ESTIMACIÓN DEL ESFUERZO BASADA EN CASOS DE USO

Mario Peralta

Centro de Ingeniería del Software e Ingeniería del Conocimiento (CAPIS)
Escuela de Postgrado. Instituto Tecnológico de Buenos Aires
Av. Madero 399 (C1106ACD), Buenos Aires – Argentina.
http://www.itba.edu.ar/capis/webcapis/planma.html
marioitba@yahoo.com.ar

Resumen: El presente artículo plantea algunas alternativas posibles para la estimación del esfuerzo en proyectos basados en Casos de Uso, utilizándose el Análisis de Puntos de Función y COCOMO II, o una variante más reciente denominada Análisis de Puntos de Casos de Uso, la cual es en cierta medida similar al Análisis de Puntos de Función.

Palabras Clave: Estimación del Esfuerzo. Casos de Uso. Puntos de Función. COCOMO II. Análisis de Puntos de Casos de Uso.

1. Introducción

La especificación de los requerimientos mediante Casos de Uso ha probado ser uno de los métodos más efectivos para capturar la funcionalidad de un sistema. Este hecho se puede apreciar en algunas metodologías actuales ampliamente difundidas, como el Proceso Unificado de Rational (Rational Unified Process) o Métrica Versión 3 (Ministerio de Administraciones Públicas de España), en las cuales se propone especificar la funcionalidad de los sistemas mediante la utilización de Casos de Uso.
El método de Casos de Uso permite documentar los requerimientos de un sistema en términos de Actores y Casos de Uso. Un Actor típicamente representa a un usuario humano o a otro sistema que interactúa con el sistema bajo análisis. Un Caso de Uso representa un gránulo funcional del sistema bajo análisis, relatado como una secuencia de acciones que uno o más actores llevan a cabo en el sistema para obtener un resultado de valor significativo.

Si bien los Casos de Uso permiten especificar la funcionalidad de un sistema bajo análisis, no permiten por sí mismos efectuar una estimación del tamaño que tendrá el sistema o del esfuerzo que tomaría implementarlo.
Para la estimación del tamaño de un sistema a partir de sus requerimientos, una de las técnicas más difundidas es el Análisis de Puntos de Función. Ésta técnica permite cuantificar el tamaño de un sistema en unidades independientes del lenguaje de programación, las metodologías, plataformas y/o tecnologías utilizadas, denominadas Puntos de Función.
Por otro lado, el SEI (del inglés, Software Engineering Institute) propone desde hace algunos años un método para la estimación del esfuerzo llamado COCOMO II. Éste método está basado en ecuaciones matemáticas que permiten calcular el esfuerzo a partir de ciertas métricas de tamaño estimado, como el Análisis de Puntos de Función y las líneas de código fuente (en inglés SLOC, Source Line Of Code).

El presente artículo plantea algunas alternativas posibles para la estimación del esfuerzo en proyectos basados en Casos de Uso, utilizándose el Análisis de Puntos de Función y COCOMO II, o una variante más reciente denominada Análisis de Puntos de Casos de Uso, la cual es en cierta medida similar al Análisis de Puntos de Función.

2. Casos de Uso y Puntos de Función

Existe una relación natural entre los Puntos de Función y los Casos de Uso. Los Puntos de Función permiten estimar el tamaño del software a partir de sus requerimientos, mientras que los Casos de Uso permiten documentar los requerimientos del software. Ambos tratan de ser independientes de las tecnologías utilizadas para la implementación.

En etapas tempranas del ciclo de vida, se identifican los Actores y los Casos de Uso del sistema, y se documenta cada uno de ellos mediante una breve descripción. Aplicando el Análisis de Puntos de Función a estos Casos de Uso, se podrá obtener una estimación grosera del tamaño y a partir de ella del esfuerzo. Esta estimación es bastante imprecisa debido principalmente a la escasa información que se tiene sobre el software al principio de un proyecto, pero permitirá obtener una idea del esfuerzo necesario para llevar adelante el mismo, y podrá ser refinada a medida que se obtenga más información.

Posteriormente se amplía la documentación de cada Caso de Uso, describiendo los Escenarios que se pro-

ducen dentro del mismo. Un Escenario relata la secuencia de pasos que efectúan los actores y el sistema durante la ejecución del Caso de Uso.
Si se aplica nuevamente el Análisis de Puntos de Función sobre estos Casos de Uso detallados, la estimación del tamaño y esfuerzo será más precisa que la anterior.

2.1. Definiciones

A continuación se adecuará la definición de los componentes utilizados en el Análisis de Puntos de Función, para su utilización en los Casos de Uso.

2.1.1 Transacciones

En la especificación de un Caso de Uso, se utiliza un escenario principal para relatar la secuencia de pasos entre el Actor y el sistema, y escenarios alternativos para relatar condiciones excepcionales o condiciones que se apartan del flujo normal de eventos. A continuación se muestra un ejemplo de especificación de Caso de Uso:

Escenario Principal
1. El usuario indica un tipo de documento y un rango de fechas desde y hasta
2. El sistema busca los documentos dados de alta en el rango de fechas indicado y que sean del tipo indicado por el usuario, y los presenta en una lista.
3. El usuario selecciona un documento de la lista
4. El sistema muestra los datos internos del documento

Escenario Alternativo

2. No existe ningún documento que cumpla con los valores indicados por el usuario
2.1 El sistema le informa al usuario que no encontró ningún documento y le da la posibilidad de reingresar los parámetros de búsqueda.

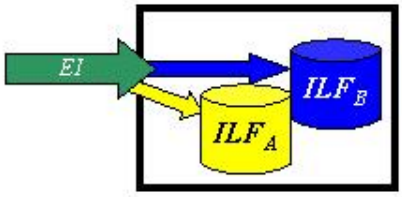
Una Transacción está representada por uno o más pasos del flujo de eventos principal del Caso de Uso, pudiendo existir más de una transacción dentro del mismo Caso de Uso. Los flujos de eventos alternativos dentro del Caso de Uso, ayudan a clarificar las transacciones. En el ejemplo del párrafo anterior se puede apreciar que existen dos transacciones diferenciadas, una vinculada a la búsqueda de documentos, y la otra vinculada a la visualización de un documento en particular.

En relación a los Puntos de Función, las transacciones se clasifican de la siguiente manera:

Entradas Externas (EI, del inglés External Inputs): se definen como un proceso elemental mediante el cual ciertos datos cruzan la frontera del sistema desde afuera hacia adentro. El Actor del Caso de Uso provee datos al sistema, los cuales pueden tratarse de información para agregar, modificar o eliminar de un Archivo Lógico Interno, o bien información de control o del negocio.

Ejemplo: un caso de uso que documente el mantenimiento de alguna entidad del sistema, por ejemplo el Documento, podría llamarse “Mantener Documentos” y estar constituído por 3 escenarios: uno que relate el alta, otro que relate la modificación y otro que relate la baja de documentos. Cada uno de éstos escenarios representaría una Entrada Externa desde el punto de vista de los Puntos de Función. De manera similar, si en vez de un caso de uso que relate el mantenimiento de Documentos como 3 escenarios, se tienen 3 casos de uso distintos, uno llamado “Agregar Documento”, otro “Modificar Documento” y otro “Eliminar Documento”, desde el punto de vista de los Puntos de Función se tienen 3 Entradas Externas.

Un ejemplo gráfico de Entrada Externa se puede ver en la siguiente figura:

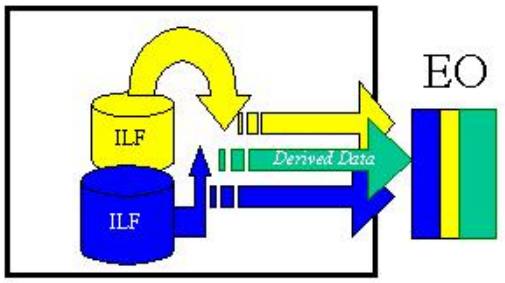


En la misma se muestra una Entrada Externa simple que modifica dos Archivos Lógicos Internos.

Salidas Externas (EO, del inglés External Outputs): se definen como un proceso elemental con componentes de entrada y de salida mediante el cual datos simples y datos derivados (esto es, datos que se calculan a partir de otros datos) cruzan la frontera del sistema desde adentro hacia afuera. Adicionalmente, las Salidas Externas pueden actualizar un Archivo Lógico Interno. Los datos crean reportes o archivos que se envían hacia el Actor del Caso de Uso (que puede ser un humano u otro sistema). Estos reportes y archivos se crean desde uno o más Archivos Lógicos Internos o Archivos de Interfaz Externos.

Ejemplo: los casos de uso que documentan reportes o estadísticas que genera el sistema para uso de los actores. La información que sale del sistema consiste fundamentalmente de datos calculados a partir de Archivos Lógicos Internos. Un ejemplo más concreto podría ser un caso de uso llamado “Generar reporte de altas”, donde se documenta la generación de un reporte de la cantidad de documentos que se dieron de alta en un período de tiempo. El actor indica el rango de fechas y el sistema genera el reporte. Este caso de uso se correspondería con una Salida Externa desde el punto de vista de los Puntos de Función.

