DeepL
Suscríbete a DeepL Pro para poder traducir archivos de mayor tamaño.
Más información disponible en www.DeepL.com/pro.

Criterios de selección de modelos de procesos de software

Linda C. Alexanderl
Centro de Ingeniería del Software
Comunicaciones-Electrónica del Ejército de EE.UU. Comando
Fort Monmouth, NJ 07703

Alan M. Davis2
Centro de Ingeniería de Sistemas de Software
Universidad George Mason
Fairfax, VA 22030

Resumen

Con el fin de mejorar la calidad del software y su producción, los investigadores han definido muchas variaciones del proceso de desarrollo de software. Éstas van desde el tradicional proceso en cascada hasta el prototipo desechable. Este documento presenta directrices para seleccionar el modelo de proceso más apropiado para un proyecto concreto. Para facilitar la selección, hemos organizado los modelos de proceso de software en una jerarquía de tres niveles. Tras aplicar todos los criterios, se genera una clasificación relativa de los modelos de proceso para orientar la selección.

1 El problema de la selección del modelo de proceso

Son muchos los factores que influyen en el éxito de un proyecto de software. Entre ellos están la habilidad de los desarrolladores de software, la capacidad de los gestores del proyecto, la disponibilidad de herramientas CASE, el entorno informático, la dificultad del problema que se resuelve, los recursos disponibles y el modelo de proceso empleado. En esta última categoría, hay una gran variedad de modelos de proceso entre los que elegir [1].

La selección de un modelo de proceso inadecuado puede dan como resultado (1) un sistema que no satisface las necesidades del usuario, (2) mayor coste, y (3) plazos de desarrollo más largos. Por ejemplo, si las necesidades del usuario no se comprenden bien o son cambiantes, el uso de un modelo de proceso convencional podría dar como resultado un sistema que no satisfaga las necesidades del usuario. Por el contrario, si las necesidades del usuario se conocen bien y son estables, la construcción de un prototipo puede aumentar innecesariamente el coste y el calendario de desarrollo [8].

Los directores de proyecto actuales eligen un modelo de proceso de software utilizando criterios ad hoc, injustificados y, a menudo, no documentados. El modelo en espiral de Boehm reconoce la necesidad de seleccionar el modelo de proceso y basa la selección en un único criterio: el riesgo [7]. Otros autores también han identificado esta necesidad [4]. No fue

Esta investigación se llevó a cabo mientras asistía a la Universidad George Maeon.

Hasta 1990 no se intentó elaborar un conjunto completo de criterios y valores para este tipo de selecciones [2]. En este documento se describen 20 criterios y un método para aplicarlos. Con este método, el gestor de proyectos evalúa un proyecto con respecto a los criterios para elaborar una lista priorizada de modelos de procesos. Utilizaremos las siguientes notaciones:

C es el conjunto de veinte criterios con elementos c_i, = 1..20 (p.ej. *-1 es* experieztce *del usuario,* c,g es ***Jequen'y*** *de cambios) -,*

Yj es el vector ordenado de elementos $v_i a$ de valores que puede cada criterio c_i (por ejemplo, *UI, V JQ* y *-13* **tienen** valores *novato,* ***experimentado*** y *experto,* respectivamente; mientras que vg_i *-82* --d *-83* ave valores *rara vez, lento* y *rápido);*

s jJ es un indicador binario de qué valor del criterio r_i se aplica a este proyecto en particular (por ejemplo, *s j j, S]2 513, 581 5(82)* --d *583* son 0, 0, 1, 1, 0 y 0 respectivamente). *Los s;J-'s* forman un proyecto matriz de características que refleje los atributos de un proyecto; y

$a_i d$ representa la aplicabilidad de un modelo de proceso a una *v* (*por ejemplo, -13* -- 1 para el modelo de proceso convencional indica que el proceso es apropiado para proyectos con usuarios expertos). *Los a;J'z* de cada modelo de proceso forman una matriz que se utiliza para determinar la calificación de un modelo de proceso con respecto a un proyecto concreto.

Este documento proporciona el conjunto de criterios, *C,* su valores posibles, H_i*'s,* y una matriz de modelo de proceso para cada modelo de proceso que contiene los nJJ-'s. ***Los s;J-'z*** son dependen del proyecto y las determina el director del proyecto.

Para determinar el modelo de proceso adecuado para un proyecto, el director de proyecto sigue un método de tres pasos.

1. Examinar los atributos del proyecto y determinar s_iJ para cada criterio que mejor describa el proyecto.

Esta investigación ha sido financiada en parte por el Mando de Comunicaciones y Electrónica del Ejército de Estados Unidos y COMPASS Corporation mediante el contrato nº DAAB07-88-B029, y por el Centro de Tecnología Innovadora de la Mancomunidad de Virginia mediante el contrato JSTC-87-0£i2.

