COCOMO II

1. Introducción

COCOMO II está compuesto por tres modelos denominados: *Composición de Aplicación, Diseño Temprano y Post-Arquitectura*.

El modelo *Composición de Aplicación***,** es el modelo de estimación utilizado en los proyectos de software que se construyen a partir de componentes pre-empaquetadas. En este caso, se emplean Puntos Objeto para estimar el tamaño del software, lo cual está acorde al nivel de información que generalmente se tiene en la etapa de planificación, y el nivel de precisión requerido en la estimación de proyectos de esta naturaleza.

El modelo *Composición de Aplicación* se emplea en desarrollos de software durante la etapa de prototipación.

El modelo *Diseño Temprano* se utiliza en las primeras etapas del desarrollo en las cuales se evalúan las alternativas de hardware y software de un proyecto. En estas etapas se tiene poca información, lo que concuerda con el uso de Puntos Función, para estimar tamaño y el uso de un número reducido de factores de costo.

El modelo *Post-Arquitectura* se aplica en la etapa de desarrollo propiamente dicho, después que se define la arquitectura del sistema, y en la etapa de mantenimiento. Este modelo utiliza:

1) Puntos Función y/o Líneas de Código Fuente[7] para estimar tamaño, con modificadores que contemplan el reuso, con y sin traducción automática, y el "desperdicio" (breakage).
2) Un conjunto de 17 atributos, denominados factores de costo, que permiten considerar características del proyecto referentes al personal, plataforma de desarrollo, etc., que tienen injerencia en los costos.
3) Cinco factores que determinan un exponente, que incorpora al modelo el concepto de deseconomía y economía de escala.

2. Estimación del Esfuerzo

El esfuerzo necesario para concretar un proyecto de desarrollo de software, cualquiera sea el modelo empleado, se expresa en meses/persona (PM) y representa los meses de trabajo de una persona fulltime, requeridos para desarrollar el proyecto.

Modelo Composición de Aplicación

La fórmula propuesta en este modelo es la siguiente:

PM = NOP / PROD

Donde:

NOP (Nuevos Puntos Objeto): Tamaño del nuevo software a desarrollar expresado en Puntos Objeto y se calcula de la siguiente manera:

NOP = OP x (100 - %reuso)/100

OP (Puntos Objeto): Tamaño del software a desarrollar expresado en Puntos Objeto

%reuso: Porcentaje de reuso que se espera lograr en el proyecto

PROD: Es la productividad promedio determinada a partir del análisis de datos de proyectos en [Banker 1994], mostrada en Tabla 1.

Experiencia y capacidad de los desarrolladores	**Muy Bajo**	**Bajo**	**Normal**	**Alto**	**Muy Alto**
Madurez y Capacidad del entorno	**Muy Bajo**	**Bajo**	**Normal**	**Alto**	**Muy Alto**
PROD	**4**	**7**	**13**	**25**	**50**

Tabla 1: Productividad para el modelo Composición de Aplicación. [Boehm 1995/2]

Modelo Diseño Temprano

Este modelo se usa en las etapas tempranas de un proyecto de software, cuando se conoce muy poco del tamaño del producto a ser desarrollado, de la naturaleza de la plataforma, del personal a ser incorporado al proyecto o detalles específicos del proceso a utilizar. Este modelo podría emplearse tanto en productos desarrollados en sectores de Generadores de Aplicación, Sistemas Integrados o Infraestructura.

El modelo de Diseño Temprano ajusta el esfuerzo nominal usando siete factores de costo. La fórmula para el cálculo del esfuerzo es la siguiente:

$$PM_{estimado} = PM_{nominal} \text{ x PRODUCTORIA}(i=1 \text{ a } 7)(\text{EM}i)$$

$$PM_{nominal} = A \text{ x } (KSLOC)^{B}$$

$$\text{B} = 0{,}91 + 1{,}01 \text{ x SUMATOIRA}(j=1 \text{ a } 5)(Wj)$$

Donde:

$PM_{Estimado}$ es el esfuerzo Nominal ajustado por 7 factores, que reflejan otros aspectos propios del proyecto que afectan al esfuerzo necesario para la ejecución del mismo.

KSLOC es el tamaño del software a desarrollar expresado en miles de líneas de código fuente.

A es una constante que captura los efectos lineales sobre el esfuerzo de acuerdo a la variación del tamaño, (***A=2.94***).

B es el factor exponencial de escala, toma en cuenta las características relacionadas con las economías y deseconomías de escala producidas cuando un proyecto de software incrementa su tamaño.

EMi corresponde a los factores de costo que tienen un efecto multiplicativo sobre el esfuerzo, llamados Multiplicadores de Esfuerzo (Effort Multipliers). Cada factor se puede clasificar en seis niveles diferentes que expresan el impacto del multiplicador sobre el esfuerzo de desarrollo. Esta escala varía desde un nivel Extra Bajo hasta un nivel Extra Alto. Cada nivel tiene un peso asociado. El peso promedio o nominal es 1.0. Si el factor provoca un efecto nocivo en el esfuerzo de un proyecto, el valor del multiplicador correspondiente será mayor que 1.0, caso contrario el multiplicador será inferior a 1.0.

