

Primer y segundo parcial

- Examen 60%
- Práctica y tareas 30%
- Participaciones 10%

Tercer parcial

- Proyecto final (Documentación y aplicación) 70%
- Participaciones 10%
- Tareas 20%

l.mendezs@ipn.mx

Atributos del Software

- Mantenimiento: El software debe construirse de tal forma que pueda evolucionar para satisfacer las necesidades cambiantes de los usuarios. El cambio de software es un requerimiento inevitable de un entorno empresarial variable.
- Confiabilidad: No causa daño físico ni económico en caso de falla del sistema.
- Seguridad: Debe evitar la posibilidad de que usuarios mal intencionados puedan acceder al sistema.
- Eficiencia: El software no tiene que desperdiciar recursos del sistema, como memoria y los ciclos del procesador. Incluye la capacidad de procesamiento, utilización de memoria, etc.
- Aceptabilidad: El software tiene que ser aceptables al tipo de usuarios para quienes se diseña. Necesita ser comprensible, utilizable y compatible con otros sistemas que ellos usan.

Exposición:

Investigación del tema

Presentación electrónica, subir a classroom, mínimo un día antes

Exposición, máximo 10 minutos, 5 preguntas.

- Portada (Nombre del tema, integrantes, logos, unidad de aprendizaje, fecha)
- Diapositivas enumeradas
- Referencias

07/10/2020

Requerimiento

Es una condición o capacidad que debe exhibir o poseer un sistema para satisfacer un contrato, estándar, especificación, u otra documentación formalmente impuesta. Es una función del sistema

que solventa las necesidades del usuario. “Es una característica del sistema o una descripción de algo que el sistema es capaz de hacer con el objeto de satisfacer el propósito del sistema.”

Documentos de requerimientos:

Existen dos documentos que emanan del análisis de requerimientos

1. **Definición de requerimientos:** Es un documento que debe escribirse en términos que el cliente puede entender. Es decir, este documento es un listado completo de todas las cosas que el cliente espera que haga el sistema propuesto. Este documento es escrito en forma conjunta por el cliente y el desarrollador.
2. **Especificación de requerimientos:** Documento que reitera la definición de los requerimientos en los términos técnicos apropiados para el desarrollador del diseño de un sistema. En la contrapartida técnica al documento de definición de requerimientos y es escrito por los analistas de requerimientos. A veces un único documento puede servir para ambos propósitos, lo que lleva a un documento aun entendimiento común entre clientes, analistas de requerimientos y diseñadores.

Clasificación:

- **Requerimientos funcionales:**

Debe llevar un verbo antes. Describen la funcionalidad o los servicios que se espera que el sistema proveerá. Dependen del tipo de software que se desarrolló y de los posibles usuarios. Cuando se expresan como requerimiento del sistema describen con detalle la función de este, sus entradas y salidas, excepciones, etc.

- **Requerimientos no funcionales:**

No afecta la funcionalidad del sistema, como las interfaces gráficas, seguridad, conectividad. Se trata de requisitos que no se refieren directamente a las funciones específicas suministradas por el sistema (características de usuario), sino a las propiedades del sistema: rendimiento, seguridad, disponibilidad.

Clasificación:

- Del producto: Especifican comportamiento del producto, Ej.: de desempeño en la rapidez de ejecución del sistema, cuanta memoria se requiere; los de fiabilidad que fijan la tasa de fallas para el sistema sea aceptable, los de portabilidad y de usabilidad
- Organizacionales: Se derivan de las políticas y procedimientos existentes en la organización del cliente y del desarrollador. Ej.: estándares en los procesos que deben utilizarse, requerimientos de implementación como los lenguajes de programación o el método de diseño a utilizar.
- Externos: Cubre todos los requerimientos que se derivan de los factores externos al sistema y de su proceso de desarrollo. Ej.: requerimientos de interoperabilidad, requerimientos legales, requerimientos éticos.

- **Requerimientos del dominio**
- **Requerimientos del usuario**

- **Requerimientos del sistema**

Técnicas de recolección:

Se utilizan una variedad de métodos a fin de recopilar los datos sobre una situación existente, como entrevistas, cuestionarios, inspección de registros y observación.

- Observación: La observación directa del caso en estudio proporciona una forma objetiva de conseguir requerimientos. No es dependiente de las personas. Se estudian los hechos son intermediarios.
- Entrevista: Una persona solicita información a otra para obtener datos sobre un hecho específico.

Puede ser:

- Estructurada: Preguntas previamente elaboradas y en el mismo orden se realizan.
- No estructurada: Preguntas abiertas de mayor libertad para el entrevistado.
- Encuesta: Se realiza en forma escrita y no requiere la presencia del entrevistador. Por lo general se compone de preguntas cerradas con varios ítems donde solo se puede escoger uno de entre ellos. Se realiza a un grupo numeroso de persona con características similares.
- Revisión documental: Esta técnica se puede considerar parte de la observación. Se consultan documentos empresariales y se analizan buscando requerimientos

Ejemplos:

- Manuales
 - Formularios
 - Hojas de cálculo
 - Documentos varios
- Cuestionario:

Preguntas: Abiertas y cerradas

¿QUÉ TIPOS DE PREGUNTAS PUEDE HABER?

- Básicamente, podemos hablar de *dos tipos de preguntas*: "cerradas" y "abiertas".
- Las preguntas "cerradas" contienen categorías o alternativas de respuesta que han sido delimitadas.
- Las preguntas "cerradas" pueden ser dicotómicas (dos alternativas de respuesta) o incluir varias alternativas de respuesta.

EJEMPLOS DE PREGUNTAS CERRADAS DICOTÓMICAS SERÍAN:

- ¿Estudia usted actualmente?
- () Sí
- () No



- Ejemplos de preguntas "cerradas" con varias alternativas de respuesta serían:
- ¿Cuánta televisión ves los domingos?
- () No veo televisión
- () Menos de una hora
- () 1 o 2 horas
- () 3 horas
- () 4 horas
- () 5 horas o más

- Hay preguntas “cerradas”, donde el respondiente puede seleccionar más de una opción o categoría de respuesta.

EJEMPLO

- Esta familia tiene:
- ¿Radio?
- ¿Televisión?
- ¿Videocasetera?
- ¿Teléfono?
- ¿Automóvil o camioneta?
- Ninguno de los anteriores
- Algunos respondientes pudieran marcar una, dos, tres, cuatro o cinco opciones de respuesta.

Las categorías no son mutuamente excluyentes. Otro ejemplo sería la siguiente pregunta:



LAS PREGUNTAS “ABIERTAS” NO DELIMITAN DE ANTEMANO LAS ALTERNATIVAS DE RESPUESTA. POR LO CUAL EL NÚMERO DE CATEGORÍAS DE RESPUESTA ES MUY ELEVADO, EN TEORÍA ES INFINITO.

EJEMPLOS

¿Por qué asiste a psicoterapia?

- _____

¿Qué opina de la novela de televisión “la tempestad”?

- _____

De qué manera la directiva de la empresa ha logrado la cooperación del sindicato para el proyecto de calidad?

- _____



¿ USAMOS PREGUNTAS ABIERTAS O CERRADAS?

- Cada cuestionario obedece a diferentes necesidades y problemas de investigación, lo que origina que en cada caso el tipo de preguntas a utilizar sea diferente. Algunas veces se incluyen solamente preguntas "cerradas",
- otras veces únicamente preguntas "abiertas" y en ciertos casos ambos tipos de preguntas. *Cada clase de pregunta tiene sus ventajas y desventajas.*
- *Las preguntas "cerradas" son fáciles de codificar y preparar para*

-
- *Las preguntas "cerradas" son fáciles de codificar y preparar para su análisis. Asimismo, estas preguntas requieren de un menor esfuerzo por parte de los respondientes.* quest5.jpg
 - *Para poder formular preguntas "cerradas" es necesario anticipar las posibles alternativas de respuesta*
 - *Las preguntas "abiertas" son particularmente útiles cuando no tenemos información sobre las posibles respuestas de las personas o cuando esta información es insuficiente.*
 - *La elección del tipo de preguntas que contenga el cuestionario depende del grado en que se puedan anticipar las posibles respuestas, los tiempos de que se disponga para codificar y si se quiere una respuesta más precisa o profundizar en alguna cuestión.*

¿CÓMO DEBEN SER LAS PRIMERAS PREGUNTAS DE UN CUESTIONARIO?

