Capítulo 10

CALIDAD DE COMPONENTES SOFTWARE

Juan Pablo Carvallo
Xavier Franch
Carmen Quer

10.1 INTRODUCCIÓN

En los últimos años se constata una tendencia creciente por parte de las organizaciones a desarrollar sus sistemas software mediante la combinación de componentes, en lugar de desarrollar dichos sistemas partiendo de cero. Esta tendencia es debida a varios factores. Entre ellos cabe destacar: la necesidad de las organizaciones de reducir los costes y el tiempo dedicados al desarrollo de los sistemas software; el crecimiento del mercado de componentes software; la reducción de la distancia entre clientes y proveedores de software gracias a la creación de nuevos canales de comunicación y marketing (p.e., www.componentsource.com); y la existencia de tecnologías que facilitan el desarrollo de sistemas basados en componentes.

El ciclo de vida del desarrollo de sistemas basados en componentes (abreviadamente, DSBC, Szyperski, 2002) incluye, entre otras, las siguientes etapas:

- Análisis: Exploración y evaluación de componentes disponibles en el mercado; estudio y especificación de la organización y de sus requisitos;

subsiguiente especificación de los modelos de proceso de negocio adecuados.

- Selección: Elección de una arquitectura de componentes que satisfaga los requisitos, de acuerdo con métodos y criterios adecuados y según la disponibilidad de componentes en el mercado y servicios de los proveedores.

- Contratación: Redacción del contrato formal según los requisitos establecidos sobre los componentes seleccionados, los resultados del análisis, su evaluación, y las condiciones de implantación.

- Implantación de los componentes: Ajuste e integración con otros componentes implantados en el sistema software bajo desarrollo.

El detalle de estas actividades puede depender de diversos factores. Uno de ellos es el tipo de componente con el que se trabaja: componentes comerciales o COTS, por su denominación inglesa "Commercial Off-The-Shelf" (Carney y Long, 2000); código abierto o FOSS, por "Free and Open Source Software" (Madanmohan y Rahul, 2004); componentes desarrollados a medida; servicios web (Papazoglou, 2008); etc.

Como cualquier otro paradigma de desarrollo de software, el estudio de la calidad de los componentes software juega un papel muy importante en el DSBC. Por ejemplo, en los procesos de selección de tales componentes es necesario conocer con detalle el comportamiento relativo a aquellos criterios que se corresponden con los requisitos del sistema en desarrollo, tanto funcionales como no-funcionales (p.e., respecto eficiencia, usabilidad, etc.). Necesidades similares aparecen en otras actividades como en el momento de integrar los componentes, de mantener el sistema, etc.

El estudio de la calidad de los componentes software presenta algunas diferencias respecto el estudio de la calidad de los sistemas software en general. Los factores que generan estas diferencias son:

- Atomicidad: los componentes son unidades indivisibles desde el punto de vista de su gestión. Consecuentemente, podemos estudiar su calidad individualmente.

- Reusabilidad: los componentes son módulos que se reusan e integran[3], en una o más aplicaciones. Ello exige un alto grado de precisión en la descripción de la calidad, especialmente en el caso de componentes Off-The-Shelf u OTS (término con el que se acostumbra a englobar los componentes COTS y FOSS, Li et al., 2008) y servicios web.

- Evolución: los componentes que integran la aplicación transitan por sucesivas versiones que no se corresponden necesariamente con las versiones de los sistemas en los que se integran, especialmente en el caso de los mencionados componentes OTS. La descripción de la calidad de los componentes debe facilitar el estudio del impacto de tales evoluciones.

Como consecuencia, el estudio de la calidad de los componentes software exige la existencia de un vocabulario exhaustivo que defina con precisión los diferentes factores que pueden influir en dicha calidad. Este vocabulario debe estar acompañado de indicaciones sobre los valores que dichos factores pueden tomar en los componentes, y cómo se pueden obtener los mismos.

Como respuesta a esta necesidad, diferentes autores y organizaciones (p.e., consultoras y fabricantes) promovieron catálogos de factores de calidad. Inicialmente, estos catálogos se configuraron en forma de una lista desestructurada, cuyos factores no estaban definidos con suficiente precisión (en particular sin indicaciones claras sobre los valores que podían tomar). Rápidamente, estos catálogos evolucionaron y finalmente tomaron la forma de modelos de calidad, entidades que solucionan los problemas mencionados. Puede decirse que actualmente, los modelos de calidad son la forma estándar de describir la calidad de componentes.

En este capítulo se proponen los modelos de calidad del software como elemento vehicular para el análisis de la calidad de los componentes software. Tras un compendio de las propuestas más conocidas, se presenta el estándar ISO/IEC 9126-1 como marco de referencia, así como nuestra propuesta de extensión de dicho estándar que incluye un refinamiento de los factores de calidad no funcionales y una extensión con factores de calidad no técnicos, de aplicación especialmente en el caso de componentes OTS. Finalmente se expone el método IQMC para la construcción de modelos de calidad para dominios particulares.

---

[3] Incluso en el caso de componentes de granularidad gruesa, como sistemas ERP o similares, y excepto en raras ocasiones, surge la necesidad de integrarlos con sistemas externos.

10.2 MODELOS DE CALIDAD

Según el estándar ISO 8402 (1986), un modelo de calidad puede definirse como el conjunto de factores de calidad, y de relaciones entre ellos, que proporciona una base para la especificación de requisitos de calidad y para la evaluación de la calidad de los componentes software. Los modelos de calidad se estructuran generalmente como una jerarquía (ya sea un árbol, ya sea un grafo dirigido), donde factores de calidad más genéricos, como eficiencia o usabilidad, se descomponen en otros más particulares, como tiempo de respuesta o facilidad de aprendizaje, probablemente en diversos niveles de descomposición.

Los modelos de calidad pueden aplicarse en diversas actividades propias del DBSC: establecer los requisitos de calidad para la selección de un componente en base a los factores de calidad del modelo; evaluar la calidad de un componente para cada uno de los factores de calidad del modelo; comparar la calidad de distintos componentes respecto a los requisitos establecidos para un proceso de selección; y redactar contratos formales, donde aparezcan explícitamente las evaluaciones de calidad de los componentes que el proveedor certifica. Normalmente, los factores de calidad que aparecen en el modelo pueden utilizarse como *checklist* para todas aquellas cuestiones relacionadas con la calidad de los componentes.

Desde que se formuló el concepto de modelo de calidad, se han presentado múltiples propuestas. Dichas propuestas intentan resolver entre otros los interrogantes siguientes: ¿Cuáles son los factores de calidad que deberían formar parte de un modelo de calidad de componentes software?; ¿Cuáles son los tipos de factores de calidad en los que tiene sentido estructurar los modelos?; ¿Cómo se estructuran los modelos?; ¿Qué tipo de relaciones pueden existir entre los factores de calidad?; ¿Cómo se evalúan los factores de calidad?

En esta sección presentamos una clasificación de los tipos de modelos de calidad, las propuestas de estándares de modelos de calidad más usadas, y una descripción y comparación de las propiedades de las distintas propuestas.

10.2.1 Tipos de modelos de calidad

Las propuestas existentes de modelos de calidad se pueden clasificar según si tienen un enfoque de modelos de calidad fijos, a medida o mixtos (v. fig. 10-1).

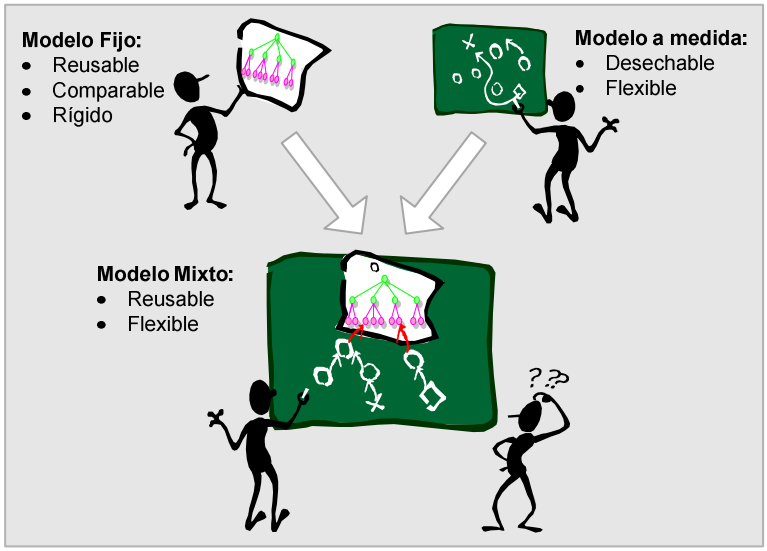


Figura 10-1. Clasificación de los modelos de calidad.