0730/3157/91/£Q¢KI/0521/$01.00 O 1991 IEEE

2. Para cada modelo de proceso **compute:**

$$VALORACIÓN = \sum_{i=1}^{20} \sum_{y=1}^{3} (s_{ij} \circ_{iJ})$$

3. El modelo de proceso con *el RATING* ***más alto*** es la mejor opción, el segundo más alto es la segunda mejor , **etc.**

En general, los modelos de procesos de desarrollo de software definen la partición, el orden y las relaciones temporales que existen entre las actividades de desarrollo de software. Sin embargo, no todos los modelos de procesos los abordan al mismo nivel de abstracción. Así, algunos modelos de proceso de software delinean amplios grupos de actividades a lo largo del ciclo de vida del software, mientras que otros son detallados hasta el punto de especificar los métodos y herramientas particulares utilizados para realizar las actividades prescritas. El método descrito anteriormente debe aplicarse a cada nivel de abstracción para seleccionar el modelo de proceso adecuado a ese nivel.

En la sección 2 de este documento se describen los criterios de selección de procesos (C) y los intervalos de valores de estos criterios (P_i's). En la sección 3 se describen algunos modelos de procesos utilizados en la actualidad y se organizan en una hiemrquía. La sección 4 establece la correspondencia entre cada valor de criterio y cada modelo de proceso *{a¿q-'s).* La sección 5 ofrece un breve ejemplo. La sección 6 resume nuestras conclusiones.

2 Criterios iniciales del proyecto

Los criterios del proyecto (C) se en las siguientes categorías geneml: personal, problema, producto, recursos y organización. El cuadro 1 muestra la gama de valores *(H_i)* para cada uno de los criterios (cj).

2.1 Criterios de personal

Los criterios de personal afectan tanto a los desarrolladores como a los usuarios. A continuación se describen los criterios de personal y sus valores.

- Experiencia de los usuarios en el dominio de la aplicación (cy): Este criterio evalúa el conocimiento de los usuarios sobre el dominio del problema. Los valores son: *V j -- (novato, experimentado, experto).* Cada extremo de esta escala puede actuar tanto a favor como en contra de un esfuerzo. Es posible que los usuarios novatos no sepan mucho sobre el área del problema, pero aceptarán mejor las soluciones nuevas o diferentes. Los usuarios expertos saben mucho sobre el problema, pero se resisten a los cambios.
- Capacidad de los usuarios para expresar sus necesidades (c2): Este criterio evalúa en qué medida los usuarios pueden comunicar sus necesidades. Los valores son: **2 - (silencioso, comunicativo, expresivo).* En el extremo silencioso del escala son usuarios que no saben decir lo que necesitan, pero se reconocerlo cuando (7) ven. Los usuarios expresivos son capaces de decir exactamente lo que creen que quieren.
- Experiencia de los desarrolladores en el dominio de aplicación (c3): Este criterio evalúa los conocimientos de los desarrolladores sobre el dominio del problema. El conocimiento de los desarrolladores puede ser el resultado de ser usuario en el dominio de la aplicación, o de desarrollar otras aplicaciones en el dominio. Los valores son: V_3 *-- (novato, experimentado, expen).*
- Experiencia en ingeniería de software de los desarrolladores (por ejemplo): Este criterio evalúa la experiencia los desarrolladores y su conocimiento de los tmls de software, métodos, técnicas y lenguajes necesarios para un esfuerzo de desarrollo. Los valores son: *Vq -- {no!'**. R !'*^* - experto).*

2.2 Criterios del problema

Los criterios del problema se refieren al problema que se pretende resolver con el desarrollo del software. A continuación se describen los criterios del problema y sus valores.