- Es conveniente iniciar con preguntas neutrales o fáciles de contestar, para que el respondiente vaya adentrándose en la situación. No se recomienda comenzar con preguntas difíciles de responder o preguntas muy directas.
- Cuando construimos un cuestionario es indispensable que pensemos en cuáles son las preguntas ideales para iniciar. Éstas deberán lograr que el respondiente se concentre en el cuestionario.

○ *La entrevista es una técnica en la que una persona (entrevistador) solicita información de otra o de un grupo (entrevistados, informantes), para obtener datos sobre un problema determinado.*

○ *Entrevista es una serie de preguntas realizadas personalmente.*

○ *Presupone, pues, la existencia al menos de dos personas y la posibilidad de interacción verbal.*

CUMPLE CON ALGUNAS DE ESTAS FUNCIONES:

- Obtener información de individuos o grupos
- Influir sobre ciertos aspectos de conductas (opiniones, sentimientos, comportamientos) o
- Ejercer un efecto terapéutico
- Con el análisis e interpretación de los resultados, el entrevistador sistematiza, ordena, relaciona y extrae conclusiones relativas al problema estudiado.

Para realizar una entrevista

- a) Entrevistas formales
- b) Entrevistas informales
- **En la entrevista formal**, dirigida o estructurada, las preguntas están preestablecidas y se recogen en un cuestionario.
- **En la informal**, libre o no estructurada, las preguntas se determinan durante el desarrollo mismo de la entrevista, obviando el carácter dirigido de la anterior.

La entrevista estará en tres categorías:

- Completa: llenada en su totalidad por el entrevistado.
- Rechazada: cuando el seleccionado se niega a responderla.
- Desechada: cuando no se localiza al seleccionado por tercera vez.

I. ORGANIZACIÓN DEL TRABAJO DE ENTREVISTAS

- *Deben planificarse pequeños detalles o instrumentos que llevan a la recopilación de datos.*
- *Hay que preparar una carta de presentación que indique los objetivos de la investigación y que determine tanto el entrevistador como al centro de investigación.*

+

cu

QUÉ CARACTERÍSTICAS DEBEN TENER LAS PREGUNTAS?

- *Las preguntas deben ser claras y comprensibles para los respondientes*
- *Las preguntas no deben incomodar al respondiente*
- *Las preguntas no deben inducir las respuestas*
- *El lenguaje utilizado en las preguntas debe ser adaptado a las características del respondiente (tomar en cuenta su nivel educativo, socioeconómico, palabras que maneja, etcétera).*

Modelos del desarrollo del software

1. Modelo en cascada
2. Modelo por prototipos

Se permite que algunas o todas las partes del sistema se construyan rápidamente y generan un acuerdo más efectivo entre los desarrolladores y los clientes de los requerimientos que se piden.

Diagramas de Flujo de Datos (DFD)

Un DFD es un diagrama en forma de red que representa el flujo de datos y las transformaciones que se aplican sobre ellos al moverse desde la entrada hasta la salida del sistema. Se utiliza para modelar las funciones del sistema y los datos que fluyen entre ellas a distintos niveles de abstracción.

Símbolos del DFD

Gane/Sarson	Yourdon/De Marco
Entidad Externa	Entidad Externa
Proceso	Proceso
Flujo de Datos	Flujo de Datos
Almacén de Datos	Almacén de Datos

Procesos

Un **proceso** puede interpretarse como una función que debe llevar a cabo el sistema. Debe ser capaz de generar los flujos de datos de salida a partir de los flujos de datos de entrada y de una información local.

Un proceso se identifica mediante un número y un nombre, que deben ser únicos en el conjunto de DFD's que representan el sistema. El nombre debe ser breve y lo más representativo posible de la función que describe. Normalmente se forma por un verbo y un sustantivo.



Almacenes de datos

Un **almacén de datos** representa información del sistema almacenada en forma temporal.

Es un depósito lógico de almacenamiento que puede representar distintos tipos de información física (una bandeja con papeles, un archivador manual, un archivo en una computadora o una base de datos).



Entidades externas

Una **entidad externa** representa un generador o consumidor de información del sistema, pero no pertenece al mismo.

Puede representar un subsistema, persona, departamento,

Flujo de datos

Se interpretan como un camino a través del cual viajan datos de composición conocida de una parte del sistema a otra.

Son el medio de conexión de los restantes componentes del DFD.

Se representan por arcos dirigidos, en donde la flecha indica la dirección de los datos.

Deben tener un nombre o rótulo que los identifique.

Entidades externas

- Normalmente, las entidades externas sólo deberían aparecer

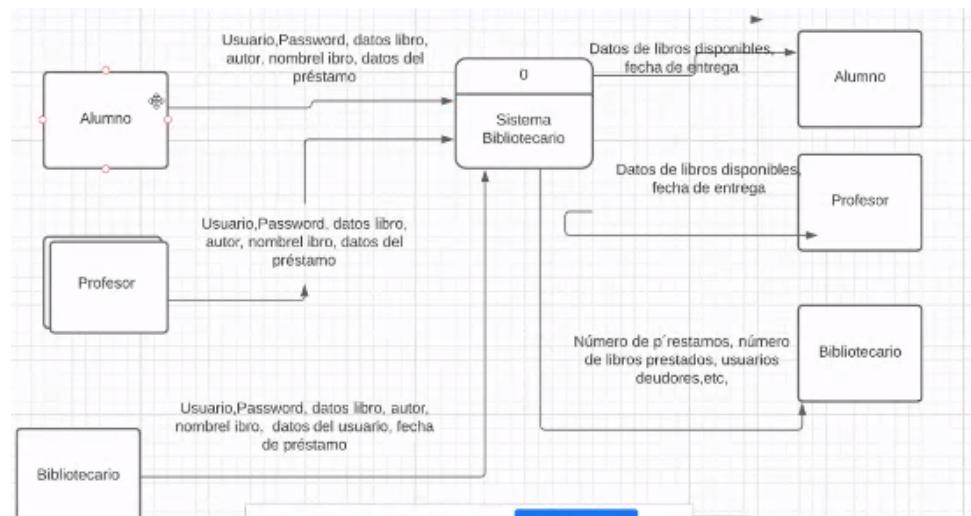
Flujo de datos

Los flujos de datos que conectan componentes de un DFD deben respetar las siguientes restricciones:

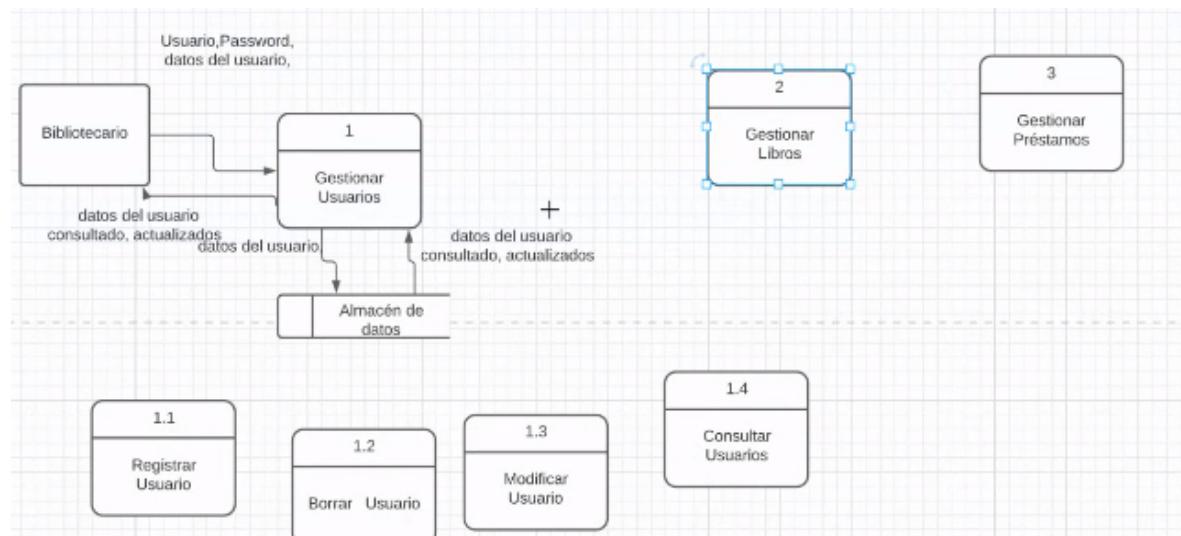
Destino Origen	Proceso	Almacén	Entidad Externa
Proceso	Si	Si	Si
Almacén	Si	No	No
Entidad Externa	Si	No	No

