1) Describa método de COCOMO con sus palabras (Págs. 66,67,68)
2) Explicar los quince factores de ajuste (Págs. 70, 71, 72)
3)Describa que entendió con sus palabras de las paginas 76 y 77.
4)Explicar:
-Factores ligados a la utilización.
-Factores ligados al mantenimiento.
-Factores ligados a la transferencia.

	Pt x Nº = Puntos
Entradas	
S	3 x ... = ...
M	4 x ... = ...
C	6 x ... = ...
Salidas	
S	4 x ... = ...
M	5 x ... = ...
C	7 x ... = ...
Ficheros	
S	7 x ... = ...
M	10 x ... = ...
C	15 x ... = ...
Interfaces	
S	5 x ... = ...
M	7 x ... = ...
C	10 x ... = ...
Consultas	
S	3 x ... = ...
M	4 x ... = ...
C	6 x ... = ...
Puntos de función brutos (PFB) =	.....
Factor de ajuste (FA) x	.....
Puntos de función ajustada (PFA) =	.....

fig. 20 – Los puntos de función: tabla de síntesis

2. EL METODO DE COCOMO

El método de COCOMO de Barry BOEHM del que hay una descripción completa en su libro de consulta Software Engineering Economics publicado en 1981 es, probablemente, el más utilizado y conocido por los especialistas, de entre todos los métodos de estimación.

Según las descripciones hechas por BOEHM en su libro, deducimos que existen tres versiones de COCOMO (COnstructive COst MOdel): el modelo COCOMO de base, el modelo COCOMO intermediario y el modelo COCOMO detallado(1).

(1) Los desarrollos que siguen se inspiran en el informe sobre la estimación de los costes de F.-G. ROUX en los números 81, 82 y 86 de *La informática profesional*

Los datos que han permitido a BOEHM diseñar COCOMO provienen de 63 proyectos realizados entre 1964 y 1979 en cualquier tipo de campo y su coste varía entre los 6 a 11.400 meses x personas, en lo que se refiere al tamaño del programa, de 2.000 instrucciones fuente a 1.150.000 instrucciones COCOMO (modelos).

Las tres versiones de COCOMO se basan en ecuaciones matemáticas más o menos similares, su diferencia estriba en que el modelo COCOMO intermedio permite el ajuste de la carga en función de 15 factores que no existen en COCOMO de base, mientras que en el caso de COCOMO detallado, se nos propone descomponer el proyecto en sub-proyectos (como en el caso de COCOMO intermedio) y los sub-proyectos en módulos a los que se les aplica las fórmulas, así como las tablas de repartición de coste.

En lo que respecta a nuestro ciclo de vida, los tres modelos COCOMO cubren el coste de las etapas comprendiendo desde la concepción funcional hasta la entrada de usuarios inclusive.

Dicho de otro modo, las etapas de estudio de viabilidad, expresión de las necesidades, puesta en marcha y mantenimiento no quedan cubiertas, por lo que es conveniente estimarlas por separado.

Para las etapas cubiertas, las actividades cubiertas son:

- la concepción funcional,
- la gestión de las modificaciones,
- el seguro y control de calidad,
- el análisis orgánico
- la programación
- las pruebas de programas, las pruebas de integración
- los planes de prueba, los juegos de ensayos,
- la documentación,
- el encuadre.

El modelo que nos interesa en esta obra es el de COCOMO intermedio, el cual se utiliza en el ámbito de los modos orgánica y semi-separada y del cual nos ocuparemos más detalladamente.

La aplicación del método COCOMO se realiza en siete etapas:

1. Determinación del tamaño en Kdsi (miles de instrucciones fuente entregadas)
2. Cálculo del coste bruto
3. Determinación del valor de los quince factores de ajuste
4. Cálculo del coste neto
5. Cálculo del plazo total normal del coste neto
6. Distribución del coste por etapa
7. Distribución del coste por actividad.

1. Determinación del tamaño del programa en Kdsi (miles instrucciones fuente entregadas) del programa

Incluso siendo un ferviente defensor de las ideas de BOEHM, no es fácil tener que aplicar un método, en este caso COCOMO, en el que lo primero que hay que hacer es estimar el número de instrucciones fuente del programa una vez desarrollado, es decir, prácticamente al final del proyecto.

Es por lo que el método de puntos de función de ALBRECHT que vimos anteriormente fracasó.

