las habilidades de un individuo y asociarlas con los requerimientos de tiempo en un proyecto. Los elementos empleados incluyen:

1) Conocimientos del lenguaje de programación
2) Experiencia con el sistema de cómputo en que el corre el sistema
3) Experiencia de programación
4) Habilidad lógica
5) Creatividad e imaginación
6) Paciencia
7) Madurez
8) Persistencia
9) Educación

1.1.5.1. CONOCIMIENTOS DEL PROGRAMADOR

A cada individuo se le concede una puntuación, de manera caractirística de 1 a 5, basándose en los atributos individuales de cada una de las categorias descritas arriba.

Por ejemplo, un programador nuevo por lo general se evalúa con 1 ó 2 en términos de experiencia de programación, en tanto un veterano con amplios antecedentes recibirá como norma general 5.

En general, los analistas deben considerar que un programador en entrenamiento necesita tres veces más tiempo para desarrollar un programa que un programador veterano. En contraste, un programador senior con mucha y variada experiencia necesita de aproximadamente de 50-75% menos tiempo que un programador veterano.

1.1.5.2. COMPLEJIDAD DEL PROGRAMA

La complejidad de un programa es una medida del nivel de las características de sistemas, como los métodos de entrada y salida de datos, y dificultad de la lógica del programa que habrá de incluirse en el software.

Por ejemplo: Un programa que lee un solo tipo de datos de entada, actualiza un archivo secuencial e imprime una lista de cada transacción procesada no es tan complejo como un sistema en línea que acepta tipos de variables de entrada de datos de terminales múltiples. Si se incluyen varios archivos o se emplea el procesamiento distribuido, la complejidad del programa es mayor.

Las variables que integran la complejidad del programa incluyen:

* Tipos de entrada de datos
* Tipos de salida de datos
* Tipos de archivos
* Lenguajes de programación

Cada factor importante para un programa recibe una puntuación en números, siendo los más bajos para las características que implican la menor complejidad y los más altos para los que la suponen mayor. Se emplea con frecuencia una escala de 1 a 5 para asignar esas puntuaciones. La figura 1.1a muestra el uso de un sistema de puntuación para un sistema que emplea archivos secuenciales indexados e informe de errores.

FIGURA 1.1a

TIEMPO DE DESARROLLO DE PROGRAMAS UTILIZANDO LA FORMULA DE ESTIMACION

CARACTERISTICAS DEL PROGRAMA	PESO
CARACTERISTICAS DE ENTRADA DE DATOS	
INDEXADO SECUENCIAL; UN TIPO DE REGISTRO	2
ARCHIVO SECUENCIAL DE TRANSACCIONES EN CINTA MAGNETICA	1
ARCHIVO SECUENCIAL DE DATOS DE ERROR DE EXCEPCION EN DISCO MAGNETICO	1
CARACTERISTICAS DE LA SALIDA DE DATOS	
LISTADO DE SALIDA DE DATOS EN FORMATO UNICO	1
LISTADO DE TRANSACCIONES EN FORMATO UNICO	1
INFORME DE ERRORES EN FORMATO UNICO	1
COMPLEJIDAD	
LENGUAJE COBOL; COMPLEJIDAD MODERADA	17

FIGURA 1.1b

CARACTERISTICAS DEL PROGRAMADOR	
FACTOR DE EXPERIENCIA DEL PROGRAMADOR	3.5
CONOCIMIENTOS DEL TRABAJO POR PARTE DEL PROGRAMADOR	0.75
FACTOR DE PERDIDA EN LA PRODUCCION	60%
FORMULA RESULTANTE	[(17+7)*(3.5*0.75)] * 1.60
	100.8 DIAS

El conocimientos que posean los programadores del sistema o del programa que deben desarrollar puede considerarse en varios niveles (ver figuras 1.2a y 1.2b).

1) Conocimiento detallado del trabajo requerido

2) Conocimiento general del trabajo requerido

3) Conocimiento general de los temas relacionados

4) Sin conocimiento del trabajo o del conocimiento general de los temas relacionados.

Las puntuaciones posteriores se evalúan dependiendo del conocimiento para realizar el trabajo.

EJEMPLO:

Una persona con alto conocimiento para un situación de conocimiento moderado será evaluada de diferente manera a persona con alto conocimiento para una situación de alto conocimiento.

1.1.5.3. CALCULO DE LA ESTIMACION DEL TIEMPO DE PROGRAMACION

Cada programa se evalúa en forma independiente al resto del sistema. Siguiendo un enfoque cuantitativo, la complejidad del programa se multiplica por la suma de la experiencia de programador y su comprensión.

Tomando como ejemplo la figura 1.1a, las características del programa producen una puntuación que arroja un total de 7. El programa en COBOL, considerado de complejidad medio utilizando las características mostradas en la figura 1.1b, tiene una puntuación de 17.

FIGURA 1.2a

LINEAS GENERALES DE PRODUCTIVIDAD PARA EL PROGRAMADOR

NIVEL DE EXPERIENCIA

POSICION DE PROGRAMACION	EXPERIENCIA GENERAL DE PROGRAMACION	DIAS DE TRABAJO DE LA PERSONA POR PROGRAMA : PESO
PROGRAMADOR SENIOR	Experiencia en la escritura puesta en marcha de muchos programas en tipos variados de equipo. Amplia experiencia en configuraciones especificas de computadoras y en sistemas de programacion.	0.50 - 0.75
PROGRAMADOR	Experiencia en la escritura puesta en marcha de programas de complejidad variada. Experiencias con configuraciones especificas y en sistemas de programacion.	1.00 - 1.50
APRENDIZ	Ha escrito y puesto en marcha varios programas. Posee experiencia limitada con configuraciones especificas y en sistemas de programacion.	2.00 - 3.00
PERSONA EN CAPACITACION	Termino el curso de la escuela de programacion. Tiene experiencia academica escrita en los programas de capacitacion. Experiencia muy limitada.	3.50 - 4.00

FIGURA 1.2b

LINEAS GENERALES DE PRODUCTIVIDAD PARA EL PROGRAMADOR

CONOCIMIENTO DEL TRABAJO DISPONIBLE	Conocimientos necesarios del trabajo		
	ALTO	ALGUNO	NINGUNO
CONOCIMIENTO DETALLADO DE ESTA LABOR	0.75	0.25	0.00
CONOCIMIENTO GENERAL BUENO DE ESTA LABOR CON CONOCIMIENTO FRAGMENTADO EN DETALLE	1.25	0.50	0.00
CONOCIMIENTO GENERAL BUENO DE ESTE TRABAJO, PERO POCO O NINGUN CONOCIMIENTO DETALLADO	1.50	0.75	0.00
SIN CONOCIMIENTO GENERAL DE LA LABOR PERO CON CONOCIMIENTO GENERAL DE ASPECTOS RELACIONADOS.	1.75	1.00	0.25
SIN CONOCIMIENTO DEL TRABAJO Y SIN CONOCIMIENTO GENERAL DE TEMAS RELACIONADOS.	2.00	1.25	0.25

El programa requiere de un conocimiento general, pero no detallado, por parte del programador, quien se considera que está en entrenamiento; además se estima que un 60% adicional se añadirá al tiempo puro de programación debido a las reuniones en el trabajo y al tiempo perdido.

Si se utiliza la fórmula indicada en la figura 1.1b, se estiman un total de 100 días para el programa muestra.

Después de la aplicación de esta fórmula, con objeto de evitar afectar la estimación con posibles errores; se utiliza una fórmula más que convina los estimados individuales del desarrollo en el tiempo mínimo, máximo y más probable teniendo como resultado un nuevo estimado.

$$\text{ESTIMADO} = \frac{\text{EL MAS PESIMISTA} + (4 * \text{EL MAS PROBABLE}) + \text{EL MAS OPTIMISTA}}{6}$$

1.2. REQUERIMIENTOS DE TIEMPO CALENDARIO

1.2.1. INTRODUCCION.

Muchas veces sucede que dada la preocupación que se tiene en determinar el tiempo total en horas para un proyecto no se da la importancia suficiente para determinar el número de días, semanas o meses necesarios para integrar el sistema.

En la mayor parte de los proyectos de sistemas se utiliza tiempo adicional en todas sus actividades ya que estas extienden los programas más alla de los estimados previos. De hecho, en los proyectos de sistemas grandes, el tiempo adicional necesario para estas actividades amplía el tiempo total de desarrollo de 50 a 100%.

Tomando en cuenta estos datos podríamos decir que para un proyecto con 250 días estimados de programación, se llevará a cabo en realidad de 375 a 500 días de programación. Un día de programación (algunas veces denominado día persona) se utiliza para medir el número de días que un proyecto requerirá basándose en la cantidad de trabajo que una persona puede realizar en un día.

Se considera que la cantidad de personas en un proyecto afecta el tiempo calendario, pero no de manera proporcional.

Por ejemplo el añadir una persona más al proyecto reduciría el tiempo total calendario. La explicación que podemos encontrar a este factor es que el tiempo destinado a presentación de status

del proyecto, comunicación con el cliente, explicaciones y entrenamiento al usuario quedarían a cargo de un número pequeño de personas en el equipo y que el resto del equipo podría dedicarse solc a las actividades del sistema como son diseño, programación, pruebas, etc.

1.2.2. METODOS PARA PLANEAR REQUERIMIENTOS DE TIEMPO CALENDARIO

En general existen tres métodos para planear los requerimientos de tiempo calendario:

* Gráficas de barras
* Gráficas de referencia
* Gráficas pert

1.2.2.1 GRAFICAS DE BARRAS

Es el tipo de planeación que utiliza barras para mostrar cada actividad de un proyecto de sistemas y el tiempo que requerirá.

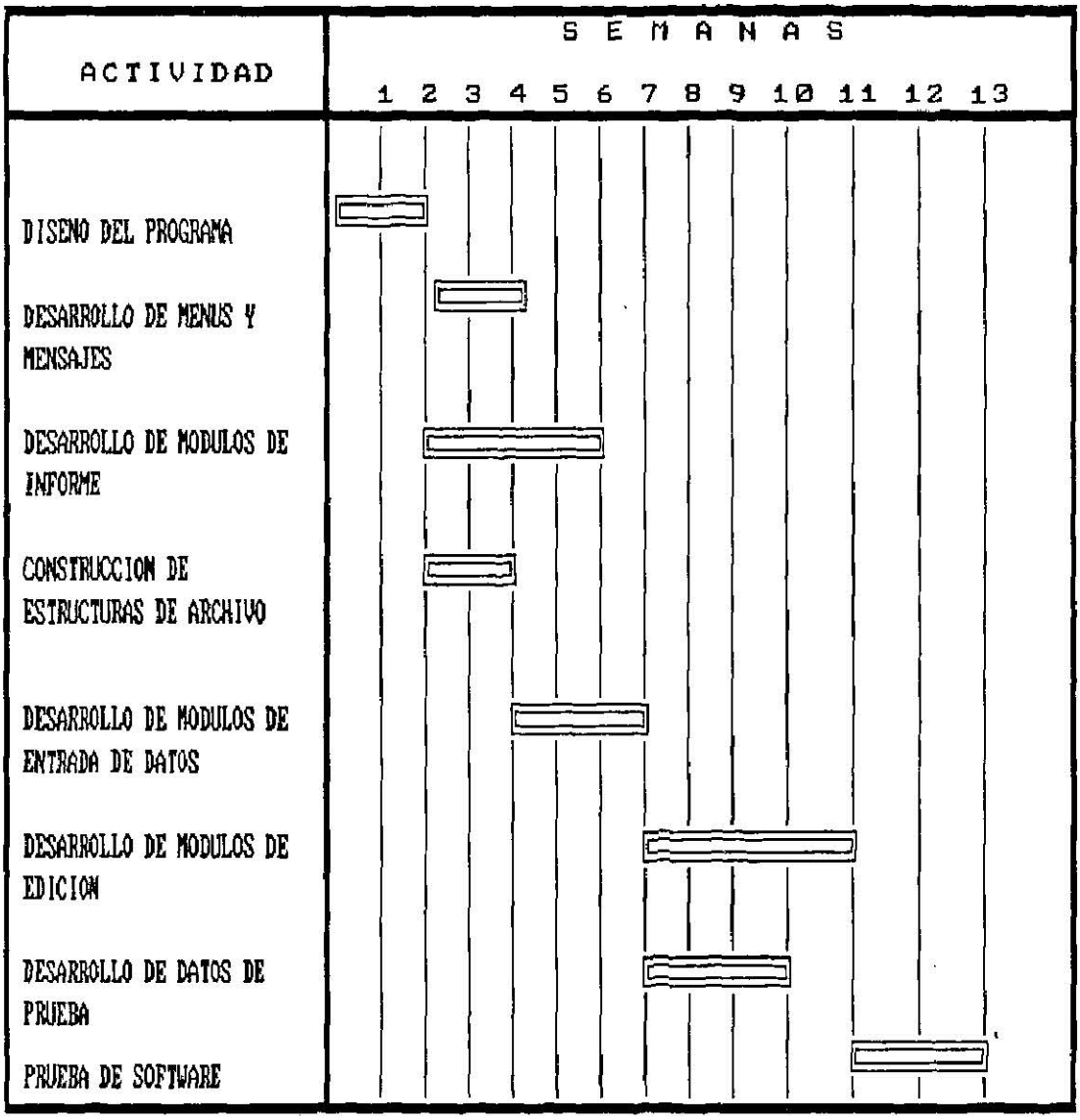
PROCEDIMIENTO

1) El análista identifica cada tarea y estima el tiempo para desempeñarla.
2) La información se transfiere a la gráfica de barras listando las tareas de arriba a abajo en la parte izquierda de la gráfica, en el orden en que se llevarán a cabo (ver figura 1.3).
3) El tiempo de calendario se muestra de izquierda a deracha.
4) Una barra horizontal se marca en la gráfica para cada tarea, indicando cuándo da principio y cuándo se espera que termine.

Este método es aplicable sólo cuando el proyecto consiste de un número pequeño de actividades, de otra manera la gráfica crece de manera desproporcionada e incluye tantas barras que es difícil utilizar la información.

FIGURA 1.3

PLAN DE DESARROLLO DE UN SISTEMA DE INFORMACION MEDIANTE UNA GRAFICA DE BARRAS.



Con frecuencia se utilizan varios niveles para comunicar la información relativa a la planeación.

* Una gráfica general de planeación que muestra las actividades generales.
* Un programador líder deseará una gráfica de las actividades concernientes a su proyecto.
* Un programador una para las actividades de su programa.

De esta manera el grado de fineza con que se muestran los detalles con que se muestran las tareas en la gráfica de barras dependerá de la manera en que la gráfica se utilize.

1.2.2.2 GRAFICAS DE PUNTOS DE REFERENCIA

Un punto de referencia es aquel momento en que el proyecto se encuentra en una situación crítica para su desarrollo. Representan etapas difíciles que deben pasarse o tareas críticas que es necesario completar en tiempo.

Las gráficas de puntos de referencia muestran los acontecimientos significativos al completar un proyecto y la secuencia con que se deben terminar. Difieren de las gráficas de barras porque representan puntos en que algo se completa y no tareas individuales que se deben llevar a cabo; por ejemplo, al completar el inicio del proyecto los puntos de referencia de un sistema incluyen la llegada del equipo, instalación, terminación de la capacitación de los usuarios, conversión de archivos y el cambio al nuevo sistema.

Los puntos de referencia son también un factor contra el cual se puede medir el progreso del proyecto y de esta manera el gerente ó jefe del proyecto puede mantenerse a tiempo.

DESVENTAJAS

* Demasiada importancia al tiempo y no a la interdependencia entre las tareas y acontecimientos de control de los costos del proyecto.

* Los puntos de referencia deben ser significativos y mensurables.

* Puntos de referencia definidos como una parte de una actividad (codificación al 50%, entrevistas al 50%, analistas familiarizados con el diseño lógico).

1.2.2.3. GRAFICAS PERT

El método más complejo para la planeación es el PERT (Proyect evaluation and Review Technique). Este método, desarrollado en 1958 por la armada de Estados Unidos y Brooze, Allen y Hamilton, una compañía consultora de gerencia se ha utilizado en muchos proyectos complejos que requieren una planeación y administración cuidadosa.

Aunque el mejor enfoque para el manejo de proyectos consiste en dividir el poyecto en partes pequeñas y manejables, existe el peligro de perder la visión global de todo el proyecto mientras se supervisan las tareas menores. Las actividades del proyecto son por lo general independientes; sin embargo, la interdependencia de las tareas no aparecen en métodos como gráficas de barras o de puntos de referencia. Las tareas críticas, las que se deben completar a tiempo y en una secuencia específica, tampoco son evidentes.

INFORMACION PROPORCIONADA POR PERT

* Indica actividades individuales
* Tiempo necesario para una actividad
* Interrelación de actividades
* Secuencia adecuada
* Estimaciones de tiempo
* Separación de áreas específicas donde los problemas potenciales o los retrasos pueden ocurrir.
* Método de verificación de progreso
* Efecto de retraso de cierta actividad a todo el proyecto
* Actividad con menor tolerancia de retraso

COMPOSICION DE UNA GRAFICA PERT

En la gráfica 1.4a se utilizan círculos (nodos) y caminos rectas o arcos) para representar la interrelación de las actividades del proyecto. Los nodos representan acontecimientos y los caminos muestran las actividades que se requieren para adelantarse de un acontecimiento al otro. Los números en paréntesis indican el tiempo necesario para llevar a cabo cada actividad.