Ejemplos

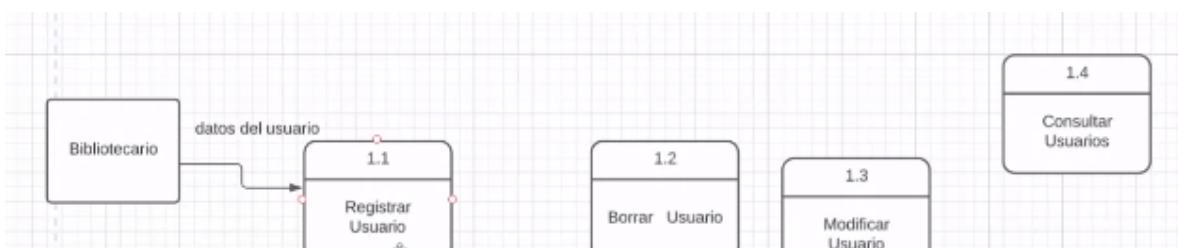
Diagrama de contexto



Nivel 1



Nivel 2



Calendarización

Es una actividad que distribuye estimaciones de esfuerzo a través de la duración planificada del proyecto, al asignar el esfuerzo a tareas específicas de ingeniería del software. Es la culminación de una actividad de planificación que es un componente principal de la gestión del proyecto de software.

Principios básicos:

- Compartimentación: El proyecto debe dividirse en compartimentos en varias actividades, acciones y tareas manejables.
- Interdependencia: Se debe determinar la interdependencia de cada actividad, acción o tarea compartimentada.
- Asignación de tiempo: A cada tarea se le debe asignar cierto número de unidades de trabajo (Ej: personas-día de esfuerzo).
- Validación del esfuerzo: El gestor del proyecto debe asegurarse de que, en un tiempo dado, no se han asignado más que el número de personas calendarizadas.
- Definición de responsabilidades: Asignar a una miembro del equipo.
- Definición de resultados: Toda tarea debe tener un resultado definido (Ej: Diseño de un módulo).
- Definición de hitos (significa tener un logro importante): Cualquier tarea o grupo de tareas debe estar asociado con un hito de proyecto. Un hito se logra cuando se ha revisado la calidad de uno o más productos de trabajos, y se ha aprobado.

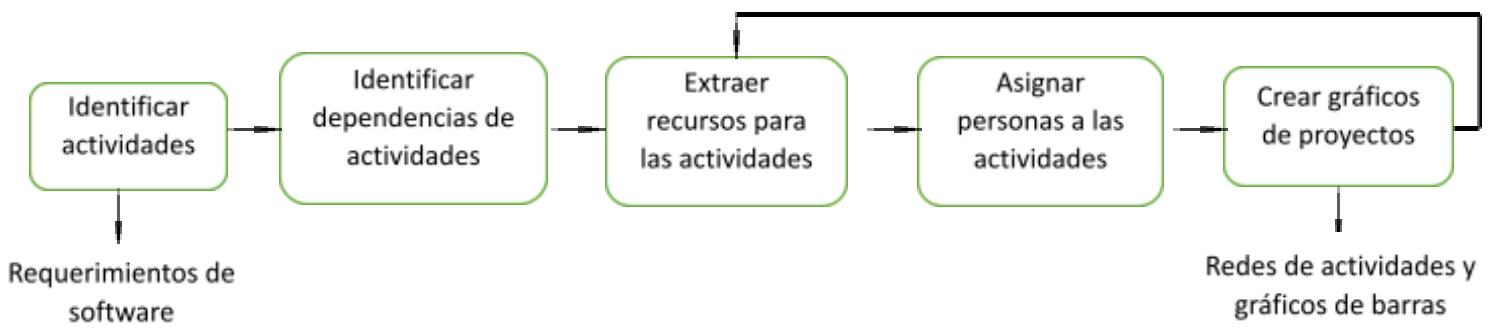
Recomendaciones:

- Parta el proyecto en tareas y estime el tiempo y los recursos requeridos para completar cada tarea.
- Organice las tareas concurrentemente, para optimizar la fuerza de trabajo.
- Minimizar las dependencias entre las tareas para evitar los retrasos.
- Se recomienda asignar entre 30% y 50% adicional al tiempo estimado por actividad.

Problemas en la calendarización:

- No se puede calcular la dificultad de un problema con exactitud y por ende tampoco el costo necesario para resolverlo.
- La productividad no es proporcional al número de personas.
- Agregar personas a un proyecto avanzado puede provocar demoras por la sobrecarga de comunicación.
- Lo inesperado siempre pasará, siempre tener planes de contingencia.

Ejemplo de calendarización:



Cronogramas:

1. Diagrama de Gantt

Muestra la programación vs tiempo calendario.

Uno por proyecto o uno por cada función.

Diamantes (rombos) marcan hitos.

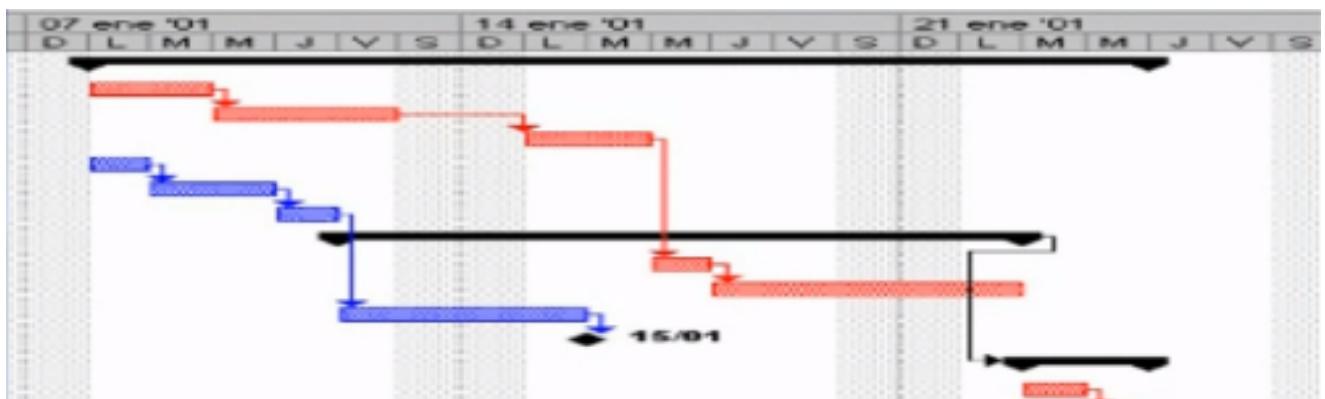
Hitos: actividades

Tareas	Semana 1	Semana 2	Semana 3	Semana 4	Semana 5
Tarea 1					
Tarea 2					
Tarea 3					

Seguimiento de la calendarización realizado:

- Reuniones para valorar el estado.
- Evaluación de resultados
- Hitos
- Comparar fechas: tentativa-real, inicio de tarea usando tabla de tareas.

