CAPÍTULO 5

Modelos empíricos de estimación.

Un modelo empírico de estimación para software puede utilizar fórmulas derivadas empíricamente para predecir el esfuerzo como una función de LDC y PF. Los valores para LDC y PF son estimados utilizando el enfoque descrito en el capítulo 3.6. pero en lugar de utilizar las tablas descritas en esas secciones, los valores resultantes para LDC y PF se unen al modelo de estimación [Len O. Ejogo '91].

Los datos empíricos que soportan la mayoría de los modelos de estimación se obtienen de una muestra limitada de proyectos. Es por eso que estos modelos de estimación no son adecuados para todas clases de software y en todos los entornos de desarrollo. Por consiguiente, los resultados obtenidos de dichos modelos se deben utilizar con prudencia.

5.1 La estructura de los modelos de estimación

Un modelo común de estimación se despega utilizando análisis de regresión sobre los datos compilados de proyectos anteriores de software. La estructura, global de tales modelos adquiere la forma de [Len O. Ejiogo '91]:

$$E = A + B * (ev)^c \qquad (5.1)$$

donde *A, B* y C son constantes conseguidas empíricamente, *E* es el esfuerzo en meses/persona, y *ev* es la variable de estimación (de LDC o PF). Además de la dependencia señalada en la ecuación, la mayoría de los modelos de estimación tienen algún componente de ajuste del proyecto que permite ajustar *E* debido a otras características del proyecto (p. ejemplo: complejidad del problema, experiencia del personal, entorno de desarrollo).

Entre los muchos modelos de estimación orientados a LDC se encuentran los siguientes:

$E = 5.2 \times (KLDC)^{0.91}$ Modelo de Walston-Felix (5.2)

$E = 5.5 + 0.73 \times (KLDC)^{1.16}$ Modelo de Bailey-Basisli (5.3)

$E = 3.2 \times (KLDC)^{1.05}$ Modelo simple de Boehm (5.4)

$E = 5.288 \times (KLDC)^{1.047}$ Modelo Doty para KLDC >9 (5.5)

También se han propuesto los modelos orientados a PF. Entre estos modelos se incluyen:

$E = -13.39 + 0.054\ PF$ Modelo de Albretch y Gaffney (5.6)

$E = 60.62 \times 7.728 \times 10\ PF^{8}$ Modelo de Kemerer (5.7)

$E = 585.\ 7 + 15.12\ PF$ Modelo de Matson, Barnett y Mellichamp (5.8)

Una rápida exploración de los modelos listados arriba indica que cada uno traerá un resultado diferente para el mismo valor de LDC y PF. Los modelos de estimación se deben evaluar para necesidades particulares.

5.2 El Modelo COCOMO

Barry Boehm [Pressman ‘93], en su libro sobre “Economía de la Ingeniería del Software”, menciona una escala de modelos de estimación de software con el nombre de COCOMO, por COnstrucive COst MOdel (MOdelo COnstructivo de COsto). La escala de modelos de Boehm incluye:

- *Modelo* 1. El modelo COCOMO *básico* calcula el esfuerzo (y el costo) del desarrollo de software en función del tamaño del programa, expresado en las líneas estimadas de código (LDC).
- Modelo 2, El modelo COCOMO *intermedio* calcula el esfuerzo del desarrollo de software en función del tamaño del programa y de un conjunto de “conductores de costo” que incluyen la evaluación subjetiva del producto, del hardware, del personal y de los atributos del proyecto.
- Modelo 3, El modelo COCOMO *avanzado* incorpora todas las características de la versión intermedia y lleva a cabo una evaluación del impacto de los conductores de costo en cada fase (análisis, diseño, etc.) del transcurso de ingeniería del software.

Los modelos COCOMO están establecidos para tres prototipos de proyectos de software que empleando la terminología de Boehm son: (1) *modo orgánico*: aquellos proyectos de software que son respectivamente pequeños y sencillos en donde trabajan pequeños equipos que poseen buena experiencia en la aplicación, sobre un conjunto de requisitos poco rígidos; (2) *modo*

semiacoplado: son los proyectos de software intermedios hablando de tamaño y complejidad, en donde los equipos tienen diversos niveles de experiencia, y además deben satisfacer requerimientos poco o medio rígidos; (3) *modo empotrado*: son proyectos de software que deben ser desarrollados en un conjunto de hardware, software y restricciones operativas muy restringido.