Un ejemplo gráfico de Salida Externa se puede ver en la siguiente figura:

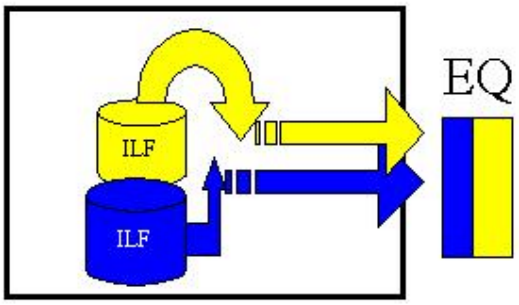


En la misma se muestra una Salida Externa compuesta por datos simples y datos derivados extraídos de 2 Archivos Lógicos Internos.

Consultas Externas (EQ, del inglés External Inquirys): se definen como un proceso elemental con componentes de entrada y de salida donde un Actor del sistema rescata datos de uno o más Archivos Lógicos Internos o Archivos de Interfaz Externos. Los datos de entrada no actualizan ni mantienen ningún archivo (lógico interno o de interfaz externo) y los datos de salida no contienen datos derivados (es decir, los datos de salida son básicamente los mismos que se obtienen de los archivos). Dentro de éste tipo de transacción entran los listados y las búsquedas de los sistemas.

Ejemplo: un caso de uso denominado "Buscar Documentos", donde se relata la interacción entre un Actor y el sistema para efectuar la búsqueda de documentos. Esta búsqueda representa una Consulta Externa desde el punto de vista de los Puntos de Función.

Un ejemplo gráfico de Consulta Externa se puede ver en la siguiente figura:



En la misma se muestra una Consulta Externa compuesta por datos simples extraídos de 2 Archivos Lógicos Internos.

2.1.2 Archivos

En relación a los Casos de Uso, los archivos están representados por las descripciones de almacenamiento de datos dentro del Caso de Uso, las cuales pueden hablar de archivos, bases de datos, u otro tipo de almacenamiento.

En relación a los Puntos de Función, los archivos se clasifican de la siguiente manera:

Archivos Lógicos Internos (ILF, del inglés Internal Logical Files): grupo de datos relacionados lógicamente e identificables por el usuario, que residen enteramente dentro de los límites del sistema y se mantienen a través de las Entradas Externas.

Ejemplo: los casos de uso que están vinculados al mantenimiento (altas, bajas, modificaciones) de entidades del sistema, están indicando la presencia de Archivos Lógicos Internos. Retomando el ejemplo del caso de uso "Mantener Documentos" en el que se documentaban 3 escenarios, uno para alta, otro para baja y otro para modificación de Documentos, se tiene que desde el punto de vista de los Puntos de Función existe un Archivo Lógico Interno que almacena la información de los documentos.

Archivos de Interfaz Externos (EIF, del inglés External Interface Files): grupo de datos relacionados lógicamente e identificables por el usuario, que se utilizan solamente para fines de referencia. Los datos residen enteramente fuera de los límites del sistema y se mantienen por las Entradas Externas de otras aplicaciones, es decir, cada Archivo de Interfaz Externo es un Archivo Lógico Interno de otra aplicación.

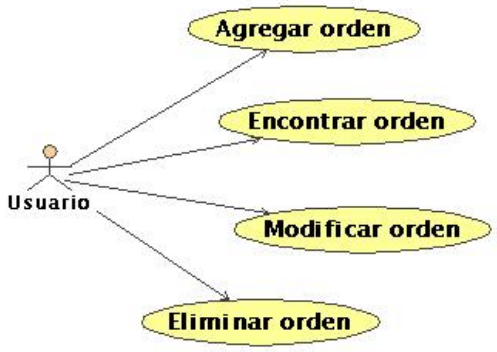
Ejemplo: un caso de uso que como parte de alguna de sus secuencias de pasos indique que el sistema debe consultar información de alguna base de datos externa y mantenida por otro sistema. Ese paso estaría dando la pauta de la existencia de un archivo de Interfaz Externo desde el punto de vista de los Puntos de Función.

2.2. Estimación inicial sobre los Casos de Uso identificados

La especificación de requerimientos mediante Casos de Uso comienza con la identificación de los Actores del sistema (usuarios u otros sistemas) y continúa con la identificación de los Casos de Uso. En ésta primera aproximación, se tiene una breve descripción de cada Caso de Uso, relatando sintéticamente la funcionalidad que brinda el mismo en beneficio de los actores.

Con ésta información, se puede efectuar una estimación inicial del tamaño en Puntos de Función, basándose en el nombre y la descripción de cada Caso de Uso. Esta aproximación es bastante grosera, ya que no existe una información completa de los requerimientos, pero puede dar una idea del tamaño del software a desarrollar.

Para ilustrar ésta situación, se presenta un breve ejemplo de Análisis de Puntos de Función a partir de un modelo UML constituído por un diagrama de Casos de Uso.
La siguiente figura muestra una parte de la funcionalidad de un sistema de administración de órdenes de compra, en la cual un Usuario (Actor) mantiene las órdenes de compra mediante cuatro Casos de Uso:



Agregar orden
Descripción: permite que el usuario efectúe el alta de una orden de compra en el sistema
Escenario principal:
1. El usuario ingresa los datos de la orden (elementos a incluir en la orden de compra, proveedor, forma de pago).
2. El sistema incorpora la orden de compra en su base de datos, asignándole un número, y le muestra al usuario la orden resultante.

Encontrar orden
Descripción: permite que el usuario ubique una orden de compra en el sistema
Escenario principal:
1. El usuario indica el número de orden de compra
2. El sistema ubica la orden de compra y la muestra al usuario

Modificar orden
Descripción: permite que el usuario modifique una orden de compra en el sistema
Escenario principal:
1. El usuario utiliza el caso de uso Encontrar orden para ubicar la orden de compra
2. El usuario ingresa los datos que desea modificar de la orden (elementos a incluir en la orden de compra, proveedor, forma de pago).
3. El sistema modifica la orden de compra en su base de datos, y le muestra al usuario la orden resultante.

Eliminar orden
Descripción: permite que el usuario elimine una orden de compra en el sistema
Escenario principal:
1. El usuario utiliza el caso de uso Encontrar orden para ubicar la orden de compra
2. El usuario confirma la eliminación de la misma.
3. El sistema elimina la orden de compra en su base de datos.

Analizando éstos casos de uso y sus descripciones, se puede ver que "Agregar orden", "Modificar orden" y "Eliminar orden" representan 3 Entradas Externas, mientras que "Encontrar orden" representa una Consulta Externa.

En todos los casos de uso está implícita la presencia de un Archivo Lógico Interno, donde se almacena la información de las órdenes de compra.

De acuerdo al Análisis de Puntos de Función, tanto las Transacciones (Entradas Externas, Salidas Externas, Consultas Externas) como los Archivos (Archivos Lógicos Internos, Archivos de Interfaz Externos) deben ser clasificados con una complejidad Baja, Media o Alta. El Apéndice A muestra un resumen de los criterios que rigen a ésta clasificación.

En éste nivel de detalle es imposible determinar la complejidad de las transacciones o los archivos utilizados para el cálculo de los Puntos de Función, con lo cual lo más razonable es asumir una complejidad media.

Entonces:
- Se tienen 3 Entradas Externas de complejidad media (valor 4, ver Apéndice A)
- Se tiene 1 Consulta Externa de complejidad media (valor 4, ver Apéndice A)
- Se tiene 1 Archivo Lógico Interno de complejidad media (valor 10, ver Apéndice A)

	Complejidad			
	Baja	Media	Alta	Aporte
Entradas Externas		3		12
Salidas Externas				
Consultas Externas		1		4
Archivos Lógicos Internos		1		10
Archivos de Interface Externos				
			Total	26

Sumando los aportes de todos los elementos se obtienen los Puntos de Función sin ajustar:

UFP (Puntos de Función sin ajustar) = 12 + 4 + 10 = 26

2.3. Estimación sobre las especificaciones de los Casos de Uso

Luego de la identificación de los Actores y Casos de Uso del sistema a desarrollar, se procede a especificar en detalle cada uno de los Casos de Uso. La forma más aceptada para la especificación de Casos de Uso consiste en la descripción de un Escenario principal que relata las acciones del actor y las del sistema durante una utilización típica, y un conjunto de Escenarios alternativos que relatan las condiciones de excepción dentro de la utilización típica, o las formas alternativas de llevar a cabo la secuencia de sucesos.

Una vez que se han especificado los casos de uso, se tiene un nivel de detalle más fino para la estimación de los puntos de función. Las secuencias que componen un escenario pueden dar lugar a una o más transacciones de Puntos de Función (Entradas Externas, Salidas Externas, Consultas Externas). Asimismo, al detallar los datos que intervienen en los escenarios, se tiene un

mejor panorama para la determinación de la complejidad de los Archivos Lógicos Internos o de Interfaz Externos.

Para esclarecer éstos conceptos, se continúa con el ejemplo anterior, tomando el caso de uso 'Encontrar orden'.