El total de los puntos de función ajustados (PFA) se calcula, gracias a la fórmula dada por Bernard Londeix en su obra *Cost Estimation for Software Development* (Addison Wesley, 1987) con el fin de evaluar el número de instrucciones fuentes entregadas COBOL (ISL) por punto de función:

$$ISL = 118.7 \times PFA - 6.490$$

Sin embargo, si nos desenvolvemos con otro lenguaje diferente al COBOL, es factible utilizar la tabla siguiente propuesta por Capers JONES en su obra *Programming Productivity* (Mc Graw Hill, 1986) que ofrece la capacidad relativa de lenguajes orientados a procedimientos calculados en cuanto a número de instrucciones fuente equivalentes.

Lenguaje	ISL
Ensamblador	320
Macro-Ensamblador	213
C	150
Cobol	106
Fortran	106
Pascal	91
RPG	80
PL/I	80
Ada	71
Basic	64
APL	32

fig. 21 – Capacidad relativa de los lenguajes en instrucciones fuente entregadas (ISL)

2. Cálculo del coste bruto

El cálculo del coste bruto se hará a partir de fórmulas que varían en función del modo de desarrollo. Es necesario determinar esto último para el proyecto.

Para el modelo COCOMO intermedio, BOEHM da las tres fórmulas siguientes según el modo de desarrollo:

- Modo orgánico: $MHB = 3{,}2 \times (KISL)^{1{,}05}$
- Modo semi-agregado: $MHB = 3{,}0 \times (KISL)^{1{,}12}$
- Modo imbricado: $MHB = 2{,}8 \times (KISL)^{1{,}20}$

BOEHM distingue tres modos de desarrollo: orgánico, intermedio e imbricado.

De todos los modos el orgánico y el intermedio o semi-agregado son probablemente los más frecuentes en la medida en que el primero se

corresponde con los pequeños y medianos proyectos desarrollados dentro de la empresa y requiere pequeños grupos (dos a ocho personas); mientras que el segundo se aplica a proyectos de talla media subcontratados a destajo a empresas de servicios.

En cuanto al modo imbricado es el que se aplica a los grandes proyectos sometidos a condiciones muy estrictas en las cuales el material y el programa están íntimamente imbricados en los trabajos de interfaces complejas.

3. Determinación del valor de los quince factores de ajuste

Se trata de determinar el valor total de los quince factores, relativamente distintos de los catorce factores de ALBRECHT, que influyen en la carga de estudio y de desarrollo. Los quince factores son los siguientes:

PRODUCTO

1 RELY: fiabilidad requerida

factor de coste cuantificado en cinco niveles según la importancia para el usuario de una anomalía de programa (de un simple inconveniente a la pérdida de vidas humanas).

2 DATA: tamaño de datos manipulados

Seis gamas propuestas de 10 Ko a más de 100 Mo.

3 CPLX: complejidad del producto

Complejidad media de algoritmos puestos en práctica, clasificada en seis niveles según las características y los tipos de módulos (control, cálculo, gestión de datos, entrada/salida).

ORDENADOR

4 TIME: condiciones de tiempos de ejecución

Descripción del medio de explotación en términos de rapidez de ejecución. Se han definido cuatro niveles según el porcentaje de disponibilidad de tiempo CPU en el ordenador de explotación cuando la aplicación sea operacional.

5 STOR: condición de cantidad de memoria

La cantidad de memoria disponible en el ordenador que generará el programa entraña unos requisitos tanto en cuanto a nivel de análisis como de la programación y de cargos suplementarios para el proyecto. Se cuantifica este factor indicando uno de los cuatro niveles de ocupación de memoria cuando la aplicación sea operacional.

6 VIRT: inestabilidad de la máquina virtual (sistema)

Aquí se tiene en cuenta el coste suplementario inducido por una modificación del entorno a considerar durante el desarrollo (en el caso de un cambio previsible de versión del sistema de explotación, por ejemplo). Se han definido cuatro niveles.

7 TURL: plazo de restitución de los trabajos

Se toman en cuenta las prestaciones y el coste del ordenador de desarrollo, esta máquina puede por supuesto ser totalmente diferente de la que asegurará la explotación del programa. Se han definido cinco niveles, del desarrollo interactivo puro (una estación de trabajo por persona con un tiempo de respuesta inferior al segundo) hasta un tiempo de restitución superior a la media jornada.