La gráfica PERT tiene gran valor cuando un proyecto se planea y se diseña. Cuando la gráfica se ha terminado se estudia para determinar la ruta crítica, es decir, el camino que es necesario llevar desde el principio hasta el fin y por el cual el tiempo total requerido será mayor que para cualquier otro camino (se muestra en la figura 1.4b).

FIGURA 1.4a

DESARROLLO DE SISTEMAS DE INFORMACION (PERT)

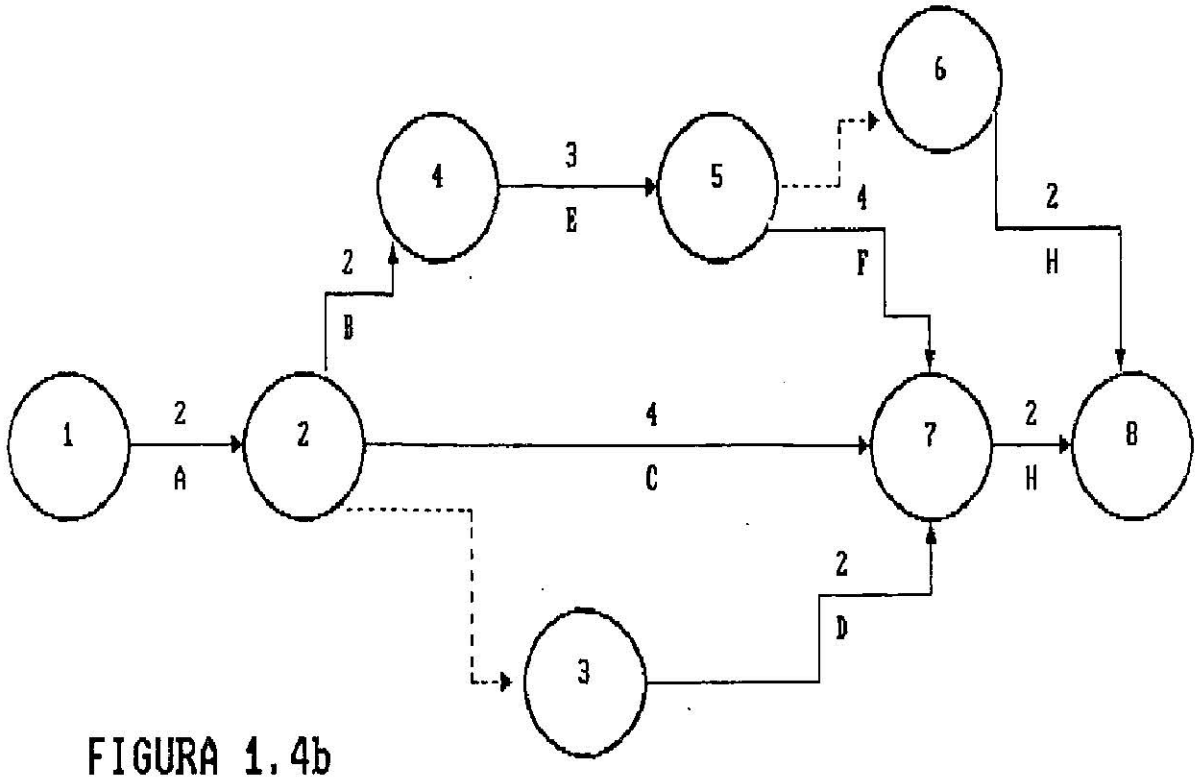
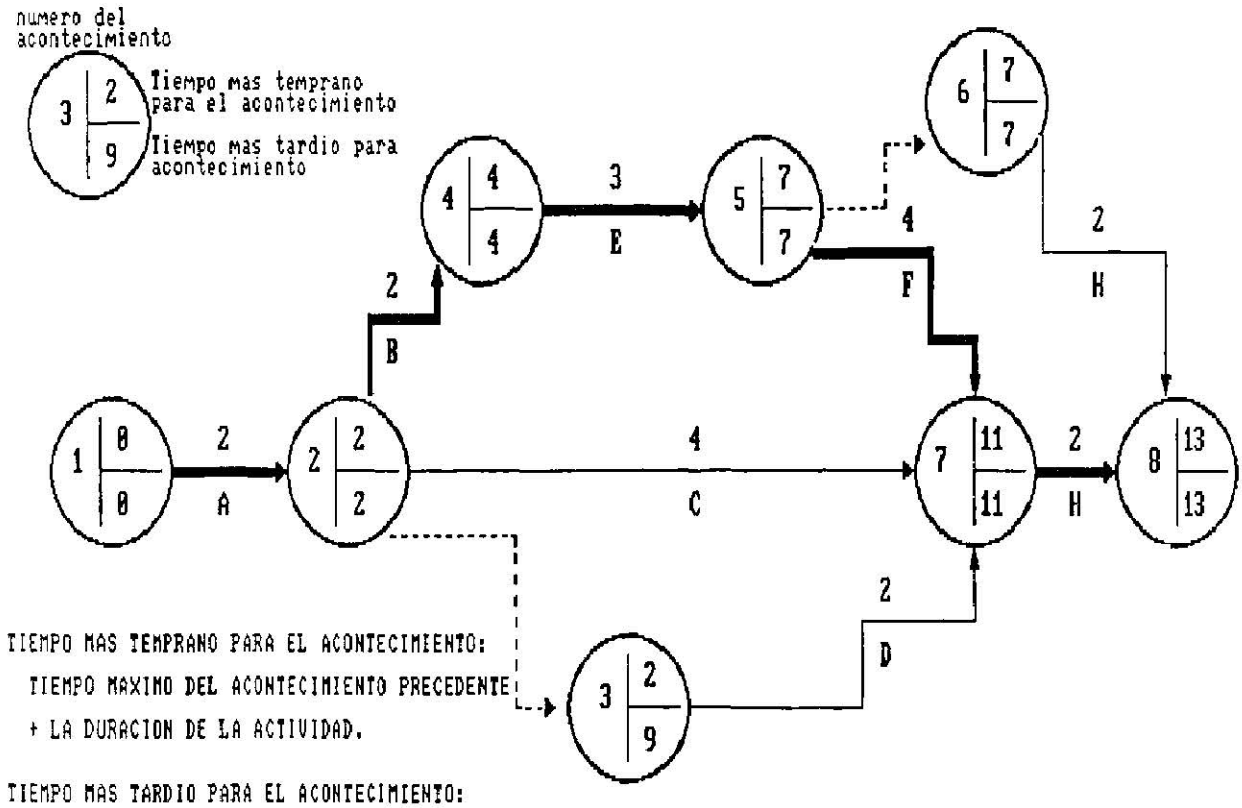


FIGURA 1.4b

TIEMPOS DE ACTIVIDAD EN LA RED (PERT)



TIEMPO MAS TEMPRANO PARA EL ACONTECIMIENTO:
TIEMPO MAXIMO DEL ACONTECIMIENTO PRECEDENTE
+ LA DURACION DE LA ACTIVIDAD.

TIEMPO MAS TARDIO PARA EL ACONTECIMIENTO:
TIEMPO MINIMO DEL ACONTECIMIENTO SIGUIENTE
- LA DURACION DE LA ACTIVIDAD.

Si las actividades que se encuentran en la ruta crítica no se terminan a tiempo, todo el proyecto se retrasará; por lo tanto la gerencia del proyecto deberá poner especial atención a estas actividades.

PROCEDIMIENTO

1) Identificación de tareas

2) Estimación de tiempos asociados a cada tarea

3) Identificación de secuencia de actividades

4) Identificación de tareas que pueden ocurrir simultáneamente con otras

El ejemplo en la figura 1.4a muestra esta información: las rectas representan actividades de A a H y el número en cada línea constituye el número de semanas que el gerente del proyecto espera que llevará esa actividad (es decir, la duración de la actividad). Las extensiones de las líneas no tienen significado. En este ejemplo, los acontecimientos (eventos) 2-4, 2-7 y 2-3 no se pueden iniciar antes de que la tarea A se halla terminado, como se indica por la ubicación de los acontecimientos de la red. La línea punteada entre los acontecimientos 2 y 3 tiene significado especial, no se ha asignado ningún tiempo a la línea y no posee una identificación de tarea (una letra o un nombre). Cuando existe una dependencia entre dos acontecimientos pero no se consumen tiempos o recursos, se denomina actividad nula. Las actividades nulas pueden utilizarse para conectar actividades paralelas.

El siguiente paso consiste en analizar el programa de tiempos del proyecto. Se desea saber:

1) Qué tan pronto puede iniciarse el acontecimiento

2) Cuánto puede tardarse en iniciar éste sin afectar el programa general del proyecto.

El tiempo más temprano para el acontecimiento (denominado TMT) es cero para el primer acontecimiento. Para los otros es la suma más alta de la duración de la actividad y el TMT de cualquier acontecimiento inmediatamente anterior.

Por ejemplo, cuatro actividades preceden de inmediato al acontecimiento 7. El TMT se calcula mediante el análisis del TMT y el tiempo de duración para cada acontecimiento que le precede.

ACTIVIDAD	TMT PRECEDENTE	DURACION	TMT CALCULADO
3-7	2	3	5
2-7	2	4	6
5-7	7	4	11
6-7	7	3	10

Ya que el acontecimiento 7 no puede iniciarse hasta que todos los acontecimientos precedentes se hayan completado, lo que interesa es el factor de tiempo mayor que en este caso es 11.

El tiempo más tardío del acontecimiento es el tiempo límite en el cual éste puede iniciar sin retrasar el proyecto.

Para determinar este tiempo es necesario trabajar respectivamente en la red iniciando desde la derecha. El tiempo más tardío constituye la diferencia menor entre el TMT del acontecimiento terminal menos el tiempo de duración de la actividad.

Por ejemplo, para determinar el TMT para el acontecimiento 7, el TMT del nodo siguiente, el número 8 en la figura, se ha designado como 13. Si se resta el tiempo de duración, el tiempo de acontecimiento máximo se determina como 13-2 = 11. El número en la parte derecha inferior en el nodo 7 se muestra como 11 (ver figura 1.4b).

Para determinar el tiempo máximo de inicio para el acontecimiento 2, los cálculos son:

ACTIVIDAD	TMT SIGUIENTE	DURACION	TMT CALCULADO
2-4	4	2	2
2-7	11	4	7

Ya que el TMT más pequeño es 2, ese tiempo se convierte en el TMT para el acontecimiento 2.

RUTA CRITICA

La ruta crítica es el conjunto de actividades que el gerente debe supervisar más cercanamente, ésto es porque en ella se identifican los acontecimientos que se deben iniciar y terminar en tiempo y que requieren no más que el tiempo de duración estimado; de otra manera todo el proyecto sufrirá un retraso.

En la figura 1.4b la ruta crítica se ilustra con líneas más gruesas. Se determinó al unir todos los nodos en donde los TMT y TMt son iguales, lo que significa que no hay posibilidades de cambio o de desviación, es decir, no hay tiempos de tolerancia u holgura.

TIEMPO DE HOLGURA

El tiempo de holgura se relaciona con un acontecimiento que se puede explicar formalmente restando el tiempo de duración y el TMT del nodo inicial del TMT del nodo terminal.

Por ejemplo, para el acontecimiento 3-7, el tiempo de holgura es de 7 semanas (11-2-2=7), lo que significa que la actividad 3-7 puede iniciarse en cualquier momento de la segunda hasta la novena semana y, si el tiempo de la segunda hasta la novena semana y, si el tiempo de duración de 2 semanas se mantiene, el proyecto estará a tiempo.

1.3. VERIFICACION DEL PROGRESO POR PARTE DE LA GERENCIA

1.3.1. INTRODUCCION.

La verificación del progreso es un proceso para asignar, medir, evaluar, y reorientar el comportamiento de un proyecto.

Este proceso incluye tres elementos básicos:

* Calendarización a corto plazo.
* Asignación del trabajo.
* Evaluación e información del grado de avance del proyecto.

1.3.2. CALENDARIZACION A CORTO PLAZO

Antes de iniciarse cada actividad dentro de un proyecto, particularmente en un proyecto grande, deberá:

1) Revisar todas y cada una de las actividades anteriores
2) Asegurar que:
   a) Cada actividad anterior sa haya concluido
   b) El personal involucrado se encuentra disponible, según lo planeado.
3) Reafirmar:
   a) Alcance
   b) Objetivos
   c) Presupuesto

4) Si existen cambios en el plan del proyecto o en las variaciones planeadas éstas deberán ser:

a) Documentadas

b) Aprobadas

Bajo este punto, la documentación deberá incluir un cambio en especificaciones para continuar con todo el ciclo de desarrollo, una vez que tales especificaciones sean aprobadas.

1.3.3. ASIGNACION DEL TRABAJO

Las asignaciones del trabajo son extensiones de la calendarización a corto plazo.

Cómo se hacen?

1) Conforme el personal involucrado concluye la ejecución de sus tareas anteriores.

2) Una vez revisada cada actividad, se identifica al personal como disponible.

Qué incluye la documentación de la asignación de trabajo?

1) Descripciones totales de todas las tareas que van a efectuarse.

2) Aprobación y revisión de documentación por parte del personal asignado a este trabajo.

3) Estimación de tiempo (cuánto le llevará hacerlo).

4) Especificación exacta de qué es lo que se quiere y cómo se quiere y debe lograr.

5) Plan de evaluación a lo que se está proponiendo. (Plan de prueba).

1.3.4. EVALUACION E INFORMACION DEL GRADO DE AVANCE DEL PROYECTO

La información sobre los proyectos de desarrollo de sistemas deberán concentrarse principalmente:

* Avance con respecto al plan
* Análisis de variaciones

La información sobre la terminación de todas las subtareas, tareas y actividades deberán estar disponibles en un resumen, en base a los distintos niveles, de acuerdo a lo aprobado en el plan del proyecto.

Los reportes sobre el grado de avance deberán elaborarse de acuerdo a dos categorías principales:

* Por unidad de trabajo
* Por persona

Las figuras 1.5, 1.6 y 1.7 muestran esta situación.

Los reportes sobre la terminación de los módulos de trabajo deberán resumir:

1) Tiempo estimado y tiempo real a la fecha

2) Variaciones a nivel de subtarea, tarea y actividad de acuerdo a su tipo:

   a) Con respecto a la planeación, reunirán cuando los planes y estimaciones originales hayan sido inadecuados.

   b) Con respecto a la actuación, representan el trabajo que lleva más o menos tiempo en horas-hombre o en el tiempo transcurrido, que el indicado el los papeles originales.

   c) Con respecto al calendario, cubren retrasos o adelantos en el nivel del trabajo, comparándolo con lo planeado.

1.3.4.1. REVISION DEL GRADO DE AVANCE

Los reportes sobre el grado de avance de un proyecto deberán ser revisados periódicamente por los usuarios y por otros responsables de la aprobación del proyecto. Estas sesiones de revisión deberán cubrir el avance, los problemas y las alternativas de solución.