En los modelos de calidad fijos existe un catálogo de factores de calidad de partida que se usa como base para la evaluación de la calidad. Este enfoque supone que el modelo de calidad contiene todos los factores de calidad posibles, y que se usará un subconjunto de dichos factores para cada proyecto concreto. En general, la propuesta típica de un modelo de calidad fijo consiste en una estructuración de los factores en una jerarquía multinivel, con un conjunto de factores de más alto nivel, unos criterios que descomponen dichos factores, y eventualmente métricas para la medida de cada criterio. Ejemplos de modelos que siguen este enfoque son los modelos de McCall et al. (1997), Boehm et al. (1978), Keller et al. (1990) y el modelo con un enfoque más industrial FURPS (Grady y Caswell, 1987). La ventaja de estos modelos fijos es que proporcionan una vista común y comparable que se reutiliza en cada proyecto (v. fig. 10-2), ya que el conjunto de factores de calidad siempre es el mismo. Ahora bien, tiene como inconveniente su poca flexibilidad (Gilb, 1988) debido a que asumen que siempre bastará con un subconjunto de sus factores para evaluar la calidad en cualquier proyecto.

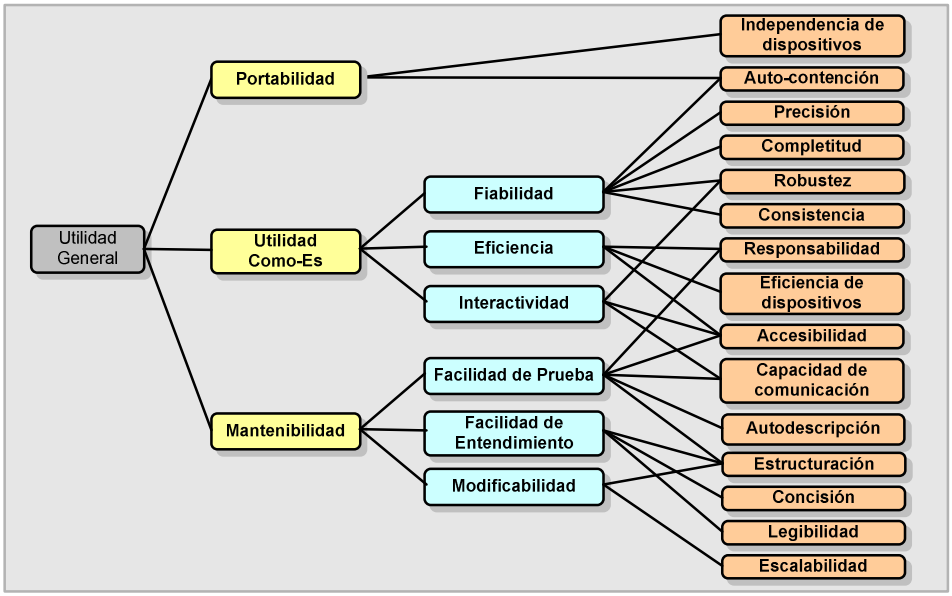
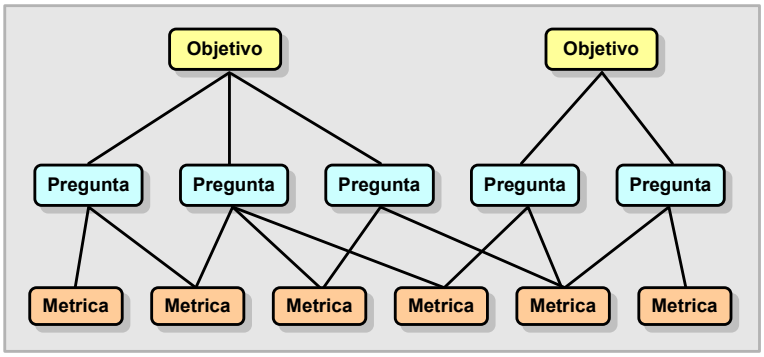


Figura 10-2. Ejemplo de modelo de calidad fijo: el modelo de Boehm et. al.

En los modelos de calidad a medida no existe ningún catálogo de factores de partida, y dichos factores deben ser identificados para cada proyecto. La idea que guía la construcción de estos modelos es que se debe partir de la identificación de los objetivos a alcanzar. Dichos objetivos serían los factores más abstractos que deben descomponerse en factores más concretos hasta llegar a hacer operativos los objetivos, de forma que pueda ser medida su consecución. Así, los modelos son creados desde cero para todo nuevo proyecto. Existen diversas propuestas de métodos para crear los modelos de calidad a medida, entre las que podemos destacar GQM (Goal-Question-Metric) de Basili et al. (1992) (v. fig. 10-3) y la del estándar IEEE 1061 (1998). La ventaja de estos modelos es su total adaptabilidad. Ahora bien, tienen como inconveniente que el coste de su construcción es muy alto comparado con el de los modelos fijos, y la reutilización de modelos de un proyecto a otro es difícil, dado que los factores identificados para un proyecto no tienen porqué ser adecuados para otro.



Objetivo	*Propósito*	Mejorar
	Objeto	Las líneas de tiempo
	Objeto (proceso)	Cambiar tiempo de proceso
	Punto de vista	Administradores
Pregunta 1	¿Cuál es la velocidad de proceso requerida actualmente?	
Métrica 1	• Tiempo promedio del ciclo • Desviación estándar • % de veces fuera del límite	
Pregunta 2	¿Está mejorando el rendimiento?	
Métrica 2	• (Tiempo promedio del ciclo actual × 100) / Tiempo promedio inicial • Calificación subjetiva de los administradores	

Figura 10-3. Ejemplo de modelo de calidad a medida: el método GQM.

Finalmente en los modelos de calidad mixtos se intenta combinar las ventajas de los dos tipos anteriores de modelos. La idea es que exista un conjunto de factores de calidad más abstractos que sean reutilizados en virtualmente todos los proyectos posibles, y que puedan ser refinados y operacionalizados para un proyecto particular. En este caso podemos destacar como propuestas de este tipo de modelos el ADEQUATE (Horgan et al., 1999), el modelo de Gilb (1988) y el modelo propuesto en el estándar ISO/IEC 9126-1 (2001) que se presenta a continuación.

10.2.2 Estándares de modelos de calidad

Podemos destacar dos estándares de modelos de calidad ya citados, el estándar IEEE 1061 y el estándar ISO/IEC 9126.

El estándar IEEE 1061 (1998) tiene como objetivo la definición de métricas de software y su uso en la evaluación de componentes software. Fue aprobado en 1992 y revisado y modificado en 1998. Propone la construcción de modelos de calidad a medida adaptados a cada proyecto. No fija ningún factor de calidad, pero sí una clasificación de los factores de los que debe constar un modelo en un nivel más alto y abstracto de factores, que deben descomponerse en subfactores, que a su vez se descomponen en métricas (v. fig. 10-4).

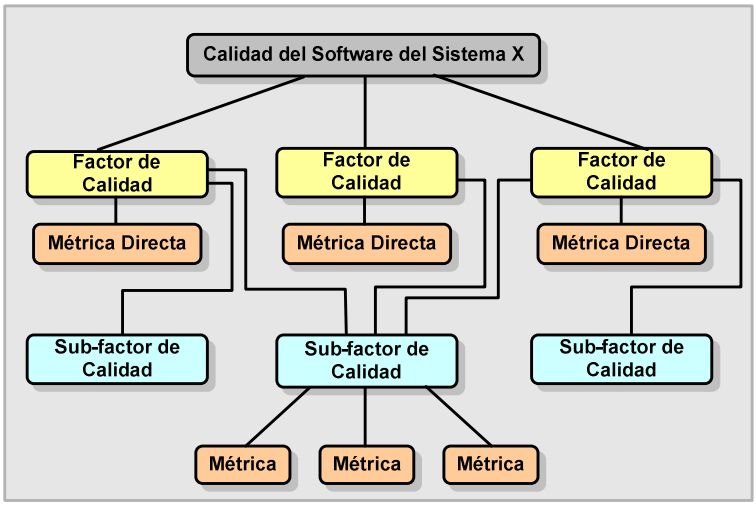


Figura 10-4. Estructura de los modelos de calidad según el estándar IEEE 1061.