- Madurez de la aplicación (cy): Este criterio evalúa el grado de conocimiento general del problema. El desarrollo de software en áreas de aplicación maduras puede beneficiarse de los esfuerzos de desarrollo previos, mientras que hay menos conocimientos previos en los dominios de aplicación más nuevos. Los valores son: *V -- (nuevo, estándar, bien establecido).* Un ejemplo de aplicación consolidada es el cálculo de nóminas. Gran parte del software de inteligencia artificial representa una nueva aplicación.
- Complejidad del problema (-6): '''s criterio mide la complejidad del problema a . Los valores son: V_d *-- (simple, d Cllr, C0 R!)* Los grandes sistemas paralelos en tiempo real tienden a ser complejos, mientras que los más pequeños, Los sistemas secuenciales suelen ser sencillos.
- Requisito de funcionalidad parcial (-7): - El criterio mide la viabilidad y/o la necesidad de ofrecen **productos intermedios que** sólo proporcionan una **parte de la** funcionalidad completa del producto **final**. Los valores son: V_7 *-- **(urgente, deseable,** no **deseable).*** El sistema de soporte vital a bordo de un transbordador espacial es un ejemplo en el que no es deseable una funcionalidad parcial. **La automatización** de funciones ofimáticas es un ejemplo en el que sería deseable **una entrega temprana** de las funciones de tratamiento de textos.
- Frecuencia de los cambios *(c_8).-* Este criterio **mide** la frecuencia con la que cambia el problema. Los valores son: *"8 -- (rápido, lento, rara vez). Un':n* ejemplo que cambia rápidamente es un sistema de armamento porque debe responder a un entorno de amenazas que cambia rápidamente. Un compilador para un lenguaje estándar es un ejemplo de un problema que cambia más lentamente.

Cuadro 1. Criterios de selección y valores Criterios y valores de selección

c_i	CRITERIOS	v_{i1}	*.i2	*i3
c_1	Experiencia del usuario	Novato	Experimentado	Experto
c_2	Expresión del usuario	Silencioso	Comunicativo	Expresivo
cy	Experiencia del	Novato	Experimentado	Experto
c_4	desarrollador en	Novato	Experimentado	Experto
c_5	aplicaciones Experiencia	Nuevo	Estándar Difícil	Establecer
'd	del desarrollador en Sw	Simple No	Deslimitado	Complejo
c_7	Madurez de la aplicación	Deseado	Lento	Urgente
c_8	Complejidad del	Rara vez	Modemte	Rápido
'9	problema Funcionalidad	Menor	Medio	Extremo
cj€I	parcial Frecuencia de los	Pequeña	Difícil	Grande
c_{11}	cambios Magnitud de los	Simple	Moderado	Complejo
c J y	cambios Tamaño del	Flexible	Significativ	Exigente
C13	producto	Menor	o Nivel	Crítico
c)4	Complejidad del	Baja-alta	Adecuado	Alto-bajo
c_{15}	producto Requisitos de	Escasa	Nivel	Amplio
Cld	"ilegalidad" Requisitos	Baja-alta	Adecuado	Alto-bajo
"J 7	de interfaz Perfil de los	Escasa	Limitado	Amplio
'18	fondos	Ninguna	Flexible	Libre
*19	Disponibilidad de	Directriz	**Intermedio**	Aplicado
c20	fondos Perfil del	Básica		Avanzado
	personal			
	Disponibilidad del			
	personal Acceso a los			
	usuarios Compatibilidad			
	de la gestión			
	Compatibilidad QA/CM			

- Magnitud de los cambios *(c9):* Este criterio evalúa la adaptabilidad y la magnitud relativa de los cambios previstos en el . requisitos de mantenibilidad.

Los valores son: **9- (extremo, moderado, menor).* Un nuevo comunicación de criterios puede dar lugar a un pequeño cambio a un sistema telefónico. La puesta en marcha de un nuevo sensor puede requerir un rediseño completo de un sistema de defensa.

- Requisitos de la interfaz humana -*13)* (E)ste protocolo de mide la criticidad de la interfaz humana. Valores son: *3- (crítico, significativo, menor).*

2.3 Criterios del producto

Los criterios de producto se refieren al desarrollo evalúa el producto que se va a desarrollar. Los criterios de producto y sus escala, los valores de financiación y personal se analizan a continuación.

- Tamaño del producto *(c j):* Este criterio mide el tamaño esperado del producto final. Dado que este La medición se realiza al inicio del de desarrollo, es sólo una estimación. Utilizando Las definiciones de Fairley [11] sobre el tamaño de los productos (mostradas en representan los paréntesis), los valores son: *V j -- (grande* (muy grande *pequeños* (triviales y pequeños)).
- Complejidad del producto *(c j j):* Este criterio mide la Bajo muestra que la mayoría de los fondos son *V j j -- (complicado, difícil, sencillo).* Perfil (c)
- Requisitos de "ilidad" (cy 2): Las "ilidades" representan el requisitos no comportamentales impuestos a un como fiabilidad, adaptabilidad, demostrabilidad, reusabilidad, y mantenibilidad. Este criterio de los valores mide la carga relativa de tales Requisitos. Los valores son: *V j y= (exigente, moderado, flexible).* Software cuyo fallo podría poner en peligro la vida un ejemplo con fiabilidad y probabilidad exigentes que para la financiación. Como requisitos. Software que se espera que esté en uso

2.4 Criterios de recursos

Los criterios relativos a los recursos se refieren a los recursos disponibles para desarrollo. A diferencia de otros criterios, en los que se los criterios de producto y su valor en una Los criterios de producto se evalúan por el perfil del recurso sobre tiempo. Los criterios de recursos y su evaluación son el se discute más adelante.