FIGURA 1.5

DEFINICION DE TRABAJO - A NIVEL DE PROYECTO

FORMA DE DESCRIPCION DE TRABAJO

FECHA __1/15/___

NIVEL DEL TRABAJO : _____ LISTA DEL PROYECTO ☐ PAGINA _1_DE_1_

LISTA DE ACTIVIDAD ☐ CODIGO DEL TRABAJO ___3_

ORGANIZACION : _____ LISTA DE TAREA ☐ CODIGO DEL PROYECTO ____

CODIGO DE ACTIVIDAD ____

No	DEFINICION DEL TRABAJO	RESPONSABILIDAD	TIPO DE PERSONAL	DIAS HOMBRE REQUERIDOS	PRESUPUESTO	FECHA DE INICIACION	FECHA DE TERMINACION PROGRAMADA
001	Planeacion de la organizacion y del proyecto	ABC x TR & Co.	-----	30			
002	Definicion del sistema de distribucion y ventas actual	ABC x TR & Co.	-----	30			
003	Sistema de facturacion	ABC x TR & Co.	-----	270			
004	Sistemas de cuentas por cobrar	ABC x TR & Co.	-----	233			
005	Administracion y control de inventarios	ABC x TR & Co.	-----	500			
006	Sistema de cuentas por pagar	ABC x TR & Co.	-----	125			

☐ SOLO NIVEL DE TAREA

PREPARADO POR : _____

APROBADO POR : _____

FIGURA 1.6

DEFINICION DE TRABAJO - A NIVEL DE TAREA

FORMA DE DESCRIPCION DE TRABAJO

FECHA ___/_/___

NIVEL DEL TRABAJO : _____ LISTA DEL PROYECTO [] PAGINA _1_DE_1_

LISTA DE ACTIVIDAD [] CODIGO DEL TRABAJO ___3_

ORGANIZACION : _____ LISTA DE TAREA [] CODIGO DEL PROYECTO _04_

CODIGO DE ACTIVIDAD _002_

No	DEFINICION DEL TRABAJO	RESPONSABILIDAD	TIPO DE PERSONAL	DIAS HOMBRE REQUERIDOS	PRESUPUESTO	FECHA DE INICIACION	FECHA DE TERMINACION PROGRAMADA
001	Disenar formatos de salida	ABC x TR & Co.	Analista	05		5/20/-	5/24/-
002	Definir archivos de informacion	ABC x TR & Co.	Analista	05		5/20/-	5/24/-
003	Definir formatos de entrada	ABC x TR & Co.	Analista	05		5/27/-	5/31/-
004	Preparar diagrama de flujo del proceso / corrida a corrida	ABC x TR & Co.	Analista o Programador	05		5/27/-	5/31/-
005	Definir funciones de corrida	ABC x TR & Co.	Analista	02		5/29/-	5/31/-
006	Establecer controles de entrada / salida y proceso	ABC x TR & Co.	Analista	04		5/31/-	6/06/-
007	Seleccionar lenguajes de programacion para la corrida	ABC x TR & Co.	Programador Encargado/ director y	01		6/06/-	6/06/-
008	Identificar nuevos procedimientos manuales	ABC x TR & Co.	Analista	03		6/06/-	6/10/-
009	Finalizar costos de operacion y ahorros del sistema	ABC x TR & Co.	Encargado del proyecto	02		6/06/-	6/07/-
010	Revisar el sistema con el personal de auditoria	ABC x TR & Co.	Encargado del proyecto	02		6/10/-	6/11/-
011	Revisar el sistema con la gerencia de ABC	ABC x TR & Co.	Encargado del proyecto y Analista	02		6/12/-	6/13/-
	Total de dias-hombre para esta actividad			36		5/20/-	6/13/-

[ ] SOLO NIVEL DE TAREA

PREPARADO POR : _____

APROBADO POR : _____

FIGURA 1.7

DEFINICION DE TRABAJO - A NIVEL DE ACTIVIDAD

FORMA DE DESCRIPCION DE TRABAJO

FECHA __1/20/___

NIVEL DEL TRABAJO : _____ LISTA DEL PROYECTO ☐ PAGINA _1_DE_1_

LISTA DE ACTIVIDAD ☐ CODIGO DEL TRABAJO ___3_

ORGANIZACION : _____ LISTA DE TAREA ☐ CODIGO DEL PROYECTO _04_

CODIGO DE ACTIVIDAD _002_

No	DEFINICION DEL TRABAJO	RESPONSABILIDAD	TIPO DE PERSONAL	DIAS HOMBRE REQUERIDOS	PRESUPUESTO	FECHA DE INICIACION	FECHA DE TERMINACION PROGRAMADA
001	Definir los requirimientos de sistema del nuevo sistema de C P/C	ABC x TR & Co.	-----	21			
002	Disenar el nuevo sistema de C P/C	ABC x TR & Co.	-----	36			
003	Preparar las especificaciones de programacion	ABC x TR & Co.	-----	20			
004	Disenar la logica de la programacion	ABC x TR & Co.	-----	20			
005	Escribir los programas de C P/C	ABC x TR & Co.	-----	60			
006	Efectuar la prueba del sistema de C P/C	ABC x TR & Co.	-----	10			
007	Planeacion de la conversion	ABC x TR & Co.	-----	10			
008	Preparar los nuevos procedimientos manuales	ABC x TR & Co.	-----	20			
009	Establecimiento y conversion de archivos	ABC x TR & Co.	-----	10			
010	Planear e implementar el programa de entrenamiento	ABC x TR & Co.	-----	10			
011	Conversion del sistema piloto de C P/C	ABC x TR & Co.	-----	06			
012	Conversion del sistema de C P/C	ABC x TR & Co.	-----	10			
	Total de dias-hombre para este proyecto			233			

☐ SOLO NIVEL DE TAREA

PREPARADO POR : _____

APROBADO POR : _____

1.3.4.2. REPORTE SOBRE LAS HORAS UTILIZADAS

Una técnica comunmente usada que ha demostrado ser apropiada para el control de proyectos es reportar sobre una base de horas utilizadas. Por lo general, en los proyectos de desarrollo de sistemas se asignan números especificos de horas a las unidades de trabajo.

Estas horas se consideran utilizadas unicamente cuando la unidad de trabajo se termina, se revisa y se acepta.

Puede darse el caso de que se haya invertido tiempo en una actividad que no se ha concluido. Dado que no habrá horas utilizadas sino hasta su terminación, las horas reales invertidas hasta el momento en que se hayan concluido serán variaciones.

Un ejemplo de documento de trabajo para registrar y apreciar los logros y las variaciones es el mostrado en la figura 1.5, 1.6 1.7 y 1.8.

Cada tarea de una actividad se lista en terminos de horas de trabajo planeadas, las horas utilizadas y las reales.

En el ejemplo que se presenta en la figura 1.8, el tiempo real invertido en la ejecución del trabajo se anota por semana. Además, existe una columna utilizada para reportar el trabajo "pendiente a realizar". Este puede ser un punto importante cuando se proyecta variaciones significativas.

El añadir la estimación o el trabajo pendiente a la ejecución real proporciona una cifra acumulada para las variaciones proyectadas para cada actividad, hasta el nivel de unidad de trabajo.

FIGURA 1.8

CONTROL DE HORAS UTILIZADAS

PROYECTO :
CLIENTE :
ANALISTA : FECHA:
PROGRAMADOR :

NO. CUENTA	PROGRAMA	RCVD/LIBR FECHA	FECHA TERMINA	ESTIMADO\ACTUAL HORAS POR PASOS					HORAS DOC	HORAS x TOTALES	STATUS TERM.	P/F/E	COMENTARIOS
				DISEÑO	EXT\DIS	INT\DIS	CODIGO	PRUEBAS					
				E									
				Ajs									
				A									

P - PROCESSO
F - FINALIZADO
E - ESPERA

Cuando las unidades de trabajo se completen, las unidades reales trabajadas podrán ser más o menos iguales al número de horas que se había estimado para la actividad. Sin embargo las grandes variaciones de horas estimadas y reales indicarán problemas, ya sea reales o potenciales.

Las condiciones que reflejan la grandes variaciones entre las horas estimadas y horas reales pueden significar que la ejecución real está llevando más tiempo que el planeado, o que se ha invertido una cantidad significativa de tiempo en unidades de trabajo que se iniciaron y no se concluyeron, o que han aparecido tiempos improductivos.

II. ESTIMACION DE COSTOS DEL SOFTWARE

2.1. INTRODUCCION.

RELACIONES DE CAMBIOS EN LOS COSTOS

En 1960, la relación de gastos en equipo y programas era aproximadamente de 80% dedicado al equipo y 20% a la programación; para 1980 la razón cambió a 20% para el equipo y 80% para la programación.

Para 1990 se espera que la programación utilice cerca del 90% de estos gastos.

En este 90% de costos por desarrollo, podemos considerar que el costo del mantenimiento representa buena cantidad del esfuerzo dedicado a la programación.

Quizás algunas de las razones para un cambio tan radical, serían los transistores, los circuitos integrados que han bajado drásticamente los precios de los equipos.

Por otro lado, la labor de programación exige gran cantidad de horas-hombre y los costos de personal se han incrementado debido a que con el transcurso del tiempo se acumulan más paquetes de programación.

La estimación de costos de un producto de programación es una de las tareas más difíciles y erráticas durante la fase de planeación de un desarrollo debido a la gran cantidad de factores desconocidos en ese momento.

Reconociendo este problema, algunas organizaciones utilizan una serie de estimadores de costo; se prepara un estudio preeliminar durante la fase de planeación y se presenta en la revisión de la factibilidad del proyecto.

La estimación mejorada se presenta en la revisión de requisitos de programación.

La estimación es un refinamiento obtenido como resultado de la actividades de trabajo desarrolladas adicionalmente; algunas veces, varias opciones de producto con sus respectivos costos, se exhiben en las revisiones; lo anterior permite que el cliente escoja una solución adecuada dentro de las posibles.

2.2. FACTORES EN EL COSTO DEL SOFTWARE

Existen muchos factores que influyen en el costo de un producto de programación. El efecto de estos factores es difícil de estimar y, por ende también lo es el costo del esfuerzo en el desarrollo o en el mantenimiento.

2.2.1. CAPACITACION DEL PROGRAMADOR

Existen dos aspectos en la capacidad.

1) Competencia general básica

2) Familiaridad con el area particular de aplicación

Se entiende por "Competencia general básica" a la capacidad básica para escribir programas de computadora correctos.

En proyectos muy grandes y extremadamente grandes, el número de programadores es tan amplio, que las diferancias individuales en la productividad tienden a equilibrar el promedio; empero, los módulos desarrollados por un programador débil pueden exhibir pobre calidad y retrasarse en su entrega.

2.2.2. COMPLEJIDAD DEL PRODUCTO

Existen tres categorías para los productos de programación:

* PROGRAMAS DE APLICACION
* PROGRAMAS DE APOYO
* PROGRAMAS DE SISTEMAS

PROGRAMAS DE APLICACION

En ellos se incluyen procesamientos de datos y programas científicos por lo común se desarrollan bajo el ambiente proporcionado por un compilador como FORTRAN o PASCAL; las interacciones con el sistema operativo se limitan a las instrucciones de control del trabajo y al llamado a las facilidades del lenguaje durante el tiempo de ejecución.

PROGRAMAS DE APOYO

Son los compiladores, ligeradores y sistemas de inventarios, se escriben con el fin de permitir al usuario ambientes de programación complicando el empleo del sistema operativo.

PROGRAMAS DE SISTEMAS

Sistemas de bases de datos, sistemas operativos y sistemas de tiempo real, interactúan directamente con el equipo; estos suelen utilizar un proceso concurrente y trabajan bajo ciertas limitaciones de tiempo de ejecución.

Los programas de apoyo son tres veces más difíciles de construir que los de aplicación, y los de sistemas que son a su vez, tres veces más difíciles que los de apoyo; los niveles para la complejidad en su clasificación son entonces 1-3-9 para aplicaciones, apoyo y sistemas respectivamente.

Estableciendo tres niveles de complejidad de un producto y ecuaciones para predecir el esfuerzo total en meses programador requerido en su desarrollo (PM), considerando como variable independiente al número de millares de instrucciones de código fuente entregadas con el producto (KDSI) de tal forma, los costos de programación de un producto pueden obtenerse con la multiplicacion del esfuerzo requerido en términos de meses de de programador por el costo unitario del mes programador. Estas ecuaciones fueron obtenidas a través del análisis de datos en registros históricos.

Programa de aplicación : PM = 2.4 * (KDSI) ** 1.05

Programa de apoyo : PM = 3.0 * (KDSI) ** 1.12

Programa de sistemas : PM = 3.6 * (KDSI) ** 1.20

Así, estas ecuaciones dicen que se requiere casi el doble de esfuerzo para desarrollar un programa de apoyo que uno de aplicación y tres veces más en el caso de un programa de sistemas.

El tiempo de desarrollo se calcula:

Programa de aplicación	:	TDEV = 2.5 * (PM) ** 0.38
Programa de apoyo	:	TDEV = 2.5 * (PM) ** 0.35
Programa de sistema	:	TDEV = 2.5 * (PM) ** 0.32

El tiempo de desarrollo es muy parecido para los tres tipos de productos. De esto se deduce que existe más oportunidades para desarrollar actividades de tipo paralelo en el desarrollo de un programa de apoyo o de sistemas, que en el desarrollo de un programa de aplicación del mismo tamaño; o quizás que la calendarización de actividades se realiza en forma diferente con los proyectos de aplicación.

Dado el número total de meses programador de un proyecto y el tiempo nominal de desarrollo requeridos, el nivel promedio de contratación puede obtenerse mediante una simple división. Para el caso de 60 KDSI, se obtiene los siguientes resultados:

Programas de aplicación:		176.6 PM / 17.85 MO = 9.9 Prog
Programas de apoyo	:	294 PM / 18.3 MO = 16 Prog
Programas de sistemas	:	489.6 PM / 18.1 MO = 2.7 Prog

La variable independiente en estas ecuaciones es el número de instrucciones finales de código, asi las estimaciones son tan buenas como la capacidad de aproximar dicho número.

Uno de los errores comunes en la estimación de líneas de código fuente de un producto de programación es subestimar la cantidad de código de servicio requerido; este código es la parte del código fuente que permite el manejo de entradas y salidas, comunicación interactiva con el usuario, interface de comunicación humana y la determinación y manejo de errores. Algunas veces el código de servicio llega a ser más del 50% e incluso hasta el 90% del código de servicio necesario; de modo que las estimaciones basadas en términos del código de cómputo suelen ser engañosas cuando se usan para estimar el código final del programa.

2.2.3 TAMAÑO DEL PRODUCTO

El tamaño del producto es un factor importante que determina el nivel del control administrativo y el tipo de herramientas y técnicas necesarias en un proyecto de programación. Las categorías que a continuación se mencionan indican el tamaño de un producto.

CATEGORIA	# DE PROGRAMADORES	DURACION	TAMAÑO DEL PRODUCTO
Trivial	1	1-4 semanas	500 líneas
Pequeno	1	1-6 semanas	1k-2k
Mediano	2-5	1-2 años	5k-50k
Grande	5-20	2-3 años	50k-100k
Muy grande	100-1000	4-5 años	1M
Extremadamente Grande	2000-5000	5-10 años	1M-10M

PROYECTOS TRIVIALES

Estos proyectos, casi no se requiere el desarrollo de análisis formal, documentación del diseño muy elaborada, pruebas piloto extensivas, ni documentación de apoyo clara; sin embargo, aún programas triviales pueden mejorarse mediante cierto tipo de análisis, un diseño sistemático, una programación estructurada y un mecánismo metódico de prubas.

PROYECTOS PEQUEÑOS

Los programas pequeños por lo regular no tienen interacciones con otros programas; ejemplos de estos programas son paquetes de aplicación científica escritos por ingenieros para resolver problemas numéricos, algunos paquetes comerciales escrito por personal de procesamiento de datos para efectuar tareas clásicas de manejo de datos y generación de informes y, por último, proyectos estudiantiles escritos en los cursos de compiladores y sistemas operativos.

PROYECTOS MEDIANOS

Un proyecto mediano requiere de dos a cinco programadores que trabajen durante uno o dos años en la generación de 10,000 a 50,000 líneas de código, y entre 250 a 1000 rutinas; estos programas tienen poca interacción con otros programas.

Entre estos proyectos se puede considerar ensambladores, compiladores, sistemas pequeños de manejo de información, sistemas de inventarios y aplicaciones de control de procesos.

PROYECTOS GRANDES

Un proyecto grande necesita de 5 a 20 programadores que trabajen durante dos o tres años para generar un producto de 50,000 a 100,000 líneas de código mediante varios subsistemas; un proyecto grande comunmente tendrá interacciones significativas con otros programas y sistemas de programación.

Como ejemplo de proyectos grandes se consideran a compiladores de gran tamaño, sistemas pequeños de tiempo compartido, paquetes de base de datos, sistemas gráficos para la adquisición y despliegue de información y proyectos para el control en tiempo real. En estos proyectos suelen surgir problemas de comunicación entre los diversos programadores, gerentes y usuarios. Para su desarrollo puede ser necesario contar con más de un grupo de programadores por ejemplo, tener 3 grupos de 5 programadores, y también tener más de un nivel en la gerencia del proyecto. Además, existe gran posibilidad de alta rotación del personal asignado al proyecto durante su desarrollo; lo anterior requiere del entrenamiento y la adoctrinación del personal nuevo, o bien, la distribución de las responsabilidades del miembro que se retira, entre los demás integrantes del grupo de trabajo.

El tamaño y complejidad de un proyecto grande dificulta, si no imposibilita, la prevención de eventualidades durante las fases de planeación y análisis; tanto el cliente como los constructores pueden modificar los requisitos del proyecto conforme este avanza. Durante un proyecto de esta magnitud, son esenciales los procedimientos sistematizados, la documentación estandard y las revisiones formales.

PROYECTOS MUY GRANDES

Un proyecto muy grande se realiza con la ayuda de 100 a 1000 programadores, durante un período de 4 a 5 años, con un resultado de cerca de 1,000,000 de líneas de código fuente y, normalmente, con varios subsistemas, cada uno de los cuales es un proyecto grande. Los subsistemas suelen tener interacciones complejas entre ellos y con otros sistemas desarrollados aparte.

Por lo general, estos proyectos comprenden sistemas de procesamiento en tiempo real, sistemas de telecomunicaciones y multiusuarios; entre ellos se pueden mencionar grandes sistemas operativos, grandes bases de datos, sistemas de control y dirección militar.

PROYECTOS EXTREMADAMENTE GRANDES

Un proyecto de esta magnitud suele incluir de 2000 a 5000 programadores durante 10 años, quienes generan entre 1,000,000 a 10,000,000 de líneas de código. Los proyectos extremadamente grandes constan de varios subsistemas de gran tamaño, que comprenden conceptos de tiempo real, telecomunicaciones, multitareas y procesamiento distribuido; estos sistemas tienen, a menudo, requisitos de confiabilidad muy altos e incurren en procesos de vida y muerte.