PERSONAL

8 ACAP: capacitación de los analistas

Cinco niveles de capacitación previstos que van desde el principiante puro hasta el experimentado operacional.

9 AEXP: experiencia en el dominio aplicativo

Se trata de la experiencia media del equipo de desarrollo definido en cinco niveles basados en la media de tiempo de experiencia de este equipo en el campo estudiado (de menos de cuatro meses a más de doce años).

10 PCAP: capacitación de los programadores

Se basa en los mismos principios de los analistas.

11 VEXP: experiencia del sistema (máquina virtual)

Cuantificación de la experiencia del equipo de desarrollo en relación al conjunto de ordenador(s) y programa(s) utilizados para el desarrollo. Cuatro niveles: de menos de un mes a más de tres años.

12 LEXP: experiencia del lenguaje de programación

Definido a partir de la experiencia media del equipo de programación en el(los) lenguaje(s) escogido(s) para el desarrollo. Cuatro niveles: de menos de un mes a más de seis años.

PROYECTO

13 MODP: utilización de métodos modernos

Existen cinco niveles de calidad para el medio metodológico de la ausencia de método en la utilización sistemática e inteligente de métodos modernos de desarrollo.

14 TOOL: utilización de herramientas

Se tienen en cuenta la capacidad y la calidad de las herramientas de desarrollo utilizadas. Se han definido siete niveles desde la utilización de las herramientas de base (ensamblador, editor de unión, 3GL...) hasta la de las herramientas del programa

15 SCED: obligaciones de los plazos

Son escasos los proyectos que no se imponen una fecha límite para su desarrollo. Así que se podrá asegurar la duración del desarrollo.

Cada factor será juzgado y anotado en función de su grado de influencia o de exigencia por el que los valores preexistentes se facilitan en la siguiente tabla proporcionada por BOEHM.

Para utilizar COCOMO, nos sería necesario recordar los diferentes modos de desarrollo o de tipo de programa en la medida en que las fórmulas de cálculo varían en función de este modo.

4. Cálculo del coste neto

Coste neto =

coste bruto x valor total de los factores de ajuste (VTA)

MHN = MHB x VTA

5. Cálculo del plazo total normal del coste neto (TDEV)

En su libro, BOEHM facilita las siguientes tres fórmulas:

- Modo orgánico

$$TDEV = 2{,}5 \times (MHN)^{0{,}38}$$

- Modo semi-agregado o intermedio

$$TDEV = 2{,}5 \times (MHN)^{0{,}35}$$

- Modo imbricado

$$TDEV = 2{,}5 \times (MHN)^{0{,}32}$$

3. LO QUE USTED HARA EN LA PRACTICA

Acabamos de examinar las dos técnicas de estimación más conocidas y prácticas. Desgraciadamente sólo se pueden aplicar al final del estudio del sistema de información, es decir cuando se describen detalladamente tanto las entradas/salidas como los datos y los tipos de proceso necesarios.

Como la mayoría de métodos y técnicas existentes, estos necesitan ser adaptados al contexto de las empresas en cuyo seno son utilizados para estimar los costes y las cargas de desarrollo de los programas.

Como usted comprenderá, no tratamos de darle un curso completo sobre las técnicas de estimación, sino de explicárselas lo más claramente posible para que pueda adaptarlas a su propia situación. Para ayudarle en esta tarea, vamos a señalarle algunas precauciones y a darle algunos consejos.

La gestión de estimación consiste, en todos los casos, en descomponer las etapas del proyecto en tantas tareas de igual forma tangibles como posibles. Para nosotros tangible significa que cada una de estas tareas debe comportar un resultado preciso en el que se puedan aplicar los criterios de calidad con el fin de examinar la validez.

En un principio, si usted ejerce la función de jefe de proyecto en una empresa que es sensible a la calidad, debe, sin muchas dificultades, tener acceso a una guía metodológica que traduzca cada una de las tareas del ciclo de vida de una aplicación.

Si no es este el caso, le será necesario efectuar usted mismo la descomposición para su proyecto, o para la etapa a su cargo.