La especificación detallada del mismo podría ser la siguiente:

Escenario principal
1. El usuario indica el número de orden de compra
2. El sistema ubica la orden de compra y la muestra al usuario

Escenario alternativo
1. El usuario indica un identificador o un nombre de proveedor
2. El sistema busca todas las órdenes de ese proveedor y muestra la lista al usuario
3. El usuario selecciona un elemento de la lista
4. El sistema busca la orden seleccionada y se la muestra al usuario

Analizando ésta especificación, se desprende lo siguiente:
- los pasos 1 y 2 del escenario principal definen una Consulta Externa
- los pasos 1 y 2 del escenario alternativo definen otra Consulta Externa
- los pasos 3 y 4 del escenario alternativo definen una Consulta Externa que es la misma que la definida en los pasos 1 y 2 del flujo principal
Como resultado de éste análisis se tiene que el Caso de Uso 'Encontrar orden', que inicialmente había sido estimado en una Consulta Externa, ahora proporciona dos Consultas Externas. De la misma manera, se tiene un nivel de detalle más elevado como para poder cuantificar la complejidad de las Transacciones y los Archivos.

En resumen, a medida que se van completando las especificaciones de los Casos de Uso, se pueden ir mejorando las estimaciones de los Puntos de Función.

2.4. De la estimación de tamaño a la estimación del esfuerzo

Una vez que se han obtenido los Puntos de Función sin ajustar del sistema a partir de los Casos de Uso, se puede estimar el esfuerzo por dos métodos diferentes.

2.4.1 Puntos de Función Ajustados y Coeficientes de Conversión

El primero de ellos consiste en el cálculo de los Puntos de Función Ajustados, siguiendo el procedimiento indicado por el Análisis de Puntos de Función, que consiste en el cálculo de un Factor de Ajuste en base a la cuantificación de ciertos coeficientes vinculados con las características deseadas del sistema (comunicación de datos, rendimiento, facilidades de instalación, de operación, frecuencia de transacciones, etc.). Los detalles para el cálculo del Factor de ajuste se muestran en el Apéndice B.

Continuando con el ejemplo del punto 2.2, calculamos el Factor de Ajuste y los Puntos de Función Ajustados.

Cálculo del Factor de Ajuste:

Característica	Descripción	Peso
Comunicación de datos	Aplicación web	3
Procesamiento distribuido de datos	No hay procesamiento distribuído, pero hay datos distribuídos	2
Rendimiento	No hay requerimientos especiales de rendimiento	0
Configuraciones fuertemente utilizadas	No hay restricciones con respecto al hardware	0
Frecuencia de transacciones	Hay un pico diario de transacciones	3
Entrada de datos on-line	Todos los datos se ingresan on-line	5
Eficiencia del usuario final	Media	3
Actualizaciones on-line	La mayoría de los archivos se actualizan on-line	3
Procesamiento complejo	No hay procesamiento lógico ni matemático complejo	0
Reusabilidad	Se pretende algún grado de reutilización	2
Facilidad de instalación	No hay restricciones	0
Facilidad de operación	Operación desatendida	5
Instalación en distintos lugares	No se requiere más de una instalación	0
Facilidad de cambio	Media	3

Con los valores asignados a las características, se obtiene el Grado Total de Influencia como:
TDI = 3 + 2 + 0 + 0 + 3 + 5 + 3 + 3 + 0 + 2 + 0 + 5 + 0 + 3 = 29

y el Factor de Ajuste como:
AF = TDI x 0.01 + 0.65 = 29 x 0.01 + 0.65 = 0.94

Finalmente, los Puntos de Función Ajustados dan:

FP = UFP x AF = 26 x 0.94 = 24.44

Luego de obtener los Puntos de Función Ajustados, se pueden aplicar coeficientes que conviertan ése valor a otros como el esfuerzo, el costo o el tiempo. Estos coeficientes se obtienen fundamentalmente de la información histórica de proyectos de la organización, aunque existen algunos valores medios disponibles, recopilados estadísticamente de la industria del software.

2.4.2 COCOMO II

El segundo de los métodos posibles es la aplicación del método COCOMO II directamente sobre los Puntos de

Función sin ajustar. Éste método es el preferido en la actualidad para la estimación del esfuerzo cuando no se tiene información histórica a la cual recurrir.

COCOMO II consiste básicamente en la aplicación de ecuaciones matemáticas sobre los Puntos de Función sin ajustar o la cantidad de líneas de código (SLOC, Source Lines Of Code) estimados para un proyecto. Estas ecuaciones se encuentran ponderadas por ciertos factores de costo (cost drivers) que influyen en el esfuerzo requerido para el desarrollo del software. El Apéndice C presenta una breve introducción al método y sus conceptos principales.

A manera de ejemplo, se aplica el método COCOMO II a la estimación inicial obtenida en el apartado 2.2. Como resultado de la estimación se tenía:

UFP (Puntos de Función sin ajustar) = 26

Para aplicar la ecuación de cálculo del esfuerzo nominal (ecuación 1 del Apéndice C), necesitamos por un lado convertir los puntos de función sin ajustar a KSLOC (Source Lines Of Code, en miles), y por otro calcular el Factor escalar B de acuerdo a las características del proyecto. Luego:

$$\mathbf{PM_{nominal} = A \times (Size)^{B}}$$

A: tomamos el valor por defecto del modelo, ajustado en 2.94

Size: se calcula como el producto de los puntos de función sin ajustar por un factor de conversión que depende del lenguaje a utilizar en el desarrollo del sistema. Supongamos que utilizamos C++ (factor de conversión = 53 SLOC/UFP). Entonces tendremos:

Size = 53 x 26 = 1378 SLOC

B: se calcula ponderando las variables escalares de acuerdo al punto b) del Apéndice C, mediante la ecuación

$$\mathbf{B = 0.91 + 0.01 \times \Sigma(W_i)}$$

donde las $\mathbf{W_i}$ se muestran en la siguiente tabla:

Variable	Descripción	Ponderación	Valor
PREC	El sistema es muy familiar	Muy Alto	1.24
FLEX	Algo de relajación en cuanto a la flexibilidad del desarrollo	Nominal	3.04
RESL	La arquitectura es sólida y los riesgos generalmente se mitigan	Alto	2.83
TEAM	La interacción del equipo es altamente cooperativa	Muy Alto	1.10
PMAT	La madurez del proceso software es baja	Bajo	6.24
		Total	1.05

Finalmente, el esfuerzo nominal resulta:

$$\mathbf{PM_{nominal} = A \times (Size)^{B}} = 2.94 \times (1.378)^{1.05} = 4.11$$

Meses-hombre

Para completar la estimación, hay que ajustar el esfuerzo nominal de acuerdo a las características del proyecto según se indica en el punto c) del Apéndice C. El ajuste se efectua aplicando la ecuación

$$\mathbf{PM_{ajustado} = PM_{nominal} \times \Pi(ME_i)}$$

donde los $\mathbf{ME_i}$ (multiplicadores de esfuerzo) varían en función del modelo de estimación seleccionado (Diseño Preliminar o Post arquitectura).
En nuestro caso vamos a aplicar el modelo de Diseño preliminar. Entonces, cuantificamos los multiplicadores de esfuerzo para éste modelo:

Multiplicador	Descripción	Ponderación	Valor
PERS	Se tienen analistas y programadores con alta eficiencia y capacidad de trabajo en equipo. Dedicación full-time.	Alto	0.83
RCPX	Las exigencias de confiabilidad, documentación y volumen de datos son moderadas, y la complejidad del producto es baja.	Nominal	1
RUSE	No se pretende reutilizar nada	Bajo	0.95
PDIF	No existen restricciones en cuando al tiempo de CPU o al consumo de memoria, la plataforma es muy estable.	Bajo	0.87
PREX	Tanto los analistas como los programadores tienen aproximadamente 6 meses de experiencia en la aplicación, la plataforma, el lenguaje y las herramientas utilizadas.	Muy Bajo	1.33
SCED	Se requiere terminar el proyecto en el tiempo estimado.	Nominal	1
FCIL	Se tienen herramientas CASE simples e infraestructura de comunicaciones básica.	Bajo	1.10
		Total	1.004

Con estos valores, el ajuste del esfuerzo resulta:

$$\mathbf{PM_{ajustado}} = 4.11 \times 1.004 = 4.13 \text{ Meses-hombre}$$

Expresando el mismo valor en Horas-hombre, y teniendo en cuenta que un mes es aproximadamente 160 horas, el esfuerzo resulta:

4.13 x 160 = 660.8 Horas-hombre

3. Puntos de Casos de Uso

La estimación mediante el análisis de Puntos de Casos de Uso es un método propuesto originalmente por Gustav Karner de Objectory AB, y posteriormente refinado por muchos otros autores. Se trata de un mé-

todo de estimación del tiempo de desarrollo de un proyecto mediante la asignación de "pesos" a un cierto número de factores que lo afectan, para finalmente, contabilizar el tiempo total estimado para el proyecto a partir de esos factores.

A continuación, se detallan los pasos a seguir para la aplicación de éste método.