Algunos ejemplos de estos sistemas son el control del tráfico aéreo, sistemas de proyectiles de defensa y sistemas de control y comando militar.

2.2.4. TIEMPO DISPONIBLE

El esfuerzo total del proyecto se relaciona con el calendario de trabajo asignado para la terminación del proyecto, varios investigadores han estudiado la cuestión del tiempo óptimo de desarrollo, y la mayoría concuerdan en que los proyectos de programación requieren más esfuerzos si el tiempo de desarrollo se reduce o incrementa más de su valor óptimo.

2.2.5. NIVEL DE CONFIABILIDAD REQUERIDO

La confiabilidad de un producto de programación puede definirse como la probabilidad de que un programa desempeñe una función requerida bajo ciertas condiciones especificadas y durante cierto tiempo.

La confiabilidad puede expresarse en términos de exactitud, firmeza, cobertura y consistencia del código fuente. Las caracteristicas de la confiabilidad pueden instrumentarse en un producto de programación, pero existe un costo asociado con el aumento del nivel de análisis, diseño, instrumentación y esfuerzo de verificación y validación que deben optarse para asegurar alta confiabilidad.

El nivel de confiabilidad deseado debe establecerse durante la fase de planeación al considerar el costo de las fallas del programa; en algunos casos, las fallas pueden causar al usuario pequeñas inconveniencias, mientras en otros tipos de productos puede generarse gran pérdida financiera e incluso poner una vida en peligro.

Factores multipicadores de esfuerzo para ajustes por confiabilidad

Categoria	Consecuencias de la falla	Factor
Muy baja	Alguna molestia menor	0.75
Baja	Las pérdidas son fáciles de recuperar	0.88
Nominal	Dificultad relativa en la recuperación	1.00
Alta	Gran pérdida financiera	1.15
Muy alta	Riesgo de una vida	1.40

2.2.6. NIVEL TECNOLOGICO

El nivel de tecnología empleado en un producto de programación se refleja en el lenguaje utilizado, la máquina abstracta (tanto el equipo como los programas de apoyo), las prácticas y las herramientas de programación utilizadas. Se sabe que el número de líneas de código fuente escritas por día es, por completo, independiente del lenguaje ocupado, y que las proporciones escritas en un lenguaje de alto nivel, en vez de un ensamblador aumenta la productividad por un factor de 5 a 10; además, las reglas de verificación de tipo de datos, los aspectos de autodocumentación de estos lenguajes mejoran la confiabilidad y la capacidad de modificación de los programas. Los lenguajes modernos de programación como el ADA brindan características adicionales para mejorar la productividad y confiabilidad del producto de programación entre estas características están la verificación fuerte de tipo de datos, la abstracción de datos, la compilación separada, el manejo de excepciones y de interrupciones, asi como los mecanismos de concurrencia.

La abstracción de la máquina en cuestión se refiere al conjunto de facilidades del equipo y de los paquetes del sistema utilizados durante un proceso de desarrollo. La familiaridad, estabilidad y facilidad de acceso a dicha abstración influyen en la productividad del programador y, por ende, el costo del proyecto. La productividad se afectará si los programadores tienen que aprender a usar un nuevo ambiente de programación como parte del proceso de desarrollo del producto, o si se planea la máquina paralelamente con los programas o también, de igual forma, si los programadores tienen un acceso restringido a la máquina.

Las prácticas modernas de programación comprenden el uso del análisis y diseño sistemático, de notaciones estructuradas de diseño, recorridos e inspecciones, de la programación estructurada, de programas básicos. Las herramientas de programación van desde las herramientas más elementales como ensambladores y depuradores, hasta compiladores y ligeradores, incluyendo editores interactivos de texto y sistemas de base de datos, asi como también lenguajes procesadores de diseño, analizadores de requisitos de programación, ambientes totales de desarrollo que incluyen, herramientas de manejo de configuraciones y verificación automática.

2.3. TECNICAS DE ESTIMACION DE COSTO DEL SOFTWARE

2.3.1. INTRODUCCION.

Dentro de la mayor parte de las organizaciones, la estimación de costos de la programación se basa en las experiencias pasadas. Lo cual significa que los datos de costos y productividad de los proyectos actuales deben ser centralizados y almacenados para un empleo posterior.

La estimación de costos puede llevarse a cabo en forma jerárquica hacia abajo o en forma jerárquica hacia arriba.

La estimación jerárquica hacia abajo se enfoca primero a los costos del nivel del sistema, asi como a los costos de manejo de configuración, del control de calidad, de la integración del sistema, del entrenamiento y de las publicaciones de la documentación. Los costos del personal relacionado se estiman mediante el examen del costo de proyectos anteriores que resulten similares.

En la estimación jerárquica hacia arriba, primero se estima el costo del desarrollo de cada módulo o subsistemas; tales costos se integran para obtener un costo total. Esta técnica tiene la vantaja de enfocarse directamente a los costos del sistema, pero se corre el riesgo de despreciar diversos factores técnicos relacionados con algunos módulos que se desarrollarán. La técnica subraya los costos asociados con el desarrollo de un sistema, aunque puede fallar al no considerar los costos del manejo de la configuración o del control de calidad.

2.3.2. JUICIO EXPERTO

La técnica más utilizada para la estimación de costos es el uso del juicio experto, que además es una técnica del tipo jerárquico hacia abajo. El juicio experto se basa en la experiencia, en el conocimiento anterior y en el sentido comercial de uno o más individuos dentro de la organización.

La mayor ventaja del juicio experto, que es la experiencia, puede llegar a ser su debilidad; el experto puede confiarse de que el proyecto sea similar al anterior; pero bien puede suceder que haya olvidado algunos factorès que ocasiona que el sistema nuevo sea significativamente diferente; o quizás el experto que realiza la estimación no tenga experiencia en ese tipo de proyecto.

Para compensar tales factores, los grupos de expertos algunas veces tratan de llegar a un consenso en el estimado; lo anterior tiende a minimizar las fallas individuales y la falta de familiaridad en proyectos particulares neutralizando las tendencias personales y el deseo de ganar un contrato a través de una estimación optimista.

2.3.3. ESTIMACION DEL COSTO POR LA TECNICA DELFI

La técnica DELFI fue desarrollada en la coorporación Rand en 1948, con el fin de obtener un consenso de un grupo de expertos sin contar con los efectos negativos de las reuniones de grupo. La técnica puede adaptarse a la estimación de costo de la siguiente manera:

1.- Un coordinador entregará a cada experto (integrante del grupo) la documentación con la definición del sistema en cuestión y una papeleta para que escriba su estimación.

2.- Cada uno de los expertos estudiará esta definición del sistema y determinará su estimación en una forma anónima.
Los expertos podrán consultar con el coordinador pero no entre ellos mismos.

3.- Posteriormente, el coordinador realizará y presentará un resumen de las estimaciones presentadas e incluirá en este, cualquier aportación extraña, idea o comentario adicional de los expertos.

4.- Se realizá una segunda ronda de estimaciones (también en forma anónima) utilizando los resultados de las primeras estimaciones. Si alguna de estas estimaciones diverge mucho de las demás, se le pedirá al estimador, exponga sus razones también en forma anónima.

5.- La ronda de estimaciones se repite tantas veces como se juzgue necesario, impidiendo las discordias del grupo y con el fin de unificar criterios.

Otro enfoque de esta técnica que aumenta la comunicación conservando el anonimato:

1.- El coordinador entregará a cada experto (integrante del grupo) la documentación con la definición del sistema en cuestión y una papaleta para que escriba la estimación.

2.- Cada experto estudiará la definición, y el coordinador llama a reunión para analizar los aspectos de la estimación con él y entre ellos mismos.

3.- Cada experto terminará su estimación, entregándola de una forma anónima.

4.- El resumen de las estimaciones es preparado por el coordinador pero no incluirá los razonamientos anexos de los expertos.

5.- Se prepará otra reunión para discutir y unificar las diferencias derivadas de las estimaciones.

6.- Por último se realiza otra ronda de estimaciones, también en forma anónima, la cual se repetirá tantas veces como sea necesario.

Si después de varias rondas de estimaciones, las variaciones de éstas son muy marcadas, el coordinador procederá a realizar investigaciones adicionales para recabar información con el objeto de unificar criterios.

2.3.4. ESTRUCTURAS DE DIVISION DEL TRABAJO

La estructura de división de trabajo es un método del tipo jerárquico hacia arriba.

Una estructura de división de trabajo es un organigrama jerárquico donde se establecen las diferentes partes de un sistema.

Una estructura de división de trabajo refleja una jerarquía de productos o bien de procesos.

UTILIZACION DE LA TECNICA PARA LA ESTIMACION DE COSTOS

Los costos se estiman mediante la asignación del costo a cada componente individual en el organigrama y luego la suma de todos.

VENTAJAS PRIMORDIALES

Identificación y contabilización de los diversos procesos y factores de productos de sistema.

Aclaración con exactitud de que costos se incluyen en la estimación.

OBJETIVO

Alcanzar un nivel en el que las unidades de trabajo lleguen a ser lo suficientemente predecibles en:

- Asignación a los individuos
- Fácil manejo
- Estimaciones de costo asociado

PUNTOS A TENER EN CUENTA AL DIVIDIR EL TRABAJO

Las definiciones de trabajo deberán hacerse de acuerdo con los niveles de experiencia para las tareas individuales.

Cada unidad de trabajo debera tener productos finales tangibles que puedan revisarse y evaluarse.

Pueden requerirse subtareas para dividir las tareas grandes en unidades de trabajo que requieran un conocimiento básico y una duración manejable, generalmente no excediendo dos semanas-hombre.

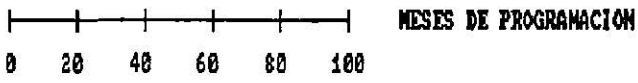
FIGURA 2.1

PAPELETA PARA LA ESTIMACION MEDIANTE LA TECNICA DELFI

PROYECTO : SISTEMA OPERATIVO FECHA : 06/06/84

ESTIMACION EN LA 3A VUELTA

SU ESTIMACION ES:



SU ESTIMACION PARA LA SIGUIENTE VUELTA ES DE: 35 PM.

RAZONES PARA SU ESTIMACION:

PARECE UN SISTEMA NORMAL DE CONTROL DE PROCESOS.

NUESTRA GENTE HA TENIDO GRAN EXPERIENCIA EN PROYECTOS SIMILARES.

NO SE ESPERA QUE HAYA PROBLEMAS CON ESTE PROYECTO.

2.3.5 MODELOS DE COSTOS POR ALGORITMOS O MODULOS (COCOMO MOdelos COnstructivos de COstos)

En los modelos de costos basados en algoritomos o modelos, los costos se estiman mediante la adición del los costos de cada uno de los módulos o subsistemas que conforman al sistemas, de modo que esta técnica es del tipo jerárquica hacia arriba.

Este método toma como base:

* Las ecuaciones presentadas en sección 2.2.2.
* Factores multiplicadores para ajustar la estimación de acuerdo:
    a) Atributos del producto
    b) Computadora a utilizar
    c) Personal dedicado al proyecto

Las ecuaciones incorporan algunas suposiciones importantes; por ejemplo, las ecuaciones para la carga nominal orgánica (programas de aplicaciones) deben usarse en las siguientes situaciones:

* Desde programas pequeños hasta medianos (2k-32k DSI)
* En una área de aplicación conocida
* Una máquina virtual estable y bien conocida

Cuando estas suposiciones van a alterarse para adecuarse a las caracteristicas del nuevo proyecto, los factores multiplicadores deberan utilizarse.

Las estimaciones mediante algoritmos cubren las siguientes actividades:

- Del diseño a pruebas de aceptación
- Costos de documentación y revisiones
- Costos del gerente del proyecto y bibliotecario de programas.

Costos incluidos en COCOMO

- Proposiciones de control de trabajo
- Código fuente
- Se considera 152 horas de programador igual a un mes programador.

Costos no incluidos en COCOMO

- Planeación
- Análisis
- Instalación
- Entrenamiento
- Secretarias, personal de limpieza, operadores de equipo de cómputo.
- Comentarios en el código fuente (documentación del código fuente)
- Rutinas de apoyo

SUPOSICIONES PARA EL PROYECTO EN ESTIMACIONES COCOMO

1) Número pequeño de personas efectúan la definición y validación de los requisitos.

2) Los requisitos permanecen constantes durante el desarrollo del proyecto.

3) Número pequeño de personas efectúan la definición y validación del diseño arquitectónico.

4) Grupos paralelos de programadores realizan, desarrollan el diseño detallado, codificación y pruebas por unidad.

5) Pruebas de integración se eleboran de acuerdo a una planeación al respecto.

6) Los errores en las interfaces se encuentran frecuentemente:

    a) recorridos através del código
    b) pruebas de integración
    c) pruebas por unidad

7) La documentación se genera como parte de proceso de desarrollo.

EJEMPLO (ver figura 2.2 y 2.3).

Proyecto: Proceso de telecomunicaciones en un microprocesador de tipo comercial

Tamaño estimado: 10 KDSI

De acuerdo a las ecuaciones:

$$PM = 2.8\ (10)\ **\ 1.20 = 44.4$$

$$TDEV = 2.5\ (44)\ **\ 0.32 = 8.4$$

AJUSTES A LAS ESTIMACIONES:

+ El proyecto es altamente complejo

- Analistas y programadores altamente calificados.

Un factor de ajuste igual 1.17 (ver figura 2.3) se obtiene de estos multiplicadores con el cual se produce una estimación de 51.9 meses de programador y 8.8 meses de tiempo de desarrollo.

Suponiendo $ 6,000 dls por mes-hombre

dólares = (51.9 PM) * (6000 por PM) = $ 311,400

FIGURA 2.2

FACTORES MULTIPLICADORES DE ESFUERZO EN COCOMO

FACTOR MULTIPLICADOR	INTERVALO DE LOS VALORES
ATRIBUTOS DEL PRODUCTO	
CONFIABILIDAD REQUERIDA	0.75 A 1.40
TAMAÑO DE LA BASE DE DATOS	0.94 A 1.16
COMPLEJIDAD DEL PRODUCTO	0.70 A 1.65
CARACTERISTICAS DE LA MAQUINA	
LIMITANTES EN EL TIEMPO DE EJECUCION	1.00 A 1.66
LIMITANTES EN LA MEMORIA PRINCIPAL	1.00 A 1.56
VOLATILIDAD DE LA VIRTUALIDAD EN LA MAQUINA	0.87 A 1.30
TIEMPO DE ENTEGA DE LOS PROGRAMAS	0.87 A 1.15
CARACTERISTICAS DEL PERSONAL	
CAPACIDAD DE LOS ANALISTAS	1.46 A 0.71
CAPACIDAD DE LOS PROGRAMADORES	1.42 A 0.70
EXPERIENCIA EN PROGRAMAS DE APLICACION	1.29 A 0.82
EXPERIENCIA EN MAQUINAS VIRTUALES	1.21 A 0.90
EXPERIENCIA EN LENGUAJES DE PROGRAMACION	1.14 A 0.95
CARACTERISTICAS DEL PROYECTO	
USO DE TECNICAS MODERNAS DE PROGRAMACION	1.24 A 0.82
USO DE HERRAMIENTAS DE PROGRAMACION	1.24 A 0.83
TIEMPO REQUERIDO PARA EL DESARROLLO	1.23 A 1.10

FIGURA 2.3

FACTORES MULTIPLICADORES DEL ESFUERZO

FACTOR MULTIPLICADOR	RAZONAMIENTO	VALOR
CONFIABILIDAD	USO LOCAL SIN PROBLEMAS DE RECUPERACION	1.00
BASE DE DATOS	20,000 BYTES (PEQUENA)	0.94
COMPLEJIDAD	PAQUETE DE TELECOMUNICACIONES (MUY COMPLEJO)	1.30
TIEMPO	ATADO AL CPU EN UN 70% (ALTO)	1.11
MEMORIA	45K DE LOS 64K DISPONIBLES (ALTO)	1.06
MAQUINA	ESTABLE, PRODUCTO COMERCIAL COMUN	1.00
TIEMPO DE RETORNO	PROMEDIO DE DOS HORAS (NOMINALES)	1.00
ANALISTAS	GENTE TITULAR DE CALIDAD (ALTO)	0.86
PROGRAMADORES	GENTE TITULAR DE CALIDAD (ALTO)	0.86
EXPERIENCIA	TRES ANOS EN LAS TELECOMUNICACIONES (NOMINAL)	1.00
EXPERIENCIA	SEIS MESES CON LA MICROCOMPUTADORA (BAJO)	1.10
EXPERIENCIA	DOCE MESES EN EL LENGUAJE (NOMINAL)	1.00
PRACTICAS	MAS DE UN ANO CON TECNICAS MODERNAS	0.91
HERRAMIENTAS	SISTEMAS BASICOS PARA UNA MICRO (BAJO)	1.10
CALENDARIO	NUEVE MESES, 8.4 ESTIMADO (NOMINAL)	1.00

Factor de Ajuste del esfuerzo = 1.17

VENTAJAS

1) Una visión intuitiva de los factores de costos dentro de la organización.

2) Los datos se pueden recolectar y analizar

3) Identificación de nuevos factores

4) Los factores multiplicadores de esfuerzo pueden ajustarse tanto como sea necesario para calibrar el COCOMO dentro del ambiente específico.