Clasificados en categorías, los 7 Multiplicadores de Esfuerzo son:

Del Producto
- RCPX: Confiabilidad y Complejidad del producto
- RUSE: Reusabilidad Requerida

De la Plataforma
- PDIF: Dificultad de la Plataforma

Del Personal
- PERS: Aptitud del Personal
- PREX: Experiencia del Personal

Del Proyecto
- FCIL: Facilidades
- SCED: Cronograma de Desarrollo Requerido

La Tabla 2 contiene los valores de estos multiplicadores:

	Bajísimo	Muy Bajo	Bajo	Normal	Alto	Muy Alto	Altísimo
RCPX	**0,73**	**0,81**	**0,98**	**1,00**	**1,30**	**1,74**	**2,38**
RUSE	**---**	**---**	**0,95**	**1,00**	**1,07**	**1,15**	**1,24**
PDIF	**---**	**---**	**0,87**	**1,00**	**1,29**	**1,81**	**2,61**
PERS	**2,12**	**1,62**	**1,26**	**1,00**	**0,83**	**0,63**	**0,50**
PREX	**1,59**	**1,33**	**1,12**	**1,00**	**0,87**	**0,71**	**0,62**
FCIL	**1,43**	**1,30**	**1,10**	**1,00**	**0,87**	**0,73**	**0,62**
SCED	**---**	**1,43**	**1,14**	**1,00**	**1,00**	**1,00**	**---**

Tabla 2: Multiplicadores de Esfuerzo del Modelo de Diseño Temprano. [COCOMO II.0]

Modelo Post-Arquitectura

Es el modelo de estimación más detallado y se aplica cuando la arquitectura del proyecto está completamente definida. Este modelo se aplica durante el desarrollo y mantenimiento de productos de software incluidos en las áreas de Sistemas Integrados, Infraestructura y Generadores de Aplicaciones.

El esfuerzo nominal se ajusta usando 17 factores multiplicadores de esfuerzo. El mayor número de multiplicadores permite analizar con más exactitud el conocimiento disponible en las últimas etapas de desarrollo, ajustando el modelo de tal forma que refleje fielmente el producto de software bajo desarrollo. La fórmula para el cálculo del esfuerzo es la siguiente:

$$PM_{estimado} = PM_{nominal} \text{ x PRODUCTORIA}(_{i=1\ a\ 17})(\text{EM}i)$$

$$PM_{nominal} = A \text{ x } (KSLOC)^{B}$$

$$B = 0{,}91 + 1{,}01 \text{ x SUMATOIRA}(j=1 \text{ a } 5)(Wj)$$

La Tabla 3 contiene los valores de estos multiplicadores:

Tipo	Factor	Muy bajo	Bajo	Normal	Alto	Muy alto	Altísimo
Producto	RELY	0,82	0,92	1,00	1,10	1,26	---
	DATA	---	0,90	1,00	1,14	1,28	---
	CPLX	0,73	0,87	1,00	1,17	1,34	1,74
	RUSE	---	0,95	1,00	1,07	1,15	1,24
	DOCU	0,81	0,91	1,00	1,11	1,23	---
Plataforma	TIME	---	---	1,00	1,11	1,29	1,63
	STOR	---	---	1,00	1,05	1,17	1,46
	PVOL	---	0,87	1,00	1,15	1,30	---
Personal	ACAP	1,42	1,19	1,00	0,85	0,71	---
	PCAP	1,34	1,15	1,00	0,88	0,76	---
	PCON	1,29	1,12	1,00	0,90	0,81	---
	AEXP	1,22	1,10	1,00	0,88	0,81	---
	PEXP	1,19	1,09	1,00	0,91	0,85	---
	LTEX	1,20	1,09	1,00	0,91	0,84	---
Proyecto	TOOL	1,17	1,09	1,00	0,90	0,78	---
	$SITE_1$	1,22	1,09	1,00	0,93	0,86	0,80
	$SITE_2$	1,22	1,09	1,00	0,93	0.86	0,80
	SCED	1,43	1,14	1,00	1,00	1,00	---

Tabla 3: Multiplicadores de Esfuerzo del Modelo Pos Arquitectónico. [COCOMO II.0] (Referencias SITE: 1.Se refiere a ubicación espacial; 2.Se refiere a comunicación)

Factores del producto

Se refieren a las restricciones y requerimientos sobre el producto a desarrollar.