A título de ejemplo, se puede imaginar que el análisis detallado de las tareas de la etapa de concepción del sistema informático en una dirección informática sigue la siguiente lista:

1 PREPARACION

2 ARQUITECTURA DETALLADA DEL SISTEMA

- 2.1 Consideración de los standards y utilidades
- 2.2 Establecimiento del diagrama de funcionamiento
- 2.3 Descripción de las cadenas de proceso

- 2.4 Establecimiento de la lista de componentes
- 2.5 Integración de los puntos de control y de los procedimientos de recuperación después de una avería

3 ESTRUCTURA DE LOS DATOS/FICHEROS

- 3.1 Descripción de tipos de ficheros empleados
- 3.2 Descripción de la estructura de ficheros permanentes y tablas
- 3.3 Descripción de la estructura de los ficheros de trabajo

4 ESPECIFICACIONES DETALLADAS DE LAS ENTRADAS/SALIDAS

- 4.1 Descripción de la estructura de los resultados
- 4.2 Descripción de la estructura de los movimientos de entrada

5 DESCRIPCION DE LOS PROCESOS TRANSACCIONALES

- 5.1 Estructura de las transacciones
- 5.2 Descripción de los procesos

6 DESCRIPCION DEL METODO DE LOS PROCESOS BATCH

7 ESPECIFICACIONES DETALLADAS DE LA SEGURIDAD DEL SISTEMA

- 7.1 Seguridad aplicada a los datos
- 7.2 Seguridad aplicada al funcionamiento

8 ESPECIFICACIONES DE VALIDACION Y DE PUESTA EN MARCHA

- 8.1 Especificaciones de los tests
- 8.2 Especificaciones para los participantes

9 ESTABLECIMIENTO DEL DOCUMENTO DE FINAL DE FASE

10 DESCRIPCION DE LAS MEDIDAS DE SEGURO DE CALIDAD

El esquema de las dependencias de origen a la fig. 24.

La figura siguiente da una idea general de la descomposición postulada por BOEHM, autor por otro lado del famoso método de estimación de las cargas COCOMO.

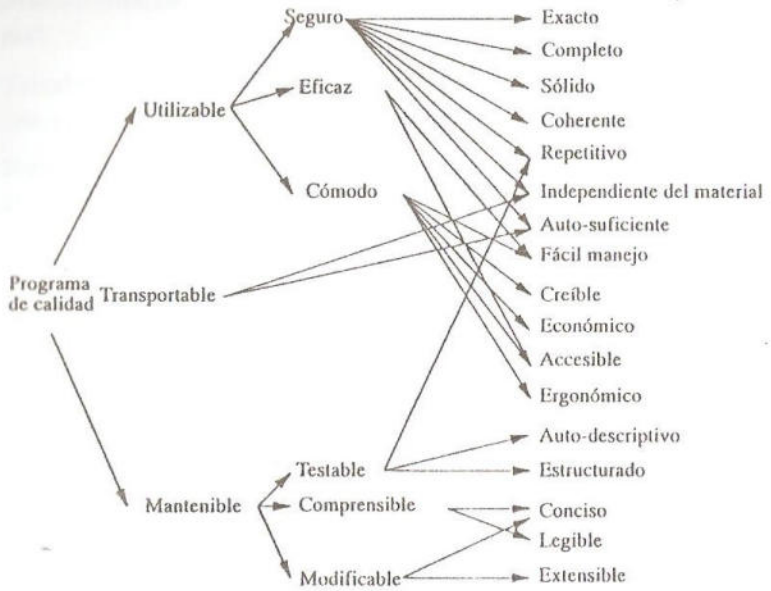

fig. 25 – Criterios de calidad

Mc CALL da un enfoque a la definición de calidad similar a la de BOEHM. Sin embargo, su definición es mejor e introduce dos niveles de abstracción:

- la cara externa: en este nivel, el programa es visto bajo el punto de vista del usuario, que lo considera como una caja negra en la que la calidad se aprecia según un cierto número de factores.

  A cada uno de estos factores va asociado un cierto número de criterios.

- la cara interna: en este nivel, el programa es considerado bajo el ángulo del realizador que lo considera como una caja blanca que comporta ciertas características internas que se aprecian según ciertos criterios.

A cada uno de estos criterios va asociado un conjunto de medidas a fin de medir la calidad del programa producido o su método de fabricación.