Diamantes (rombos) marcan hitos.



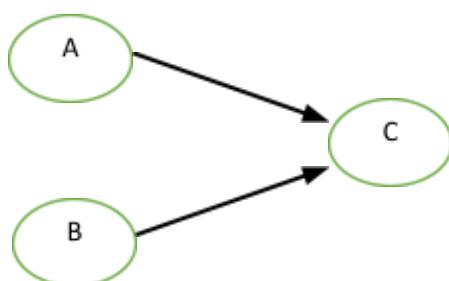
Red de tareas o actividades

Red de tareas:

Representación gráfica del flujo de tareas de un proyecto. Muestra las principales tareas de la ingeniería de software, sus dependencias y si se pueden ejecutar en paralelo.

Dos posibles enfoques:

- Calendarización macroscópica
- Calendarización detallada



A y B deberán completarse antes de que se inicie la actividad C.

Red PERT

Es una técnica que nos permite planificar la consecución de un objeto de un proyecto en general. No solo es planificación si no también se puede medir el avance de su proyecto u objeto. Una vez evaluado se puede tomar decisiones y tomar correctivos, permite planificar y mejorar lo planificado.

Aplicaciones:

- Construcción
- Desarrollo de sistemas informáticos
- Controles y auditorías financieras
- Instalación de plantas industriales

Requisitos para la aplicación de Red PERT:

- Proyectos grandes que tengan gran cantidad de actividades
- Que sean proyectos dinámicos, que estén sujetos a cambios continuos
- Proyectos por lo general de gran inversión económica (costosos)
- En proyectos urgentes

Proyectos no aplicables (Donde no se puede usar PERT):

- En proyectos que tengan una trayectoria lineal u horizontal
- Proyectos que sean pequeños

Ventajas:

- Conocer el tiempo de inicialización u finalización del proyecto
- Saber el tiempo y costo mínimo de un proyecto
- Permite una flexibilidad y un refinamiento en los proyectos
-

Terminología:

- Actividades:
 - Se representan por una flecha
 - No importa la magnitud de la flecha ni su dirección.
 - Lo importante es la secuencia o la relación de las actividades.
 - Toda actividad tiene duración y es una parte del proyecto.
- Eventos:
 - Están representados por círculos
 - Los eventos no tienen duración, llamados también HITOS.
 - Permiten marcar puntos en el tiempo
 - Existen eventos iniciales y finales

- Se les asigna un número
- iii. Enumeración de eventos
- Existen normas para enumerar los eventos
 - Es preferible enumerar de izquierda a derecha y de arriba hacia abajo
 - El evento de finalización debe ser mayor al de inicio
 - Existen reglas para enumerar eventos
 - Para enumerar un evento deberá enumerarse antes los eventos que están en los extremos de las flechas o de las actividades que concurren o llegan a dicho evento

Relaciones:

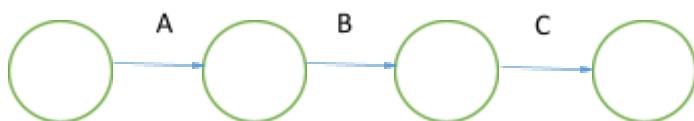
La gráfica permite claramente la secuencia de las relaciones y estas pueden ser:

1. Precedencia o antecedencia

A la actividad A no le antecede ninguna actividad

A la actividad B le antecede la actividad A

A la actividad C le antecede la actividad B



2. Secuencia

A la actividad A le sigue ninguna actividad

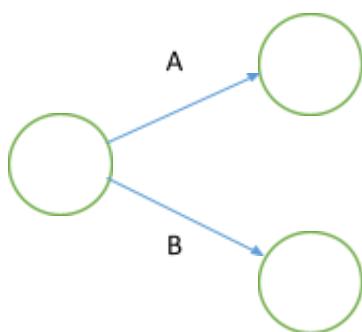
A la actividad B le sigue la actividad A

A la actividad C no le sigue ninguna actividad.

3. Concurrencia

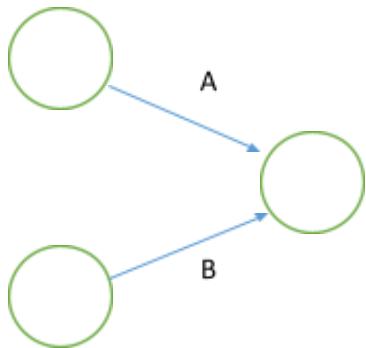
A y B salen del mismo evento

Salida



Llegada

A y B llegan al mismo evento



4. Actividades ficticias

No tienen duración o su tiempo de duración es igual a cero

Es un artificio gráfico, sirve para representar relaciones complejas en una Red

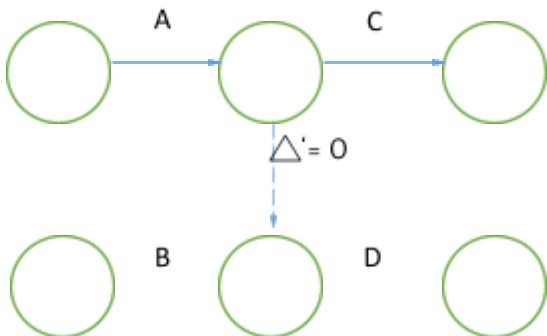
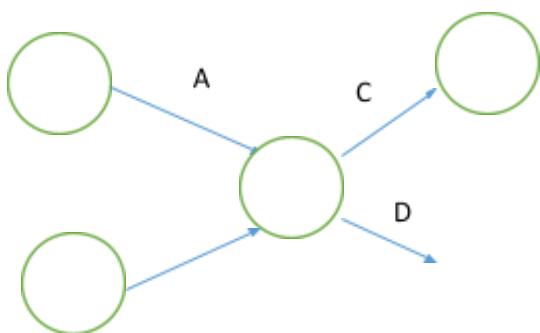
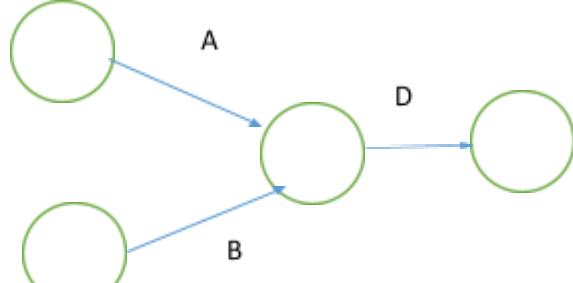
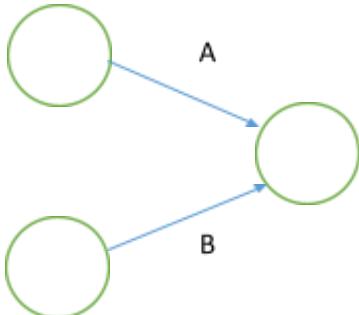
Ejemplo: Condiciones de las relaciones

A y B son concurrentes de llegada

A D le antecede A y B

C y D tienen concurrencia de salida

A C le antecede solo A



B
PERT/TIEMPO



Es importante el cálculo de tiempos estimados para cada una de las actividades del proyecto para determinar la duración total del mismo y así tener la aplicación o la negación de su realización.

Tiempo esperado (Te): Es el tiempo de duración de cada actividad y estos tiempos son proporcionales por especialistas en cada una de las materias.

Se obtiene a través de estadística y la probabilidad (BETA)

$$Te = (a + 4m + b) / 6$$

Donde:

a: Tiempo más optimista, es el tiempo en que puede durar una actividad en las mejores condiciones posibles.

b: Tiempo más pesimista, que es el tiempo más largo que puede demorarse una actividad en las peores condiciones posibles.

m: Tiempo más probable o medio, es el tiempo en el que puede desarrollarse una actividad en condiciones normales.