RELY: Confiabilidad requerida
Este factor mide la confiabilidad del producto de software a ser desarrollado, esto es, que el producto cumpla satisfactoriamente con la función que debe realizar y respete el tiempo de ejecución que se fijó para el mismo. Los niveles de escala para este factor son Muy Bajo, Bajo, Nominal, Alto y Muy Alto. Si el efecto de la falla del software produce inconvenientes solamente al desarrollador, quien debe solucionarla, el valor de RELY es Bajo. Si por el contrario, la falla atenta contra la vida humana el valor que adopta es Muy Alto.

DATA: Tamaño de la base de datos
El esfuerzo requerido para desarrollar un producto de software está relacionado con el tamaño de la base de datos asociada. Un ejemplo que marca la importancia de esta influencia es el esfuerzo que insume la preparación de los lotes de prueba que se usan en el testeo del producto. El valor de DATA se determina calculando la relación entre el tamaño de la base de datos y el tamaño del programa.

$$\frac{D}{P} = \frac{Tamaño\_BasedeDatos(Bytes)}{Tamaño\_Programa(SLOC)}$$

CPLX: Complejidad del producto
CPLX analiza la complejidad de las operaciones empleadas en el producto, clasificadas en operaciones: de control, computacionales, dependientes de los dispositivos, de administración de datos y de administración de interfaz de usuario. El nivel que adopta este factor es el promedio del nivel de cada una de las cinco áreas o tipo de operaciones involucradas (Operaciones de Control, Operaciones computacionales, Operaciones dependientes de los dispositivos, Operaciones de administración de datos, y Operaciones de administración de interfaces de usuario)

RUSE: Requerimientos de reusabilidad
Este factor considera el esfuerzo adicional necesario para construir componentes que puedan ser reusadas dentro de un mismo proyecto o en futuros desarrollos. El incremento del esfuerzo se debe a que se incorporan tareas inherentes al reuso, tales como: creación de diseños genéricos de software, elaboración de mayor cantidad de documentación, testeo intensivo para asegurar que las componentes estén debidamente depuradas, etc.

DOCU: Documentación acorde a las diferentas etapas del ciclo de vida
Varios modelos de costo de software tienen un factor de costo para representar el nivel de documentación requerida. En COCOMO II este factor se evalúa en función de la adecuación de la documentación del proyecto a las necesidades particulares en cada etapa del ciclo de vida. Los posibles valores de DOCU van desde Muy Bajo (documentación que no cubre varias necesidades) hasta Muy Alto (documentación excesiva de acuerdo a las necesidades).

Factores de la plataforma

Estos factores analizan la complejidad de la plataforma subyacente.

La plataforma es la infraestructura base de hardware y software, lo que también recibe el nombre de máquina virtual. Si el software a desarrollar es un sistema operativo la plataforma es el hardware, si en cambio se trata del desarrollo de un administrador de base de datos se considerará como plataforma el hardware y el sistema operativo. Por ejemplo, la plataforma puede incluir cualquier compilador o ensamblador empleado en el desarrollo del software.

PVOL: Volatilidad de la plataforma
Este factor se usa para representar la frecuencia de los cambios en la plataforma subyacente.

STOR: Restricción del almacenamiento principal
Este factor es una función que representa el grado de restricción del almacenamiento principal impuesto sobre un sistema de software. Cuando se habla de almacenamiento principal se hace una referencia al almacenamiento de acceso directo, tales como circuitos integrados, memoria de núcleos magnéticos, excluyendo discos, cintas, etc. EL valor de STOR está expresado en términos de porcentaje del almacenamiento principal que usará el sistema. El rango posible de valores va desde Nominal hasta Extra Alto.

TIME: Restricción del tiempo de ejecución
Este factor representa el grado de restricción de tiempo de ejecución impuesta sobre el sistema de software. EL valor de TIME está expresado en términos de porcentaje de tiempo de ejecución disponible que usará el sistema. El rango posible de valores va desde Nominal hasta Extra Alto.

Factores del personal

Estos factores están referidos al nivel de habilidad que posee el equipo de desarrollo.

ACAP: Capacidad del analista
Se entiende por analista a la persona que trabaja con los requerimientos, en el diseño global y en el diseño detallado. Los principales atributos que deberían considerarse en un analista son la habilidad para el diseño, el análisis, la correcta comunicación y cooperación entre sus pares. En este análisis no se tiene en cuenta el nivel de experiencia.

PCAP: Capacidad del programador

Las tendencias actuales siguen enfatizando la importancia de la capacidad de los analistas. Sin embargo, debido a que la productividad se ve afectada notablemente por la habilidad del programador en el uso de las herramientas actuales, existe una tendencia a darle mayor importancia a la capacidad del programador. También se evalúa la capacidad de los programadores para el trabajo en equipo más que para el trabajo individual, resaltando las aptitudes para comunicarse y cooperar mutuamente.

PCON: Continuidad del personal
Este factor mide el grado de permanencia anual del personal afectado a un proyecto de software. Los posibles valores que puede adoptar PCON van desde 48% (muy bajo) al 3% (muy alto).