Esta jerarquía de relación entre factores, criterios y medidas viene representado a continuación:

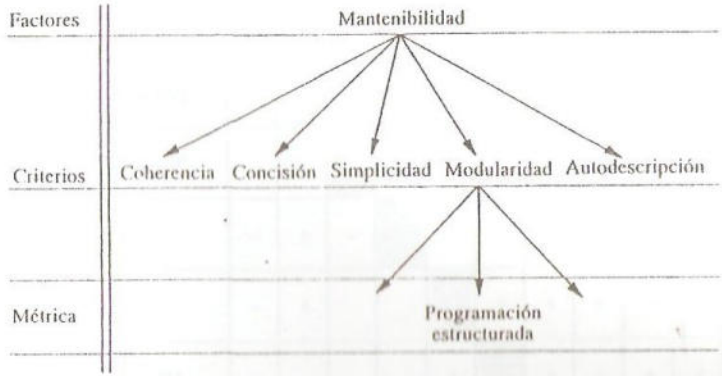

fig. 26 – Jerarquía de las relaciones entre factores, criterios y métricas

Mc CALL ha enumerado a lo largo de sus trabajos un total de 55 factores potencialmente utilizables, de los cuales sólo 11 han sido seleccionados por los usuarios, que los han reagrupado a su vez dentro de 3 tipos: utilización/mantenimiento/transferencia.

1.1. Factores ligados a la utilización

Conformidad: ¿Hace el programa lo que solicita el usuario?

Fiabilidad (robustez): ¿Hace el programa lo que se le solicita en cualquier circunstancia?

Eficacia: ¿Consume el programa sólo los recursos que necesita?

Integridad: ¿Está protegido el programa frente a errores de manipulación e intrusiones?

Facilidad de uso: ¿Es el programa de fácil manejo?

1.2. Factores ligados al mantenimiento

Mantenibilidad: ¿Se pueden aportar fácilmente correcciones al programa?

Testabilidad: ¿Se pueden emplear pruebas que permitan verificar el funcionamiento correcto del programa después de las modificaciones?

Flexibilidad (adaptabilidad): ¿Se pueden suprimir funciones constantes o añadir fácilmente nuevas funcionalidades?

1.3. Factores ligados a la transferencia

Portabilidad: ¿Es el programa fácilmente utilizable en otro ordenador?

Compatibilidad: ¿Puede ser el programa conectado fácilmente a otros programas?

Reutilizabilidad: ¿Puede ser una parte del programa reutilizada en el marco de otra aplicación?

Nos encontramos algunas veces con autores que sólo seleccionan 8 de los 11 factores antes mencionados, reagrupándolos dentro de 2 tipos y no de 3.

Lejos de considerar que su decisión es interesante, preferimos dejar en sus manos la selección de más o menos factores según su propio juicio.

En cambio, lo que nos interesa mostrar es cómo utilizar estos trabajos para sacar adelante su proyecto y satisfacer las necesidades de sus usuarios.

Para llevarlo a cabo, les daremos la lista de los 23 criterios, a priori unánimemente reconocidos por la profesión, cuya combinación, juiciosamente escogida, permite concurrir en la obtención de un factor de calidad.

Pero antes, examinaremos la figura 27 a fin de meditar sobre la relación existente entre los diferentes factores. Esta, puede ser neutra, positiva o negativa.

Sacar adelante un proyecto cuyos factores de calidad exigidos se contradicen unos con otros es seguramente más difícil que tener que llevar a buen puerto un proyecto cuyos factores de calidad van en el mismo sentido. Pero no se haga demasiadas ilusiones.

	1	2	3	4	5	6	7	8	9	10	11
REUTILIZABILIDAD (01)				–	–	+	–	+		+	
CONFORMIDAD (02)					+	+		+	+	+	
ACOPLAMIENTO (03)				–			–				
EFICACIA (04)	–	+	–			–	–	–	–	–	
FIABILIDAD (05)	–	+				+	+	+	+	+	
FLEXIBILIDAD (06)	+			–	+		+	+	+	+	
INTEGRIDAD (07)	–		–	–		–			+		
MANTENIBILIDAD (08)	+	+		–	+	+			+	+	
FACILIDAD DE USO (09)		+		–	+	+	+	+		+	
TESTABILIDAD (10)	+	+		–	+	+	+	+	+		
PORTABILIDAD (11)	+		+	–			+	+		+	

Leyenda:

+ : Compatibilidad

– : Antagonismo

Blanco: Neutralidad

fig. 27 – Relaciones entre factores de calidad