3.1. Cálculo de Puntos de Casos de Uso sin ajustar

El primer paso para la estimación consiste en el cálculo de los Puntos de Casos de Uso sin ajustar. Este valor, se calcula a partir de la siguiente ecuación:

UUCP = UAW + UUCW

donde,
- **UUCP**: Puntos de Casos de Uso sin ajustar
- **UAW**: Factor de Peso de los Actores sin ajustar
- **UUCW**: Factor de Peso de los Casos de Uso sin ajustar

3.1.1 Factor de Peso de los Actores sin ajustar (UAW)

Este valor se calcula mediante un análisis de la cantidad de Actores presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Actores se establece teniendo en cuenta en primer lugar si se trata de una persona o de otro sistema, y en segundo lugar, la forma en la que el actor interactúa con el sistema. Los criterios se muestran en la siguiente tabla:

Tipo de Actor	Descripción	Factor de Peso
Simple	Otro sistema que interactúa con el sistema a desarrollar mediante una interfaz de programación (API, Application Programming Interface)	1
Medio	Otro sistema que interactúa con el sistema a desarrollar mediante un protocolo o una interfaz basada en texto	2
Complejo	Una persona que interactúa con el sistema mediante una interfaz gráfica	3

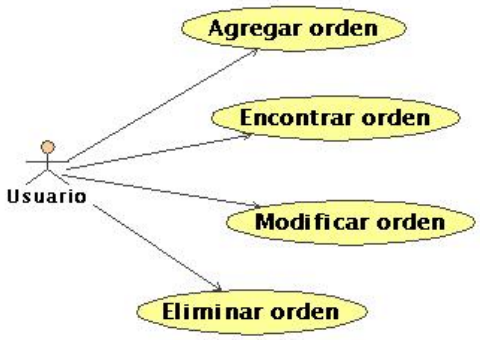
3.1.2 Factor de Peso de los Casos de Uso sin ajustar (UUCW)

Este valor se calcula mediante un análisis de la cantidad de Casos de Uso presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Casos de Uso se establece teniendo en cuenta la cantidad de transacciones efectuadas en el mismo, donde una transacción se entiende como una secuencia de actividades atómica, es decir, se efectúa la secuencia de actividades completa, o no se efectúa ninguna de las actividades de la secuencia. Los criterios se muestran en la siguiente tabla:

Tipo de Caso de Uso	Descripción	Factor de Peso
Simple	El Caso de Uso contiene de 1 a 3 transacciones	5
Medio	El Caso de Uso contiene de 4 a 7 transacciones	10
Complejo	El Caso de Uso contiene más de 8 transacciones	15

Ejemplo

Para aclarar los conceptos vistos hasta el momento, podemos retomar el sistema de administración de órdenes de compra tratado en el ejemplo del apartado 2.2



Aplicando el análisis de Puntos de Casos de Uso sin ajustar, se tiene:

Factor de Peso de los Actores sin ajustar (UAW)

El Usuario constituye un actor de tipo complejo, ya que se trata de una persona utilizando el sistema mediante una interfaz gráfica, al cual se le asigna un peso 3. Luego, el factor de peso de los actores sin ajustar resulta:

$$UAW = 1 \times 3 = 3$$

Factor de Peso de los Casos de Uso sin ajustar (UUCW)

Cada uno de los casos de uso “Agregar orden”, “Modificar orden” y “Eliminar orden” consisten de una única transacción, y el caso de uso “Encontrar orden” consiste de dos transacciones (como se vió en el ejemplo del apartado 2.3). Se tienen entonces 4 casos de uso tipo simple (peso 5), con lo cual el factor de peso de los casos de uso sin ajustar resulta:

$$UUCW = 4 \times 5 = 20$$

Finalmente, los Puntos de Casos de Uso sin ajustar resultan

$$UUCP = UAW + UUCW = 3 + 20 = 23$$

3.2. Cálculo de Puntos de Casos de Uso ajustados

Una vez que se tienen los Puntos de Casos de Uso sin

ajustar, se debe ajustar éste valor mediante la siguiente ecuación:

UCP = UUCP x TCF x EF

donde,
UCP: Puntos de Casos de Uso ajustados
UUCP: Puntos de Casos de Uso sin ajustar
TCF: Factor de complejidad técnica
EF: Factor de ambiente

3.2.1 Factor de complejidad técnica (TCF)

Este coeficiente se calcula mediante la cuantificación de un conjunto de factores que determinan la complejidad técnica del sistema. Cada uno de los factores se cuantifica con un valor de 0 a 5, donde 0 significa un aporte irrelevante y 5 un aporte muy importante. En la siguiente tabla se muestra el significado y el peso de cada uno de éstos factores:

Factor	Descripción	Peso
T1	Sistema distribuído	2
T2	Objetivos de performance o tiempo de respuesta	1
T3	Eficiencia del usuario final	1
T4	Procesamiento interno complejo	1
T5	El código debe ser reutilizable	1
T6	Facilidad de instalación	0.5
T7	Facilidad de uso	0.5
T8	Portabilidad	2
T9	Facilidad de cambio	1
T10	Concurrencia	1
T11	Incluye objetivos especiales de seguridad	1
T12	Provee acceso directo a terceras partes	1
T13	Se requieren facilidades especiales de entrenamiento a usuarios	1

El Factor de complejidad técnica se calcula mediante la siguiente ecuación:

$$\mathbf{TCF = 0.6 + 0.01 \times \Sigma (Peso_i \times Valor\ asignado_i)}$$

3.2.2 Factor de ambiente (EF)

Las habilidades y el entrenamiento del grupo involucrado en el desarrollo tienen un gran impacto en las estimaciones de tiempo. Estos factores son los que se contemplan en el cálculo del Factor de ambiente. El cálculo del mismo es similar al cálculo del Factor de complejidad técnica, es decir, se trata de un conjunto de factores que se cuantifican con valores de 0 a 5.

En la siguiente tabla se muestra el significado y el peso de cada uno de éstos factores.

Factor	Descripción	Peso
E1	Familiaridad con el modelo de proyecto utilizado	1.5
E2	Experiencia en la aplicación	0.5
E3	Experiencia en orientación a objetos	1
E4	Capacidad del analista líder	0.5
E5	Motivación	1
E6	Estabilidad de los requerimientos	2
E7	Personal part-time	-1
E8	Dificultad del lenguaje de programación	-1

- Para los factores E1 al E4, un valor asignado de 0 significa sin experiencia, 3 experiencia media y 5 amplia experiencia (experto).
- Para el factor E5, 0 significa sin motivación para el proyecto, 3 motivación media y 5 alta motivación.
- Para el factor E6, 0 significa requerimientos extremadamente inestables, 3 estabilidad media y 5 requerimientos estables sin posibilidad de cambios.
- Para el factor E7, 0 significa que no hay personal part-time (es decir todos son full-time), 3 significa mitad y mitad, y 5 significa que todo el personal es part-time (nadie es full-time).
- Para el factor E8, 0 significa que el lenguaje de programación es fácil de usar, 3 medio y 5 que el lenguaje es extremadamente difícil.

El Factor de ambiente se calcula mediante la siguiente ecuación:

$$\mathbf{EF = 1.4 - 0.03 \times \Sigma (Peso_i \times Valor\ asignado_i)}$$

Ejemplo

Continuando con el ejemplo del apartado 3.1, se calculan los Puntos de Casos de Uso ajustados.

Factor de complejidad técnica (TCF)

Factor	Descripción	Peso	Valor asignado	Comentario
T1	Sistema distribuído	2	0	El sistema es centralizado
T2	Objetivos de performance o tiempo de respuesta	1	1	La velocidad es limitada por las entradas provistas por el usuario
T3	Eficiencia del usuario final	1	1	Escasas restricciones de eficiencia
T4	Procesamiento interno complejo	1	1	No hay cálculos complejos
T5	El código debe ser reutilizable	1	0	No se requiere que el código sea reutilizable
T6	Facilidad de instalación	0.5	1	Escasos requerimientos de facilidad de instalación
T7	Facilidad de uso	0.5	3	Normal
T8	Portabilidad	2	0	No se requiere que el sistema sea portable
T9	Facilidad de cambio	1	3	Se requiere un costo moderado de mantenimiento
T10	Concurrencia	1	0	No hay concurrencia
T11	Incluye objetivos especiales de seguridad	1	3	Seguridad normal
T12	Provee acceso directo a terceras partes	1	5	Los usuarios web tienen acceso directo
T13	Se requieren facilidades especiales de entrenamiento a usuarios	1	1	Pocos usuarios internos, sistema fácil de usar

El Factor de complejidad técnica resulta:

TCF = 0.6 + 0.01 x 17 = 0.77

Factor de ambiente (EF)

Factor	Descripción	Peso	Valor asignado	Comentario
E1	Familiaridad con el modelo de proyecto utilizado	1.5	4	El grupo está bastante familiarizado con el modelo
E2	Experiencia en la aplicación	0.5	4	La mayoría del grupo ha trabajado mucho tiempo en ésta aplicación
E3	Experiencia en orientación a objetos	1	4	La mayoría del grupo programa en objetos
E4	Capacidad del analista líder	0.5	5	Se contrató a un especialista
E5	Motivación	1	5	El grupo está altamente motivado
E6	Estabilidad de los requerimientos	2	2	Se esperan cambios
E7	Personal part-time	-1	0	Todo el grupo es full-time
E8	Dificultad del lenguaje de programación	-1	3	Se usará lenguaje C++

El Factor de ambiente resulta:

EF = 1.4 - 0.03 x 19.5 = 0.82

Finalmente, los Puntos de Casos de Uso ajustados resultan:

UCP = 23 * 0.77 * 0.82 = 14.52

3.3. De los Puntos de Casos de Uso a la estimación del esfuerzo

Karner originalmente sugirió que cada Punto de Casos de Uso requiere 20 horas-hombre. Posteriormente, surgieron otros refinamientos que proponen una granularidad algo más fina, según el siguiente criterio:

- Se contabilizan cuántos factores de los que afectan al Factor de ambiente están por debajo del valor medio (3), para los factores E1 a E6.

