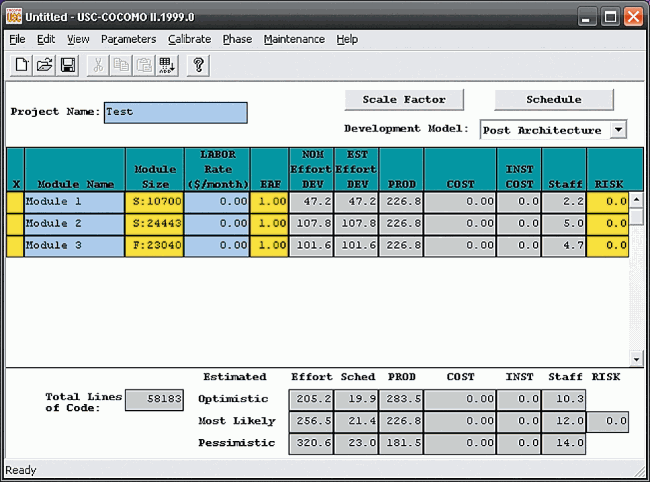
**COCOMO II**

**INTRODUCCIÓN**

COCOMO II es una herramienta utilizada para la estimación de esfuerzo, tiempo y costos generados en el desarrollo de software. Es una una evolución de un modelo económico, por lo que no sirve para cuestiones de diseño, sino exclusivamente para quienes vayan a trabajar con el desarrollo de alguna aplicación. Me permite saber el esfuerzo y poder estimar cuánto se va a cobrar al cliente.

Actualmente, en la mayoría de los proyectos en los cuales se requiere una determinación rápida del coste y esfuerzo de un proyecto promedio de las líneas de código, se utiliza Cocomo II, ya que es de mucha ventaja cuando el proyecto es de gran tamaño y no se lo puede determinar de manera rápida con otras formas de estimación.

**FUNCIONAMIENTO**

La pantalla principal es la siguiente:

Se comienza ingresando un Modulo al proyecto. Luego de esto, se presiona “Module size” para seleccionar a partir de que parámetros se medirá el esfuerzo y coste del proyecto. Los resultados la herramienta los presenta en la parte inferior. Allí se determinan tanto los valores más probables del proyecto, los valores optimistas (suponiendo un proyecto ideal), y los valores pesimistas que corresponderían al peor de los casos (en un proyecto muy conflictivo).

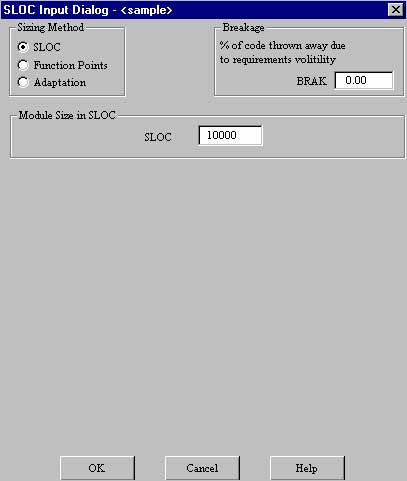
Los campos amarillos son los que se pueden configurar, el más importante es el tamaño del modulo.

Los atributos calculados serán los siguientes:

* Effort: esfuerzo más probable estimado para el módulo.
* Sched: cuántas semanas llevará el proyecto (schedule). Sirve para ir dimensionando el cronograma. Esto estima qué tan complejo será el desarrollo.
* PROD: productividad.
* Cost: esfuerzo intelectual, cuánto cuesta hacer el programa. Esto sin tener en cuenta costos indirectos como luz, renta, papel, transporte, costo de librería. Estos costos no entran aquí, ni se tienen en cuenta en el programa, ya que Cocomo estima solo costos para software.
* INST: coste por instrucción. El coste más probable por línea de código.
* STAFF: cuántas personas se necesitarán para el desarrollo.