- Perfil de financiación *(c j4):* Este criterio mide el esfuerzo cantidad y disponibilidad de fondos para el desarrollo esfuerzo. Los gráficos de la Figura 1 y extremadamente grande), *medio* (medio y), perfiles. El perfil (a) Bajo-Alto muestra la cantidad de fondos aumenta al principio del esfuerzo, con complejidad del software a . Valores Perfil (b) Alto- disponibles al principio del esfuerzo. nivel indica un nivel constante de financiación sobre software. Así, *V jq -- (Bajo-Alto, Alto-Bajo, Nivel).* producto,
- Disponibilidad de fondos *{cj f;).'* Este criterio mide la usabilidad adecuación de los fondos disponibles para un esfuerzo. El son: *V j$ -- (escasos, adecuados, amplios).*
- Perfil del personal *(c j d)* Este criterio evalúa la de las personas disponibles a lo largo del tiempo para el . es Los perfiles para la dotación de personal son los mismos

52.3

que se muestra en la Figura 1, *V d -- (Bajo-Alto, Alto-Bajo, **Level**).*

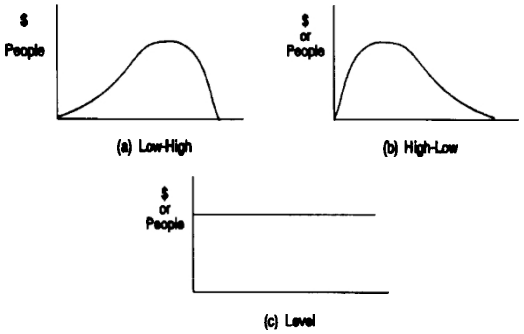

Figura 1 Perfiles de financiación y personal

- Disponibilidad de personal *(-i 7):* Este criterio mide la suficiencia del personal disponible para un proyecto. Los valores son: **i 7- (escaso, adecuado, amplio).*
- Accesibilidad de los usuarios (cJ,g): Este criterio mide la cantidad acceso que los desarrolladores tienen a los usuarios. Los valores son: *V j g -- (sin acceso, acceso limitado, acceso libre).*

2.5 Criterios de organización

Los criterios organizativos se refieren a cómo la organización

políticas influyen en un esfuerzo de desarrollo. La organización se discuten a continuación.

- Compatibilidad de gestión *-i9)"* Este criterio **segmentos** de **mide** el grado en que un proceso de software Estos modelos son compatibles con **el modelo** de partición de **las necesidades de** los usuarios**.** Este desarrollo requisitos. valores son: *V j9* - convencionales, *(de aplicación estricta, flexible, sólo orientativa).*
- Garantía de calidad y gestión de la configuración Capacidad *(cy):* Este criterio **evalúa** el compatibilizar un modelo de **proceso** concreto y de calidad de la organización y la **configuración** procedimientos de gestión. Los valores son: *Vy -- (baSiC, avanzado).*

3 Jerarquía de modelos de procesos

La variedad de modelos de proceso que se utilizan hoy en día es

El nivel de abstracción al que cada uno trata el proceso de software es una variante significativa. El nivel superior de la jerarquía contiene modelos de procesos **que delimitan** amplios grupos de actividades. El siguiente nivel

contiene modelos de procesos que afinan aún más la agrupación y las relaciones entre las actividades prescritas. En el siguiente nivel se encuentran los modelos de procesos que especifican herramientas y métodos concretos para realizar las actividades. La figura 2 muestra un ejemplo de esta jerarquía. En esta figura, el nivel superior (nivel 3) muestra un modelo de proceso concreto:

determinar el problema, el desarrollo y las opemciones y el mantenimiento, realizados secuencialmente. El nivel 2 muestra un proceso concreto para llevar a cabo el desarrollo. El nivel 1 muestra un conjunto particular de herramientas y técnicas para realizar las actividades de requisitos del nivel 2. En las secciones siguientes se describen algunos de los modelos de proceso de los niveles 3 y 2. Los métodos y herramientas del nivel 1 quedan fuera del alcance de este . La selección de métodos y tmls de nivel 1 para requisitos se ha discutido en [9].