Ejemplo:

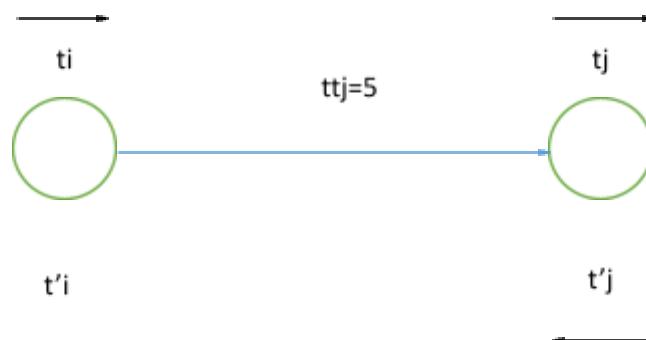
$$a = 4 \text{ h} \quad Te = (4 + 4*8 + 16) / 6$$

$$b = 16 \text{ h} \quad Te = 8.7 \text{ h}$$

$$m = 8 \text{ h}$$

Tiempos más próximos y más tardíos de una actividad:

- Tiempo más próximo: Es la fecha más temprana de inicio de la actividad y se calcula de izquierda a derecha o del inicio al final.
- Tiempo más tardío: Es fecha más lejana del inicio de una actividad, se calcula del final al inicio.



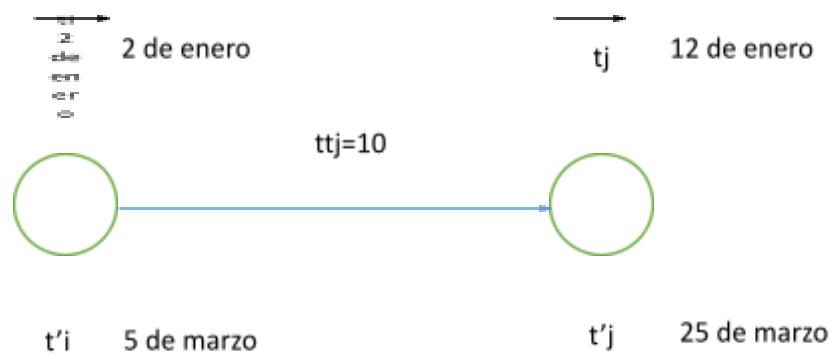
t_i = tiempo más próximo de inicio

$t'i$ = tiempo más tardío de inicio

t_j = tiempo más próximo de finalización

$t'j$ = tiempo más tardío de finalización

Ejemplo:



Holgura de eventos: Es la diferencia entre el tiempo más tardío menos el tiempo más próximo de un evento.

Ejem.:

$$\begin{array}{ll} t_3 = 8 & H_i = 15 - 8 \\ t'3 = 15 & H_i = t'i - t_i \end{array}$$

EVENTO	$t'i$	t_i	H_i	SITUACIO	
				C	NC
1	0	0	0	X	
2	20	14	6		X
3	15	8	7		X
4	20	20	0	X	
5	29	29	0	X	
6	65	40	25		X
7	45	45	0	X	
8	82	82	0	X	

Holgura de actividades: Se define como la flexibilidad de realización de ciertas actividades, cuando una actividad puede iniciar lo más pronto posible o concluir lo más tarde posible.

$$H_{ij} = t'j - \{ t_i + t_{ij} \}$$

ACTIVIDADES		t _{ij}	t _i	t'j	t _{ij} - { t _i + t _{ij} }	H _{ij}	SITUACION	
Código	(l)						C	NC
A	1,2	14	0	20	20-{0+14}	6		X
B	1,3	9	0	15	15-{0+9}	7		X
C	1,4	20	0	20	20-{0+20}	0	X	
D	1,7	17	0	45	45-{0+17}	28		X
A'	2,4	0	14	20	20-{14+0}	6		X
E	2,6	6	14	65	65-{14+6}	45		X
F	3,5	14	8	29	29-{8+14}	7		X
G	3,7	21	8	45	45-{8+21}	16		X
H	4,5	9	20	29	29-{20+9}	0	X	
I	5,6	11	29	65	65-{29+11}	25		X
J	5,7	16	29	45	45-{29+16}	0	X	
K	5,8	15	29	82	82-{29+15}	38		X
L	6,8	17	40	82	82-{40+17}	25		X
M	7,8	37	45	82	82-{45+37}	0	X	7

$$\text{Ruta crítica de actividades (RCA)} = C - H - J - M$$

Calendarización

Pueden utilizar técnicas/herramientas calendarización de proyectos.

- PERT (Técnica de evaluación y revisión de programa)
- CPM (Método de la Ruta Crítica)

Identificar todas las actividades que involucra el proyecto, lo que significa, determinar relaciones de precedencia, tiempos técnicos para cada una de las actividades.

Construir una red con base en nodos y actividades (o arcos, según el método más usado), que implican el proyecto.

Analizar los cálculos específicos, identificando las rutas críticas y las holguras de los proyectos.

En términos prácticos la ruta crítica se interpreta como la dimensión máxima que puede durar el proyecto y las diferencias con las otras rutas que no sean la crítica, se denominan tiempos de holgura.

Información etapas tempranas:

- Estimación de esfuerzo
- Descomposición de la función del producto
- Selección del modelo de proceso y conjunto de tareas apropiadas.
- Descomposición de Tareas

Modelo COCOMO

Los costos se pueden calcular como una función matemática basada en atributos de productos, proyectos y procesos, cuyos valores son calculados por administradores de proyectos. La función se basa en un estudio histórico de datos de costos.

LOC (tamaño de código) utiliza estimación de costos.

Fue desarrollado y presentado en 1981 por Barry W. Boehm.

Trata de establecer una relación matemática que permita estimar el esfuerzo (hombre-mes) y tiempo requerido para desarrollar un proyecto.

Basado en una base de datos de costos (con más de 60 proyectos diferentes).

Existen tres niveles:

Nivel Básico

Nivel Intermedio

Nivel Detallado

Existen tres modelos:

Orgánico

Semilibre

Fuertemente restringido

Nivel Básico

Es adecuado para realizar estimaciones de forma rápida, aunque son de gran precisión.

No tiene en cuenta los diferentes atributos que afectan al proyecto como: calidad, experiencia del personal, restricciones de hardware, utilización de técnicas modernas y herramientas de desarrollo.

Nivel Intermedio

Los factores antes mencionados se consideran como adicionales al costo total del proyecto.

Nivel Detallado

Se considera cómo estos factores afectan dentro de las diferentes fases individuales que componen el proyecto.

El modelo COCOMO

- El factor principal sobre el que se basan las estimaciones es el tamaño del producto, es decir, el *número de instrucciones fuente desarrolladas*.**
- La cantidad de instrucciones fuente se deben estimar por experiencia, por analogía con otros proyectos semejantes, o por otros datos que se posean.**

El modelo COCOMO

- En el modelo de desarrollo de software se planifican solo las fases comprendidas desde el análisis hasta la implantación, (La fase de estudio preliminar no se considera).
- Los parámetros estimados no incluyen los correspondientes a las actividades de formación de los usuarios, planificación de las instalaciones y trabajos de conversión.

El modelo COCOMO

- Los indicadores de planificación que se pueden obtener con este método son:

- **Esfuerzo** (hombre-mes)
- **Tiempo de desarrollo** (meses)
- **Personal necesario** (hombres)
- **Productividad** (inst/hombre-mes)
- **Costo** (pesos)

El modelo COCOMO

La unidad de esfuerzo Hombre-Mes supone un total de 152 horas de trabajo por persona, en base a la experiencia práctica y a consideraciones sobre vacaciones, permisos, enfermedad, etc.

Hombres-Mes x 152 = Hombres-Hora

Hombres-Mes x 19 = Hombres-Día

Hombres-Mes / 12 = Hombres-Año

El modelo COCOMO

Modelos de desarrollo de software:

Modelo Orgánico o Familiar



Modelo Semilibre

Modelo Fuertemente restringido

Modelo Orgánico o Familiar (1)

- El equipo de desarrollo es relativamente pequeño y se desenvuelven en un entorno altamente familiar.
- La gran mayoría de la gente relacionada con el proyecto tiene una amplia experiencia en otros proyectos relacionados con la misma organización
- Tienen un buen conocimiento de cómo el sistema bajo desarrollo, contribuirá a los objetivos de su organización.