Las ecuaciones del COCOMO básico tienen la siguiente forma: [Norman E. Fenton'91]

$$E = a_b \, KLDC^{b}_{b} \qquad (5.9)$$

$$D = C_b \, E^{d}_{b} \qquad (5.10)$$

donde *E* es el esfuerzo aplicado en personas-mes, *D* es el tiempo de desarrollo en meses cronológicos y KLDC es el número estimado de líneas de código distribuidas (en miles) para el proyecto. Los coeficientes a_b y C_b y los exponentes $^{d}_{b}$ y $^{b}_{b}$, con valores constantes se muestran en la Tabla 5.1 [Norman E. Fenton '91].

Modelo COCOMO básico

Proyecto de Software	a_b	$^{b}_{b}$	c_b	$^{d}_{b}$
Orgánico	2.4	1.05	2.5	0.38
Semiacoplado	3.0	1.12	2.5	0.35
Empotrado	3.6	1.20	2.5	0.32

Tabla 5.1 Valores Constantes [Norman E. Fenton '91].

La ecuación del modelo COCOMO intermedio toma la forma:

$$E = a_i KLDC^{bi} * FAE \qquad (5.11)$$

donde *E* es el esfuerzo aplicado en personas-mes y LDC es el número estimado de líneas de código distribuidas para el proyecto. El coeficiente a_i y el exponente b_i se muestran en la Tabla 5.2

Modelo COCOMO Intermedio

Proyecto de Software	**a_i**	**b_i**
Orgánico	3.2	1.05
Semiacoplado	3.0	1.12
Empotrado	3.8	1.20

Tabla 5.2 Valores Constantes [Norman E. Fenton '91].

COCOMO es el modelo empírico más completo para la estimación del software publicado hasta la fecha. Sin embargo, deben tenerse en cuenta los propios comentarios de Boehm [Pressman '98] sobre COCOMO (y por extensión, sobre todos los modelos):

Hoy en día un modelo de estimación de costos de software está bien fundado si puede evaluar tanto los costos de desarrollo de software en un 20 por ciento de los costos reales, así como un 70 por ciento del tiempo y ello en su propio terreno (o sea dentro de la clase de proyectos para los cuales ha sido calibrado), en realidad ésta no es la exactitud que aspiramos, pero es más que

suficiente para facilitar el análisis económico de la ingeniería del software y también en la toma de decisiones.

5.3 La ecuación del software

La *ecuación del software* [Norman E. Fenton'91] es un modelo multivariable que asume una distribución específica del esfuerzo a lo largo de la vida de un proyecto de desarrollo de software. El modelo se ha obtenido a partir de los datos de productividad para unos 4,000 proyectos actuales de software. Un modelo de estimación tiene esta forma:

$$E=[LDC * B^{0.333} / P]^3 * (1/t^4) \qquad (5.12)$$

donde E = esfuerzo en personas-mes o en personas-año

t = duración del proyecto ya sea en meses o años

B = "factor especial de destrezas, en donde incrementa a medida que crecen la necesidad de integración, pruebas, garantía de calidad, documentación y habilidad de administración" [Fenton'91]. Para programas pequeños (KLDC= 5 a 15), B = 0.16. Para programas mayores de 70 KLDC, B = 0.39.

P = "parámetro de productividad" que refleja:

- Madurez global del proceso y de las prácticas de administración
- La amplitud hasta donde se utilizan correctamente las normas e la ingeniería del software.
- El nivel de los lenguajes de programación utilizados

- Las habilidades y la experiencia del equipo del software.
- La complejidad de la aplicación.

Es importante señalar que la ecuación del software tiene dos parámetros dependientes: (1) una estimación del tamaño (en LDC) y (2) una indicación de la duración del proyecto en meses o años. Mediante la ecuaciones 5.12 tendrá con valor de P = 12.000 (valor recomendado para software científico).