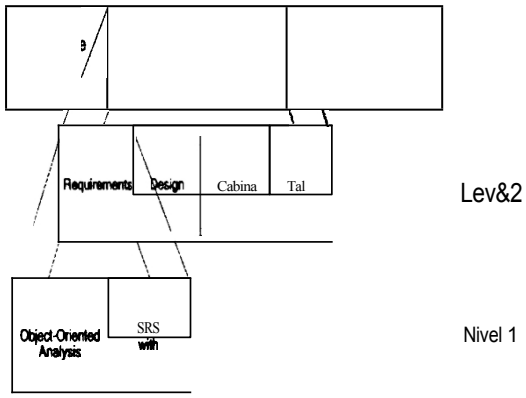

Figura 2. Jerarquía de modelos de procesos de software
Jerarquía de los modelos de procesos de software

3.1 Modelos de procesos de nivel 3

bien 3 modelos de proceso asignan etiquetas **a tiempo para** caracterizar las actividades que tienen lugar. **Los** modelos de proceso también **abordan cualquier**

trabajo se considerarán tres ejemplos:

incremental y evolutiva.

En el modelo de proceso de software convencional, la problema no está subdividido. Se desarrolla un producto para "el en su totalidad. El ciclo de vida del software es la garantía dividido **en tres segmentos: determinar el** problema a resolver, desarrollar el producto, y opemte e *infemiediate,* mantener. Véase la figura 3.

Figura 3. Modelo convencional de proceso de software

En el modelo de proceso de software *incremental* existe una decisión consciente de retrasar el desarrollo de una parte del producto de software objetivo [5]. El problema en su totalidad se subconjunta antes de desarrollar cualquier producto intermedio. El producto de cada desarrollo provisional se utilizará en el entorno operativo, y más probable es que no se le aplique ningún esfuerzo de mantenimiento. La figura 4 muestra el modelo de proceso de software incremental.

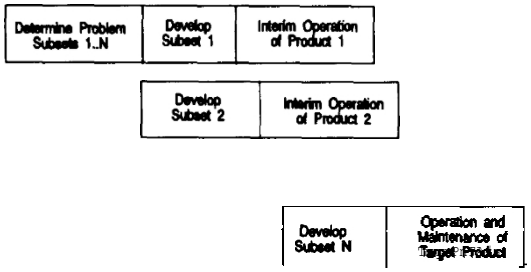

Figura 4. Modelo de proceso de software incremental

El modelo de proceso de software *evolutivo* es similar al incremental en el sentido de que existe una decisión consciente de retrasar el desarrollo de una parte del producto de software objetivo [12]. Sin embargo, a diferencia del desarrollo incremental, los productos intermedios se desarrollan antes de determinar el siguiente subconjunto. Véase la Figura 5.

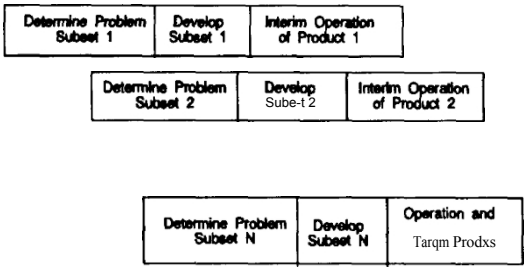

Figura S. Modelo de proceso de software evolutivo

3.2 Modelos de procesos de nivel 2

Los modelos de proceso de nivel 2 especifican las actividades que deben realizarse en cada uno de los segmentos temporales de un modelo de proceso de nivel 3. Estos modelos también abordan las distintas representaciones del producto que se construirá. Este trabajo considerará tres ejemplos: cascada, prototipado híbrido y especificación opcional.

El modelo de *cascada* **fue** introducido **por primera** vez por Royce en 1970 [15]. La figura 6 muestra un software típico en cascada
modelo de proceso [61

En *el prototipado híbrido,* los prototipos se construyen inicialmente. En **fases** posteriores, **partes** de los prototipos pueden descartarse selectivamente, mientras que otras partes (por ejemplo, objetos, código reutilizable) se incorporan al producto (Luqi utiliza el término prototipado rápido, pero también puede referirse a la construcción rápida de prototipos **desechables** [14]). **No debe** confundirse con el prototipado **de usar y tirar.**
[13] que suele utilizarse método en un modelo de proceso de nivel 1 para realizar algunas de las actividades prescritas. La figura 7 muestra un modelo de proceso de software de nivel 2.