El estándar ISO/IEC 9126 tiene como objetivo la definición de un modelo de calidad y su uso como marco para la evaluación de software. Como ya se ha mencionado, los modelos de calidad concordantes con este estándar pertenecen a la categoría de modelos mixtos, ya que el estándar propone una jerarquía de factores de calidad clasificados como características, subcaracterísticas y atributos según su grado de abstracción, entre los que se propone un conjunto de factores de partida compuestos de 6 características y 27 subcaracterísticas.

El estándar ISO/IEC 9126 distingue entre calidad interna y calidad externa, e introduce también el concepto de calidad en uso. La calidad interna tiene como objetivo medir la calidad del software mediante factores medibles durante su desarrollo. La calidad externa pretende medir la calidad del software teniendo en cuenta el comportamiento de este software en un sistema del cual forme parte. Finalmente, la calidad en uso corresponde a la calidad del software desde el punto de vista de un usuario.

El ISO/IEC 9126 original fue substituido en 2001 por dos estándares relacionados, el ISO/IEC 9126 de calidad del software y el ISO/IEC 14598 de evaluación de productos software (v. fig. 10-5). La versión de 2001 del ISO/IEC 9126 consiste de cuatro partes: 9126-1 (2001), presenta un modelo de calidad, que es común para medir la calidad interna y externa, y uno distinto para medir la calidad en uso; 9126-2 (2003), presenta posibles métricas externas para atributos de calidad externos; 9126-3 (2003), presenta posibles métricas para atributos de calidad internos; y 9126-4 (2004), presenta posibles métricas para evaluar atributos de calidad en uso. Cabe destacar que en este cambio, las subcaracterísticas mencionadas anteriormente pasaron de ser recomendadas en un anexo, a formar parte del estándar.

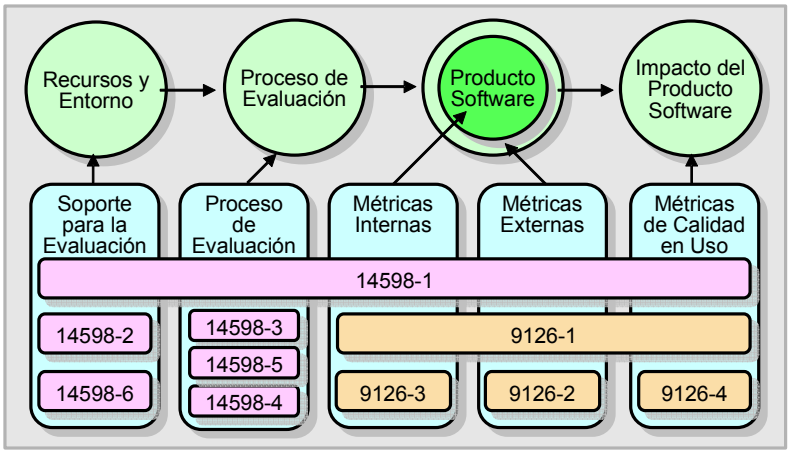


Figura 10-5. Relación entre los estándares 9126 y 14598 de ISO/IEC.

Actualmente, este estándar está en proceso de revisión, esperándose como resultado su aprobación y su inclusión en la nueva serie de estándares ISO/IEC 25000, también referenciados como Software Quality Requirements and Evaluation (abreviadamente, SQUARE, v. ISO, 2005).

10.2.3 Propiedades de los modelos de calidad

Del estudio de las diferentes propuestas de modelos de calidad existentes, se desprenden algunas propiedades estructurales importantes (v. fig. 10.6).

Número de capas

En general, el número de capas de un modelo de calidad puede ser utilizado como una medida para determinar el nivel de detalle con el que describe

el dominio de software para el cual ha sido construido: a más niveles, mayor descomposición y por tanto, una descripción más detallada del tipo de componente a evaluar. Los modelos a la medida tienden a estructurarse en jerarquías con más niveles de descomposición que los modelos fijos.

Tipos de elementos del modelo

En general todas las propuestas incluyen elementos de alto nivel, utilizados con propósitos de clasificación, y elementos de bajo nivel, utilizados con propósitos de descripción detallada y evaluación de características observables de los componentes. Eso sí, se observa una falta de uniformidad en la nomenclatura utilizada en diversos estándares (se usan indistintamente términos como "factor", "atributo", "característica", etc.).

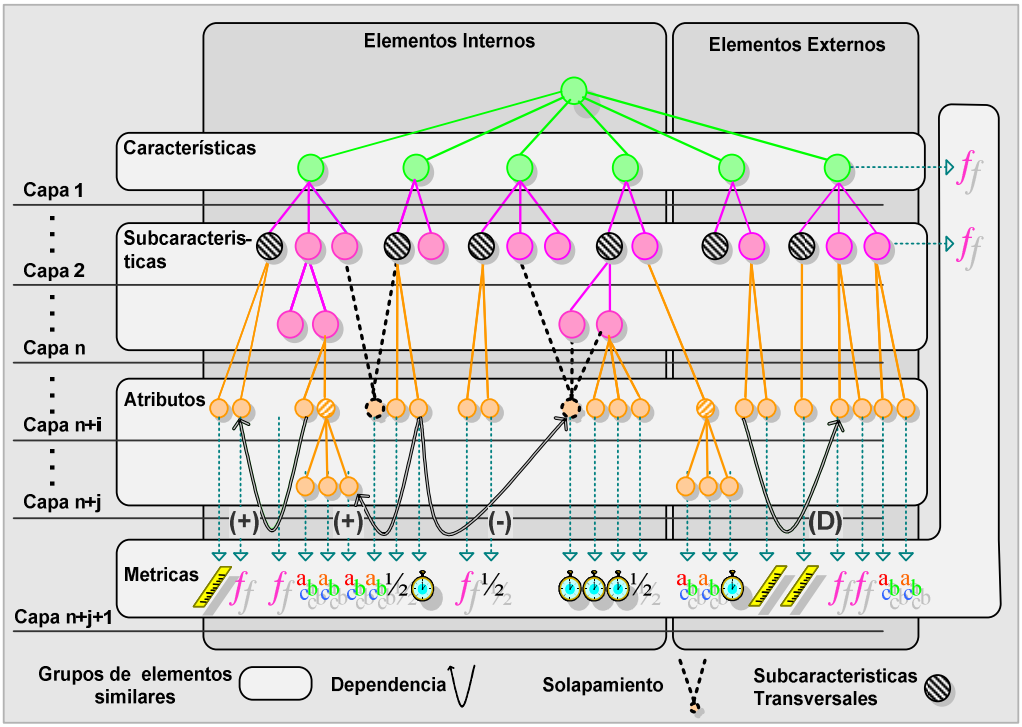


Figura 10-6. Propiedades estructurales de los modelos de calidad del software.

Normalmente encontramos una relación entre el número de capas y los tipos de elementos, dando un número fijo de capas por tipo de elemento (p.e., una capa de ele-mentos de nivel más abstracto), o estableciendo ciertas restricciones

adicionales (p.e., un elemento de un tipo no puede descomponerse en elementos de más de un tipo diferente).

Propósito del modelo

Al construir modelos de calidad es necesario considerar al menos dos dimensiones: la dimensión especifico/general y la dimensión reutilizable/descartable

Los modelos generales carecen de información específica de un producto o proyecto, y por tanto son menos complejos en su estructura (menos capas, elementos de calidad y relaciones entre ellos), y son usualmente utilizados como modelos fijos. Por su parte los modelos específicos son construidos a la medida para un producto o proceso y un contexto organizacional dado, y por tanto con una mayor cantidad de información disponible, por lo que su estructura final suele ser más compleja

La dimensión reutilizable/descartable por su parte, categoriza los modelos de calidad respecto a su nivel de reusabilidad. Los modelos a la medida son definidos en base a requisitos específicos de un contexto o aplicación particular, lo cual restringe su reutilización. Por el contrario, los modelos fijos no son construidos para un contexto o aplicación particular, son más generales, lo cual favorece su reutilización. Los modelos a la medida pueden ser reutilizados en proyectos similares, o las partes específicas del proyecto pueden ser eliminadas para crear modelos de calidad reutilizables. En cambio, los modelos fijos son usualmente muy generales y requieren ser completados para que sean utilizables en proyectos específicos. Los modelos mixtos son una alternativa a los dos casos anteriores, pero nuevamente, su reusabilidad depende de su nivel de detalle.

Es difícil considerar una de las dimensiones sin considerar las implicaciones sobre la otra. En general se puede decir que la reusabilidad de un modelo de calidad se reduce al hacerlo mas específico y se incrementa al hacerlo más general.