DESVENTAJAS

1) El uso de los factores multiplicadores parte de la suposición de que son independientes entre sí.

2) La modificación de un factor puede variar otro pero en ocasiones el porqué de la variación no suele ser clara.

PROCEDIMIENTO

1.- Identificar: Subsistemas y Módulos.

2.- a) Estimar el tamaño de cada módulo

b) Calcular el tamaño de cada subsistema

c) Calcular el tamaño total del sistema

3.- Especificar los factores multiplicadores para cada módulo.

4.- Calcular el esfuerzo y tiempo de desarrollo para cada módulo (para este punto utilizar las fórmulas vistas).

5.- Especificar los once multiplicadores restantes para cada módulo.

6.- Efectuar un análisis de sensibilidad sobre la estimación, estableciendo comparaciones para diversos casos.

7.- Sumar los costos de planeación y análisis que no se hallan incluido.

8.- Hacer una comparación con otra estimación obtenida a partir de la técnica DELFI, identificando y corrigiendo diferencias.

2.4. ESTIMACION DE LOS COSTOS DEL MANTENIMIENTO DEL SOFTWARE

2.4.1. INTRODUCCION.

El período de vida de un producto de programación es de tres años en su desarrollo y de cinco a quince en su uso, incluyendo el mantenimiento.

La distribución del esfuerzo entre desarrollo y mantenimiento se ha informado como a razón de 40/60, 30/70 e incluso de 10/90.

Mantenimiento contempla tres tipos:

1) Mejoramiento de las capacidades del producto
2) Adaptación del producto a nuevos ambientes de cómputo
3) Depuración de errores

La distribución de esfuerzo resulta pues de la siguiente manera

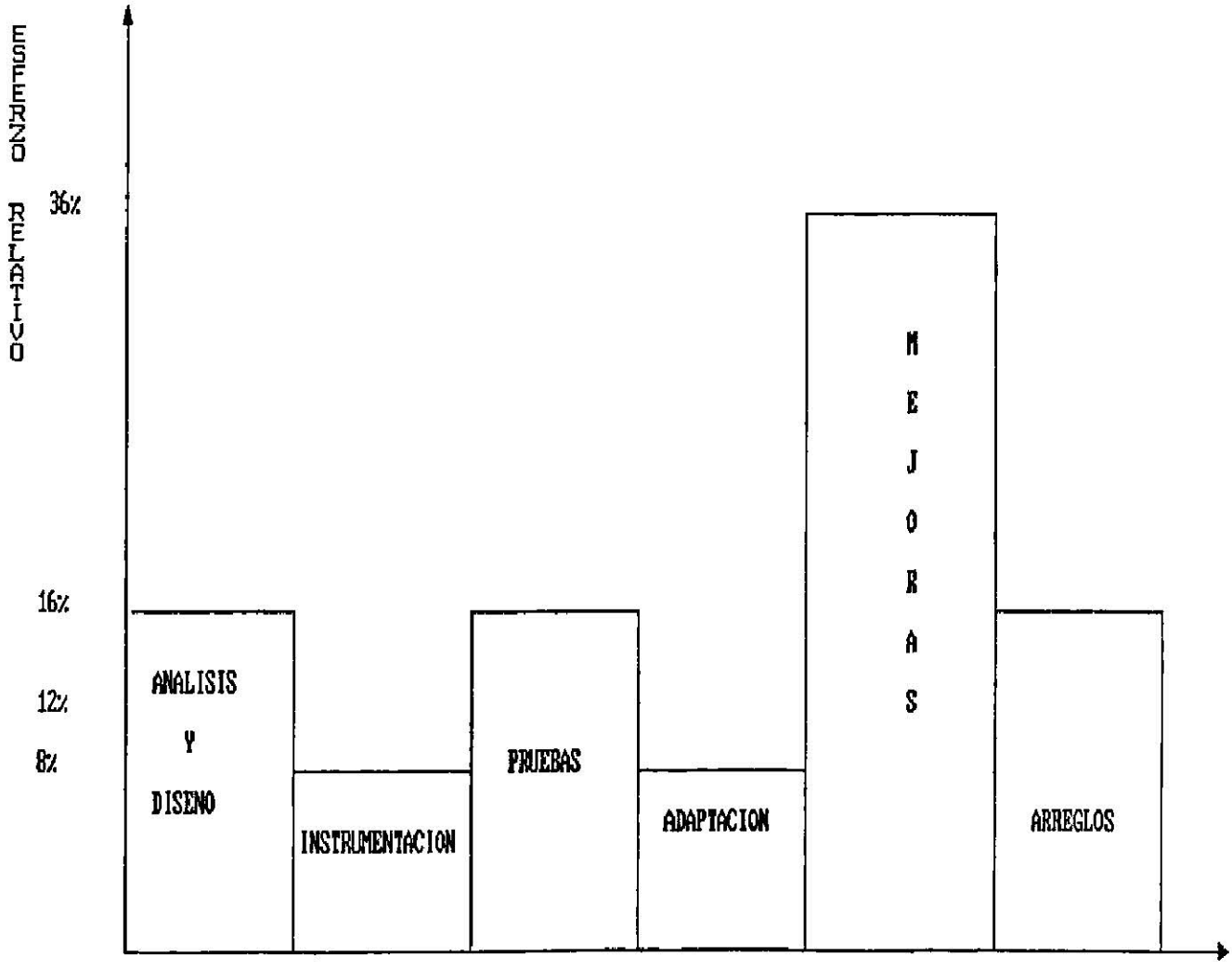
1) 60%
2) 20%
3) 20%

Durante la fase de desarrollo de un producto la distribución del esfuerzo resulta:

40 análisis y diseño
20 instrumentación, depuración y pruebas
40 integración y pruebas de aceptación

Si la distibución global del esfuerzo entre desarrollo y mantenimiento se considera como 40/60 se obtiene el diagrama de barras en la figura 2.3.

FIGURA 2.4



ACTIVIDADES EN EL CICLO DE DESARROLLO

HECHOS DESPRENDIDOS DE LA GRAFICA 2.4.

1) Las actividades de mantenimiento gastan más recursos que las actividades de desarrollo.

2) El mantenimiento se disminuye a través de un largo período, por lo tanto, el esfuerzo por unidad de tiempo es menor.

3) Un gran procentaje se dedica a mejoras de productos

4) Las actividades de pruebas consumen cerca del 50% del tiempo dedicado al desarrollo del producto.

5) 3/4 del esfuerzo total son dedicados a pruebas, mejoras y adaptación del producto.

6) La importancia del análisis, diseño e instrumentación es aligerar la carga de prueba y mantenimiento del producto.

2.4.2. MEJORAMIENTO DEL MANTENIMIENTO DURANTE EL DESARROLLO

Muchas actividades realizadas durante el desarrollo del software mejoran el mantenimiento del producto.

* Actividades de análisis

  - Desarrollo de estándares.
  - Fijar logros en los documentos de apoyo.
  - Especificar procedimientos de control de calidad.
  - Identificar probables mejoras.
  - Determinar recursos requeridos para el mantenimiento.
  - Estimar costos de mantenimiento.

* Actividades de diseño arquitectónico

  - Subrayar la claridad y modularidad como criterios de diseño.
  - Diseñar para facilitar probables mejoras.
  - Usar notaciones estandarizadas para documentar flujos de datos, abstracción de datos y descomposición jerárquica hacia abajo.

* Actividades de diseño detallado

  - Uso de notaciones estandarizadas, para especificar algoritmos estructuras de datos y procedimientos para especificar interfaces.

  - Proporcinar directorios con referencia cruzada.

* Actividades de implementación

  - Usar estructuras de una sola entrada y una sola salida.

  - Un estilo de codificación simple y clara.

  - Una indentación estándar en las estructuras.

  - Usar constantes simbólicas para asignar parámetros a las rutinas.

  - Proporcinar documentación estándar en cada subrutina.

* Otras Actividades

  - Desarrollar guías de mantenimiento

  - Desarrollar casos de pruebas

  - Documentación de casos de pruebas

2.4.3. ESTIMACION DEL COSTO DEL MANTENIMIENTO

El mantenimiento del software suele necesitar de 40 a 60% y en algunos casos hasta el 90% del esfuerzo total durante el ciclo de vida del proyecto.

Una regla útil muy usada para la distribución del esfuerzo de las actividades de mantenimiento es asignar 60% al tiempo de mejoras y 20% al de adaptación o corrección de problemas.

El nivel de esfuerzo característico dedicado al mantenimiento de los programas es de aproximadamente 50% del tiempo total del ciclo de vida del proyecto.

ACTIVIDAD	% DE ESFUERZO
Mejoras	
mayor eficiencia	4.0
mejor documentación	5.5
mejoras para el usuario	41.8
Adaptación	
datos de entrada y archivos	17.4
equipo y sistema operativo	6.2
Correcciones	
arreglos de emergencia	12.4
arreglos programados	9.3
Otros	3.4

La gran preocupación durante la fase de planeación con respecto al mantenimiento es estimar el número de programadores de mantenimiento requeridos y especificar las facilidades necesarias para llevarlo acabo.

Un estimado muy usual, es, relacionar el total de líneas de código que pueden ser mantenidas por cada programador en forma individual; un programador común en el ambiente computacional puede mantener hasta 32k instrucciones.

Un estimado del total de personal de tiempo completo que puede ser requerido es relacionar el número total de líneas que ha de mantenerse entre el total de líneas que un programador puede mantener (este número está relacionado al tipo de aplicación).

Por ejemplo. Si un programa cuenta con 64 KDSI y cada programador puede mantener hasta 32 KDSI se requiere pues dos programadores para efectuar esta tarea.

$$FSPm = (64\ KDSI) / (32\ KDSI/FSP) = 2\ FSPm$$

El esfuerzo de mantenimiento puede estimarse mediante el empleo de un cociente de actividad, que se calcula como el número de instrucciones de código fuente que serán agregadas o modificadas durante un periodo, divididas entre el total de instrucciones:

$$ACT = (DSI\ agregadas + DSI\ modificadas) / DSI\ totales$$

Una vez obtenido este cociente se multiplica por el número de meses de programador empleado durante el período específico de desarrollo, con el fin de determinar el número de meses programador requeridos para el período de mantenimiento.

$$PMm = ACT * EAF * MMdev$$

Por último se aplican las ecuaciones para obtener el costo, que son las mencionadas anteriormente.

2.5. RAZONES DE FALLAS O CAUSAS DE RIESGOS

2.5.1. INTRODUCCION.

Los riesgos que encontramos en las aplicaciones y en los proyectos de sistemas no surge de nada; cada uno es causado.

Las causas de riesgo más comunes son:

* Evaluación económica incompleta

* Especificaciones inadecuadas

* Errores en el diseño de sistemas

* Personal de diseño incompetente

* Vanidad técnica

* Comunicación pobre

* Ausencia de puntos de "liquidación" del proyecto

* Aplicaciones que no pueden mantenerse

* Dirección incoherente

2.5.2. EVALUACION ECONOMICA INCOMPLETA

Algunas organizaciones utilizan el enfoque de "bola de cristal" para planear sistemas, otras, cometen frecuentemente el error de tratar de justificar los beneficios que traerá el nuevo sistema.

Una metodología más eficaz puede ser el hecho de tratar que el personal de planeación predizca el efecto del sistema propuesto tomando como base hechos concretos, esto es, pueden utilizarse presupestos económicos de sistemas previos muy parecidos al sistema que actualmente se desea desarrollar.

Una evaluación económica incompleta puede, también, poner en desventaja a la compañía ante la competencia, a esta razón es necesario de parte de la compañía el desarrollo de sistemas de aplicación adecuados para satisfacer los requerimientos cambiantes del mercado.

Una vez realizada la primera evaluación económica, será necesario en algún momento dado realizar otras conforme se resuelvan dudas ó se obtenga más información.

2.5.3. ESPECIFICACIONES INADECUADAS

Las especificaciones técnicas y del usuario deben desarrollarse tanto por el usuario como por el personal técnico, de otro manera puede caerse en dos errores:

1) Presentación de requerimientos en forma incorrecta

2) Exagerar las capacidades requeridas

En la presentación de los requerimientos en forma incorrecta, el usuario, puede muchas veces creer que el personal técnico está enterado de todas las acciones necesarias bajo ciertas situaciones no esperadas, esto es, las especificaciones pueden estar excluyendo algún detalle importante necesario para el programador, que en un momento dado detenga su trabajo, para esperar respuesta de solución por parte del usuario. Por otro lado la presentación incorrecta puede deberse a mero error de redacción de los requerimientos deseados.

Muchos usuarios no están familiarizados con el ambiente computacional de tal manera que tienen la idea de que el sistema resolverá todos sus problemas; bajo este criterio suele ser exageradamente costoso, si no es que imposible, cumplir con los requerimientos establecidos.

Por lo tanto para cualquier especificación con requerimientos inadecuados, exagerados ó con cualquiera característica anterior, llevará a una aplicación que incluirá uno o varios riesgos.

Dentro del proceso de desarrollo de sistemas, los requerimientos inadecuados o exagerados resultarán en costos excesivos.

2.5.4. PERSONAL DE DISEÑO INCOMPETENTE

El éxito de un trabajo de desarrollo de sistemas depende, en gran medida, de la competencia de la gerencia del usuario, los analistas de sistemas y los programadores. Debido a que el diseño de sistemas efectivo requiere un pensamiento claro y racional, ningún sistema de aplicación alcanzará su implementación efectiva si el personal involucrado no posee estas capacidades. A menudo, el personal incompetente que realiza funciones que son esenciales puede causar cualquiera de los riesgos, ya sea en el diseño del sistema o en la operación de la aplicación.

2.5.5. VANIDAD TECNICA

Esta es una frecuente causa de riesgo. Los analistas de sistemas a menudo se sienten tentados a diseñar un sistema para ejercer sus propios impulsos creativos e incorporar técnicas muy sofisticadas. Los análistas de sistemas y los programadores se consideran a sí mismos básicamente como creadores. Sin embargo, los sistemas de aplicación que se diseñan son para el uso de los usuarios, no de los diseñadores.

Los analistas de sistemas algunas veces llegan a creer que conocen las necesidades del negocio de un usuario específico mejor que el usuario mismo. Otras veces, el analista busca satisfacer las necesidades del usuario creando una aplicación que está más allá de las capacidades que tiene el usuario para operarlo. Tal vanidad normalmente ocasiona que un sistema no funcione del todo y obliga al usuario a seguir dentro de un "sistema sombra", y normalmente no es más que gastos inútiles.

2.5.6. COMUNICACION DEFICIENTE

En relación con todo lo anterior, el éxito del proceso de diseño de sistemas depende de la comunicación efectiva de los requerimientos de la aplicación, a aquellos que implantarán estos requerimientos.

Cualesquier deficiencia significativa en la comunicación entre el equipo de desarrollo del proyecto, los departamentos usuarios y la alta gerencia originarán fácilmente alguno o todos los riesgos de desarrollo de sistemas y de aplicación.

2.5.7. AUSENCIA DE PUNTOS PARA "LIQUIDAR" EL PROYECTO

Una vez iniciados, algunos proyectos de desarrollo de sistemas pasan inexorablemente a su implantación, independientemente de que obstáculos se presenten. Una vez que las ruedas están girando, pueden seguir sin que se consideren nuevas estimaciones de costos, cambios en la necesidades de la organización.

Debe concebirse algunas formas para detener los proyectos cuando llega a ser evidente que los beneficios ya no justifican los costos. El riesgo que existe si no se permiten tales puntos de "liquidación" es el de costos de desarrollo excesivos.

2.5.8. APLICACIONES QUE NO PUEDEN MANTENERSE

Una aplicación que no puede mantenerse es un sistema que no puede ser modificado técnica o económicamente para satisfacer las necesidades cambiantes de la organización. Tal sistema de aplicación estanca a toda la organización en el estado en que encuentra en un determinado momento. Debido a que un cambio en los requerimientos podría de alguna manera permitir que surgiera cualquier otro riesgo de aplicación que podría ser resultado final de un sistema que no puede mantenerse.

2.5.9. DIRECCION INCOHERENTE

Cuando participan varias personas en el diseño e implementación de una nueva aplicación, cada una de ellas puede formarse una idea un tanto distinta sobre qué deben lograr y cómo deben relacionarse al respecto. Sus trabajos diferirán, a menos que una fuerza unificadora los guíe continuamente hacia una dirección común.