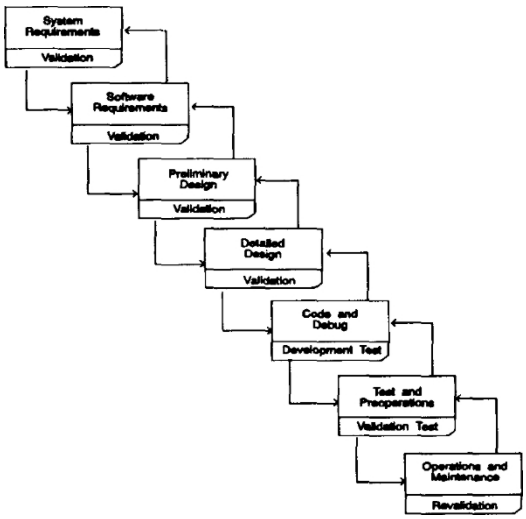

Figura 6. El modelo de proceso de software en cascada @ 1976 IEEE

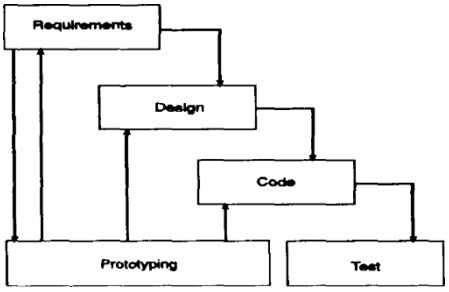

Figura 7. Modelo de proceso de creación de prototipos híbridos

La especificación operativa proporciona una representación ejecutable del producto directamente a partir de las especificaciones. Véase la figura 8 [1]. La primera fase describe el comportamiento externo de un sistema de soluciones. Las fases posteriores lo transforman en un sistema operativo más robusto.

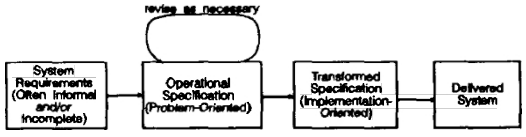

Figura 8. Modelo de proceso de especificación operativa 1986 IEEE

3.3 Resumen

Los modelos de procesos presentados anteriormente no son los únicos que existen. Por ejemplo, los modelos

prototyping [10] y concurrent software modeling [16] ofrecen modelos alternativos de nivel 3. El modelo de proceso transformacional [3] es un modelo alternativo de nivel 2.

4 Tablas de selección de modelos de proceso

Esta sección contiene cuadros que indican la adecuación de a un proceso modelo a proyectos valores particulares $v_i J$- para los criterios c_i. Estos se desarrollaron teniendo en cuenta las características de los modelos de procesos y las características necesarias en situaciones concretas de proyectos. Las principales características consideradas fueron:

1) la oportunidad de modificar, validar y/o verificar el software;
2) el grado de funcionalidad suministrado y cuándo; y
3) el nivel de actividad con respecto al tiempo.

Una entrada de 1 en una tabla indica que el modelo de proceso es adecuado para los proyectos que presentan ese valor de un criterio. Una entrada de 0 indica que el modelo de proceso es inadecuado. Así, la tabla (?) muestra que el modelo de proceso convencional es adecuado para proyectos con usuarios *expertos*, usuarios *expresivos*, aplicaciones *establecidas*, etc. Las tablas (?) a (?) muestran la misma información para los demás modelos de proceso.

5 Ejemplo

Esta sección contiene un ejemplo de utilización de las matrices de modelos de procesos. El ejemplo se desarrolló a partir de un proyecto real, el Terminal Prototipo de Vigilancia Oceánica (POST) [10], con las características del proyecto determinadas por un miembro del equipo de desarrollo.

Paso 1: Examinar el proyecto y determinar "e $_{ij}$'s para cada criterio ci_ El objetivo de este proyecto era colocar estaciones de trabajo de sobremesa a bordo de buques para analizar información de inteligencia estratégica. La aplicación era nuevo tanto para los usuarios como para los desarrolladores e implicaba una problema complejo y en rápida evolución. Era necesario entregar rápidamente un producto con cualquier capacidad. La interfaz de usuario representaba una parte crítica de los requisitos. El cuadro 8 recoge las características de este proyecto especificando los f_iJ-.

Paso 2: Comput9 Jjie rating para cada modelo de proceso utilizando la fórmula:

$$VALORACIÓN = \sum_{i=1}^{20} \sum_{j=1}^{3} (s_{ij} * a_{ij})$$

La tabla 9 muestra los resultados de realizar este paso para el modelo de proceso convencional. Cada entrada de la tabla es el resultado de la multiplicación *{s;J- * a_id).* La dirección números de la columna de la derecha son los resultados de la suma sobre y. El total en la parte inferior de la columna de la derecha es el resultado de la suma sobre *i.* Para este proyecto, el modelo de proceso convencional recibió una puntuación de 8. Los resultados para todos modelos de proceso se muestran en la tabla 10.