- Se contabilizan cuántos factores de los que afectan al Factor de ambiente están por encima del valor medio (3), para los factores E7 y E8.
- Si el total es 2 o menos, se utiliza el factor de conversión 20 horas-hombre/Punto de Casos de Uso, es decir, un Punto de Caso de Uso toma 20 horas-hombre.
- Si el total es 3 o 4, se utiliza el factor de conversión 28 horas-hombre/Punto de Casos de Uso, es decir, un Punto de Caso de Uso toma 28 horas-hombre.
- Si el total es mayor o igual que 5, se recomienda efectuar cambios en el proyecto, ya que se considera que el riesgo de fracaso del mismo es demasiado alto.

El esfuerzo en horas-hombre viene dado por:

E = UCP x CF

donde,
E: esfuerzo estimado en horas-hombre
UCP: Puntos de Casos de Uso ajustados
CF: factor de conversión

Se debe tener en cuenta que éste método proporciona una estimación del esfuerzo en horas-hombre contemplando sólo el desarrollo de la funcionalidad especificada en los casos de uso.

Finalmente, para una estimación más completa de la duración total del proyecto, hay que agregar a la estimación del esfuerzo obtenida por los Puntos de Casos de Uso, las estimaciones de esfuerzo de las demás actividades relacionadas con el desarrollo de software. Para ello se puede tener en cuenta el siguiente criterio, que estadísticamente se considera aceptable. El criterio plantea la distribución del esfuerzo entre las diferentes actividades de un proyecto, según la siguiente aproximación:

Actividad	Porcentaje
Análisis	10.00%
Diseño	20.00%
Programación	40.00%
Pruebas	15.00%
Sobrecarga (otras actividades)	15.00%

Obviamente, éstos valores no son absolutos sino que pueden variar de acuerdo a las características de la organización y del proyecto.

Con éste criterio, y tomando como entrada la estimación de tiempo calculada a partir de los Puntos de Casos de Uso, se pueden calcular las demás estimaciones para obtener la duración total del proyecto.

Ejemplo

Aplicando éstos criterios al ejemplo que se venía desarrollando en éste apartado, se obtiene el esfuerzo necesario para el desarrollo de los casos de uso como:

E = 14.52 * 20 = 290.4 Horas-Hombre

Si además se considera que este esfuerzo representa un porcentaje del esfuerzo total del proyecto, de acuerdo a los valores porcentuales de la tabla anterior, se obtiene:

Actividad	Porcentaje	Horas-Hombre
Análisis	10.00%	72.6
Diseño	20.00%	145.2
Programación	40.00%	290.4
Pruebas	15.00%	108.9
Sobrecarga (otras actividades)	15.00%	108.9
Total	100.00%	726

Comparando éste resultado con el obtenido en el párrafo 2.4.2 (estimación por COCOMO II) vemos que resultan similares.

4. Conclusiones

Se ha aplicado los métodos presentados a lo largo del artículo para estimar el esfuerzo en algunos proyectos de su ámbito laboral, obteniendo resultados satisfactorios en su mayoría, en cuanto a la precisión de las estimaciones con respecto a la cantidad de información disponible.

En la experiencia del mismo, se pueden destacar las siguientes apreciaciones:

- La estimación a partir de Puntos de Función ajustados y Coeficientes de Conversión es difícil de realizar si no se cuenta con una base histórica de proyectos que provea los coeficientes de conversión. Los valores estadísticos son difíciles de encontrar.

- La estimación por COCOMO II (con Puntos de Función sin ajustar como entrada), resulta muy útil para estimar un proyecto en forma global, cuando se tiene un conjunto de Casos de Uso bastante amplio (del orden de 50) y con escaso nivel de detalle. Utilizando la herramienta del SEI (Software Engineering Institute), se puede refinar la estimación a medida que se va adquiriendo más información sobre el proyecto. Cabe aclarar la herramienta mencionada no está calibrada para proyectos menores a 2000 líneas de código, con lo cual no es aplicable a proyectos muy pequeños.

- La estimación por Puntos de Caso de Uso resulta muy efectiva para estimar el esfuerzo requerido en el desarrollo de los primeros Casos de Uso de un sistema, si se sigue una aproximación iterativa como el Proceso Unificado de Rational. En éste tipo de aproximación, los primeros Casos de Uso a desarrollar son los que ejercitan la mayor parte de la arquitectura del software y los que a su vez ayudan a mitigar los riesgos más significativos (iteraciones de Elaboración en el Proceso Unificado). Fuera de éste contexto, el método tiende a sobredimensionar el esfuerzo requerido por lo cual el autor no lo recomienda para estimar el esfuerzo global de un proyecto.

Si bien ninguno de éstos métodos es la panacea, todos ellos aportan a la formación del Ingeniero de Software, y la aplicación sistemática de los mismos permite obtener mediciones y puntos de comparación que ayudan a ampliar la experiencia profesional en la estimación de proyectos de software.

Apéndice A: Clasificación de Transacciones y Archivos en Análisis de Puntos de Función.

La complejidad de las Transacciones y los Archivos en el Análisis de Puntos de Función, se puede clasificar y cuantificar de acuerdo con los criterios que se muestran a continuación:

Transacciones

a) Clasificación de las Entradas Externas

Para las Entradas Externas, la clasificación está dada por la siguiente tabla:

Archivos referenciados	Elementos de datos		
	1-4	5-15	>15
0-1	Baja	Baja	Media
2	Baja	Media	Alta
3 o más	Media	Alta	Alta

En la misma, "Archivos referenciados" representa el número de Archivos Lógicos Internos mantenidos por la Entrada Externa, y "Elementos de datos" representa la cantidad de elementos que componen la Entrada Externa.

b) Clasificación de las Salidas Externas y Consultas Externas

Para las Salidas Externas y las Consultas Externas, la clasificación está dada por la siguiente tabla:

Archivos referenciados	Elementos de datos		
	1-5	6-19	>19
0-1	Baja	Baja	Media
2-3	Baja	Media	Alta
>3	Media	Alta	Alta

En la misma, "Archivos referenciados" representa el número de Archivos Lógicos Internos o Archivos de Interfaz Externos vinculados con la Salida Externa o la Consulta Externa, y "Elementos de datos" representa la cantidad combinada de elementos de datos de entrada y de salida que componen la Salida Externa o Consulta Externa.

c) Asignación de valores numéricos

Los valores numéricos que se asignan a cada complejidad (Baja, Media o Alta), se muestran en la siguiente tabla, para cada uno de los tipos de transacción (Entrada Externa, Salida Externa, Consulta Externa):

Clasificación	Valores		
	Salidas Externas	Consultas Externas	Entradas Externas
Baja	4	3	3
Media	5	4	4
Alta	7	6	6

Archivos

a) Clasificación de los Archivos Lógicos Internos y Archivos de Interfaz Externos

Para los Archivos Lógicos Externos y los Archivos de Interfaz Externos, la clasificación está dada por la siguiente tabla:

Tipos de registro	Elementos de datos		
	1-19	20-50	>50
1	Baja	Baja	Media
2-5	Baja	Media	Alta
>5	Media	Alta	Alta

donde "Tipos de registro" representa un subgrupo de elementos de datos reconocibles por el usuario, y

"Elementos de datos" representa la cantidad de elementos de datos básicos (campos únicos) que componen el Archivo.

El concepto de "Tipos de registro" se puede ver mejor mediante un ejemplo:

Supongamos que se almacena la información de un CD de música en un Archivo Lógico Interno. La información asociada al CD es el Cantante, el Productor, el Título, la Fecha y las Canciones. Cada canción a su vez tiene como información asociada un Nombre, un Autor y una Duración.

En este caso se tienen 2 "Tipos de registro", la información del CD y la información de la canción. A su vez, hay 4 "Elementos de datos" para la información del CD (Cantante, Productor, Título y Fecha), y 3 "Elementos de datos" para la información de la canción (Nombre, Autor y Duración).

b) Asignación de valores numéricos
Los valores numéricos que se asignan a cada complejidad (Baja, Media o Alta), se muestran en la siguiente tabla, para cada uno de los tipos de archivo (Archivo Lógico Externo, Archivo de Interfaz Externo):

Clasificación	Valores	
	Archivo Lógico Interno	Archivo de Interface Externo
Baja	7	5
Media	10	7
Alta	15	10

Síntesis del Apéndice:

El conteo de Puntos de Función comienza con la identificación de las Transacciones (Entradas Externas, Salidas Externas, Consultas Externas) y los Archivos (Lógicos Internos o de Interface Externos).
Una vez identificados éstos elementos, se utilizan los criterios mostrados de manera de cuantificar el aporte de cada uno de ellos a la suma total de Puntos de Función.