Separación entre elementos internos y externos

Los factores externos son todos aquellos factores que pueden ser directamente percibidos por los usuarios y que afectan su trabajo (usualmente relacionadas a la funcionalidad y usabilidad), mientras que los factores internos hacen referencia a las características constructivas de los componentes, que son tan solo accesibles y controlables por sus fabricantes. No todas las propuestas existentes insisten en esta separación, que sí podemos encontrar p.e. en el estándar

ISO/IEC 9126 (2001-2004) y en los modelos resultantes de aplicar el método SQUID (Bøegh et al., 1999).

Relaciones entre factores de calidad

- Además de la pura descomposición jerárquica, los factores de calidad se encuentran relacionados por otros criterios. Citamos:

- Solapamiento: un factor de calidad participa en la descomposición jerárquica de varios otros de niveles superiores. Cabe citar que dicho factor puede evaluarse con métricas diferentes para cada uno los factores que descompone.

- Transversalidad: es una relación de solapamiento donde no sólo cambia la métrica, sino también la definición. Este es el caso de las seis subcaracterísticas de Cumplimiento asociadas a cada una de las características incluidas en el modelo de calidad del estándar ISO/IEC 9126-1 (2001).

- Dependencia: un factor de calidad se relaciona con otros factores, generalmente del mismo nivel. Por ejemplo, Chung et al. (2000) identifican diversos tipos de dependencia (*makes*, *breaks*, etc.) dependiendo del tipo de relación (favorecer vs. perjudicar) y del grado de intensidad de la misma (total o parcial). El número de dependencias puede llegar a ser muy elevado, aunque como señalan Egyed y Grünbacher (2004), muchas de ellas pueden no ser relevantes.

Relación de las métricas con los factores de calidad

Todas las propuestas de modelos de calidad existentes incluyen métricas asociadas al menos al nivel más detallado de descomposición, aunque en algunos casos (p.e., el estándar IEEE 1061), requieren explícitamente que las métricas sean también aplicadas a los niveles más altos o abstractos de la jerarquía. En el caso del estándar ISO/IEC 9126, las partes 2, 3 y 4, incluyen conjuntos completos de atributos y métricas explícitamente concebidos para su uso en modelos construidos en base a este estándar.

10.3 EL ESTÁNDAR DE CALIDAD ISO/IEC 9126-1

En este capítulo nos centramos en la parte del catálogo de factores de calidad del estándar ISO/IEC 9126 que goza de un reconocimiento más amplio por la comunidad. Ésta es la parte 1 del estándar (ISO, 2001), y concretamente dentro

de la misma, nos centramos en los modelos de calidad para la evaluación de la calidad externa del software. Esta decisión se fundamenta en el contexto de utilización de los modelos que presentamos en este capítulo: la evaluación de la calidad de componentes software existentes en el mercado. Para dichos componentes no es factible la evaluación de su calidad interna, debido al desconocimiento sobre cómo han sido desarrollados, así como tampoco una evaluación de su calidad en uso, ya que ésta depende del uso de los componentes una vez seleccionados.

ESTRUCTURA DEL ESTÁNDAR ISO/IEC 9126-1

En la sección anterior ya se ha presentado la estructura general de los modelos de calidad concordantes con el estándar ISO/IEC 9126-1. Resumiendo, el modelo se estructura como una jerarquía multinivel de factores de calidad. El nivel más alto de la jerarquía corresponde a características generales del software, que son descompuestas en subcaracterísticas y que a la vez son descompuestas en atributos. Los atributos del nivel inferior de la jerarquía deben ser atributos medibles, cuyo valor se puede calcular aplicando una métrica. La figura 10-7 presenta las seis características definidas para la evaluación de la calidad externa y su descomposición en subcaracterísticas, tal y como aparecen en el estándar

Características/ Subcaracterísticas	Definición
Funcionalidad	Capacidad del producto software para proporcionar las funcionalidades que satisfacen las necesidades explicitas e implícitas cuando el software se usa bajo unas ciertas condiciones
Adecuación	Capacidad del producto software para proporcionar un conjunto de funciones apropiado para unas ciertas tareas y objetivos de usuario
Exactitud	Capacidad del producto software para proporcionar los resultados o efectos correctos o acordados, con el grado necesario de precisión
Interoperabilidad	Capacidad del producto software para interactuar con uno o más sistemas
Seguridad	Capacidad del producto software para proteger información y datos de manera que las personas o sistemas no autorizados no puedan leerlos o modificarlos, al tiempo que no se deniega el acceso a las personas o sistemas autorizados
Cumplimiento funcional	Capacidad del producto software para adherirse a normas, convenciones o regulaciones en leyes y prescripciones similares relacionadas con la funcionalidad
Fiabilidad	Capacidad del producto software para mantener un nivel especificado de prestaciones cuando se usa bajo unas cierta condiciones
Madurez	Capacidad del producto software para evitar fallar como resultado de fallos en el software

Características/ Subcaracterísticas	Definición
Tolerancia a fallos	Capacidad del software para mantener un nivel especificado de prestaciones en caso de fallos software o de infringir sus interfaces
Capacidad de recuperación	Capacidad del producto software para reestablecer un cierto nivel de prestaciones y de recuperar los datos directamente afectados en caso de fallo
Cumplimiento de la fiabilidad	Capacidad del producto software para adherirse a normas, convenciones o regulaciones relacionadas con al fiabilidad
Usabilidad	Capacidad del producto software para ser entendido, aprendido, usado y ser atractivo para el usuario, cuando se usa bajo condiciones especificadas
Capacidad para ser entendido	Capacidad del producto software que permite al usuario entender si el software es adecuado y cómo puede ser usado para unas tareas o condiciones de uso particulares
Capacidad para ser aprendido	Capacidad del producto software que permite al usuario aprender sobre su aplicación
Capacidad para ser administrado	Capacidad del producto software que permite al usuario administrarlo y controlarlo
Capacidad de ser atractivo	Capacidad del producto software para ser atractivo al usuario
Cumplimiento de la usabilidad	Capacidad del producto software para adherirse a normas, convenciones, guías de estilo o regulaciones relacionadas con la usabilidad

Figura 10-7. Características y subcaracterísticas del estándar ISO/IEC 9126-1.

Características/ Subcaracterísticas	Definición
Eficiencia	Capacidad del producto software para proporcionar prestaciones apropiadas, relativas a la cantidad de recursos usados, bajo condiciones determinadas
Comportamiento temporal	Capacidad del producto software para proporcionar tiempos de respuesta y de proceso y índices de respuesta al realizar sus funciones bajo unas ciertas condiciones
Utilización de recursos	Capacidad del producto software para usar las cantidades y tipos de recursos adecuados cuando el software lleva a cabo su función bajo condiciones determinadas
Cumplimiento de la eficiencia	Capacidad del producto software para adherirse a normas o convenciones relacionadas con la eficiencia
Mantenibilidad	Capacidad del producto software para ser modificado. Las modificaciones podrían incluir correcciones, mejoras o adaptación del software a cambios en el entorno, y requisitos y especificaciones funcionales

Características/ Subcaracterísticas	Definición
Capacidad de ser analizado	Capacidad del producto software para serle diagnosticadas deficiencias o causas de los fallos en el software, o para identificar las partes que han de ser modificadas
Capacidad para ser cambiado	Capacidad del producto software que permite que una determinada modificación sea implementada
Estabilidad	Capacidad del producto software para evitar efectos inesperados debidos a modificaciones del software
Capacidad para ser probado	Capacidad del producto software que permite que el software modificado sea validado
Cumplimiento de la mantenibilidad	Capacidad del producto software para adherirse a normas o convenciones relacionadas con la mantenibilidad
Portabilidad	Capacidad del producto software para ser migrado de un entorno a otro
Adaptabilidad	Capacidad del producto software para ser adaptado a diferentes entornos, sin aplicar acciones o mecanismos distintos de aquellos proporcionados para este propósito por el propio software
Instalabilidad	Capacidad del producto software para ser instalado en un cierto entorno
Coexistencia	Capacidad del producto software para coexistir con otro software independiente, en un entorno común, compartiendo recursos comunes
Capacidad para reemplazar	Capacidad del producto software para ser usado en lugar de otro producto software, para el mismo propósito, en el mismo entorno
Cumplimiento de la portabilidad	Capacidad del producto software para adherirse a normas o convenciones relacionadas con la portabilidad

Figura 10-7. Características y subcaracterísticas del estándar ISO/IEC 9126-1 (cont.).