Para simplificar el proceso de estimación y utilizar una forma más común para su modelo de estimación, Putnam y Myers sugieren un conjunto de ecuaciones obtenidas de la ecuación del software. Un tiempo mínimo de desarrollo se define como [Norman E. Fenton '91]:

$$t_{min} = 8{,}14\ (\text{LDC/PP})^{0.43} \text{ en meses para } t_{min} > 6 \text{ meses} \qquad (5.13)$$

$$E = 180Bt^3 \text{ en personas-mes para } E >= 20 \text{ personas-mes} \qquad (5.14)$$

Se tendrá en cuenta que t en la ecuación 5.14 se representa en años.

5.4 El modelo CMM (Modelo de Capacidad y Madurez)

El desarrollo del modelo CMM fue encargado por el departamento de defensa de los Estados Unidos, como una ramificación de los problemas experimentados durante la obtención del software., ya que ellos perseguían un medio para evaluación de los contratistas potenciales. Pero en el firme proceso de realizar progresos en los resultados de la calidad de software, adonde se ve envuelto el mejoramiento del proceso de desarrollo de este, ha llevado al Instituto

de Ingeniería de Software (Software Engineering Institute, SEI) en Carneige Mellom el desarrollo del Modelo de Capacidad y Madurez (Capability Maturity Model, CMM), que constituye de cinco niveles del desarrollo del software, organizando así al proceso de maduración:

Por medio de un amplio cuestionario, seguido de entrevistas y recolección de evidencias, EI software de la organización puede ser clasificado/ categorizado en uno de los cinco niveles de madurez principalmente en base a la rigurosidad de su proceso de desarrollo. Con excepción del nivel uno, cada nivel esta caracterizado por un conjunto de áreas de proceso clave (Kevs Process Areas, KPA' s). Por ejemplo las KPA's para el nivel dos son: *administración de requerimientos, planeación de proyectos, rastreo de proyectos, administración de subcontratistas, confianza en la calidad y administración de la configuración*. Las KPA' s para el nivel cinco son: prevención de errores, administración en el cambio de tecnología, administración en el proceso de cambio.

Idealmente las empresas suponen que se deben encontrar en el nivel tres al menos, para poder ganar grandes contratos. Esta importante motivación comercial es la razón de que el CMM esta contando con una alta aceptación. Pocas empresas tienen que administrarse para alcanzar el nivel tres; la mayoría están en nivel uno.

La siguiente información presenta un punto de vista de los tipos de medición sugeridos para cada nivel de madurez, donde la selección depende de la cantidad de información visible y disponible en el nivel de madurez.

- Las mediciones en el nivel uno dan una línea base de comparación en la búsqueda del mejoramiento del proceso y el producto.
- Las mediciones en el nivel dos están enfocadas en la administración del proyecto.
- Las mediciones en el nivel tres están enfocadas en el producto durante el desarrollo.
- Las mediciones en el nivel cuatro registran las características del mismo proceso de desarrollo para permitir un control individual de las actividades del proceso.
- En el nivel cinco el proceso es lo suficiente maduro y se administra cuidadosamente para que se permitan mediciones que nos retroalimentarán para cambios dinámicos del proceso durante el desarrollo de un proyecto en particular.

En los últimos años se ha hecho hincapié en la “madurez del proceso”, es por eso que el SEI ha desarrollado un modelo completo que se basa en un conjunto de funciones de ingeniería del software que deberían estar presentes conforme las organizaciones alcanzan diferentes niveles de madurez de software. Para determinar el estado actual de madurez del proceso, el SEI utiliza un cuestionario de evaluación y un esquema de cinco grados, en donde el esquema de grados determina la conformidad del *modelo de capacidad de madurez* [Addison Wesley ‘95] que definen las actividades clave que se requieren en los diferentes niveles de madurez del proceso.