Apéndice B: Cálculo del Factor de Ajuste en Análisis de Puntos de Función.

El Análisis de Puntos de Función plantea el ajuste de los Puntos de Función calculados a partir de las Transacciones y Archivos, mediante la evaluación de 14 características generales del sistema. A cada una de estas características se le asigna un factor de peso (un valor entre 0 y 5) que indica la importancia de la característica para el sistema bajo análisis. El significado del valor asignado a cada característica es el siguiente:

0 No presente o sin influencia
1 Influencia incidental
2 Influencia moderada
3 Influencia media
4 Influencia significativa
5 Fuerte influencia

La siguiente tabla muestra las características a tener en cuenta:

Característica	**Descripción**
Comunicación de datos	Cuántas facilidades de comunicación hay disponibles para ayudar en el intercambio de información"con la aplicación o el sistema?
Procesamiento distribuido de datos	Cómo se manejan los datos y las funciones de procesamiento distribuído?
Rendimiento	Existen requerimientos de velocidad o tiempo de respuesta?
Configuraciones fuertemente utilizadas	Qué tan intensivamente se utiliza la plataforma de hardware donde se ejecutará la aplicación o el sistema?
Frecuencia de transacciones	Que tan frecuentemente se ejecutan las transacciones? Diariamente, semanalmente, mensualmente?
Entrada de datos on-line	Qué porcentaje de la información se ingresa on-line?
Eficiencia del usuario final	Se designa la aplicación para maximizar la eficiencia del usuario final?
Actualizaciones on-line	Cuántos Archivos Lógicos Internos se actualizan por una transacción on-line?
Procesamiento complejo	Hay procesamientos lógicos o matemáticos intensivos en la aplicación?
Reusabilidad	La aplicación se desarrolla para suplir una o muchas de las necesidades de los usuarios?
Facilidad de instalación	Qué tan difícil es la instalación y la conversión al nuevo sistema?
Facilidad de operación	Que tan efectivos o automatizados son los procedimientos de arranque, parada, backup y restore del sistema?
Instalación en distintos lugares	La aplicación fue concebida para su instalación en múltiples sitios y organizaciones?
Facilidad de cambio	La aplicación fue concebida para facilitar los cambios sobre la misma?

Cada una de éstas características aporta un valor entre 0 y 5, de acuerdo a la importancia que tenga en el sistema. Luego se suman los aportes de cada una de las características, obteniendo el grado total de influencia (TDI, del inglés Total Degree of Influence), y se calcula el Factor de Ajuste como:

AF = (TDI x 0.01) + 0.65

Finalmente, los Puntos de Función Ajustados se obtienen como el producto de los Puntos de Función sin ajustar por el Factor de Ajuste:

FP = UFP x AF

Síntesis del Apéndice:

El Análisis de Puntos de Función plantea que luego de la contabilización de los Puntos de Función sin ajustar, se puede realizar un ajuste sobre el valor obtenido.
Este ajuste tiene en cuenta un conjunto de características del sistema no contempladas en el conteo de las Transacciones y los Archivos.
Las características (que forman un total de 14) se ponderan con un valor entre 0 y 5, y permiten obtener un factor de ajuste que se aplica a los Puntos de Función sin ajustar para obtener los Puntos de Función ajustados.

Apéndice C: Introducción al modelo de estimación COCOMO II.

a) Descripción general
El método de estimación COCOMO II está basado dos modelos: uno aplicable al comienzo de los proyectos (Diseño preliminar, en inglés Early Design) y otro aplicable luego del establecimiento de la arquitectura del sistema (Post arquitectura, en inglés Post Architecture).

El modelo de **Diseño preliminar (Early Design)** contempla la exploración de las arquitecturas alternativas del sistema y los conceptos de operación. En esta etapa no se sabe lo suficiente del proyecto como para hacer una estimación fina. Ante ésta situación, el modelo propone la utilización de Puntos de Función como medida de tamaño y un conjunto de 7 factores (cost drivers) que afectan al esfuerzo del proyecto. Estos 7 factores son agrupaciones de los factores que se utilizan en la otra variante del modelo (Post Arquitectura).

El modelo **Post arquitectura (Post Architecture)** contempla el desarrollo y el mantenimiento de un producto software. Esta estrategia es más precisa si se ha desarrollado una arquitectura del sistema, la cual haya sido validada y establecida como base para la evolución del producto. Ante ésta situación, el modelo propone la utilización de Líneas de código fuente y/o Puntos de Función como medidores del tamaño, modificadores para indicar el grado de reutilización y descarte del software, un conjunto de 17 estimadores de costo, y un conjunto de 5 factores que afectan de manera exponencial en el esfuerzo del proyecto.

En ambos modelos, la estimación del esfuerzo se realiza tomando como base la siguiente ecuación:

$$\mathbf{PM_{nominal} = A \times (Size)^{B}} \quad (1)$$

donde
$\mathbf{PM_{nominal}}$: es el esfuerzo nominal requerido en meses-hombre
Size: es el tamaño estimado del software, en miles de líneas de código (KSLOC) o en Puntos de Función sin ajustar (convertibles a KSLOC mediante un factor de conversión que depende del lenguaje y la tecnología).
A: es una constante que se utiliza para capturar los efectos multiplicativos en el esfuerzo requerido de acuerdo al crecimiento del tamaño del software. El modelo la calibra inicialmente con un valor de 2.94
B: es una constante denominada **Factor escalar**, la cual tiene un impacto exponencial en el esfuerzo y su valor está dado por la resultante de los aspectos positivos sobre los negativos que presenta el proyecto.

b) Valoración del Factor escalar B
El factor escalar B se calcula a partir de la sumatoria de los aportes de distintas Variables escalares, las cuales son variables que indican las características que el proyecto presenta en lo que a su complejidad y entorno de desarrollo se refiere. Las Variables escalares de COCOMO II son las siguientes:

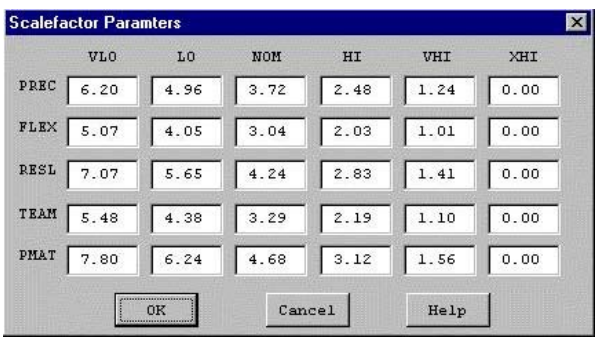
- PREC, variable de precedencia u orden secuencial del desarrollo
- FLEX, variable de flexibilidad del desarrollo
- RSEL, indica la fortaleza de la arquitectura y métodos de estimación y reducción de riesgos
- TEAM, esta variable refleja la cohesión y madurez del equipo de trabajo
- PMAT, relaciona el proceso de madurez del software

Cada una de estas variables se cuantifica con un valor desde Muy Bajo hasta Extra Alto.

La siguiente tabla muestra los criterios y niveles de cuantificación para cada una de éstas variables:

Factor Escalar (W_i)	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
PREC	Completa	Completa	Algo	Familiar	Muy Familiar	Absolutamente Familiar
FLEX	Riguroso	Ocasional	Algo de relajación	Generalmente conforme	Algo de conformidad	Objetivos generales
RESL	Poco (20%)	Algo (40%)	A menudo (60%)	Generalmente (75%)	Mayormente (90%)	Totalmente (100%)
TEAM	Interacción muy difícil	Algo de dificultad de interacción	Básicamente hay interacción cooperativa	Cooperativa	Altamente cooperativa	Interacción total
PMAT	Promedio de respuestas afirmativas en el cuestionario de CMM	Promedio de respuestas afirmativas en el cuestionario de CMM	Promedio de respuestas afirmativas en el cuestionario de CMM	Promedio de respuestas afirmativas en el cuestionario de CMM	Promedio de respuestas afirmativas en el cuestionario de CMM	Promedio de respuestas afirmativas en el cuestionario de CMM

Los valores que asumen cada uno de éstos factores en cada nivel se pueden ver en la siguiente figura:



Luego de la ponderación de éstas variables, el Factor escalar se calcula mediante la siguiente ecuación:

$$\mathbf{B = 0.91 + 0.01 \times \Sigma(W_i)} \quad (2)$$

c) Ajuste del esfuerzo nominal
El esfuerzo calculado en la ecuación (1) es un valor nominal y debe ser ajustado en base a las características del proyecto. COCOMO II obtiene los datos necesarios para el ajuste del esfuerzo nominal considerando un conjunto de **Multiplicadores de Esfuerzo (ME)**, los cuales representan las características del proyecto y expresan su impacto en el desarrollo total del producto de software.