AEXP: Experiencia en la aplicación
Este factor mide el nivel de experiencia del equipo de desarrollo en aplicaciones similares. El rango de valores posibles de ***AEXP*** va desde Muy Bajo, representando una experiencia menor a 2 meses, hasta Muy Alto, experiencia de 6 o más años.

PEXP: Experiencia en la plataforma
COCOMO afirma que existe gran influencia de este factor en la productividad. Reconociendo así la importancia del conocimiento de nuevas y potentes plataformas, interfaces gráficas, base de datos, redes, etc. El rango de valores posibles de ***PEXP*** va desde Muy Bajo, representando una experiencia menor a 2 meses, hasta Muy Alto, experiencia de 6 o más años.

LTEX: Experiencia en el lenguaje y las herramientas
Este factor mide el nivel de experiencia del equipo en el uso del lenguaje y herramientas a emplear. El desarrollo de software, hoy en día, incluye el uso de herramientas que soportan tareas tales como representación de análisis y diseño, administración de la configuración, extracción de documentación, administración de librerías, y chequeos de consistencia. Es por ello que, no sólo es importante la experiencia en el manejo del lenguaje de programación sino también en el uso de estas herramientas, ya que influye notablemente en el tiempo de desarrollo. El rango de valores de posibles de ***LTEX*** va desde Bajo, representando una experiencia menor a 2 meses hasta Muy Alto representando una experiencia de 6 o más años.

Factores del proyecto

Estos factores se refieren a las condiciones y restricciones bajos las cuales se lleva a cabo el proyecto.

TOOL: Uso de herramientas de software
Las herramientas de software se han incrementado significativamente desde la década del 70. El tipo de herramientas abarca desde las que permiten editar y codificar hasta las que posibilitan una administración integral del desarrollo en todas sus etapas. El rango de valores posibles de ***TOOL*** va desde Muy Bajo, que corresponde al uso de herramientas sólo para codificación, edición y depuración, hasta Muy Alto, que incluye potentes herramientas integradas al proceso de desarrollo.

SITE: Desarrollo multisitio
La determinación de este factor de costo involucra la evaluación y promedio de dos factores, ubicación espacial (disposición del equipo de trabajo) y comunicación (soporte de comunicación).

SCED: Cronograma requerido para el desarrollo
Este factor mide la restricción en los plazos de tiempo impuesta al equipo de trabajo. Los valores se definen como un porcentaje de extensión o aceleración de plazos con respecto al valor nominal. Acelerar los plazos produce más esfuerzo en las últimas etapas del desarrollo, en las que se acumulan más temas a determinar por la escasez de tiempo para resolverlos tempranamente. Por el contrario una

relajación de los plazos produce mayor esfuerzo en las etapas tempranas donde se destina más tiempo para las tareas de planificación, especificación, validación cuidadosa y profunda. El rango de valores posibles de ***SCED*** va desde 75% al 160%.

3. Factor Exponencial de Escala

Los modelos de estimación de costos analizan dos aspectos antagónicos que influyen notablemente en los procesos de estimación, la economía y deseconomía de escala. La economía de escala abarca factores que hacen más eficiente la producción de software en gran escala. Es frecuente lograr economía en proyectos de gran envergadura, gracias a la inversión en software de propósitos específicos que mejoran la productividad, tales como herramientas de testeo, librerías de programas, preprocesadores, postprocesadores. Ahora bien, estamos frente a una deseconomía de escala cuando al incrementarse el tamaño del producto se produce una considerable disminución de la productividad. El aumento de la cantidad de personas que conforman el equipo de desarrollo generalmente provoca problemas de integración, que sumados a los conflictos personales, las diferencias en la filosofía y hábitos de trabajos producen deseconomía de escala.

Los modelos de estimación de costos frecuentemente tienen un factor exponencial para considerar las economías y deseconomías de escala. En particular, COCOMO II captura esos efectos en el exponente ***B***:

$$B = 0{,}91 + 1{,}01 \times \text{SUMATOIRA}(j=1 \text{ a } 5)(Wj)$$

Si ***B*** < 1.0, el proyecto exhibe economía de escala. Es decir si un producto aumenta el doble su tamaño el esfuerzo del proyecto es menos del doble. Esto significa que la productividad del proceso de desarrollo de software incrementa a medida que aumenta el tamaño del proyecto.

Si el ***B*** = 1.0 las economías y deseconomías de escala están en equilibrio. Este modelo lineal se usa siempre en la estimación de costos de proyectos pequeños.

Si el ***B*** > 1.0 el proyecto muestra deseconomía de escala. Esto generalmente se debe a dos factores principales: el crecimiento de las comunicaciones interpersonales y el de la integración de sistemas. Integrar un producto pequeño como parte de otro requiere no sólo el esfuerzo de desarrollar el producto sino también el esfuerzo de diseñar, mantener, integrar y testear interfaces con el resto del software. La productividad del proceso de desarrollo de software disminuye a medida que aumenta el tamaño del proyecto.