Para comenzar a estimar costos, se da *click* a “Module size” y podemos elegir entre: cantidad de líneas de código, puntos de función y adaptación o reuso. Los tres cuadros de diálogo permiten la configuración del parámetro BRAK (% de código que se descarta debido a la volatilidad de los requisitos)

**Modulo estimado en base a la cantidad de líneas de código**

****

Con tan solo ingresar cantidad de líneas de código, el software calcula resultados. Se debe tener en cuenta que hay una restricción, si el programa es de menos de 2000 líneas de código, Cocomo arrojará error y los datos brindados serán falsos.

A modo de ejemplo, si ingresamos 3000 lineas de código, arrojará los siguientes resultados:

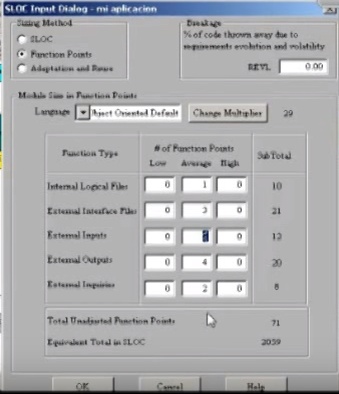
* Duracion aproximada: 7.6 semanas
* Staff: 1.3 personas

El Costo permanecerá en blanco hasta no brindar el dato de “salario mensual”. El mismo varía todos los semestres y años, por lo que se lo debe actualizar constantemente. Suponiendo un salario de $6000, el costo que el software estima en el peor de los casos será de $73807, y 8 semanas de tiempo.

**Estimación según los puntos de función**

Es más específico porque pide información acerca de los flujos de información en los puntos de intersección de la información.

1. En que voy a programar: habitualmente lenguajes OO.
2. Tipos de flujos de información: cuántos grupos de entrada de información se tendrán (archivos internos lógicos: block de notas, archivos complementarios, tablas)
3. Archivos externos de interfaz: cuántas caratulas se harán que estarán haciendo manejo de la aplicación (ejemplos: front de usuario, front de administrados, front de superusuario)
4. Datos de entrada: materias, calificaciones, etc.
5. Reportes
6. Consultas

Ejemplo:

Datos ingresados:

1. Lenguaje Orientado a objetos.
2. 1
3. 3
4. 3
5. 4
6. 2

Con este ejemplo tendría 2050 líneas de código, apenas arriba de las 2000 líneas aceptadas. Con estos datos y con costo de salario de 6000, varían los resultados obtenidos anteriormente. Baja el costo y la cantidad de semanas, por ser mas específico y tener menos líneas de código (en el otro ejemplo eran 3000).

**Basado en adaptación o reuso**

Esta opción es elegida cuando ya se tiene el código fuente (puede haberlo descargado, y estar en cualquier lenguaje) Simplemente se altera el código existente, definiendo cuanto se cambia el diseño original, el código, cuántos recursos se destinarán a la integración, qué tanto se entiende la codificación, asimilación de los conceptos básicos, qué tan familiarizado se está con este lenguaje de programación, cuántos componentes se utilizarán en su totalidad.

Finalmente, hay dos parámetros más para brindar mayor exactitud a la estimación. Estos son:

* “Development model”: Hay dos opciones de desarrollo. Una de ellas es cuando se está haciendo el modelo y no se tiene nada (“early begin”). La otra se selecciona cuando se parte de un diseño de una arquitectura prácticamente ya establecida (“post architecture”). En esta última opción las variantes son mínimas porque ya se tiene algo de conocimiento, lo cual hace que el producto baje de costo.
* “Scale factor”: Se puede ajustar la escala de factores, es un ajuste fino. Se le dice a la aplicación cuantos programas he hecho, tipo de diseño (ej: altamente flexible), si la arquitectura es clara o es cambiante (el cliente no sabe lo que quiere), equipo de trabajo (si llevan mucho tiempo trabajando juntos o nunca), maduración del proceso (claro o no muy claro). Estos datos cambiarán un poco los resultados finales.