Modelo Orgánico o Familiar (2)

- La mayoría de las personas pueden contribuir de forma efectiva a la terminación puntual de cada una de las etapas sin generar grandes necesidades de comunicación para determinar con precisión las tareas que cada uno debe desarrollar en el proyecto
- El equipo de trabajo puede negociar con facilidad la modificación de algunas de las especificaciones para hacer más fácil este desarrollo.

Modelo Orgánico o Familiar (3)

- Entorno de desarrollo estable, con poco desarrollo concurrente de nuevo Hardware asociado.
- Mínimas necesidades de introducir algoritmos innovadores o nuevas arquitecturas de proceso.
- Un trabajo de proyecto relativamente pequeño. Muy pocos proyectos desarrollados de modo orgánico sobrepasan los 50 MF (50 000 instrucciones fuente).
 - Proyectos en Modo Orgánico de mayor tamaño pueden desarrollarse utilizando software ya existente.

Modelo Semilibre (1)

- Representa un estado intermedio entre el modo orgánico y el modo fuertemente restringido
- Todos los miembros del equipo de diseño tienen un nivel medio de experiencia en sistemas relacionados con el proyecto
- El equipo de desarrollo esta formado por una mezcla de gente experta e inexperta.

Modelo Fuertemente restringido (1)

- Debe desarrollarse sometido a fuertes restricciones.
- El producto debe operar en entornos de software y hardware fuertemente acoplados.
- En estos proyectos no existe la posibilidad de negociar fácilmente cambios en el software y en tal caso precisará un mayor tiempo para acomodar o asegurar que los cambios cumplan las especificaciones (mayor costo de verificación, validación y de gestión de la configuración).

COCOMO Básico

$$E = a_b (KLOC)^{b_b}$$

E= esfuerzo (hombre/mes)

KLOC= número (miles) estimado de líneas de código del proyecto.

COCOMO Básico

Proyecto de software	a_b	b_b	c_b	d_b
Orgánico	2.4	1.05	2.5	0.38
Semi-acoplado	3.0	1.12	2.5	0.35
Fuertemente restringido	3.6	1.20	2.5	0.32

COCOMO Básico

Supongamos que una empresa cualquiera desea diseñar un proyecto que gestione sus inventarios y decide desarrollarlo mediante su propio equipo de analista y programadores que anteriormente y durante muchos años, vienen desarrollando aplicaciones similares en la misma empresa.

Si un estudio inicial determina que el tamaño del producto en alrededor de 32 000 líneas de programa fuente (32 KLOC). Cuales serán las características del proyecto?.

COCOMO Básico

- Esfuerzo:

$$E = a_b (KLOC)^{b_b}$$

$$E = 2.4 (32)^{1.05} \quad E = 91 \text{ hombres-mes}$$

- Tiempo de desarrollo:

$$D = c_b (E)^{d_b} \quad D = 2.5 (91)^{0.38} = 14 \text{ meses}$$

COCOMO Básico

- Número de personas trabajando en el proyecto:

$$N = 91/14 = 6.5 \text{ hombres}$$

La cantidad de hombres nos da una medida del número equivalente de personas trabajando a tiempo completo en el proyecto.

Modelos Empíricos de Estimación

a) Modelo COCOMO:

COCOMO Básico:

- Los coeficientes ab y cb y los exponentes db y bb , con valores constantes se muestran en la Tabla siguiente:

Modelo COCOMO básico				
Proyecto de Software	a_b	b_b	c_b	d_b
Orgánico	2.4	1.05	2.5	0.38
Semiacoplado	3.0	1.12	2.5	0.35
Empotrado	3.6	1.20	2.5	0.32

Otros elementos para calcular esfuerzo, número de persona y tiempo:

Puntos de Función (1)

- Establecer el esfuerzo que conlleva el desarrollo de un producto de software, ha sido sin duda una necesidad creciente de la industria informática.
- Muchas métricas se han visto en el transcurso de los años, pero la que ha logrado resultados más relevantes ha sido el conteo de los Puntos de Función.
- Las bases de la métrica Punto de Función, fueron desarrollados en un período superior a los 5 años comenzando en 1974 por el Departamento de Servicios Computacionales de IBM.

Puntos de Función (2)

- Allan J. Albrecht (IBM, White Plains) fue el primero en publicar la técnica de Punto de Función en 1979.
- Desde fines del año 1982 el Punto de Función, ha sido usado en los Estados Unidos, Inglaterra, Nueva Zelandia, Australia y Canadá.
- Algunas Empresa que utilizan la técnica son: IBM, UNISYS, Bank of America, Bell, ITT, Xerox, General Motors, otros.

Puntos de Función (3)

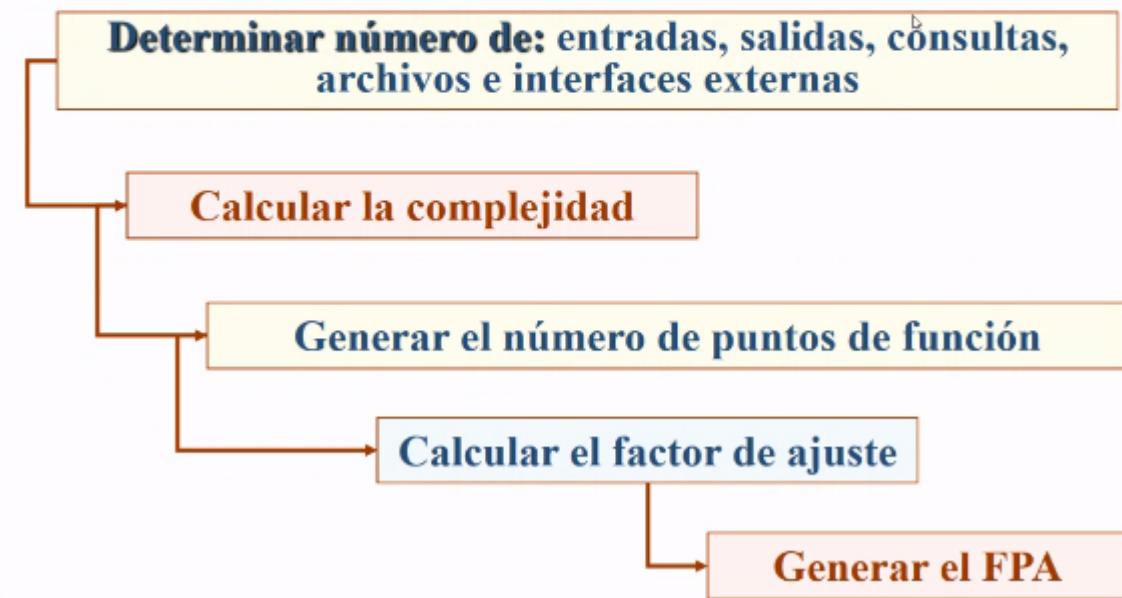
- Los objetivos de los puntos de función son:
 - Medir lo que el usuario pide y lo que el usuario recibe.
 - Medir independientemente de la tecnología utilizada en la implantación del sistema.
 - Proporcionar una métrica de tamaño que dé soporte al análisis de la calidad y la productividad.
 - Proporcionar un medio para la estimación del software.

Puntos de Función (4)

- El análisis de los puntos de función se desarrolla considerando cinco parámetros:
 - Número de Entradas de usuario
 - Número de Salidas de usuario
 - Número de Consultas (peticiones del usuario)
 - Número de archivos
 - Número de interfaces externas

Puntos de Función (5)

- Cálculo:



Puntos de Función (6)

- **Entradas de usuario:** Se cuenta cada entrada de usuario que proporciona diferentes datos orientados a la aplicación.