El cálculo del Factor Exponencial de Escala ***B*** está basado en factores que influyen exponencialmente en la productividad y esfuerzo de un proyecto de software. Estos factores toman valores dentro de un rango que va desde un nivel ***Muy Bajo*** hasta uno ***Extra Alto,*** tal como muestra la Tabla 4. Cada nivel tiene un peso asociado ***Wj,*** y ese valor específico es el que se denomina factor de escala.

Factor de Escala Wj		**Muy Bajo**	**Bajo**	**Normal**	**Alto**	**Muy Alto**	**Extra**
Precedencia PREC	Descripción	Completamente sin precedentes	Ampliamente sin precedentes	Algún precedente	Generalmente Familiar	Ampliamente Familiar	Completamente Familiar
	Valor	**6,20**	**4,96**	**3,72**	**2,48**	**1,24**	**0,00**
Flexibilidad en el desarrollo FLEX	Descripción	Rigurosa	Relajación Ocasional	Alguna Relajación	Conformidad en general	Alguna Conformidad	Metas generales
	Valor	**5,07**	**4,05**	**3,04**	**2,03**	**1,01**	**0,00**

Arquitectura/ Resolución de riesgo RESL	Descripción	Poca (20%)	Alguna (40%)	Siempre (60%)	Generalmente (75%)	Principalmente (90%)	Completo (100%)
	Valor	**7,07**	**5,65**	**4,24**	**2,83**	**1,41**	**0,00**
Cohesión de Equipo TEAM	Descripción	Interacciones Difíciles	Interacciones con alguna dificultad	Interacciones básicamente cooperativas	Ampliamente Cooperativas	Altamente Cooperativas	Interacciones Sin Fisuras
	Valor	**5,48**	**4,38**	**3,29**	**2,19**	**1,10**	**0,00**
Madurez del Proceso PMAT	Descripción	Ver Tabla 5					
	Valor	**7,80**	**6,24**	**4,68**	**3,12**	**1,56**	**0,00**

Tabla 4: Factores de Escala. [Boehm 1995/2]

Precedencia y Flexibilidad en el Desarrollo (PREC Y FLEX)

El factor de precedencia (***PREC***) toma en cuenta el grado de experiencia previa en relación al producto a desarrollar, tanto en aspectos organizacionales como en el conocimiento del software y hardware a utilizar.

El factor de flexibilidad (***FLEX***) considera el nivel de exigencia en el cumplimiento de los requerimientos preestablecidos, plazos de tiempos y especificaciones de interface.

Arquitectura y Determinación del Riesgo (RESL)

Este factor involucra aspectos relacionados al conocimiento de los ítems de riesgo crítico y al modo de abordarlos dentro del proyecto.

Cohesión del Equipo (TEAM)

El factor de escala denominado Cohesión del Equipo tiene en cuenta las dificultades de sincronización entre los participantes del proyecto: usuarios, clientes, desarrolladores, encargados de mantenimiento, etc. Estas dificultades pueden surgir por diferencias culturales, dificultad en la conciliación de objetivos, falta de experiencia y familiaridad con el trabajo en equipo.

Madurez del Proceso (PMAT)

El procedimiento para determinar el factor ***PMAT*** se basa en el Modelo de CMM propuesto por el Software Engineering Institute. Existen dos formas de calcularlo:

1) La primera captura el nivel de madurez de la organización, resultado de la evaluación según CMM y asignándole el valor correspondiente según Tabla 5.

2) La segunda está basada en las dieciocho Áreas de Procesos Claves (KPAs) del modelo del SEI. El procedimiento para determinar el ***PMAT*** es establecer el porcentaje de cumplimiento de cada una de las Áreas evaluando el grado de cumplimiento de las metas correspondientes.

Nivel de CMM	PMAT
1 – Mitad inferior	**Muy Bajo**
1 – Mitad superior	**Bajo**
2	**Nominal**
3	**Alto**
4	**Muy Alto**
5	**Extra Alto**

Tabla 5: Factor PMAT de acuerdo al nivel de CMM. [COCOMO II.0]

4. Métricas de Software

En la estimación del tamaño de software COCOMO II utiliza tres técnicas: *Puntos Objeto*, *Puntos Función No Ajustados* y *Líneas de Código Fuente*. Además se emplean otros parámetros relativos al tamaño que contemplan aspectos tales como: reúso, reingeniería, conversión, y mantenimiento.

Puntos Objeto

A pesar de que la estimación a través de Puntos Objeto es un enfoque de medición de tamaño de software relativamente nuevo, es apropiado para las aplicaciones con componentes y para estimar esfuerzos en las etapas de prototipación. En estas circunstancias, se lo ha comparado con la estimación de Puntos Función. Un experimento, diseñado por Kaufman y Kumar en 1993, involucró a 4 administradores de proyecto experimentados usando Puntos Objeto y Puntos Función, para estimar el esfuerzo requerido de dos proyectos terminados en 3.5 y 6 meses-persona respectivamente. Como base, se emplearon las descripciones disponibles al comienzo de tales proyectos. El experimento permitió determinar que:

1) Los Puntos Objeto y los Puntos Función produjeron resultados igualmente precisos (ligeramente más exacto con Puntos Objetos, pero no estadísticamente significativo).