Tabla 2. Selección convencional Matriz

@£	V	V	Vi3
c j	0	0	1
c2	0	0	1
Cy	0	0	1
eg	1	i	i
-5	0	0	1
*d	1	0	0
*7	1	0	0
*8	1	0	0
C_9	1	0	0
*10		0	0
c j j	1	0	0
cy2	1	1	1
*13	1	0	0
c j,f	1	0	0
c15		'	
*16		o	0
*17	i	0	I
*18	0	1	1
*19	0		
cy	'		
	1	1	1

Tabla 3. Matriz de selección incremental

Criterios	*i1	*i2	*i3
c j	0	1	1
c_2	0	1	1
*3	0	1	1
ct	0	1	1
cy	0	1	1
*d		1	0
*7	0	1	0
*8	1	1	0
C_9	1	1	0
'10	1	1	1
*11	1	'1	
*12	1	1	
cly	1	1	0
cj,f	0	1	0
llor	0	1	0
ar	0	1	1
Cld	0	1	0
C17	0	1	1
*18		1	1
C19	0		0
cy			1

Tabla 4. Selección evolutiva Matriz

Criterios	vil	*i2	*i3
c j	1	1	1
c2	1	1	1
C_3			
*4	0	1	1
c5	1	1	1
cg	1	1	1
*7	0	1	1
C_8	1	1	1
*9	1	1	1
*10			
*11			0
c j2	1	0	0
*13	1	1	1
*14	o		o
c15		i	i
$c j_c$	0	1	0
*17		1	1
C18	0	0	1
C19	1	1	0
cy	0	1	1

Cuadro S. Matriz de selección en cascada

Criteri® *i1		*i2	*i3
c j	0	0	1
c2	0	0	1
cy	0	1	1
c,f	1	1	1
c$	0	0	1
Cd	1	0	0
ct	0	0	0
C8	1	0	0
'9	0	0	0
cj	0	0	0
'11	1	o	o
cjy	0	0	0
*13	1	0	0
°21	1	o	o
*15	0	0	0
c j g	1	0	0
C17	0	0	0
"18		1	1
*19		¡!	¡!
C20	¡ !1	1	1

Tabla 6. Prototipos híbridos

Matriz de selección

Criteria	v_{i1}	v_{i2}	v_{i3}
c_1	1	1	1
c2	1	1	1
ct	1	1	1
cq	0	1	1
cff	1	1	1
ct	1	1	1
C7	0	0	0
*8	1	1	0
*9	0	0	0
cj	0	0	0
*11			
cjy	¡!	¡!	
C13	0	0	0
cjq	1	1	1
cjs	0	1	0
*ld	0	0	0
cU	0	1	0
cjg	0	0	0
C19	0	0	1
*20	1	1	0
		1	1

Tabla 7. Matriz de selección de especificaciones operativas

Criteria	v_{i1}	v_{i2}	v_{i3}
c_1	0	1	1
C	0	1	1
C3	0	1	1
€q	0	1	1
Cs	0	1	1
d		1	0
C7	0	0	0
C8	1	1	0
9	0	0	0
Cyb	0	0	0
11	1	0	0
Cjy	0	0	0
C]3	1	1	0
Cyb	0	1	0
C]s	0	0	0
16	0	1	0
17	0	0	0
18	o	o	'
C]9	1	0	0
20	0	1	1

Tabla 8. Características del proyecto ejemplo

Criterios	*i1	*i2	*i3
c_1	1	0	0
c_2	1	0	0
C3	!		
cq	0	0	1
c5	1	0	0
ct	0	0	1
^7			¡!
C8			!
C9		!	
cJ	1	0	0
cjf	0	1	0
c12			'
*13			¡!
cyq	1	0	O
c15			
*10			
*J7		¡!	
cjff	0	0	1
C19			
cy	1	0	0
	1	0	0

Tabla 9. Cálculo de la clasificación para el modelo de proceso convencional

Criteria	$(s_{i1} * a_{i1})$	$(s_{i2} * a_{i2})$	$(s_{i3} * a_{i3})$		
c)	0	0	0	=>	0
c2	0	0	0	=>	0
'3	0	0	0	=>	0
C4	0	0	1	=>	1
's	0	0	0	=>	0
ci	0	0	0	=>	0
'7	0	0	0	=>	0
^8	*'	0	0	=>	0
c9	0	0	0	=>	0
c)	1	0	0	=6	1
cij	0	0	0	=>	0
'J2	0	0	¡!	""	¡!
C]3	0	0	0	=>	0
^14					
c $j	1	0	0	=>	1
*Id	0	0	0	=>	0
*17		0	0	=>	1
*18	0	0	0	=>	0
	0	0	1	=T	1
*19	1	0	0	"	'
c20	1				8

Tabla 10. Calificaciones del modelo de proceso para el proyecto de ejemplo

Modelo de proceso	Clasificación
Nivel 3	
Convencional	8
Incremental	8
Evolución	16
Nivel 2	
Cascada	6
Prototipos híbridos	10
Especificaciones opcionales	3

Paso 3: Seleccionar el modelo de proceso con la punta más alta Para los modelos de proceso de nivel 3, las puntuaciones indican que el modelo de proceso evolutivo sería la opción más adecuada para este proyecto. Para los modelos de proceso de nivel 2, el modelo de proceso de prototipado híbrido es la mejor selección en este nivel de abstmción.