10.3.1 Concreción de conceptos ambiguos

Tal como hemos comentado anteriormente, uno de los factores que hace al estándar atractivo es su flexibilidad. Sin embargo, esta voluntad de flexibilidad ha provocado que exista una falta de precisión en la definición de algunos puntos importantes que detallamos a continuación, y para los que proponemos un cierto refinamiento. Para plasmar gráficamente estas decisiones y con el objetivo de trasmitir sin ambigüedades los conceptos incluidos en el estándar, se presenta en la fig. 10-8 un modelo conceptual de datos expresado en el lenguaje UML (Botella et al., 2004).

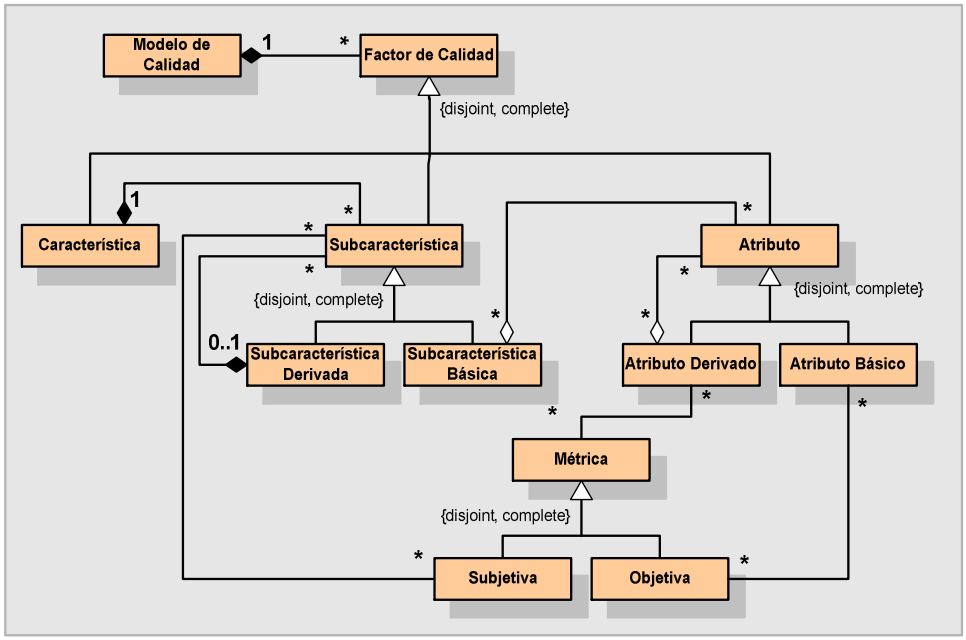


Figura 10-8. Modelo conceptual en UML del estándar ISO/IEC 9126-1

- Renunciamos a limitar el número de características a las 6 recogidas en el estándar. Ello permite extender el modelo no sólo en profundidad, como está previsto, sino también en anchura, y en concreto podremos así añadir nuevas dimensiones de calidad como son los factores no técnicos, estudiados más adelante en este mismo capítulo. Del mismo modo, es posible añadir nuevas subcaracterísticas si se considera apropiado.

- Permitimos la definición de jerarquías intermedias de subcaracterísticas y atributos en los modelos de calidad. Así, pues, por un lado, las subcaracterísticas pueden ser descompuestas en otras subcaracterísticas o alternativamente en atributos. Por otro lado, los atributos pueden ser derivados o básicos. Los atributos derivados pueden ser descompuestos en otros atributos, pero los básicos no.

- Consideramos que las características son factores de calidad no medibles, que se usan solamente como una forma de clasificar el nivel más alto de subcaracterísticas. Por el contrario, permitimos que las subcaracterísticas sean medibles si es necesario, aunque consideramos que no pueden ser medibles con una métrica objetiva (es decir, una métrica formal y precisa que permita asignarles un valor dependiendo del valor de las subcaracterísticas o atributos en los que se descomponen), sino con una

métrica subjetiva (es decir, con una métrica cuyo valor dependa de la percepción de las personas involucradas en el proceso de medida). De todos modos, cabe comentar que en nuestras experiencias raramente hemos medido subcaracterísticas, usando éstas al igual que las características, como elementos de clasificación.

- La jerarquía de características y subcaracterísticas se define como una jerarquía perfecta, debido a su papel primordial como criterio de clasificación. Por el contrario, permitimos solapamientos en la jerarquía de atributos y sus relaciones con las subcaracterísticas, pues se observa con relativa frecuencia que un atributo puede contribuir a diversos factores más abstractos.

- Las métricas para atributos básicos deben ser objetivas, ya que deben ser medidos de una formal y precisa, independiente de la percepción que puedan tener las personas involucradas en el proceso de medida. Para los atributos derivados algunas veces no es posible encontrar métricas objetivas para medirlos en función del valor de los atributos en que se descomponen. En estos casos se necesita una métrica subjetiva.

10.4 EL MODELO ISO/IEC 9126-1 EXTENDIDO

El objetivo del catálogo de factores de calidad de partida del modelo ISO/IEC 9126-1 es ofrecer un conjunto de factores de calidad que tienen sentido para cualquier dominio de componente software, y por tanto contiene factores de calidad que serán reusados entre distintos proyectos. En nuestra experiencia en el uso de dicho catálogo ha sido así. En todos los proyectos en que nos hemos visto involucrados, hemos usado el 100% de las características del catálogo y aproximadamente un 80% de las subcaracterísticas. Las subcaracterísticas que no se han usado siempre han sido las relacionadas con el cumplimiento de cada característica (p.e., Cumplimiento de la fiabilidad, Cumplimiento de la portabilidad, etc.). Estas subcaracterísticas están relacionadas con la capacidad del software de cumplir con unas normas, convenciones o regulaciones relacionadas con cada una de las características. En la mayoría de casos no se usaron debido a la falta de dichas regulaciones.

Sin embargo, también en dichas experiencias hemos observado que otros factores de calidad que usábamos para descomponer los propios del estándar, eran reutilizados regularmente. Este hecho hizo que construyéramos una extensión del modelo ISO/IEC 9126-1 original a la que llamamos modelo ISO/IEC 9126-1 extendido. Dicho modelo añade 60 nuevas subcaracterísticas, 34 de las cuales en el

tercer nivel de la jerarquía y 26 en el cuarto. De los nuevos factores de calidad añadidos, un 40% se usaron en todas nuestras experiencias de construcción y uso de modelos de calidad. También destacamos que decidimos mantener algún factor que apareció en pocos proyectos si consideramos que su existencia debía ser comprobada explícitamente

En la figura 10-9 incluimos la parte del catálogo extendido que descompone la característica Funcionalidad. Destacamos que la subcaracterística Adecuación no ha sido descompuesta dada su alta dependencia del dominio concreto de componentes de cada proyecto. En otras palabras, los factores que descomponen Adecuación pueden reutilizarse de un proyecto sobre sistemas de gestión documental a otro de la misma temática, pero no a un tercero referente a sistemas CRM

Características/ Subcaracterísticas	**Definición**		
Funcionalidad	Capacidad del producto software para proporcionar las funcionalidades que satisfacen las necesidades explicitas e implícitas cuando el software se usa bajo unas ciertas condiciones.		
Adecuación	Capacidad del producto software para proporcionar un conjunto de funciones apropiado para unas ciertas tareas y objetivos de usuario		
Exactitud	Capacidad del producto software para proporcionar los resultados o efectos correctos o acordados, con el grado necesario de precisión		
	Verificabi-lidad	Capacidad del producto software para proporcionar recursos para permitir monitorizar y verificar la correctitud de los resultados o efectos acordados	
		Historial de cambios	Capacidad del producto software para proporcionar una historia de los cambios en los datos gestionados
		Versión de datos	Capacidad del producto software para almacenar/ proveer versiones de los datos gestionados
		Capacidad de logging	Capacidad del producto software de proveer mecanismos de anotación o log
	Efecti-vidad	Capacidad del producto software para proporcionar mecanismos para determinar la correctitud de los resultados o efectos acordados	
		Resultados de autotest	Capacidad del producto software para proporcionar mecanismos para realizar pruebas directas de los resultados o efectos acordados
		Resultados de pruebas publicados	Capacidad del producto software según reportes publicados por terceras organizaciones sobre los resultados de pruebas de los resultados o efectos acordados
Interoperabilidad	Capacidad del producto software para interactuar con uno o más sistemas		
	Interopera-bilidad directa	Capacidad del producto software para interoperar con unos ciertos sistemas	
		Por medio de protocolos	Capacidad del producto software para interoperar directamente con otros sistemas por medio de unos determinados protocolos
		Por medio de APIs	Capacidad del producto software para interoperar directamente con otros sistemas por medio de unas ciertas librerías API
	Interopera-blidad indirecta	Capacidad del producto software para interoperar directamente con otros sistemas por medio de mecanismos indirectos	