FIGURA 2.5

CAUSAS DE RIESGO Y RIESGOS RELATIVOS AL DESARROLLO DE SISTEMAS

IMPACTO DE LAS CAUSAS

3—Es muy probable
2—Es probable que ocuura
1—Podria ocurrir
Blancos—Generalmente de poco efecto

CAUSAS DE RIESGO									
EVALUACION INCOMPLETA ECONOMICA	ESPECIFICACIONES INADECUADAS	ERRORES EN EL DISEÑO DEL SISTEMA	PERSONAL DE DISEÑO INCOMPETENTE	UNIDAD TECNICA	COMUNICACION DEFICIENTE	PARA LIQUIDAR PROYECTO AUDIENCIA DE PUNTOS	APLICACION QUE NO PUEDE MANTENERSE	DIRECCION INCOHERENTE	RIESGOS
	2	3	3		3		2	3	RIESGOS DE APLICACION (SI SE IMPLEMENTA EL PROYECTO) CONTABILIDAD ERRONEA
	2	2	2		2		2	2	CONTABILIDAD INACEPTABLE
	1	1	1		1		1	1	INTERUPCION DEL NEGOCIO
	3	2	3		2		2	2	DECISIONES ERRONEAS DE LA GERENCIA
	1	1	1		1			1	FRAUDE
	1	1	1		1	1	2	1	SANCIONES LEGALES
	3	3	3	3	2	2	1	2	COSTOS EXCESIVOS / INGRESOS DEFICIENTES
	1	1	1		1			1	PERDIDA O DESTRUCCION DE ARCHIVOS
	2	2	2	1	2	1	2	2	DESVENTAJAS ANTE LA COMPETENCIA
	3	3	2		3			3	RIESGOS DE PROYECTO DESICIONES ERRONEAS DE LA GERENCIA
	3	3	3	3	3	2	3	3	COSTOS EXCESIVOS
2	2	2	2	1	2	1	2	2	DESVENTAJAS ANTE LA COMPETENCIA
2	2	2	3	2	2		2	3	INTERRUPCION DEL NEGOCIO

III. DOCUMENTACION

3.1. INTRODUCCION

La documentación en el desarrollo de sistemas desempeña un papel muy importante ya que ésta es tomada como un medio para asugurar:

* Evaluaciones económicas completas
* Especificaciones adecuades
* Comunicación eficiente
* Revisión efectiva por parte de:
    - Gerencia
    - Usuarios
    - Supervisores técnicos
    - Auditores
* Corrección efectiva de errores en el diseño
* Soporte para futuros mantenimientos

3.2. DOCUMENTACION DEL DISEÑO DETALLADO

Normalmente, la documentación del diseño detallado incluirá:

* Sumario
* Estructura
* Diagrama jerárquico
* Lista de programas
* Rutinas comunes
* Bases de datos
* Diccionario de datos
* Reporte de referencia cruzada
* Lista de mensajes
* Especificaciones de programa

3.2.1. SUMARIO

Es el tipo de documento en el que se explica o narra el sistema en forma completa.

Debe incluirse la explicación para cada módulo que compone el sistema; describiendo entradas, salidas y procesos; además pueden ser mencionadas las interrelaciones entre ellos así como con otros sistemas.

El lenguaje de codificación también debe ser especificado en este documento.

3.2.2. ESTRUCTURA

Este tipo de documento muestra por medio de diagramas de bloques todos los módulos del sistema, así como las bases de datos utilizadas y todas las interrelaciones entre estos elementos.

Se puede considerar que este documento es un soporte del SUMARIO.

3.2.3. DIAGRAMA JERARQUICO

El diagrama jerárquico es un esquema que define la estructura del sistema para dar visibilidad de las relaciones entre pantallas y las forma en que cada programa es accesado desde su menú.

FIGURA 3.1

SUMARIO DETALLADO

A continuacion una breve descripcion del sistema completo:

SISTEMA DE INVENTARIOS

.
.
.
.

A continuacion una breve descripcion de cada modulo

SUBSISTEMA ARTICULOS

.
.
.
.
.

SUBSISTEMA DE VENTAS

.
.
.
.
.

FIGURA 3.2a

ESTRUCTURA DETALLADA

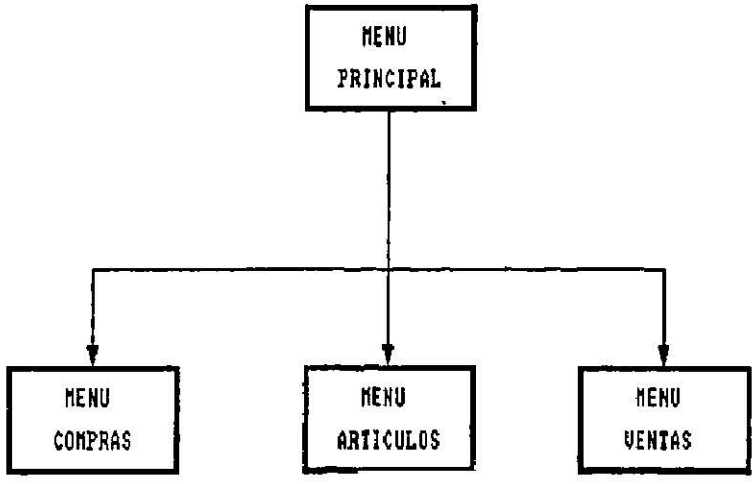


FIGURA 3.2b

ESTRUCTURA DETALLADA

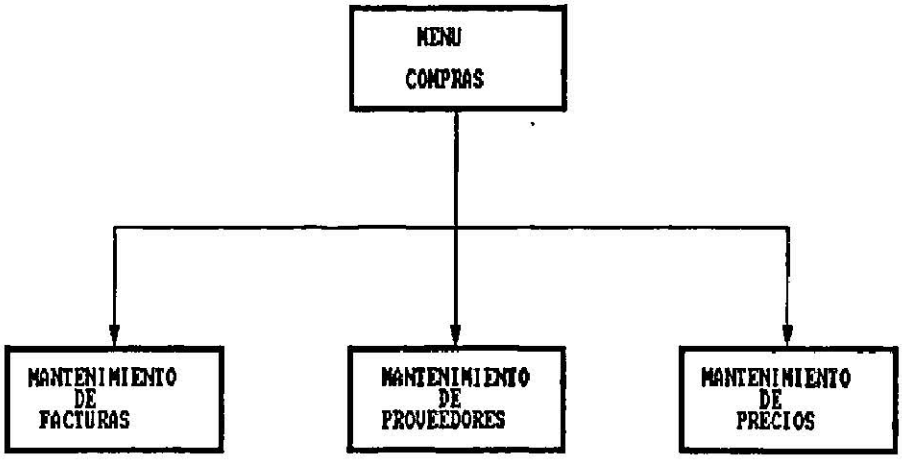


FIGURA 3.2c

ESTRUCTURA DETALLADA

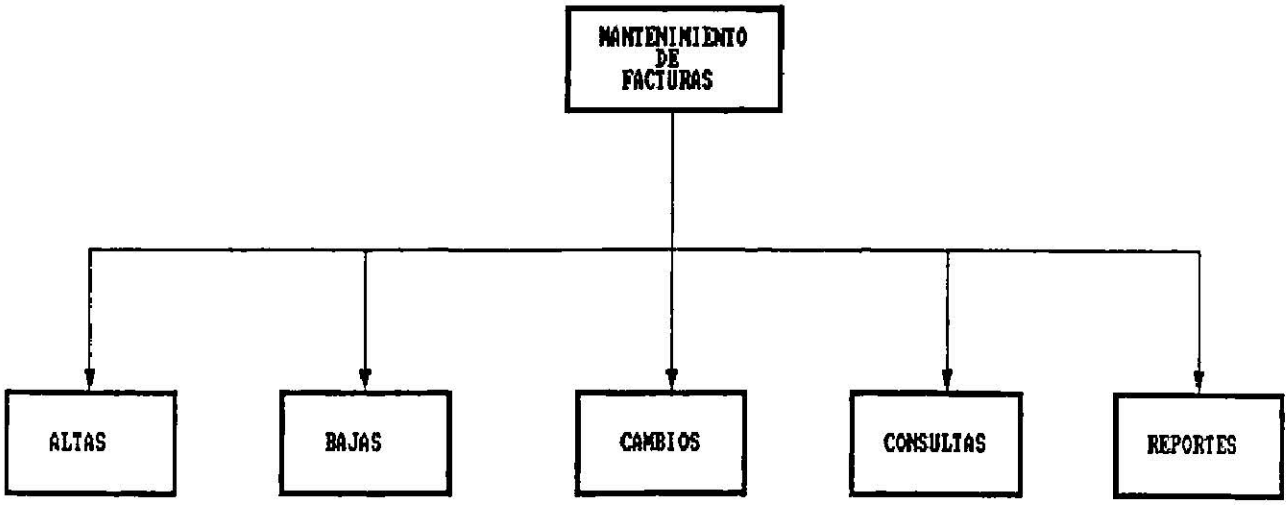
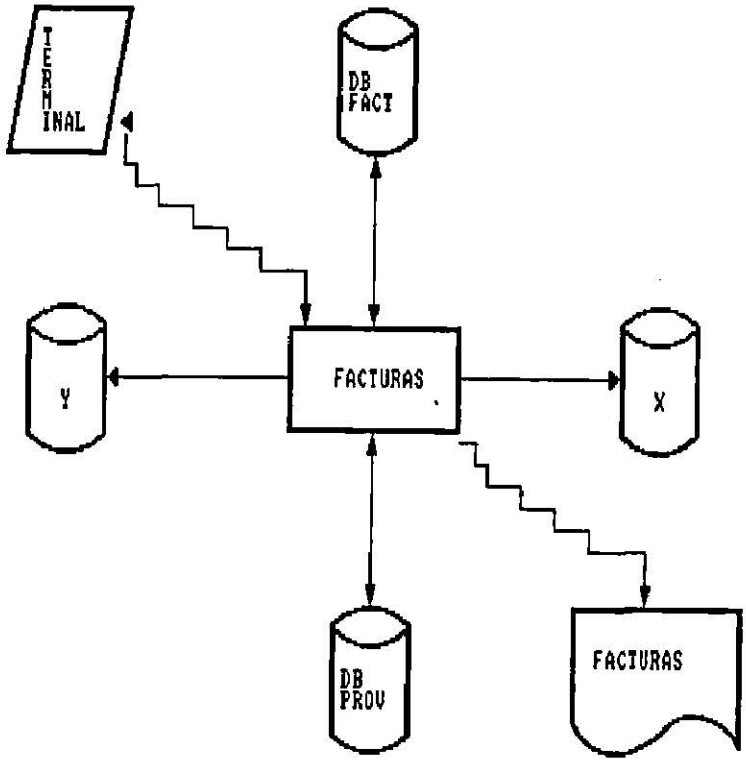
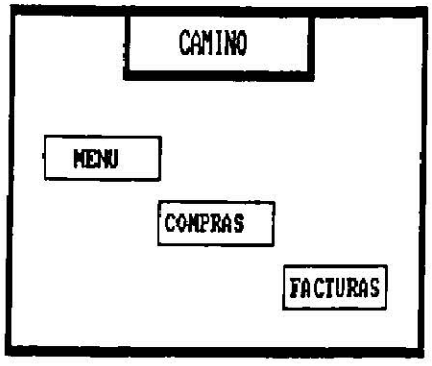


FIGURA 3.2d

ESTRUCTURA DETALLADA





En este diagrama, es necesario incluir información en un formato específico para cada programa. Esta información es:

* Identificación estándar de programas
* Breve descripción de los programas
* Programa llamador
* Información de entrada/salida
* Bases de datos que accesa el programa

3.2.4. LISTA DE PROGRAMAS

Este documento consiste de una tabla, la cual deberá incluir información acerca del propósito de cada uno de los programas que integran los módulos del sistema, además deberá anexarse la información siguiente:

* Nombre del programa
* Descripción
* Lenguaje
* Tipo de programa
  - Menú
  - Reporte
  - Proceso
  - Batch

FIGURA 3.3a

DIAGRAMA JERARQUICO

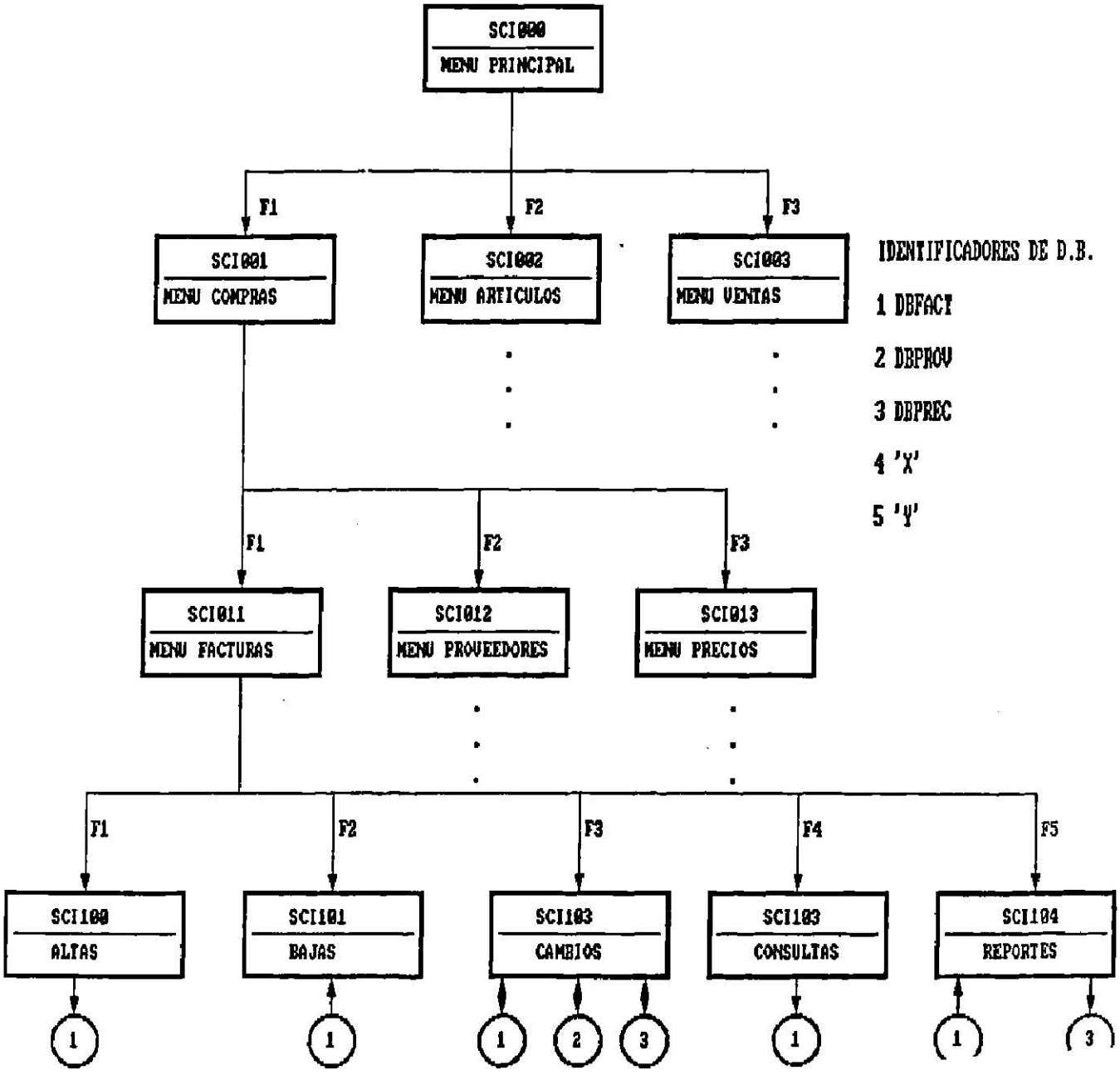


FIGURA 3.3b

DIAGRAMA JERARQUICO

DESCRIPCION DE PROGRAMAS

IDENTIFICADOR: SCI000

DESCRIPCION : Este programa es el menu principal del sistema. Muestra la primer pantalla en el sistema solicitando y checando un password. Ademas encadenara a los programas de los otros modulos dependiendo de la seleccion hecha por el usuario.

LLAMADO POR : N/A

ENTRADAS : N/A

SALIDAS : N/A

BASES DE USADAS : NINGUNA

IDENTIFICADOR: SCI010

DESCRIPCION : Este programa es el munu de compras para el modulo de COMPRAS y puede accesar al menu de facturas, proveedores y precios.

LLAMADO POR : SCI000

ENTRADAS : N/A

SALIDAS : N/A

BASES DE USADAS : NINGUNA

FIGURA 3.4a

LISTA DE PROGRAMAS

PROGRAMA	DESCRIPCION	LENGUAJE	TIPO	FRECUENCIA	MODULO
SCI000	MENU PRINCIPAL	COBOL	MENU		
SCI001	MENU DE COMPRAS	COBOL	MENU		COMPRAS
SCI002	MENU DE ARTICULOS	COBOL	MENU	REQUERIDA	ARTICULOS
SCI003	MENU DE VENTAS	COBOL	MENU	REQUERIDA	VENTAS
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.

TIPO = REPORTE, MENU, PROCESO, BATCH, ETC.

FECUENCIA = REQUERIDA, DIARIA, SEMANAL, ETC.

LENGUAJE = ESTE ES LLANADO CUANDO EXISTEN LENGUAJES DIFERENTES USADOS EN EL SISTEMA.

FIGURA 3.4b

LISTA DE PROGRAMAS

MODULO : COMPRAS

PROGRAMA	DESCRIPCION	LENGUAJE	TIPO	FRECUENCIA	MODULO
SCI001	MENU DE COMPRAS	COBOL	MENU		COMPRAS
SCI010	MENU DE FACTURAS	COBOL	MENU	REQUERIDA	COMPRAS
SCI011	MENU DE PROVEEDORES	COBOL	MENU	REQUERIDA	COMPRAS
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.