El enfoque del SEI proporciona una medida de la efectividad global de las prácticas de ingeniería del software de una compañía y establece cinco niveles de madurez del proceso, que se definen de la forma siguiente: [Addison Wesley '95]

Nivel 1 : Inicial.- El proceso del software se caracteriza según el caso, y ocasionalmente incluso de forma caótica. Se definen pocos procesos, y el éxito depende del esfuerzo individual.

Nivel 2: Repetible.- Se establecen los procesos de administración del proyecto para hacer seguimiento del costo, de la planificación y de la funcionalidad. Para repetir éxitos anteriores en proyectos con aplicaciones similares se aplica la disciplina necesaria para el proceso.

Nivel 3: Definido.- El proceso del software de las actividades de administración y de ingeniería se documenta, se estandariza y se integra dentro de un proceso de software de toda una organización. Todos los proyectos utilizan una versión documentada y aprobada del proceso de la organización para el desarrollo y mantenimiento del software. En este el nivel se incluyen todas las características, definidas en el nivel 2

Nivel 4: Administrando.- Se recopilan medidas detalladas del proceso del software y de la calidad del producto. Mediante la utilización de medidas detalladas, se comprenden y se controlan cuantitativamente tanto el producto como el procero del software. El nivel 4 se incluye todas las características definidas para el nivel 3.

Nivel 5. Optimización.- Mediante un desarrollo cuantitativo del proceso y de las ideas y tecnologías innovadoras se posibilita una mejora del proceso. En el nivel se incluyen todas las características definidas en el nivel 4.

Los cinco niveles definidos por el SEI se obtienen como consecuencia de evaluar las respuestas de un cuestionario de evaluación basado en el CMM. Los resultados del cuestionario se filtran en un único grado numérico que proporciona una indicación de la madurez del proceso de una organización.

A pesar de la aceptación internacional, el CMM cuenta con criticas. Las críticas más serias son referentes a la validez de la escala de cinco niveles ya que no existe evidencia convincente de que las empresas que se encuentran en el nivel mas alto (nivel cinco) produzcan software de mejor calidad.

5.5 Modelo de productividad.

El modelo de productividad de Fenton [‘91] presentada en la figura 5.1 hace énfasis en el hecho de que la productividad es una función tanto del valor como de costo. En donde el valor incluye factores de calidad como de cantidad, los costos incluyen factores de recursos, complejidad y de personal.

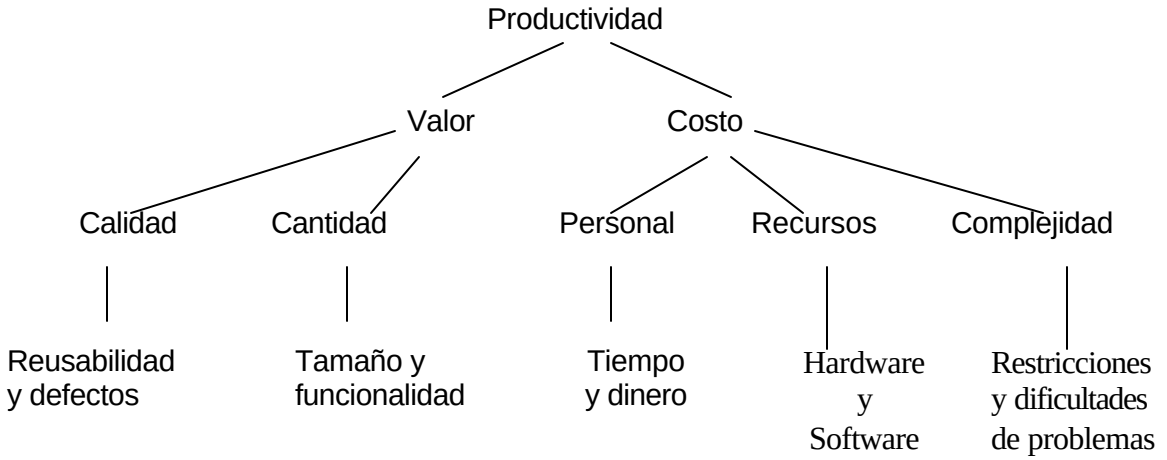

Figura 5.1 Modelo de Productividad [Fenton ´91]