<table>
<tr><th>Características/ Subcaracterísticas</th><th colspan="3">Definición</th></tr>
<tr><td rowspan="6">Seguridad</td><td colspan="3">Capacidad del producto software para proteger información y datos de manera que las personas o sistemas no autorizados no puedan leerlos o modificarlos, al tiempo que no se deniega el acceso a las personas o sistemas autorizados</td></tr>
<tr><td rowspan="3">Seguridad en la aplicación</td><td colspan="2">Capacidad del producto software para proporcionar mecanismos para prevenir el acceso no-autorizado deliberado o accidental a la funcionalidad del producto</td></tr>
<tr><td>Gestionada por la aplicación</td><td>Capacidad del producto software para proporcionar mecanismos para prevenir el acceso no-autorizado deliberado o accidental a la funcionalidad del producto</td></tr>
<tr><td>Gestionada por terceros</td><td>Capacidad del producto software según reportes publicados por terceras organizaciones para proporcionar mecanismos para prevenir el acceso no-autorizado deliberado o accidental a la funcionalidad del producto</td></tr>
<tr><td rowspan="3">Seguridad en los datos</td><td colspan="2">Capacidad del producto software para proporcionar mecanismos para prevenir el acceso no-autorizado deliberado o accidental a los datos gestionados</td></tr>
<tr><td>Datos almacenados</td><td>Capacidad del producto software para proporcionar mecanismos para prevenir el acceso no-autorizado deliberado o accidental a los datos almacenados</td></tr>
<tr><td>Datos transmitidos</td><td>Capacidad del producto software para proporcionar mecanismos para prevenir el acceso no-autorizado deliberado o accidental a los datos transmitidos</td></tr>
<tr><td>Cumplimiento funcional</td><td colspan="3">Capacidad del producto software para adherirse a normas, convenciones o regulaciones en leyes y prescripciones similares relacionadas con la funcionalidad</td></tr>
</table>

Figura 10-9. Descomposición de la característica Funcionalidad del ISO/IEC 9126-1.

10.5 EL MODELO ISO/IEC 9126-1-NT Y SU EXTENSIÓN

La extensión ISO/IEC 9126-1, como caso particular de extensión del modelo ISO/IEC 9126-1, propone un catálogo detallado de factores de calidad del software que responden a un denominador común: se refieren a la calidad desde el punto de vista técnico, intrínseca del componente software como tal. No obstante, en ciertos contextos, y en particular al reutilizar componentes OTS, el estudio de la calidad de un componente debe incluir otro tipo de factores que denominamos factores no técnicos, relativos al proveedor, al coste, etc.

La figura 10-10 ilustra esta necesidad. Por ejemplo, en la etapa inicial del DSBC, es necesario normalmente establecer el presupuesto estimado que se puede dedicar a la adquisición de componentes OTS. Posteriormente, en la etapa de selección, deben considerarse usualmente factores tales como el posicionamiento en el mercado de los proveedores potenciales de componentes, su experiencia en el dominio del proyecto, etc.

Como consecuencia, los factores no técnicos no pueden soslayarse en el contexto del DSBC. De hecho, puede constatarse que este tipo de factores responden a principios similares que los técnicos: pueden definirse a diversos

niveles de abstracción, presentan interrelaciones entre ellos, en general deben medirse, etc. Por lo tanto, aparece de manera natural la necesidad de integrar los factores técnicos y no técnicos bajo un mismo paraguas, es decir, nos planteamos la inclusión de los factores no técnicos en los modelos de calidad concordantes con ISO/IEC 9126-1 hasta ahora reservados a factores técnicos.

En la figura 10-11 aparece una propuesta de los dos primeros niveles que formarían lo que hemos venido a denominar modelo ISO/IEC 9126-1-NT (Carvallo et al, 2006, 2007). Hemos tratado que el resultado fuera uniforme respecto el modelo original, por lo que obtenemos una ratio características-subcaracterísticas similar: 15 subcaracterísticas para 3 características. El contenido del catálogo es como sigue (v. Carvallo et al. 2006b, para una descripción detallada de los factores individuales):

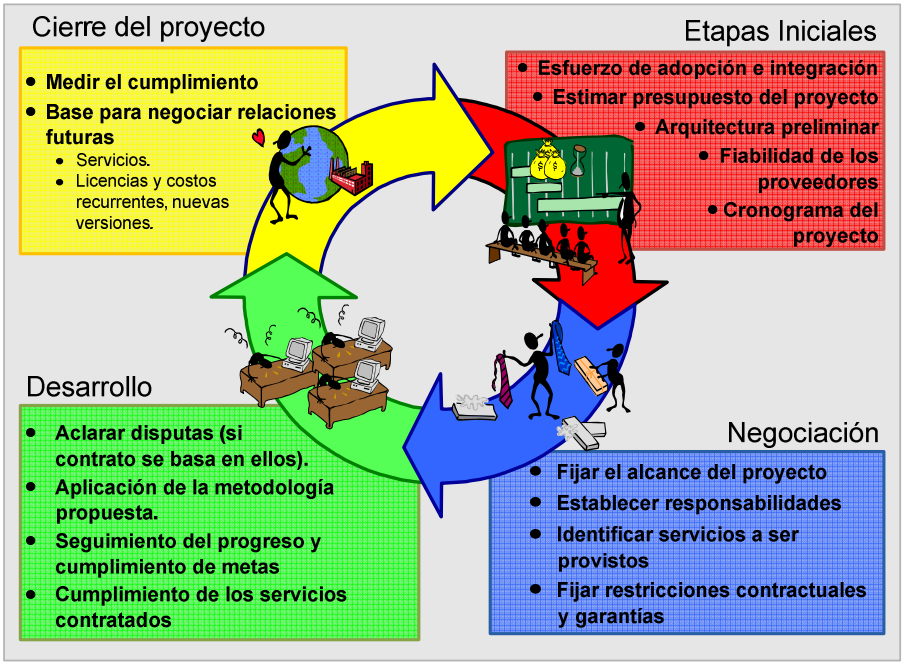


Figura 10-10. Importancia de los factores no-técnicos en el DSBC

Características Subcaracterísticas	Definición
Proveedor	Características del proveedor que pueden influenciar en la calidad del producto software
Estructura organizacional	Descripción de la estructura de la organización de la empresa del proveedor
Posicionamiento	Descripción del nivel y orientación de la empresa del proveedor en el mercado
Reputación	Reconocimiento de la capacidad del proveedor de realizar proyectos similares basado en certificaciones y experiencias pasadas
Servicios ofrecidos	Descripción de los servicios ofrecidos por el proveedor
Soporte	Descripción de los mecanismos de soporte ofrecidos por la empresa del proveedor
Negocio	Características del contrato entre el proveedor y el cliente que pueden influenciar la calidad del producto software
Esquema licencia	Descripción de las opciones de licencia posibles para el producto
Propiedad	Descripción de aspectos referentes a los derechos de propiedad intelectual del producto
Garantía	Detalle de las garantías que se ofrece para el producto
Costes de licencia	Descripción de los costes de los componentes del producto, y el coste total de propiedad según los tipos de licencia disponibles
Costes de plataforma	Estimación de costes para la plataforma de explotación requerida
Costes de implantación	Estimación de los costes de implantación del producto basada en anteriores experiencias
Coste de red	Estimación de los costes adicionales para la operación de red
Producto	Características de los aspectos comerciales del producto software que pueden influenciar su calidad
Historia	Evolución del producto desde que apareció en el mercado
Entregables	Detalle de los entregables que se ofrecen una vez implantado el producto
Parametrización ycustomización	Descripción del esfuerzo inicial requerido para preparar el producto para su puesta en explotación

Figura 10-11. Características y subcaracterísticas del modelo ISO/IEC 9126-1-NT.

- Proveedor: Los factores agrupados en esta característica tratan todos aquellos aspectos relacionados con los proveedores de componentes OTS

que pueden afectar a su calidad. Estos factores ayudan a estimar la capacidad del proveedor para responder al proyecto de implantación/integración de los componentes seleccionados, e identificar potenciales riesgos del proyecto en relación a los recursos humanos (p.e., carencia de personal cualificado, equipo de consultores muy reducido, etc.) y la estructura de la organización (p.e., excesiva dependencia de terceras organizaciones, carencia de áreas requeridas, etc.). También proporcionan una medida de la fiabilidad que pueden tener los componentes en tanto a su proveedor.