TIPO = REPORTE, MENU, PROCESO, BATCH, ETC.

FECUENCIA = REQUERIDA, DIARIA, SEMANAL, ETC.

LENGUAJE = ESTE ES LLANADO CUANDO EXISTEN LENGUAJES DIFERENTES USADOS EN EL SISTEMA.

* Frecuencia de programa

- Requerido
- Diario
- Semanal

* Módulo al que pertenece

3.2.5. RUTINAS COMUNES

En este tipo de documento se eleborará una lista de rutinas usadas frecuentemente en el sistema.

Estas rutinas suelen ser llamadas a bases de datos, rutinas de mensajes de error, procedimientos comunes, etc; todas las rutinas que una empresa puede elaborar con el fin de estandarizar los procesos de codificación y diseño de los programas.

El objetivo principal de este documento es preveer consistencia dentro del sistema y através de todos los módulos.

Cada rutina debe ser descrita en forma correcta, de tal manera que la función que se lleva a cabo pueda dejar una idea clara.

Algunos ejemplos de rutinas son:

* Conversión de fechas (interna, desplegado/impresión)
* Cálculos comunes
* Transferencia de programas
* Rutinas de error, etc.

FIGURA 3.5

RUTINAS COMUNES

NOMBRE DE LA RUTINA: MENSAJERO.PRC

DESCRIPCION : ESTA RUTINA MANEJA TODOS LOS POSIBLES ERRORES ENCONTADOS CUANDO CUALQUIER BASE DE DATOS ES ACCESADA.

ALGORITMO:

1) Inicio
2) recibir numero de error
3) identificar el mensaje
4) leer el mensaje
5) variable = mensaje
6) regresar
7) fin

PARAMETROS DE ENTRADA/INFORMACION(FORMATO INCLUIDO):

NUMERO DE ERROR (NUMERICO)

PARAMETROS DE SALIDA/INFORMACION(FORMATO INCUIDO):

UN MENSAJE PARA EL ERROR. (ALFANUMERICO 20 CARACTERES).

3.2.6. BASES DE DATOS

En este documento se podrá describir ya sea las bases de datos usadas ó archivos tradicionales a utilizar en el sistema.

Se deberá mencionar las características de las bases de datos las cuales se pueden resumir en:

1) DESCRIPCION GENERAL

Incluirá la información necesaria para desarrollar y crear el sistema de base de datos.

a) Nombre
b) Identificador
c) Layout
   - tipo de acceso
   - medida
   - tipo de campos
   - llaves

d) Origen

   - programas que mantienen esta base de datos

e) Descripción
f) Tipo de base de datos
   - relacional
   - Jerárquica
   - sequencial, etc.

g) Relaciones

Cuáles bases de datos están relacionadas con ésta en forma

- directa

  Bases de datos complementarias

- indirecta

  Bases de datos que obtienen ó dan información de ésta.

- referencia

  Esta relación es solamente para checar información necesaria para continuar con una operación especifica.

h) Llaves

- Primaria/Secundaria
- Duplicable/No duplicable
- Modificable/No modificable

i) Locación de la base de datos

- Directorio
- Diskette, disco, cinta, etc.

j) Clasificación de seguridad

- Que nivel de seguridad es necesario para esta base de datos.
- Usuarios autorizados
- Resplado requerido
- Frecuencia de resplado
- Frecuencia de purgación

k) Consideraciones Especiales

Cualquier cosa no mencionada en los puntos anteriores que deba tomarse en cuenta.

2) Descripción del registro

* Identificador
* Longitud
* Tipo de registro
  - único o variable

    Si es de algun tipo de jerarquía
  - padre
  - hijo
  - nivel de jerarquía
  - vista utilizada
* Número de campos
* Volumen estimado
* Descripción

FIGURA 3.6

BASE DE DATOS

IDENTIFICADOR :

NOMBRE :

LAYOUT :

ORIGEN :

DESCRIPCION :

TIPO DE BASE DE DATOS:

FIGURA 3.7

BASE DE DATOS: REGISTRO

IDENTIFICADOR DE BASE DE DATOS: DBCPROV.DAT

NOMBRE DEL REGISTRO : PROVWS-01

LONGITUD DEL REGISTRO : 128

TIPO DE REGISTRO : UNICO

NUMERO DE CAMPOS : 10

VOLUMEN ESTIMADO DEL REGISTRO : INICIALMENTE : 60 RANGO DE INCREMENTO: 10 POR MES

DESCRIPCION DEL REGISTRO : ESTE REGISTRO ES UNICO EN LA BASE DE DATOS.

DEFINICION DEL REGISTRO:

NUMERO DE CAMPO	NOMBRE	DESCRIPCION	POSICION INICIO	POSICION FIN	LONGITUD (BYTES)	FORMATO
1	PROV-NUME	NUMERO DE PROVEEDOR	1	4	4	9(6)
2	PROV-CONT	NUMERO DE CONTRATO	5	9	5	9(5)
3	PROV-TIPO	TIPO DE PROVEEDOR	10	10	1	X(1)
4	PROV-NOMB	NOMBRE DEL PROVEEDOR	11	29	20	X(20)
5	PROV-DIRE	DIRECCION DEL PROVEEDOR	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.					

3.2.7. DICCIONARIO DE DATOS

Documento que lista todos los nombres de los campos de cada base de datos dentro del sistema incluyendo la descripción y todos los posibles valores que pueden tomar.

El propósito principal de este documento es asegurar consistencia en los nombres convencionales através del sistema y proveer una fuente de información para todos los campos o elementos.

3.2.8. REPORTE DE REFERENCIA CRUZADA

El reporte de referencia cruzada provee un mecanismo efectivo para identificar el impacto sobre los campos, archivos o programas cuando algún cambio será implementado durante la vida de un sistema.

FIGURA 3.8

DICCIONARIO DE DATOS

NOMBRE DEL CAMPO	BASE DE DATOS	DESCRIPCION	VALORES DE EDICION	REFERENCIAS	
				COMPLETA	ABREVIADA
PROV-NUME	DBPROV.DAT	NUMERO DE PROVEEDOR	NUMERICO	NO. PROVEEDOR	NO. PROV.
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
FACT-NUME	DBFACT.DAT	NUMERO DE FACTURAS	NUMERICO	FACTURA	FACT.
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.

En general, existen cuatro tipos de reportes de referencia cruzada:

* Programas / Bases de datos
* Programas / Campos
* Programas / Programas
* Bases de datos / Bases de datos

El primero y el segundo proveen toda la información relacionada a los elementos o bases de datos. La tercera mostrará las relaciones entre los programas, y la última las relaciones entre las bases de datos.

3.2.9. LISTA DE MENSAJES

Este documento contendrá todos los mensajes que serán utilizados en el en el sistema. Estos mensajes serán identificados por un número y será necesario especificar la razón por la cual el mensaje será utilizado.

Para este punto, puede ser declarado los tipos de mensajes ha ser utilizados e identificarlos por algún dígito especial o tal vez una línea estándar de despliegue.

FIGURA 3.9a

REPORTE DE REFERENCIA CRUZADA
PROGRAMA/BASES DE DATOS

BASES DE DATOS / PROGRAMAS	(1) DBFACT	(2) DBPROD	(3) DBPREC	(4) DBXXXX	(5) DBCCCC	.	.	.	.
SCI000									
SCI100									
SCI200									
SCI300									
SCI400									
SCI500									
SCI600									
SCI700									
SCI800	R/W	R/W	R	W					

ACCIONES:

'R' - EL PROGRAMA LEE DE LA BASE DE DATOS

'W' - EL PROGRAMA ESCRIBE DE LA BASE DE DATOS

'R/W' - EL PROGRAMA LEE DE Y ESCRIBE DE LA BASE DE DATOS

FIGURA 3.9b

REPORTE DE REFERENCIA CRUZADA
PROGRAMA/CAMPOS

BASE DE DATOS: DBPROV TIPO DE REGISTRO: UNICO

CAMPOS / PROGRAMAS	(1) PROV NUME	(2) PROV CONT	(3) PROV TIPO	(4) PROV NOMB	(5) PROV DIRE	.	.	.	.
SCI000									
SCI100									
SCI200	I	A	A	I	U				
SCI300									
SCI400									
SCI500									
SCI600									
SCI700									
SCI800									

ACCIONES:

'A' - EL PROGRAMA SUMA EL CAMPO 'I' - EL PROGRAMA DESPLIEGA EL CAMPO

'U' - EL PROGRAMA ACTUALIZA EL CAMPO 'D' - EL PROGRAMA BORRA EL CAMPO

FIGURA 3.9c

REPORTE DE REFERENCIA CRUZADA
PROGRAMA/PROGRAMA

PROGRAMA LLAMADOR / PROGRAMAS LLAMADOS	(1) SCIMENU1	(2) SCIMENU2	(3) SCIMENU3	(4) SCIMENU4	(5) SCIMENU5	.	.	.	.
SCI000									
SCI100	*								
SCI200	*								
SCI300	*								
SCI400	*								
SCI500	*								
SCI600		*							
SCI700		*							
SCI800		*							

ACCIONES:

'*' - PROGRAMA LLAMADOR 'LLAMA' A PROGRAMA LLAMADO

FIGURA 3.10

LISTA DE MENSAJES

NUMERO	MENSAJE	RAZON
0001	EL REGISTRO YA EXISTE	ESTA TRATANDO DE DUPLICAR UN REGISTRO
0002	IMPRESORA FUERA DE LINEA	ENVIA UN DOCUMENTO A LA IMPRESORA Y ESTA NO ESTA PREPARADA PARA IMPRIMIR
0003	NUMERO DE CLIENTE A DAR DE ALTA	SE DESEA PROCESAR SIN INFORMACION
0004	ARCHIVO VACIO	NO EXISTE INFORMACION PARA CONSULTAR
0005	. . . . .	. . . . .
0006		
. . . . .		

En general se pueden identificar cuatro tipos de mensajes.

* Informativos ("impresora fuera de línea")
* Error ("número de empleado inválido")
* Status ("reporte en proceso. . .")
* Petición ("número de empleado a dar de alta")

3.2.10. ESPECIFICACIONES DEL PROGRAMA

El propósito de este documento es proveer una descripción completa de cada programa que integra el sistema.

Incluye las siguientes partes:

1) IDENTIFICACION Y CONTROL DE INFORMACION

Información general necesaria para proveer el identificador estándar por el cual el programa es conocido, el nombre del programa y cualquier información para el control del programa.

2) NARRACION DEL PROGRAMA

Es una descripción detallada de qué es lo que se espera que el programa produzca. Debe ser lo suficientemente detallada para que una breve instrución al programador pueda enterarlo sin dejar muchas dudas al respecto:

3) RUTINAS COMUNES

Todas las rutinas a utilizar así como su localización

4) BASES DE DATOS/ ELEMENTOS UTILIZADOS

Incluye todas las bases de datos a utilizar, los campos a accesar en cada una de ellas e inclusive la operación sobre los campos, cualquier aspecto relevante como alguna condición especial para escribir o leer algún elemento.

5) CALCULOS

Si algún cálculo va a ser utilizado, debe ser mencionado así como cualquier aspecto relevante sobre estos cálculos; por ejemplo cuando este cálculo tome cierto valor tomar una decisión particular.

6) FORMATO DE PANTALLAS/REPORTES

El formato de la pantalla o del reporte utilizado sobre alguna forma estándar, de tal manera que las líneas y columnas en donde el texto y las variables deben aparecer o aceptarse sean definidos.

7) DIAGRAMA DE FLUJO

Diagrama que mostrará el flujo lógico del programa en forma breve por parte del jefe de proyecto y en forma detallada por el programador. Deberá incluir todos los cálculos primarios y todas las decisiones tomadas através del programa. Todas las lecturas desde o escrituras a archivos o bases de datos serán definidos también.

8) CASOS GENERALES DE PRUEBAS/ RESULTADOS DE PRUEBAS

Tiene como propósito mostrar el funcionamiento del programa através de algunos casos generales de pruebas y sus resultados. Ejemplos:

* Desplegado correcto de encabezados
* Fin de archivo
* Campos numéricos con contenidos alfanuméricos
* Fechas incorrectas
* Desplegado de mensajes correcto
* Elección de opciones invalidas
* Registros no existentes
* Archivos vacios

IV. A P E N D I C E

4.1. ESPECIFICACIONES DEL USUARIO

Las especificaciones del usuario se realizan mediante esfuerzos conjuntos de los usuarios y los analistas de sistemas y tiene como objeto el planeamiento y la resolución de un problema que afecta su negocio.

Uno de los objetivos importantes de las especificaciones es que el analista se familiarice con el medio del usuario ya que una vez que esto ocurre el usuario podrá expresar sus necesidades.

Los analistas de sistemas se dedican a las actividades de examinación y estudio de todos los procedimientos manuales y computarizados utilizados por el usuario en su actividad diaria.

Además se revisan todas las relaciones entre la aplicación que se está desarrollando y otras aplicaciones que puedan estar afectando directa o indirectamente a ésta.

La documentación que se debe desarrollar en esta etapa refejará la jerarquía y los efectos de un contrato entre el usuario y el personal de desarrollo. Este compromiso tendrá validez tanto para la primera liberación del proyecto a desarrollar como para sus futuros mantenimientos.

Por lo tanto, esta documentación debe presentar:

* Las acciones que el usuario va a seguir
* Las reglas de decisión que van a aplicarse
* Los servicios que va a ofrecer el proyecto
* Los métodos y calendarios para interacciones entre los usuarios y el equipo de desarrollo.

Debido a que estas especificaciones están orientadas a la resolución de problemas del negocio los esfuerzos mayores son por parte del usuario y de los analistas, solo se realizan pequeñas consultas al jefe de programación con respecto a la viabilidad técnica, estimaciones de tiempo y costos.

4.2. ESPECIFICACIONES TECNICAS

Esta actividad define el enlace entre los niveles técnicos y la actividad en el negocio.

El proyecto es llevado hacia el punto donde se genera la lógica del procesamiento y el manejo de archivos que requerirá tanto el el programador como el computador.

La documentación que resulta de esta fase cubre toda una variedad de restricciones técnicas y operacionales del sistema.

Esto quiere decir que todas las limitantes técnicas para el desarrollo del sistema deberán ser documentadas con el objeto de hacerle conocer al usuario y al gerente del proyecto la viabilidad del mismo.

Además la documentación deberá incluir una serie completa de especificaciones que le permitan al programador trabajar sin riesgo de errores.

Dependiendo de los resultados de este trabajo deberán anexarse los calendarios de tiempo en que se espera se implemente el trabajo, lo cual servirá para posteriores reevaluaciones.

4.3. ESPECIFICACIONES DEL PROGRAMA

Formalmente efectuada significa:

1) El análista tendrá la oportunidad de reconsiderar la lógica del diseño de sistemas con un nivel detallado.

2) Es un medio para recolectar ideas con uno o más programadores, los que requiera el programa.

3) Es un registro permanente de los requerimientos cambiantes de cada programa del sistema.

En general se especifican las siguientes fases de desarrollo de programa.

1) Programación

2) Hoja de control de programa

3) Diagrama de flujo general

4) Formatos de pantallas y reportes

5) Lógica, tablas y variables del usuario

6) Datos de prueba

PROGRAMACION

Consiste en definir el propósito y alcance del programa, la secuencia de la lógica, su implementación en código así como la depuración del mismo; la búsqueda de casos de prueba así como su depuración y por último la documentación de todos estos pasos.

HOJA DE CONTROL DEL PROGRAMA

Este documento define todo lo relacionado con la historia y estadísticas del programa; ejemplo:

Identificador del programa, nombre, versión, frecuencia de duración, cargos al cliente, fecha programada de terminación, estimación en horas programadas.

DIAGRAMA DE FLUJO

Ilustración general del propósito y alcance del programa con el objeto de lograr una comprensión más clara por parte del programador.

FORMATOS DE PANTALLAS Y REPORTES

Este documento tiene el propósito específico de ahorrar al programador rediseños de formas anteriores y de perder tiempo.

VARIABLES DEL USUARIO

Instrucciones con respecto a cómo deben aparecer totales (jerarquización), cuántas veces debe aparecer una impresión, cuántas copias de reporte deben aparecer, ésto es, elementos importantes cuando el programa tiene múltiples posibilidades en forma de procesamiento.

DATOS DE PRUEBA

Esto permite al programador ahorrar tiempo en la búsqueda de datos de prueba y al analista en la revisión del funcionamiento del programa, ya que él conoce los resultados que se deberán obtener porque es él quien tiene una visión global de los casos de prueba que el programa incluye.

4.4. ANALISIS DE COSTO EFECTIVIDAD

Se le llama análisis de costo/efectividad al proceso de justi - ficación de inversión de capital contra la efectividad derivada de- los beneficios. En otras palabras, el análisis determina si el sis- tema propuesto aportará beneficios que superen el costo.

Normalmente este análisis se lleva a cabo para varias propues - tas de sistema, el objetivo de hacer esto es que la administración pueda elegir entre varias opciones.

En general, son cuatro las áreas que se trabajan en un análisis de costo efectividad para probar la viabilidad de un sistema.