Los Multiplicadores de esfuerzo se cuantifican con una escala que va desde Extra Bajo a Extra Alto, y cada multiplicador tiene un valor asociado a cada nivel de la escala.
Cada uno de los modelos de estimación (Diseño preliminar y Post arquitectura) tiene un conjunto de Multiplicadores de esfuerzo, los cuales son acordes con la información que se maneja en cada uno de estos modelos.

En ambos modelos, el esfuerzo ajustado se calcula mediante la siguiente ecuación:

$$\mathbf{PM_{ajustado} = PM_{nominal} \times \Pi(ME_i)} \quad (3)$$

d) Multiplicadores de esfuerzo en el modelo Post arquitectura
Para este modelo, los multiplicadores son 17, agrupados en las siguientes categorías: producto, plataforma, personal y proyecto. A continuación se muestran los multiplicadores, con una breve descripción de su significado.

Multiplicadores que afectan al producto:

RELY: Confiabilidad requerida del software. Mide el impacto que tiene una falla en el software.

	Muy Bajo	**Bajo**	**Nominal**	**Alto**	**Muy Alto**
RELY	Inconvenientes imperceptibles	Bajo, y con perdidas fácilmente recuperables	Moderado, con perdidas de fácil recuperación	Altas pérdidas financieras	Riesgo para la vida humana

DATA: Tamaño de la base de datos. Se mide como el tamaño de la base en bytes sobre el tamaño del programa en LOC. Se utiliza para dimensionar el esfuerzo requerido para el control y la generación de datos de prueba.

	Muy Bajo	**Bajo**	**Nominal**	**Alto**	**Muy Alto**
DATA		D/P < 10	10 <= D/P < 100	100 <= D/P < 1000	D/P >= 1000

CPLX: Complejidad del producto. La complejidad se divide en cinco áreas: Operaciones de Control, Operaciones de Cálculo, Dependencia de Dispositivos, Manejo de Datos e Interfaces de Usuario.

	Operaciones de Control	**Operaciones de Cálculo**	**Dependencia de Dispositivos**	**Manejo de Datos**	**Interfaces de Usuarios**
Muy Bajo	Programación lineal, con muy pocas estructuras no anidadas: DO, IF, CASE. Composición de módulos simples a través de procedimientos y funciones	Evaluaciones de expresiones simples.	Escrituras y grabaciones con formatos simples.	Manejo de vectores simples en memoria. Manejo de consultas y accesos a la base de datos en forma sencilla.	Formularios de ingreso sencillos. Generación de reportes simples.
Bajo	Estructuras bien anidadas	Evaluaciones moderadas de cálculos. Ej. Raíz cuadrada, exponentes	Sin particularidades o dependencias del procesador de E/S. Se manejan a través de GET y PUT de conjuntos de datos	Manejo de datos sin archivos intermedios, sin edición. COTS-DB, consultas y actualizaciones moderadas	Uso de GUI (grafic user interface, interfaces graficas de usuario) simples
Nominal	La mayoría de las estructuras son anidadas. Con controles de intercambio simple de datos entre módulos a través de parámetros, tablas de decisiones, soporte para procesamiento distribuido de baja complejidad	Uso de rutinas básicas y estándar. Manejo básico de vectores y matrices	Operaciones de E/S que permiten seleccionar el dispositivo, haciendo un control del estado de los mismos y procesando errores	Ingreso con formatos múltiples de datos, pero conservando un único formato de salida. Cambios estructurales simples con ediciones. Uso de consultas y COTS-DB complejos	Uso simple de un conjunto de parámetros de definición de interfaz del usuario.
Alto	Total uso de estructuras anidadas. Manejo de pilas y colas. Desarrollos para un único procesador	Análisis numérico sencillo: interpolación, ecuaciones diferenciales. Uso de redondeo y trunc.	Operaciones de E/S a niveles físicos usando direccionamiento para la lectura y búsqueda. Overlap optimizado para éstas operaciones	Activación de disparadores a través de datos de la DB. Reestructuraciones complejas de datos.	Uso de parámetros de definición de interfaces. Uso simple de características multimediales (entrada a través de la voz)
Muy Alto	Código recursivo. Manejo de interrupciones, y sincronización de tareas complejas. Procesamiento distribuido heterogéneo. Un único procesador controlado en tiempo real	Cálculos numéricos difíciles pero estructurados: ecuaciones matriciales, diferenciales	Rutinas para el diagnóstico de interrupciones. Manejo de la línea de comunicación entre dispositivos. Uso intensivo del manejo de performance	Uso complejo de disparadores. Optimización de consultas. Coordinación de bases de datos distribuidas	Uso moderado de 2 y 3 dimensiones. Habilidades gráficas complejas y multimediales.
Extra alto	Operaciones de control múltiples con cambios dinámicos de prioridades. Control a nivel microcódigo. Control en tiempo real del hardware distribuido.	Análisis numérico complejo y no estructurado. Cálculos complejos en paralelo	Codificación de dispositivos asincrónica, con control crítico de performance y procesos.	Uso de lenguajes de manejo de datos, y técnicas de objetos así como relacionales	Uso complejo de multimedia y realidad virtual

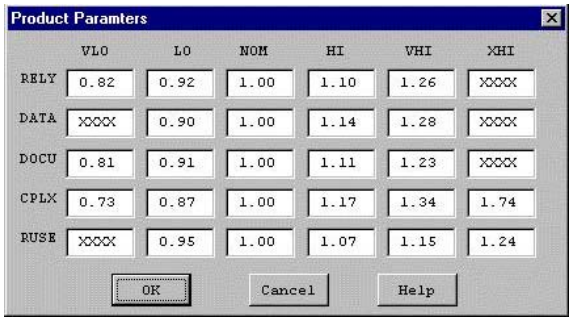
RUSE: Reusabilidad del código. Mide el costo adicional requerido para diseñar componentes más genéricos, mejor documentados y más confiables, de manera de reutilizarlos en otros proyectos.

	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
RUSE		Nada	Por Proyecto	Por Programa	Por línea de Producto	Por múltiples líneas de productos

DOCU: Documentación. Evalúa los requerimientos de documentación a lo largo del ciclo de vida del proyecto.

	Muy Bajo	Bajo	Nominal	Alto	Muy Alto
DOCU	Muchas etapas del ciclo de vida están sin documentación	Algunas	De acuerdo a las necesidades exactas de las etapas del ciclo de vida	Excesiva	Muy Excesiva

Los valores que asume cada uno de éstos multiplicadores en cada nivel se pueden ver en la siguiente figura:



Multiplicadores que afectan a la plataforma:

TIME: Restricciones de tiempo de ejecución. Se expresa en términos de porcentaje de disponibilidad de tiempo de ejecución que será usado por el sistema, versus los recursos disponibles.

	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
TIME			<= 50% de uso de los recursos disponibles	70%	85%	95%

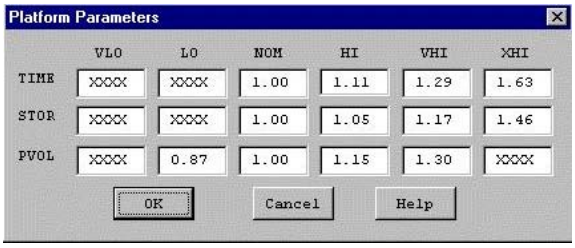
STOR: Restricciones de almacenamiento principal. Similar al multiplicador anterior, pero relacionadas con el espacio principal de almacenamiento.

	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
STOR			<= 50% de uso del espacio disponible	70%	85%	95%

PVOL: Volatilidad de la plataforma. Expresa la velocidad de cambio del hardware y el software usados como plataforma.

	Muy Bajo	Bajo	Nominal	Alto	Muy Alto
PVOL		Cambios mayores cada 12 meses, y cambios menores cada mes	Mayores : 6 meses, Menores : 2 semanas	Mayores : 2 meses, Menores : 1 semana	Mayores : 2 semanas, Menores : 2 días

Los valores que asume cada uno de éstos multiplicadores en cada nivel se pueden ver en la siguiente figura:



Multiplicadores que afectan al personal:

ACAP: Capacidad de los analistas. Se considera la capacidad de análisis y diseño, eficiencia, habilidad para comunicarse y trabajar en equipo. No se considera el nivel de experiencia.

	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
ACAP	15%	35%	55%	75%	90%	

PCAP: Capacidad de los programadores. Se considera la capacidad de trabajo en equipo, eficiencia y habilidad para comunicarse. No se considera el nivel de experiencia.

	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
PCAP	15%	35%	55%	75%	90%	

AEXP: Experiencia en aplicaciones. Contempla el nivel de experiencia del grupo de desarrollo (principalmente analistas) en aplicaciones equivalentes.

	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
AEXP	2 meses	6 meses	1 año	3 años	6 años	

PEXP: Experiencia en la plataforma. Refleja la experiencia del grupo de desarrollo (principalmente programadores) en el uso de herramientas de software y hardware utilizado como plataforma.