- Transacciones : datos introducidos para mantener archivos lógicos internos.
- Pantallas de entrada: Hay que añadir una unidad por cada función (añadir, cambiar, borrar) que mantiene un archivo lógico interno

Puntos de Función (7)

- **Salidas de usuario:** Se cuenta cada salida que proporcione al usuario información orientada a la aplicación.

- Informes
- Pantallas
- Mensajes de error/configuración
- La transferencia de datos a otras aplicaciones
- Cada gráfico distinto (tabla, diagrama de barras: se cuenta como 2 salidas)

Puntos de Función (8)

- **Consultas:** Una entrada interactiva que produce la generación de alguna respuesta del software inmediata en forma de salida interactiva.

- Búsqueda inmediata de datos
- Tutoriales
- Las Ayudas
- Las pantallas de logon que proporcionarían seguridad
- Las pantallas de menú que proporcionan una selección de pantallas y entradas para la búsqueda de datos para la pantalla llamada.

Puntos de Función (9)

• **Número de archivos:** se cuenta cada archivo maestro lógico (un grupo lógico de datos que puede ser una parte de una gran base de datos o un archivo independiente).



- Archivos maestros
- Mensajes Help actualizados por la aplicación
- Mensajes de error actualizados por la aplicación
- Archivos lógicos internos mantenidos por más de una aplicación.

Puntos de Función (10)

• **Interfaces externas:** Se cuentan todas las interfaces legibles por la máquina (archivos de datos de cinta o disco) que se utilizan para transmitir información a otro sistema.



- Bases de datos compartidas
- Archivos lógicos internos utilizados por otra aplicación
- Lista de parámetros compartidos

Líneas de Código y Puntos de Función

La relación entre las líneas de código y los puntos de función depende del lenguaje de programación que se utilice para implementar el software y de la calidad del diseño.

LOC → - Errores por LOC
- \$ por LOC
- LOC por persona-mes

PF  - Errores por PF
- \$ por PF
- PF por persona-mes

Líneas de Código y Puntos de Función

Lenguaje de programación	LDC/PF (media)
Ensamblador	320
C	128
Cobol	105
Fortran	105
Pascal	90
Ada	70
Lenguajes OO	30
...	...



Ejemplo de Puntos de Función

La Especialización en Ingeniería de Software requiere de un Sistema de Información que permita llevar el control escolar de sus alumnos:

- Se necesita un módulo de Actualizaciones (alumnos, materias y calificaciones) el cual permitirá realizar altas, bajas y cambios para cada uno de ellos.
- Otro módulo es el de consultas (alumnos, materias y calificaciones).
- El último módulo será el de reportes: solo se emitirán los reportes de "alumnos inscritos", "catálogo de materias" y "alumnos con sus calificaciones".

Ejemplo de Puntos de Función

- Los archivos que se utilizarán son: alumnos, materias y cardex.
- No. entradas:
 - 3 altas (alumnos, materias y calificaciones)
 - 3 bajas (alumnos, materias y calificaciones)
 - 3 cambios (alumnos, materias y calificaciones)
- No. salidas:
 - 3 reportes: "alumnos inscritos", "catálogo de materias" y "alumnos con sus calificaciones".

Ejemplo de Puntos de Función

- No. consultas: 3 alumnos, materias y calificaciones
 - 4 pantalla de menú de selección (pantalla principal, actualizaciones, consultas y reportes)
- No. archivos: 3 alumnos, materias y cardex
- No. interfaces externas: 0

Se considera que el sistema tiene una complejidad media en cada uno de los factores antes mencionados.

Puntos de función
Complejidad media

entradas	$9 \times 4=36$
salidas	$7 \times 5=35$
consultas	$3 \times 4=12$
Archivos lógicos	$3 \times 10=30$
Archivos de interfaz	$0 \times 7=0$
Total	113
Multiplicador	1.13
Total de PF ajustados	128 PF

- 1- **5** Requiere el sistema copias de seguridad y de recuperación fiables ?
- 2- **2** Se requiere comunicación de datos ?
- 3- **0** Existen funciones de procesamiento distribuido ?
- 4- **4** Es crítico el rendimiento?
- 5- **4** Se ejecutará el sistema en un entorno operativo existente y fuertemente utilizado?
- 6- **4** Requiere el sistema entrada de datos interactiva ?
- 7- **4** Requiere la entrada de datos interactiva que las transacciones de entrada se lleven a cabo sobre múltiples pantallas u operaciones?
- 8- **5** Se actualizan los archivos maestros de forma interactiva ?
- 9- **5** Son completas las entradas, las salidas, los archivos o las peticiones ?

- 10- **1** Es complejo el procesamiento interno ?
11- **3** Se ha diseñado el código para ser reutilizable ?
12- **3** Están incluidas en el diseño la conversión y la instalación?
13- **3** Se ha diseñado el sistema para soportar múltiples instalaciones en diferentes organizaciones ?
14- **5** Se ha diseñado la aplicación para facilitar los cambios y para ser fácilmente utilizada por el usuario ?

TOTAL: 48

**0 No influencia 1 Incidental 2 Moderado 3 Medio
4 significativo 5 Esencial**

$$\text{PF} = \text{cuenta-total} \times (0.65 + 0.01 \times (f_1 + f_2 + \dots + f_{14}))$$



$$\text{PF} = 113 \times (0.65 + 0.01 \times 48)$$



$$\text{PF} = 128$$

Puntos de función Complejidad media

entradas	$9 \times 4 = 36$
salidas	$7 \times 5 = 35$
consultas	$3 \times 4 = 12$
Archivos lógicos	$3 \times 10 = 30$
Archivos de interfaz	$0 \times 7 = 0$
Total	113
Multiplicador	1.13
Total de PF ajustados	128 PF

Componentes de la calidad

La calidad de un sistema informático puede descomponerse en factores que contribuyen a la misma.

Wilkin y Castleman (2003) describen un instrumento multidimensional (denominado QUALIT) capaz de medir la calidad de los sistemas de información entregados y diferencia los siguientes componentes:

- **Calidad del sistema.**

Juicio global sobre el grado en que los componentes técnicos del mismo proporcionan la calidad de la información y servicio requerido por los *stakeholders*.

- **Calidad de la información proporcionada a los stakeholders**(usuarios del sistema)
- **Calidad del servicio.**
-Proporcionado por el departamento de SI y el personal de soporte

Factores de calidad ISO 9126

- Funcionalidad
- Confiabilidad
- Usabilidad
- Eficiencia
- Facilidad de recibir Mantenimiento
- Portabilidad

Funcionalidad.

- Capacidad del producto de SW para proporcionar funciones que satisfacen necesidades declaradas e implícitas cuando se usa bajo condiciones especificadas.

Funcionalidad

- Esta característica se subdivide en:

- **Adecuación**

Proporcionar un conjunto apropiado de funciones para tareas y objetivos de usuario específicos.

- **Exactitud**

Proporcionar los resultados y efectos correctos o acordados, con el grado necesario de precisión.

Funcionalidad

- **Interoperabilidad**

Interactuar con uno o más sistemas especificados.

- **Seguridad de Acceso**

Proteger información y datos de manera que las personas o sistemas no autorizados no puedan leerlos o modificarlos, al tiempo que no se deniega el acceso a las personas o sistemas autorizados.

- **Cumplimiento funcional**

Adherirse a normas, convenciones o regulaciones en leyes y prescripciones similares relacionadas con la funcionalidad

Confiabilidad

- **Madurez**

Evitar fallas como resultado de fallos en el SW.

- **Tolerancia a fallos**

Mantener un nivel de prestaciones en caso de fallos de SW o de infringir sus interfaces especificados.

- **Capacidad de recuperación**

Reestablecer un nivel de prestaciones especificado y de recuperar los datos directamente afectados en caso de fallo.

- **Cumplimiento de la fiabilidad.**

Adherirse a normas, convenciones o regulaciones relacionadas con la fiabilidad.