2) El tiempo promedio para producir una estimación con Puntos Objeto fue alrededor del 47% del tiempo promedio necesario para las estimaciones con Puntos Función. Además, los administradores consideraron que el método de Puntos Objeto era más fácil de usar.

De esta manera, aunque estos resultados no están respaldados estadísticamente, parecen suficientemente prometedores como para justificar el uso de Puntos Objeto como punto de partida en el modelo de estimación de *Composición de Aplicación* de COCOMO II.

A continuación se describe el procedimiento para determinar Puntos Objeto en un proyecto de software:

Primero: Determinar Cantidad de Objetos: Estimar la cantidad de pantallas, reportes, componentes de 3GL que contendrá la aplicación.

Segundo: Clasificar cada instancia de un objeto según sus niveles de complejidad (simple, media o difícil) de acuerdo a la Tabla 6.

Tercero: Dar el peso a cada objeto según el nivel de complejidad. Los pesos reflejan el esfuerzo relativo requerido para implementar una instancia de ese nivel de complejidad. Tabla 7.

Cuarto: Determinar la cantidad de Puntos Objeto, sumando todos los pesos de las instancias de los tipos de objetos especificados.

Para Pantallas			
Cantidad de Vistas Contenidas	**Cantidad y fuente de las tablas de datos**		
	Total < 4 (< 2 servidor < 3 cliente)	**Total < 8 (< 2 - 3 servidor < 3 - 5 cliente)**	**Total 8 + (> 3 servidor < 5 cliente)**
< 3	**Simple**	**Simple**	**Media**
3 - 7	**Simple**	**Media**	**Difícil**
> 8	**Media**	**Difícil**	**Difícil**

Para Reportes			
Cantidad de Vistas Contenidas	**Cantidad y fuente de las tablas de datos**		
	Total < 4 (< 2 servidor < 3 cliente)	**Total < 8 (< 2- 3 servidor < 3-5 cliente)**	**Total 8 + (> 3 servidor < 5 cliente)**
0 o 1	**Simple**	**Simple**	**Media**
2 o 3	**Simple**	**Media**	**Difícil**
4 +	**Media**	**Difícil**	**Difícil**

Tabla 6: Esquema de Clasificación de Puntos Objetos. [Boehm 1995/2]

Tipo de Objeto	Complejidad - Peso		
	Simple	**Media**	**Difícil**
Pantalla	**1**	**2**	**3**
Reporte	**2**	**5**	**8**
Componente 3GL	**---**	**---**	**10**

Tabla 7: Peso de un Punto Objeto. [Boehm 1995/2]

Puntos Función

El modelo COCOMO II usa Puntos Función y/o Líneas de Código Fuente (SLOC) como base para medir tamaño en los modelos de estimación de *Diseño Temprano* y *Post-Arquitectura*

Los Puntos Función procuran cuantificar la funcionalidad de un sistema de software. La meta es obtener un número que caracterice completamente al sistema. Son útiles estimadores ya que están basados en información que está disponible en las etapas tempranas del ciclo de vida del desarrollo de software.

La fórmula para calcular los puntos función, es la siguiente:

$$FP = UFP \times TCF$$

Donde

UFP: Puntos Función no Ajustados

TCF: Factor de Complejidad Técnica

Para calcular los UFP, se deben identificar los siguientes tipos de ítems:

1) Entradas Externas (Inputs): Entrada de datos del usuario o de control que ingresan desde el exterior del sistema para agregar y/o cambiar datos a un archivo lógico interno.

2) Salidas Externas (Outputs): Salida de datos de usuario o de control que deja el límite del sistema de software.

3) Archivo Lógicos Internos (Archivos): Incluye cada archivo lógico, es decir cada grupo lógico de datos que es generado, usado, o mantenido por el sistema de software.

4) Archivos Externos de Interfase (Interfases): Archivos transferidos o compartidos entre sistemas de software.

5) Solicitudes Externas (Queries): Combinación única de entrada-salida, donde una entrada causa y genera una salida inmediata, como un tipo de solicitud externa.

Una vez identificados los ítems se clasifican de acuerdo al grado de complejidad en: bajo, promedio o alto. Se asigna un peso a cada ítem según el tipo y el grado de complejidad correspondiente. Finalmente los UFP son calculados mediante la sumatoria de los pesos de todos los ítems identificados.

$$UFP = SUMATORIA(j=1 \text{ a } 5)(Cantidad\_Item\_Tipo_j)(Peso_j)$$

La Tabla 8 muestra cómo se determinan los niveles de complejidad de cada tipo de ítem en función del número y tipo de elementos de datos y archivos involucrados.