- Negocio: Los factores agrupados en esta característica evalúan aspectos que usualmente preocupan a las organizaciones en el momento de adquirir un componente COTS. Son de interés, entre otras cosas, los derechos que el proveedor otorga al cliente, los costes de cada tipo de licencia, y las garantías que el proveedor extiende sobre el componente y su operación futura.

- Producto: Los factores agrupados en esta característica tratan todos aquellos aspectos relacionados con el historial del producto y la facilidad para ponerlo a operar, los cuales permiten estimar el esfuerzo requerido para su adopción y mantenimiento dentro de una organización.

Al igual que sucedía con el modelo de calidad estándar ISO/IEC 9126-1, este modelo general que proponemos para factores no técnicos debe descomponerse en subcaracterísticas adicionales y atributos. En nuestras experiencias en procesos de selección de componentes OTS (descritas en Carvallo et al., 2006), así como mediante el estudio exhaustivo de propuestas en la misma dirección (detallado en Carvallo et al., 2007), hemos identificado diversos factores adicionales que aparecen con alta frecuencia, en concreto 144 nuevos factores entre subcaracterísticas y atributos. Con ellos hemos configurado el modelo ISO/IEC 9126-1-NT extendido. En la figura 10-12 se muestran algunos atributos identificados.

Atributo no-técnico	Definición	Métrica	Ejemplo
Producto en el mercado	Tiempo transcurrido desde que el componente está disponible en el mercado	Tiempo = Años Años = Integer	5 años
Versiones del producto	Versiones del componente que actualmente están disponibles en el mercado	Versiones = Conjunto(Versión)Versión = Tupla(Número Versión, Tiempo)Número Versión = StringTiempo = AñosAños = Integer	(V1, 2 años) (V2, 1 año)
Producción propia	Indica si el componente ha sido desarrollado por el propio proveedor	Desarrollo propio = Boolean	Cierto

Figura 10-12. Ejemplo de factores de la extensión del modelo ISO/IEC 9126-1-NT.

10.6 EL MÉTODO IQMC PARA LA CONSTRUCCIÓN DE MODELOS DE CALIDAD

La construcción de modelos de calidad viene dificultada por distintas circunstancias relacionadas con: (1) el equipo que realiza la construcción del modelo, en el caso de que este equipo no tenga experiencia en la construcción de modelos de calidad o bien en el contexto del dominio del componente objeto; (2) el dominio para el que se construye el modelo, para el que en muchas ocasiones no existe una terminología común; (3) factores metodológicos, ya que es difícil conocer el nivel de profundidad hasta el que es necesario descomponer los modelos, y por tanto cuándo se puede decir que un modelo de calidad se ha finalizado. La existencia de un método que proporcione unas pautas para la construcción de los modelos de calidad puede ayudar a paliar estas dificultades. En esta sección presentamos el método Individual Quality Model Construction (IQMC) que proporciona un conjunto de guías y técnicas para la identificación de los factores de calidad apropiados que deben ser incluidos en un modelo de calidad que permita analizar la calidad de componentes pertenecientes a un cierto dominio de software.

IQMC adopta un enfoque de construcción mixto. El catálogo de partida para la construcción de los modelos es el constituido por nuestros ISO/IEC 9126-1 e ISO/IEC 9126-1-NT extendidos (v. fig. 10-13). El método propone unos pasos para el refinamiento de dichos catálogos que conducen a la construcción de modelos de calidad para componentes software de un cierto dominio. Los modelos que se obtienen se estructuran según la concreción del estándar presentada anteriormente en este capítulo.

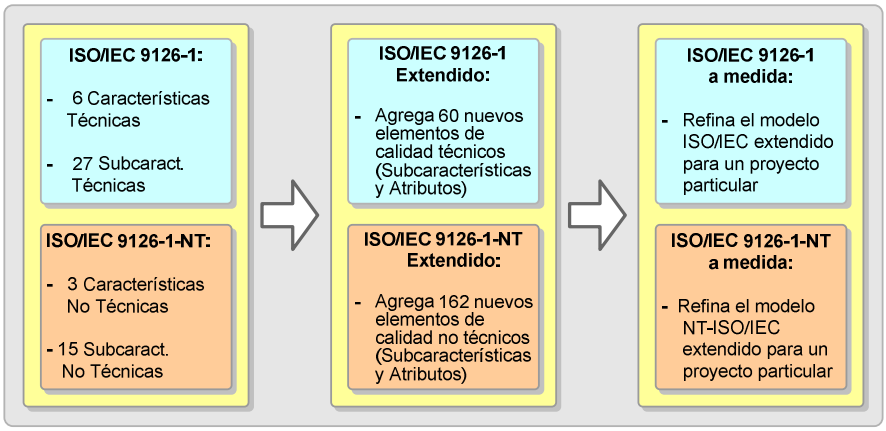


Figura 10-13. Extensión del ISO 9126-1 y su refinamiento para un contexto particular.

El método IQMC consiste de siete pasos (v. fig. 10-14) que, aunque se presentan como si fueran secuenciales, pueden ser simultaneados y/o iterados si se considera oportuno. En el primer paso, el ámbito de calidad es explorado en profundidad y, a continuación, los seis pasos restantes conducen la construcción del modelo de calidad partiendo de las características de calidad, y su descomposición en subcaracterísticas del catálogo ISO/IEC 9126-1 extendido. Es importante hacer hincapié en la diferencia de detalle que existe dentro de este catálogo entre lo que son los factores de calidad técnicos y los no-técnicos. Esta diferencia hace que en el método que se describe a continuación no se realiza un mismo enfoque para los unos que para los otros. En concreto, cuando se habla de refinamiento de algún tipo de factor de calidad, para los factores técnicos nos referimos más bien a una descomposición de dichos factores en factores de más bajo nivel de abstracción, y sólo ocasionalmente a la eliminación o modificación de algún factor ya existente. En cambio, cuando hablamos de refinamiento de factores de calidad no-técnicos, nos referimos más bien a la eliminación de factores que no sean relevantes para el contexto para el que se construye el modelo, y sólo ocasionalmente a añadir o modificar algún factor ya existente. En otras palabras, se puede llegar a considerar que la parte técnica de nuestro catálogo pertenece a la categoría de modelos de calidad mixtos, mientras que la parte no-técnica más bien parece un modelo de calidad fijo.

- Paso 0. Estudio del ámbito del software. Este paso consiste en realizar un estudio del ámbito al cual pertenecen los componentes software para los que se quiere evaluar la calidad. Es un paso opcional que puede soslayarse en caso de poseer el conocimiento suficiente, de ahí su numeración. Es recomendable revisar fuentes de información que describan dicho ámbito (Ayala y Franch, 2009). Por otra parte también puede ser interesante realizar algún tipo de modelización del ámbito para realizar una unificación de la terminología identificada en las distintas fuentes de información de cara a los pasos siguientes.

- Paso 1. Determinación de subcaracterísticas de calidad. Teniendo en cuenta que partimos del catálogo ISO/IEC 9126-1 extendido, el añadido de subcaracterísticas no será muy habitual y lo que puede pasar es que alguna de las existentes deba reformularse ligeramente para adaptarla al dominio de interés, o eliminarse en el caso de subcaracterísticas no técnicas.

- Paso 2. Refinamiento de la jerarquía de subcaracterísticas. Se descomponen las subcaracterísticas del más bajo nivel de abstracción formando jerarquías de subcaracterísticas. En lo que se refiere a las subcaracterísticas técnicas, al igual que en el paso anterior, el añadido de subcaracterísticas no será muy habitual, excepto en el caso de la

descomposición de la subcaracterística Adecuación perteneciente a la características Funcionalidad, pues como se ha comentado anteriormente, esta subcaracterística depende del dominio concreto para el cual se construye el modelo. En lo que se refiere a las subcaracterísticas no-técnicas, lo que se realizará es un purgado de las subcaracterísticas que no interesen para el proyecto en cuestión (notemos que en el caso de los factores no-técnicos el énfasis es en el proyecto y no en el dominio, dada su naturaleza independiente del dominio, más acusada incluso que en el caso de los factores no-funcionales).

- Paso 3. Refinamiento de subcaracterísticas en atributos. Este refinamiento tiene como objetivo llegar a tener descompuestas las subcaracterísticas en atributos medibles ya sea de forma directa o indirecta a partir del valor de otros atributos básicos.

- Paso 4. Refinamiento de atributos derivados en básicos. Se descomponen los atributos complejos (derivados) hasta obtener atributos básicos, los cuales pueden ser medidos de forma directa.

- Paso 5. Establecimiento de relaciones entre factores de calidad. Se establecen las relaciones entre factores de calidad que permiten conocer las dependencias entre los distintos factores de calidad del modelo.