Usabilidad

- Capacidad del producto del SW para ser entendido, aprendido, usado y ser atractivo para el usuario, cuando se usa bajo condiciones especificadas.

Usabilidad

- Esta característica se subdivide en:
- **Capacidad para ser entendido**

Permite al usuario entender si el SW es adecuado y cómo puede ser usado para unas tareas o condiciones de uso particulares.

- **Capacidad para ser aprendido**

Permite al usuario aprender sobre su aplicación.

- **Capacidad para ser operado**

Permite al usuario operarlo y controlarlo.

- **Capacidad de atracción**

• Capacidad del producto de software para ser atractivo al usuario.

- **Cumplimiento de la usabilidad**

Adherirse a normas, convenciones, guías de estilo o regulaciones relacionadas con la usabilidad.

Eficiencia.

- Capacidad del producto de SW para proporcionar prestaciones apropiadas, relativas a la cantidad de recursos usados, bajo condiciones determinadas.

Eficiencia.

- Esta característica se subdivide en:
- **Comportamiento temporal**
- Proporcionar tiempos de respuesta, tiempos de proceso y potencia apropiados, bajo condiciones determinadas.
- **Utilización de recursos**
- Usar las cantidades y tipos de recursos adecuados cuando el SW lleva a cabo su función bajo condiciones determinadas.
- **Cumplimiento de la eficiencia**
- Adherirse a normas o convenciones relacionadas con la eficiencia.

Mantenibilidad

- Capacidad del producto de SW para ser modificado. Las modificaciones podrían incluir correcciones, mejoras o adaptación del SW a cambios en el entorno, y requisitos y especificaciones funcionales.

Mantenibilidad

- Esta característica se subdivide en:
- **Capacidad para ser analizado**
- Capacidad para ser diagnosticadas deficiencias o causas de los fallos en el SW, o para identificar las partes que han de ser modificadas.
- **Capacidad para ser cambiado**
- Permite que una determinada sea implementada.
- **Estabilidad**
- Evitar efectos inesperados debidos a modificaciones de software.
- **Capacidad para ser probado**
- Permite que el SW modificado sea válido.
- **Cumplimiento de la mantenibilidad**
- Adherirse a normas o convenciones relacionadas con la mantenibilidad

Portabilidad.

- Capacidad del producto de SW para ser transferido de un entorno a otro.

Portabilidad.

- Esta característica se subdivide en:
- **Adaptabilidad**
- Adaptarse a diferentes entornos especificados, sin aplicar restricciones o mecanismos distintos de aquellos proporcionados para éste propósito por el propio SW considerado.
- **Instalabilidad**
- Capacidad para ser instalado en un entorno especificado.
- **Coexistencia**
- Para coexistir con otro SW independiente, en un entorno común, compartiendo recursos comunes.
- **Capacidad para ser reemplazado**
- Ser usado en lugar de otro producto de software, para el mismo propósito, en el mismo entorno.
- **Cumplimiento de la portabilidad**
- Adherirse a normas o convenciones relacionadas con la portabilidad.

Estándares de Calidad

- ISO 9000
- ISO 25000
- IEEE Std 1061-1998
- ISO/IEC 15939
- Moprospect

Exposición, ISO 9000: Desarrollo de software

Sistema de información:

Sistema: Es un conjunto de elementos con relaciones de interacción e interdependencia que le confieren entidad propia al formar un todo unificado. Conjunto de partes o elementos organizados y relacionadas que interactúan entre sí para lograr un objetivo.

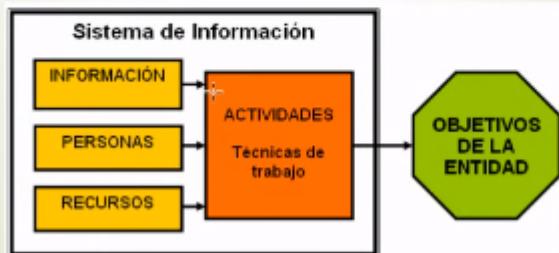
Información: Es un conjunto organizado de datos procesados que constituyen un mensaje que cambia el estado de conocimiento del sujeto o sistema que recibe dicho mensaje.

► Concepto de Sistema de Información (SI)

- Es un conjunto de elementos orientados al tratamiento y administración de datos e información, organizados y listos para su uso posterior, generados para cubrir una necesidad u objetivo.
- Es un conjunto de elementos que interactúan entre sí con el fin de apoyar las actividades de una empresa o negocio.

► Sistema de Información

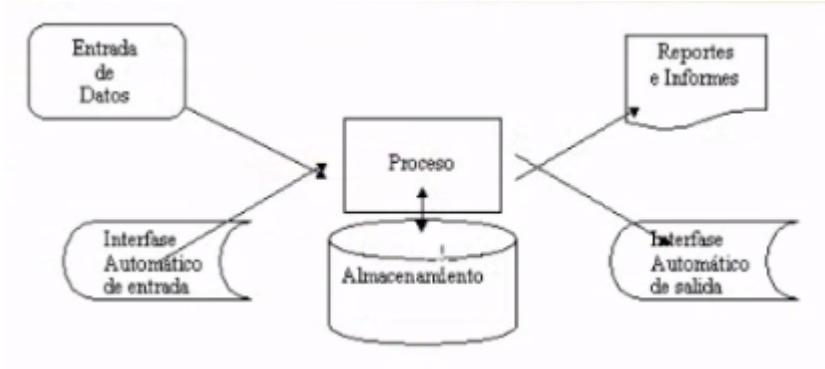
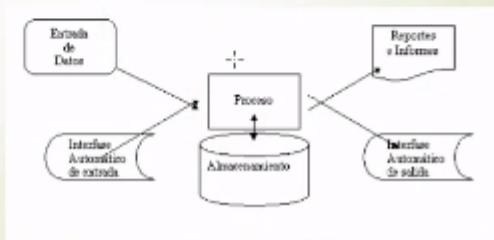
Los sistemas de información proporcionan servicio a todos los demás sistemas de una organización y enlazan todos sus componentes en forma tal que éstos trabajen con la eficiencia para alcanzar el mismo objetivo.



Sistema de Información

Actividades Básicas de un sistema:

- Entrada
- Almacenamiento
- Procesamiento
- Salida



Actividades SI

- **Entrada:** Es el proceso mediante el cual el Sistema de Información toma los datos que requiere para procesar la información. Las entradas pueden ser manuales o automáticas.
- **Almacenamiento:** El almacenamiento es una de las capacidades más importantes que tiene una computadora, ya que a través de esta propiedad el sistema puede recordar la información guardada en la sección o proceso anterior.
- **Procesamiento:** Es la capacidad del Sistema de Información para efectuar cálculos de acuerdo con una secuencia de operaciones pre-establecidas.
- **Salida:** La salida es la capacidad de un Sistema de Información para sacar la información procesada o bien datos de entrada al exterior.

Calidad de Sistemas de información

Es un método organizado para recolectar, almacenar y reportar la información sobre la calidad para ayudar a tomar decisiones en todos los niveles dentro de una empresa u organización.

Calidad de Sistemas de información

- Los sistemas de información constituyen plataformas de apoyo para el desarrollo de las actividades en cualquier organización.
- Recolección de datos
- Evaluación de la calidad y relevancia de los datos
- Manipulación o proceso de los datos
- Almacenamiento
- Distribución
- Presentación

► Elementos



Calidad de un sistema de información

- "Concordancia con los requisitos funcionales y de rendimiento explícitamente establecidos los estándares de desarrollo explícitamente documentados y con las características implícitas que se espera de todo software desarrollado profesionalmente." [Pressman]



¿Quién define la calidad?

La calidad del software la define o avala una gestión de la calidad del software por ejemplo: ISO 9000, esto como política de calidad, se entiende como un conjunto de actividades de la función general de la dirección que determina la calidad, los objetivos, el control de calidad.

Algunos de varios estándares para software provienen de ISO 9000 quién rige la calidad mundial.

¿Quién define la calidad?