1) Evaluación del equipo.

2) Identificación de Costo

3) Efectividad

4) Métodos de Estimación
   (Comparación de Costo y Efectividad).

Los costos se definen primero: por tipo, por comportamiento, por función y por el tiempo.

DEFINICION DE COSTOS SEGUN SU TIPO

1) Costos Directos: Son los relacionados como provinientes del-equipo propuesto.

2) Costos Indirectos: Son todos aquellos que son necesarios para llevar a cabo el sistema y que no pueden ser identificados fácilmente con él.

Todos estos quedan repartidos en la organización y como ejemplo de ellos están: la renta del local, el seguro, los impuestos, lòs sueldos de los administradores y los empleados.

DEFINICION DE COSTOS SEGUN SU COMPORTAMIENTO

Costos Variables: Son aquéllos que sufren un impacto directo con la variación del volumen de operaciones. Por ejemplo si aumentó el nivel de trabajo, el computador será más utilizado y por ende la energía eléctrica necesaria se incrementará y con ello su costo aumentará.

Costos no Variables: Son aquellos que aunque tienen cambios, estos no suelen ser tan frecuentes, se encuentran en este caso; la renta, los impuestos, los sueldos.

DEFINICION DE COSTOS SEGUN SU FUNCION

Costos de Desarrollo: Son los que están relacionados con el fin de realizar algún producto. Caen en este caso, el costo de programación, entrenamiento, etc.

Costos de Operación: Son aquellos que son necesarios para lograr el funcionamiento del área relacionado con el sistema, como ejemplo tenemos: los operadores del equipo de cómputo, técnicos en hardware y soportes de software.

Costos de Mantenimiento: Su nombre es obvio, son todos aquellos que son necesarios para que el área siga funcionando, ésto es; reparación de equipo, mano de obra y componentes y partes.

DEFINICION DE COSTOS DE ACUERDO AL TIEMPO

Costos Periódicos: Ocurre en intervalos regulares de tiempo; sueldos, rentas.

Costos no Periódicos: El costo de programación es un costo no - periódico.

MEDICION DE EFECTIVIDAD

La efectividad de un sistema se mide en términos de los beneficios:

1) Beneficios directos ó tangibles.

2) Beneficios indirectos ó intangibles.

Beneficios Directos: Son ahorros en los costos debidos a la eliminación de alguna operación o eficiencia de ésta.

Ejemplo: El costo actual del sistema es $ 2.00 para procesar cada transacción, mientras que el sistema propuesto procesará la misma transacción con un gasto de $ 1.50. Si se procesan 300,000 transacciones al año, el ahorro en el costo apartado por el nuevo sistema será de $ 150,000.00.

Beneficios Indirectos: Son beneficios de naturaleza intangibles sin embargo pueden expresarse en términos cuantitativos en la medida que puedan identificarse.

Ejemplo: el análisis de las ventas puede indicar que la empresa está perdiendo el 5% de sus ventas brutas cada año, debido al agotamiento de las existencias.

El sistema actual tiene un nivel de servicio al cliente del 85%, mientras que el nuevo sistema, debido a la implantación de nuevos métodos de control de inventario, logrará un nivel de servicio aumentará las ventas anuales en un 3%, por que habrá menor probabilidad de quedarse sin existencias.

METODOS DE ESTIMACION

Para estimar el costo y la efectividad básicamente se emplean dos métodos:

1) Cálculo Objetivo: Este cálculo consiste en la verificación de las listas de precios presentadas por los vendedores de equipo, software, etc. y la comparación de todas ellas.

2) Estimaciones: Consiste en utilizar un método matemático ó hacer uso de registros históricos para estimar el costo y la efectividad del sistema propuesto.

EVALUACION DE EQUIPO

Este proceso de evaluación de equipo empieza con una lista presentada por los vendedores al analista de sistemas; con base en un equipo específico.

El proceso tiene 2 fases principales:

1) El analista determinará simplemente si el equipo ofrecido por los vendedores reune las características imperativas. En este momento aceptará o rechazará el equipo ofrecido.

A) Costo inferior a X dollares

B) Compatibilidad

C) Computador

D) Velocidad de impresora

2) Del equipo que halla aceptado procederá a realizar una comparación para aceptar aquel equipo que le ofrezca más.

G L O S A R I O

Análisis de Costo Efectividad: Proceso de justificación de inversión de capital contra la efectividad derivada de los beneficios.

Actividad Nula: Actividad que no consume tiempo ni recursos.

Actividades Paralelas: Actividades que por ser independientes pueden ser desarrolladas en el mismo tiempo.

Archivos: Colección de cierto tipo de datos que satisfacen ciertas necesidades de información.

Bases de Datos: Colección de archivos de datos integrados y organizados en un solo sistema de archivo general.

Beneficios Directos: Ahorros en los costos por eliminación ó eficiencia de operaciones inadecuadas.

Beneficios Indirectos: Beneficios intangibles, pueden expresarse cuantitativamente en la medida que pueden identificarse.

Bibliotecario de programas: Persona encargada de llevar el el control de las versiones de los programas, su almacenamiento en localizaciones conocidas y reportes sobre ellos.

Casos de Prueba: identificación de situaciones especiales que el progrma deberá resolver y obtener alguna aclaración ó solución para el usuario.

Cliente: Persona que paga un servicio computacional.

COCOMO: Modelo contructivo de costos.

Código Fuente: Es el código escrito por el programador.

Codificar Software: Es la imagen del diseño lógico en un lenguaje que podrá ser traducido por una máquina.

Conversión de Archivos: Es el cambio de la información utilizada a otra que pueda sèr utilizada por el nuevo sistema a implantar.

Compilador: Es un traductor de intrucciones codificadas por un programador a un lenguaje que la máquina podrá entender.

Complejidad: Medida del nivel de dificultad para producir un resultado.

Costos de Desarrollo: Relacionados con la realización de un producto.

Costos Directos: Relacionados como provenientes del equipo propuesto.

Costos indirectos: Son necesarios para desarrollar un sistema (Renta, Impuestos, Sueldos, etc.)

Costos de Mantenimiento: Relacionado con el funcionamiento continuo de un sistema computacional.

Costos de Operación: Relacionados con el logro del funcionamiento de un sistema computacional.

Costos Periódicos: Ocurridos en intervalos regulares de tiempo.

Costos Variables: Sus impactos no son frecuentes.

Datos de Prueba: Datos utilizados para probar la funcionalidad de un programa.

Delfi: Tipo de estimación que realiza concenso con el fin de minimizar diferencias en las diversas opiniones de los expertos involucrados y obtener una estimación optimizada.

Diccionario de Datos: Tipo de documento que refleja información de todas las variables y campos de las bases de datos usadas por el sistema.

Diagrama de Flujo: Tipo de documento que por medio de diagrama de bloques plasmará la secuencia de intrucciones que un programa deberá especificar para obtener un resultado específico.

Diagrama Jerárquico: Documento que por medio de diagramas muestra las interrelaciones entre pantallas y programas.

Diseño detallado: Definición de programas, tipo de archivos usados, pantallas, reportes, etc., que un sistema utilizará en su desarrollo.

Diseño Lógico: Conjunto de figuras que unidas como un todo definirán la manera para solucionar un problema.

Documento Bases de Datos: Documento que refleja la información sobre todas las bases de datos utilizadas en el sistema y las características más importantes.

Efectividad: Nivel de beneficios obtenidos.

Ensamblador: Convertidor o traductor de un programa de computadora de un lenguaje simbólico a un lenguaje de máquina.

Especificaciones: Documentación que define cúales serán las decisiones a ser tomadas en el desarrollo de un producto de programación y cómo será dessarrollado.

Especificaciones Técnicas: Documento que define el enlace entre los niveles técnicos y la actividad del negocio de un usuario.

Especificación de programa: Tipo de documento que provee información detallada de cada programa que integra un sistema.

Especificaciones del usuario: Documento que plasma el planeamiento y la solución de un problema que afecta el negocio de un usuario.

Estimación: aproximación de horas, días, meses ó costos que se necesitan para producir un sistema.

Estimación jerárquica hacia abajo: Los costos de esta estimación son obtenidos tomando en cuenta el sistema integrado.

Estimación jerárquica hacia arriba: Los costos de esta estimación es obtenido a partir de cada módulo del sistema.

Estructura: Documento basado en diagramas de bloques para definir un sistema. (Define cada módulo como archivos utilizados e interrelaciones).

Estructuras de division: Organigrama jerárquico.

Evaluación del Equipo: Proceso de selección del equipo a ser utilizado por el sistema a desarrollar.

Factibilidad del Proyecto: Fase en la que se estudia si es redituable desarrollar un sistema de información computarizado que apoye las operaciones de un negocio.

Factores Multiplicadores: Son los factores tomados en cuenta en una estimación de proyectos de software para determinarla con más exactitud.

Gráficas de Barras: Tipo de planeación que utiliza barras para mostrar actividades a desarrollar.

Gráficas de Referencia: Momento en el que el proyecto se encuentra en una situación crítica para su desarrollo.

Gráficas Pert: Gráficas de planeación de proyectos que utilizan círculos y rectas para representar interrelaciones de actividades del proyecto.

Hoja de Control de un Programa: Documento que define la historia de un programa.

Intalación: Preparación del ambiente y del equipo para poner a funcionar un sistema computacional.

Integración del Sistema: Visión del sistema como un todo.

Jucio Experto: Tipo de estimación que se centra en el juicio y el sentido comercial de los individuos involucrados en ella.

KDSI: Número de millares de instrucciones de código fuente entregadas con el producto.

Lenguaje de alto nivel: Lenguaje de programación cuya estructura está destinada a las aplicaciones y es independiente de la estructura de la computadora.

Liberación de Proyecto: Fecha en que un sistema o proyecto es puesto en marcha.

Lista de Mensajes: Documento que refleja todos los mensajes a utilizar en un sistema.

Manejo de Configuración: Manejo necesario para adoptar el ambiente al equipo y al sistema a desarrollar (Máquinas; memoria y equipo periférico. CPU).

Mantenimiento: Correcciones y mejoras a un sistema existente.

Módulo: Parte de un sistema que define una parte especifica y que por sus características es identificado como tal.

Paquetes de Programación: Son pequeños sistemas de programación que facilitan las tareas para los proyectos a desarrollar.

Procesamiento Distribuido: Es aquel tipo que centraliza la información en un lugar pero ésta puede ser usada por procesos fuera de ese lugar.

Programa: unidad más pequeña de un sistema de información que tiene por objeto resolver un problema específico.
Esta formado de una colección de instrucciones que la la máquina traducirá y entenderá para producir un producto.

Programador: Persona encargada de desarrollar todas las tareas derivadas de la unidad más pequeña de un sistema de información llamado programa.

Programador Senior: Nivel de experiencia de un programador que cumple con 5 ó 6 años de variadas aplicaciones y tipos de lenguajes trabajados.

PM: Esfuerzo mensual por programador requerido en el desarrollo de un producto.

Prueba de integración: Prueba de todos los programas que integran un sistema.

Prueba de Software: Es una lectura del código de programa para probar que es entendible y que cumple con los estandares y la calidad requerida.

Prueba por unidad: Prueba de un programa.

Puesta en Producción: Fecha en que un sistema computacional es instalado.

Puntos de liquidación: Momento en el cual los beneficios de un sistema no justifica los costos.

Recorridos: Lectura del código del programa para probar que sea entendible, mantenible y cumpla con los estandares.

Requerimientos: necesidades.

Reporte de Referencia Cruzada: Documento que refleja información para identificar los campos, archivos y programas utilizados en el sistema.

Resultado de Pruebas: Todas las aclaraciones y soluciones dadas por el programa provenientes de algún caso de prueba.

Ruta Crítica: Camino que reune las actividades de mayor tiempo y la cual requiere de inspección cuidadosa ya que si alguna de estas actividades se retrasa el proyecto se retrasará.

Rutinas de apoyo: Son pequeños programas comunes en un sistema que se utilizan con el fin de estandarizar los procesos.

Rutinas Comunes: Procesos comunes utilizados por un sistema.

Sistemas Interactivos: Sistema de interface con el usuario necesita respuesta del usuario através de pantalla de captura.

Sistema en línea: Es el tipo de sistema del cual se obtine una respuesta inmediata através de pantallas.

Sistema de Tiempo Real: Son aquellos sistemas que responden a una solicitud en un tiempo inmediato bajo circunstancias que ocurren en el ambiente real.

Sistemas de telecominicaciones: Relativos a la transmisión de datos a grandes distancias por medio de instaladores telefónicos.

Sistema Operativo: Conjunto de intrucciones que definen la interfase entre un programador y la máquina para disponer de los recursos como: memoria, dispositivos perifericos, etc.

Sistemas Multiusuarios: Sistemas que dan la facilidad de compartir datos sin riesgo de alterar la información.

Software: Es la parte intangible de la computación.

Sumario: Tipo de documento que narra el sistema en forma completa.

Tiempo Calendario: Son todas las actividades adicionales pero necesarias para producir un proyecto de software.

Tiempo Compartido: El uso de un dispositivo para dos o más objetivos intercalados que operan simultáneamente.

Tiempo de ejecución: Tiempo que un programa necesita para producir resultados.

Tiempo Estimado: Tiempo aproximado en que una tarea específica habrá de terminarse.

Tiempo de Holgura: Tiempo en el que un acontecimiento puede prolongarse como punto máximo.

Tiempo Real: Es el tiempo en que en la fecha actual ha consumido alguna actividad.

TMt: Tiempo más tardío para un acontecimento.

TMT: Tiempo más temprano de ocurrencia.

Usuarios: Personas que utilizan el sistema de computación para ayuda en sus negocios.

Variables de Desarrollo: Factores que afectan de alguna manera el desarrollo de un proyecto de computación.

Variaciones: Cambios en los planes del proyecto que no fueron tomados en cuenta.

CONCLUSIONES

Con el propósito de cumplir el programa de desarrollo, los proyectos se deben de planear cuidadosamente. El tiempo que se requiere para desarrollar un producto de programación puede estimarse con métodos tales como los registros históricos del esfuerzo invertido en proyectos anteriores muy similares al actualmente planeado. En otros casos, la experiencia constituye la base para elaborar la estimación y como tercer método tenemos el empleo de una fórmula matemática que relaciona las características del producto y las del personal involucrado.

Todos estos métodos nos determinan el tiempo en horas para cada actividad; sin embargo para distribuir estas horas en el calendario de trabajo, podemos utilizar métodos tales como: gráficas de barras, gráficas de puntos de referencia ó utilizar el método más completo que permite identificar actividades, tiempo en horas, interrelaciones, identificación de ruta crítica que deberá ser supervisada cuidadosamente por parte de la gerencia para terminar el proyecto en tiempo, este método es conocido como PERT.

La sesión de revisión constituye una actividad de equipo; sin embargo, como regla general, la dirección de la gerencia no participan directamente en la sesiones. En lugar de esto, se entregan informes a los gerentes que resumen estas sesiones.

Estos informes detallan la naturaleza de la revision, indican responsable, variaciones según lo planeado, problemas presentados y acciones que se ha decidido emprender.

Se concluye que los métodos para estimar costos tienen dos divisiones principales de acuerdo a la jerarquización con la que se conciba, a saber: estimación jerarquíca hacia arriba y jerarquíca hacia abajo.

Los métodos para estimar costos tanto para la fase de desarrollo como para la fase de mantenimiento son conocidos como COCOMO, DELFI, de nuevo la experiencia JUICIO EXPERTO y por último se concluye que la actividad de asignación de trabajo es utilizada también como un método para estimar costos.

Como resultado de cada fase de desarrollo de un producto se tendrá la generación de la documentación correspondiente, la cual podrá ser utilizada en caso de tener algún problema con el usuario, con el personal técnico así como con el diseñador.

Las estimaciones en proyectos de software son la base para el éxito de la organización así como la proyección profesional del individuo que la elabora, se espera que esta pequeña introducción sea de gran ayuda para todas las personas que no tienen una idea clara de esta importante parte del estudio de factibilidad de un proyecto de software.

BIBLIOGRAFIA

TITULO : ANALISIS Y DISEÑO DE SISTEMAS DE INFORMACION
AUTOR : JAMES A. SENN
EDICION : 1982

TITULO : CONTROL Y AUDITORIA DEL COMPUTADOR
EDICION : INSTITUTO MEXICANO DE CONTADORES

TITULO : INGENIERIA DE SOFTWARE
AUTOR : RICHARD FAIRLEY
EDITORIAL : MC. GRAW HILL
EDICION : MEXICO D.F. 1985

TITULO : MANAGEMENT OF COMPUTER OPERATIONS
AUTOR : ISRAEL BOROVITS
EDITORIAL : PRENTICE HALL
EDICION : 1984

TITULO : SISTEMAS DE INFORMACION, TEORIA Y PRACTICA
AUTOR : JOHN G. BURCH Jr. & FELIX R. STRATER Jr
EDITORIAL : LIMUSA
EDICION : MEXICO D.F. 1985

TITULO : THE RISE OF MANAGERIAL COMPUTING
AUTOR : JOHN ROCKART AND CHRISTIANE V. BOLLEN.

Ing. Aurelio Ramírez Granados
tel. 52-25-25