	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
PEXP	2 meses	6 meses	1 año	3 años	6 años	

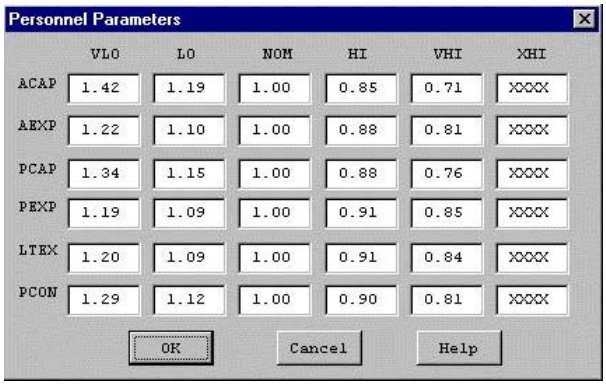
LTEX: Experiencia en el lenguaje y herramientas de desarrollo. Refleja la experiencia del grupo de desarrollo en el lenguaje de programación y las herramientas de desarrollo utilizadas.

	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
LTEX	2 meses	6 meses	1 año	3 años	6 años	

PCON: Continuidad del personal. Expresa el porcentaje de rotación anual del personal afectado al proyecto.

	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
PCON	48% al año	24% al año	12% al año	6% al año	3% al año	

Los valores que asume cada uno de éstos multiplicadores en cada nivel se pueden ver en la siguiente figura:



Multiplicadores que afectan al proyecto:

TOOL: Uso de herramientas de software. Contempla el uso de herramientas, desde la edición hasta el manejo de todo el ciclo de vida.

	Muy Bajo	Bajo	Nominal	Alto	Muy Alto
TOOL	Edición y codificación con debug	CASE simple y de poca integración	Herramientas básicas para todo el ciclo de vida con moderada integración	Potentes herramientas a ser usadas en todo el ciclo de vida con integración moderada	Herramientas potentes y proactivas, muy bien integradas con el proceso, los métodos y la reusabilidad

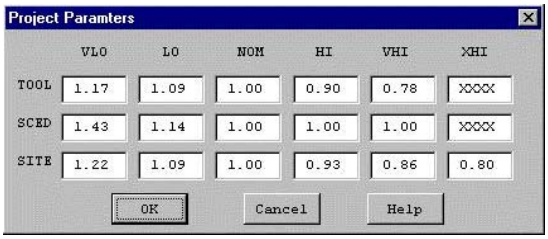
SITE: Desarrollo en múltiples ubicaciones. Involucra la ubicación física y el soporte de comunicaciones.

	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
SITE	Algo de teléfono y mail	Fax y teléfonos individuales	Red de correo electrónico interno	Comunicaciones electrónicas que cubren todas las ubicaciones	Comunicaciones electrónicas que cubren todas las ubicaciones con la posibilidad de video conferencias ocasionales.	Multimedia

SCED: Requerimientos de calendario de desarrollo. Refleja las restricciones impuestas al grupo de desarrollo sobre la agenda nominal estimada del proyecto.

	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
SCED	75% del nominal	85%	100%	130%	160%	

Los valores que asume cada uno de éstos multiplicadores en cada nivel se pueden ver en la siguiente figura:



e) Multiplicadores de esfuerzo en el modelo de Diseño preliminar

Para este modelo, los multiplicadores son 7, y se obtienen como combinaciones de los multiplicadores del modelo Post arquitectura.
Estos multiplicadores son:

PERS: Capacidad del personal. Está dado por la suma o la combinación porcentual de los multiplicadores ACAP, PCAP y PCON.

	Extra Bajo	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
Suma de ACAP, PCAP, PCON	3,4	5,6	7,8	9	10,11	12,13	14,15
Combinación de ACAP y PCAP	20%	39%	45%	55%	65%	75%	85%
Rotación anual del personal	45%	30%	20%	12%	9%	5%	4%

RCPX: Complejidad del producto. Está dado por la combinación de los multiplicadores RELY, DATA, CPLX y DOCU.

	Extra Bajo	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
Suma de RELY, DATA, CPLX y DOCU	5,6	7,8	9-11	12	13-15	16-18	19-21
Énfasis en la documentación	Muy poca	Poca	Algo	Básica	Fuerte	Muy fuerte	Extrema
Complejidad del Producto	Muy simple	Simple	Algo	Moderada	Compleja	Muy compleja	Extremadamente compleja
Tamaño de la base de datos	Pequeña	Pequeña	Pequeña	Moderada	Grande	Muy grande	Muy grande

RUSE: Reusabilidad. Está dado por el mismo multiplicador RUSE del modelo Post arquite ctura.

	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
RUSE		Nada	Por Proyecto	Por Programa	Por línea de Producto	Por múltiples líneas de productos

PDIF: Dificultad de la plataforma. Está dado por la combinación de los multiplicadores TIME, STOR y PVOL.

	Bajo	Nominal	Alto	Muy Alto	Extra Alto
Suma de TIME, STOR y PVOL	8	9	10-12	13-15	16,17
Restricciones de TIME & STOR	50%	50%	65%	80%	90%
Volatilidad de la plataforma	Muy estable	Estable	Algo volátil	Volátil	Muy volátil

PREX: Experiencia del personal. Está dado por la combinación de los multiplicadores AEXP, PEXP y LTEX

	Extra Bajo	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
Suma AEXP, PEXP y LTEX	3,4	5,6	7,8	9	10,11	12,13	14
Experiencia en la aplicación, plataforma, lenguaje y herramientas utilizadas	<= 3 meses	5 meses	9 meses	1 año	2 años	4 años	6 años

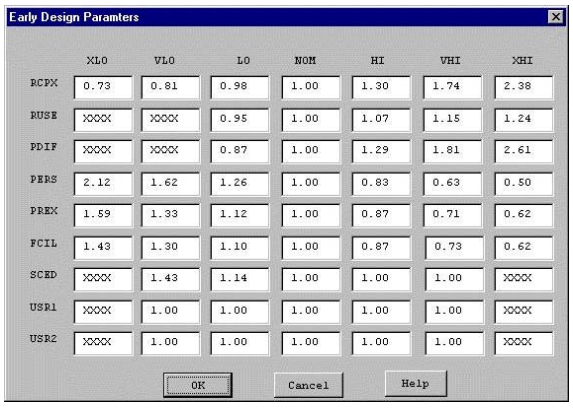
SCED: Calendario. Está dado por el mismo multiplicador SCED del modelo Post arquitectura.

	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
SCED	75% del nominal	85%	100%	130%	160%	

FCIL: Facilidades. Está dado por la combinación de los multiplicadores TOOL y SITE.

	Extra Bajo	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
Suma de TOOL y SITE	2	3	4,5	6	7,8	9,10	11
Soporte de herra-mientas de software	Mínimo	Algo	CASE Simples	Herramientas básicas según el ciclo de vida	Buenas, mode-rada-mente integradas	Muy buenas, moderadamente integradas	Muy buenas totalmente integradas
Múltiples lugares de desarrollo	Soporte débil para el desarrollo complejo	Algo de soporte para desarrollos complejos	Algo de soporte para desarrollos moderados	Soporte básico para desarrollos complejos	Fuerte soporte para el desarro-llo de procesos moderados	Fuerte soporte de desarrollos de procesos simples	Muy fuerte soporte para el desarrollo de procesos complejos

Los valores que asume cada uno de éstos multiplicado-res en cada nivel se pueden ver en la siguiente figura:



Referencias bibliográficas

- Métrica Versión 3, Ministerio de Administraciones Púbilcas Espa-ñol, http://www.map.es/csi/metrica3/index.html
- Use Cases and Function Points, artículo disponible en el site de Longstreet Consulting, http://www.softwaremetrics.com/Articles/usecases.htm
- Estimación de esfuerzo, Unidad 3 del Máster en Ingeniería del Software, ITBA.
- COCOMO II Model Manual, disponible en el CSE Center for Software Engineering, http://sunset.usc.edu/research/COCOMOII/cocomo_main.html#downloads
- USC COCOMO II User's Manual, disponible en el CSE Center for Software Engineering, http://sunset.usc.edu/research/COCOMOII/cocomo_main.html#downloads
- Artículo "Modelo COCOMO II", Magistrando Carlos G. Rivero Bianchi, publicado en julio del 2001 en la revista del Instituto Tecno-lógico de Buenos Aires (ITBA).
- Function Point Traninig Manual, disponible en el site de Longstreet Consulting, http://www.softwaremetrics.com/freemanual.htm
- Rational Unified Process, documentación online disponible con los productos de Rational.
- Time estimation in software development projects, Thomas Fihlman, disponible en http://www.callista.se/ITPartner/timeart.htm
- Test Effort Estimation Using Use Case Points, Suresh Nageswaran, Quality Week 2001, San Francisco, California, USA, June 2001, disponible en http://www.cts-corp.com/cogcommunity/presentations/Test%20Effort%20Estimation%20Using%20Use%20Case%20Points.pdf
- Understanding RET's, artículo disponible en el site de Lognstreet Consulting, http://www.softwaremetrics.com/ Articles/ret.htm