Para archivos lógicos internos y archivos externos de interface				**Para salidas y consultas externas**				**Para entradas externas**			
Elementos de Registro	**Elementos de datos**			**Tipos de archivos**	**Elementos de datos**			**Tipos de archivos**	**Elementos de datos**		
	1-19	**20-50**	**51+**		**1-5**	**6-19**	**20+**		**1-4**	**5-15**	**16+**
1	**Bajo**	**Bajo**	**Prom.**	**0 ó 1**	**Bajo**	**Bajo**	**Prom.**	**0 ó 1**	**Bajo**	**Bajo**	**Prom.**
2-5	**Bajo**	**Prom.**	**Alto**	**2-3**	**Bajo**	**Prom.**	**Alto**	**2-3**	**Bajo**	**Prom.**	**Alto**
6+	**Prom.**	**Alto**	**Alto**	**4+**	**Prom.**	**Alto**	**Alto**	**3+**	**Prom.**	**Alto**	**Alto**

Tabla 8: Puntos Función. Determinación del Peso. [Boehm 1995/2]

La Tabla 9 muestra las ponderaciones asociadas a cada tipo de ítem. Estas ponderaciones han sido derivadas y validadas empíricamente mediante la observación de una gran variedad de proyectos.

Tipo de función	**Peso del Factor de Complejidad**		
	Bajo	**Promedio**	**Alto**
Entradas Externas (Inputs)	**3**	**4**	**6**
Salidas Externas (Outputs)	**4**	**5**	**7**
Archivo Lógicos Internos (Archivos)	**7**	**10**	**15**
Archivos Externos de Interface (Interfaces)	**5**	**7**	**10**
Consultas Externas (Queries)	**3**	**4**	**6**

Tabla 9: Peso del Factor de Complejidad. [Boehm 1995/2]

Para el cálculo del Factor de Complejidad Técnica, TCF, se considera la siguiente fórmula:

$$TCF = 0{,}65 + 0{,}01 \text{ x } SUMATOIRA(i=1 \text{ a } 14)(F_i)$$

Donde los F_i corresponden a los pesos asignados a los siguientes factores:

F1: Mecanismos de recuperación y back-up confiables
F2: Comunicación de Datos
F3: Funciones de Procesamiento Distribuido
F4: Performance
F5: Configuración usada rigurosamente
F6: Entrada de datos on-line
F7: Factibilidad Operativa
F8: Actualización de archivos on-line
F9: Interfaces Complejas
F10: Procesamiento Interno Complejo
F11: Reusabilidad
F12: Fácil Instalación
F13: Soporte de múltiples instalaciones
F14: Facilidad de cambios y amigabilidad

Los pesos se consideran dentro de una escala de 0 a 5, descripta a continuación:

0: Sin influencia
1: Incidental
2: Moderado
3: Medio
4: Significativo
5: Esencial

Estas 14 características consideran aspectos como reusabilidad, performance, complejidad, confiabilidad, etc., contemplados por COCOMO II a través de los factores de costo. Es por ello que este modelo utiliza los UFP como métrica de determinación de tamaño.

Líneas de Código Fuente

COCOMO II considera a la sentencia fuente lógica como línea standard de código. Ahora bien, definir una línea de código es difícil debido a que existen diferencias conceptuales cuando se cuentan sentencias ejecutables y de declaraciones de datos en lenguajes diferentes. El objetivo es medir la cantidad de trabajo intelectual puesto en el desarrollo de un programa.

Para determinar el esfuerzo nominal en el modelo COCOMO II los puntos función no ajustados tienen que ser convertidos a líneas de código fuente considerando el lenguaje de implementación (assembler, lenguajes de alto nivel, lenguajes de cuarta generación, etc.). Esto se realiza para los modelos Diseño Temprano y Post Arquitectura teniendo en cuenta la Tabla 10.

Lenguaje	SLOC/Punto Función
C	128
C++	64
C#	64
Fox Pro	35
J2EE	50
Java	53
Java Script	55
JSP	59
.NET	60
Perl	57
FORTRAN	106
COBOL	106
ASP	56
Python	67
Delphi	29
Assembler	320
Lisp	64
Modula 2	80
Pascal	91
Prolog	64

Tabla 10: Conversión de UFP a SLOC. [COCOMO II.0]

5. Estimación del Cronograma

La versión inicial de COCOMO II provee un modelo de estimación del cronograma similar al presentado en COCOMO' 81 y ADA COCOMO. La ecuación inicial para los tres modelos de COCOMO II es:

$$TDEV = \left[3.0 \times PM_{*}^{(0.33+0.2\times(B-1.01))}\right] \times \frac{SCED\%}{100}$$

Donde:

TDEV es el tiempo calendario en meses que transcurre desde la determinación de los requerimientos a la culminación de una actividad que certifique que el producto cumple con las especificaciones.