6 Conclusiones

Los criterios de selección determinados en este trabajo no son **en** absoluto definitivos. Su definición es el resultado de una investigación bibliográfica y de la observación informal de muchas iniciativas de desarrollo de software. Las correlaciones válidas entre las características del proyecto y el proceso alternativo
modelos sólo puede hacerse **tras** realizar la correlación análisis entre las características, el uso de varios modelos de proceso y los éxitos y fracasos de un gran número de proyectos.

Es necesario seguir estudiando el significado real de la puntuación para un modelo de proceso. Puede resultar beneficioso calcular puntuaciones **"negativas**" que indiquen la penalización por ignorar determinados criterios o valores. Este podría ser útil para desempatar o para evaluar la coste de no utilizar el modelo de proceso recomendado. Los criterios con una influencia "negativa" significativa también podrían **incitar a los planes a** evitar problemas cuando no se utilice el modelo de proceso recomendado o a adaptar un modelo de proceso para que se ajuste a todas las evaluaciones.

Referencias

[1] Agresti, W., "¿Qué son los nuevos paradigmas?", Tutorial: *New* Paradigms for Software Development, W. Agresti, ed., Washington, D.C.: IEEE Computer Society Press, 1986, pp. 6-10.

[2] Alexander, L., *Selection Criteria for Alternate Software Life Cycle Process Models,* Software Systems Engineering M.S. Thesis, Fairfax, Virginia: Universidad George Mason, 1990.

[3] Balzer, R., T. Cheatham, Jr. y C. Green, "Software Technology in the 1990's: Using a New Paradigm", *IEEE Computer,* 6, 11 (noviembre de **1983),** pp. **39-45.**

[4] Basili, V. y H. Rombach, "Tailoring the Software Process to Project Goals and Environments", *Proceedings Ninth International* ***Conference on Software Engineering,*** Washington, D.C.: IEEE Computer Society Press, 1987.

[5] Bersoff, E., B. Gregor y A. Davis, 'A New Look at the C3I Software Life Cycle', *SIGNAL,* 41, 8 (abril de 1987), pp. 85-93.

[6] Boehm, B., 'Software Engineering', *IEEE Transactions on Computers,* 25, 12 (diciembre de 1976), pp. 1226-1241.

[7] Boehm, B., "A Spiral Model Of Software Development and Enhancement", *IEEE Computer,* 21, *5* (mayo de 1988), pp. 61-72.

[8] Davis A., E. Bersoff y E. Comer, "A Strategy for Comparing Alternative Software Development Life Cycle Models", *IEEE Transactions on Software Engineering,* 14, 10 (octubre de 1988).

[9] Davis Alan M., Requisitos del software: Analys3 and Specification, Englewood Cliffs, Nueva Jersey, Prentice Hall, 1990.

[10] Davis, A., "Operational Prototyping: The POST Story", presentado a *IEEE Software,* enero de 1991.

[11] Fairley, R., Software Eneineerinz Concepts, Nueva York, Nueva York: McGmw-Hill, Inc., 1985.

[12] Gilb, T. , 'Evolutionary Delivery versus the Waterfall Model", AC3f *Software Engineering Notes,* 10, 3 (julio de 1985), pp. 49-61.

[13] Gomaa, H., "Impact of Rapid Prototyping on Specifying User Requirements", *ACM SIGSOFT Software Engineering Notes,* 8, 2 (abril de 1983).

[14] Luqi, "Software Evolution Through Rapid Prototyping", *Computer,* 22, 5 (mayo de 1989).

[15] Royce, W., "Managing the Development of Lerge Software Systems: **Concepts and Techniques",** *Proceedings, WESCON,* agosto de 1970, reimpreso *en: Proceedings Ninth International* ***Conference on*** *Software Engineering,* Washington, D.C.: IEEE Computer Society Press, 1987, pp. 328-338.

[16] Sitaram, P., *Concurrent Modeling of Software,* Software Systems Engineering M.S. Thesis, Fairfax, Virginia: George Mason University, 1990.