PM* es el esfuerzo expresado en meses personas, calculado sin tener en cuenta el multiplicador de esfuerzo ***SCED.*** Ver Tabla 3.

B es el Factor de Escala

SCED% es el porcentaje de compresión/expansión del cronograma.

6. Procedimiento de estimación

Se propone usar un formulario denominado CLEF (Component Level Estimating Form. Formulario de Estimación al nivel de componente), este formulario es de gran ayuda para la estimación manual. El formulario puede verse en la Tabla 11.

Los pasos del proceso de estimación de esfuerzo y tiempo de desarrollo son:

1) Identificar los módulos que conforman el sistema, asignarles un número y un nombre e ingresarlos en las columnas ***1*** y ***2***, respectivamente.

2) Determinar el tamaño de cada módulo expresado en SLOC, líneas de código fuentes liberadas, y registrarlo en la columna ***3***.

3) Determinar el tamaño en SLOC del Sistema, sumando el tamaño de los módulos que lo componen. Anotarlo en la celda ***28***.

4) Calcular el Factor Exponencial de Escala (B), considerando los 5 factores W_j (PREC, FLEX, RESL, TEAM y MAT) en un nivel nominal.

5) Calcular el Esfuerzo Nominal requerido para desarrollar el sistema, **$PM_{Nominal}$**, en la celda ***29*** y la Productividad del Proyecto en la celda ***30***.

6) Calcular y registrar en la columna ***22*** el Esfuerzo Nominal por Módulo(**$PM_{Nominal,Módulo}$**), que se obtiene como el cociente entre el tamaño del módulo (columna ***3***) y la Productividad del Proyecto (celda ***30***).

7) Analizar las características de cada módulo y determinar, con la ayuda de la Tabla 3, en qué nivel se encuentra cada uno de los factores de costo. Según el nivel determinado (Muy Bajo, Bajo, Nominal, Alto, Muy Alto) asignar los valores de los multiplicadores de esfuerzo correspondientes, y completar las columnas ***4*** a ***20***.

8) Multiplicar los multiplicadores de esfuerzo de la columna ***4*** a la ***20*** para cada fila y así obtener el Factor de Ajuste del Esfuerzo **EAF** para cada módulo. Ingresar los resultados en la columna ***21.***

9) Calcular el Esfuerzo Estimado por Módulo, $PM_{Estimado,Módulo}$, en la columna ***23***, multiplicando el valor de $PM_{Nominal,Módulo}$, columna ***22,*** por el correspondiente Factor de Ajuste EAF_M de la columna ***21***.

10) Sumar los valores calculados en el ítem anterior para determinar el Esfuerzo Estimado del Sistema Total $PM_{Estimado}$, registrar este valor en la celda ***31.***

11) Determinar el Tiempo de Desarrollo Estimado del proyecto **TDEV** y anotarlo en la celda ***34.***

12) Anotar en la columna ***24*** el Costo del Mes-Persona para cada módulo, expresado en miles de $. Posteriormente multiplicar estos costos por los $PM_{Estimado,Módulo}$ correspondientes (columna ***23***), encontrando así el Costo Estimado de cada módulo y registrarlo en la columna ***25***.

13) Calcular el Costo Total del Sistema sumando los valores obtenidos en el ítem anterior y registrarlo en la celda ***32***.

14) Para cada módulo determinar y registrar en la columna ***26*** el Costo por instrucción en $, el cual se calcula como el cociente entre el Costo de Desarrollo (columna ***25***) y el Tamaño del Módulo (columna ***3***).

15) Para cada módulo determinar y registrar en la columna ***27*** la Productividad, calculada como el cociente entre el Tamaño del Módulo (columna ***3***) y el Esfuerzo Estimado por módulo

			Producto					Plataforma			Personal						Proyecto									
Nro. Módulo	**Nombre Módulo**	**SLOC**	**RELY**	**DATA**	**CPLX**	**RUSE**	**DOCU**	**TIME**	**STOR**	**PVOL**	**ACAP**	**PCAP**	**PCON**	**AEXP**	**PEXP**	**LTEX**	**TOOL**	**SITE**	**SCED**	**EAF**	**PM Nominal**	**PM Estimado**	**Costo [Mes-Pers]**	**Costo**	**Costo x Instrucción**	**Productividad SLOC/Mes-Pers**
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**
1	**M1**																									
2	**M2**																									
3	**M3**																									
4	**M4**																									
5	**M5**																									
														Esfuerzo Estimado											**31**	
28			**Total SLOC**											**Costo Total Estimado**											**32**	
29			**Esfuerzo Nominal**											**Productividad Total Estimada**											**33**	
30			**Productividad Nominal**											**Tiempo de Desarrollo TDEV**											**34**	

Tabla 11: Formulario para la estimación de esfuerzo y tiempo de desarrollo utilizando COCOMO II.