- Paso 6. Determinación de métricas para los atributos. Se determinan las métricas para los atributos identificados.

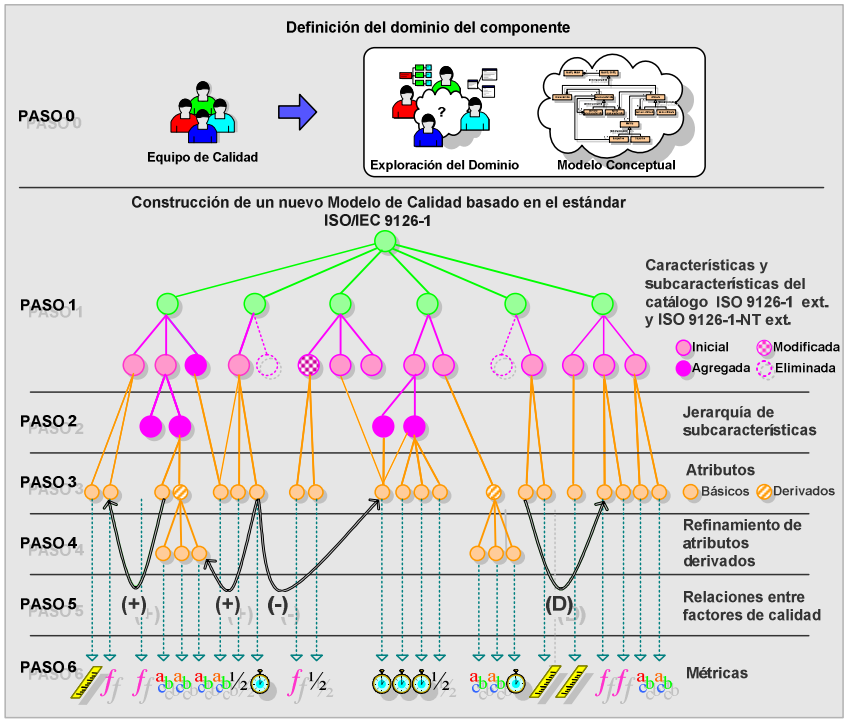


Figura 10-14. Pasos del método IQMC.

10.7 CONCLUSIONES

En este capítulo hemos abordado la problemática del estudio de la calidad de los componentes de software usando modelos de calidad. Hemos centrado nuestro estudio en el estándar ISO/IEC 9126-1 de calidad y hemos presentado nuestras propias aportaciones en forma de extensiones del catálogo, tanto en amplitud (considerando los factores de calidad no-técnicos) como en profundidad (produciendo niveles adicionales de factores de calidad), y de un método de construcción de modelos de calidad para un dominio o proyecto determinado. Estas extensiones y método han sido profusamente validadas en diversos casos industriales en los que los autores han participado, y también se han realizado algunos estudios académicos en profundidad. Puede encontrarse más información en las referencias siguientes: (Botella et al., 2003; Franch y Marco, 2004; Carvallo et al. 2003, 2004b, 2005, 2006; Franch et al., 2008). Asimismo, puede consultarse la página web http://www.lsi.upc.edu/~gessi/QMTool/CQM/ que contiene información actualizada sobre los modelos, las experiencias y software desa-

rrollado (en este último apartado, citamos la herramienta DesCOTS, Grau et al. 2004).

Los resultados presentados aquí pueden ser utilizados:

- Al desarrollar un componente que deba satisfacer unos requisitos dados, para verificar el grado de satisfacción de los mismos.
- Al desarrollar un componente genérico, para describir con precisión la calidad que ofrece.
- Al seleccionar un componente para su integración en un sistema, para guiar el proceso de evaluación y selección.
- Al integrar un componente en un sistema, para asegurar la compatibilidad de ambos tipos de software.
- Al evolucionar un componente en el tiempo, para facilitar el análisis del impacto de los cambios.
- Al redactar documentos con entidad legal (p.e., contratos), para disponer de una guía que identifique los criterios cuya aparición debe ser considerada.

No obstante, algunos puntos referentes al uso de modelos de calidad son todavía objeto de estudio. Entre ellos destacamos el cálculo de la calidad del sistema como combinación de los modelos de calidad de los componentes que lo integran tal y como se planteó ya en (Carvallo et al., 2004). Disponer de este algoritmo de cálculo, que por supuesto debe tener en cuenta un gran número de condicionantes y que es altamente dependiente de cada factor, representaría un avance considerable del estado del arte.

10.8 AGRADECIMIENTOS

Este trabajo ha sido parcialmente apoyado por el proyecto de investigación TIN2007-64753 subvencionado por el Ministerio de Ciencia e Innovación.

10.9 BIBLIOGRAFÍA

AYALA, C.P. y FRANCH, X. (2009). "Gestión *Sistemática de la Calidad de la Información en los Procesos de Selección de Componentes de Software*". En:

Procs. XII Conferencia Iberoamericana de Ingeniería de Requisitos y Ambientes de Software (IDEAS'09).

BASILI, V.R., CALDIERA, G. y ROMBACH, H.D. (1994). *"Goal Question Metric Paradigm*". En: Encyclopedia of Software Engineering 1, John Wiley & Sons.

BØEGH, J., DEPANFILIS, S., KITCHENHAM, B. y PASQUINI. A. (1999). "*A Method for Software Quality Planning, Control, and Evaluation*". En: IEEE Software, 23(2).

BOEHM, B.W., BROWN, J.R., KASPAR, H., LIPOW, M., MACLEOD, G.J. y MERRITT, M.J. (1978). "*Characteristics of Software Quality*". North Holland Publishing Company.

BOTELLA, P., BURGUÉS, X., CARVALLO, J.P., FRANCH, X., GRAU, G., MARCO, J. y QUER, C. (2004). "ISO/IEC 9126 in *Practice: what do we need to know?"*. En: Procs. 1st Software Measurement European Forum (SMEF'04), Ed. Instituto di Recerca Internazionale.

BOTELLA, P., BURGÚES, X., CARVALLO, J.P., FRANCH, X., PASTOR, J.A. y QUER, C. (2003). *"Towards a Quality Model for the Selection of ERP Systems"*. En: Component-Based Software Quality, Methods and Techniques, LNCS 2693, Springer-Verlag.

CARNEY, D. y LONG, F. (2000). "*What do you mean by COTS? Finally, a useful Answer*". En: IEEE Software, 17(2).

CARVALLO, J.P., FRANCH, X., GRAU, G. y QUER, C. (2004). "*COSTUME: A Method for building Quality Models for composite COTS-based Software Systems*". En: Procs. 4th IEEE International Conference on Quality Software (QSIC'04).

CARVALLO, J.P., FRANCH, X., QUER, C. y RODRÍGUEZ, N. (2004). "*A Framework for selecting Workflow Tools in the Context of Composite Information Systems*". En: Procs. 15th International Conference Database and Expert Systems Applications (DEXA'04), LNCS 3180, Springer-Verlag.

CARVALLO, J.P., FRANCH, X. y QUER, C. (2005). "*A Quality Model for Requirements Management Tools*". En: Requirements Engineering for Sociotechnical Systems, Idea Group Publishing.

CARVALLO, J.P., FRANCH, X., y QUER, C. (2006). *"Managing Non-Technical Requirements in COTS Components Selection*". En: Procs. 14th IEEE International Requirements Engineering Conference (RE'06).

CARVALLO, J.P., FRANCH, X., y QUER, C. (2006b). "*Un Catálogo de Factores de Calidad para la Definición de Requisitos No-Técnicos en la Selección de Componentes COTS*". En: Procs. 9º Worshop em Engenharia de Requisitos (WER'06).

CARVALLO, J.P., FRANCH, X., y QUER, C. (2007). "*Towards a Unified Catalogue of Non-technical Quality Attributes to Support COTS-Based Systems Lifecycle Activities*". En: Procs. 6th IEEE International Conference on COTS-Based Software Systems (ICCBSS'07).

CHUNG, L., NIXON, B., YU, E. y MYLOPOULOS, J. (2000). "*Non-functional Requirements in Software Engineering*". Kluwer Academic Publishers.

EGYED, A. y GRÜNBACHER. P. (2004) "*Identifying Requirements Conflicts and Cooperation: how Quality Attributes and Automated Rraceability can help.*" En: IEEE Software, 21(6).

FRANCH, X. y MARCO, J. (2004). "*A Quality Model for the Ada Standard Container Library*". En: 8th Ada-Europe International Conference on Reliable Software Technologies, LNCS 2655, Springer-Verlag.

FRANCH, X., QUER, C., CANTÓN, J.A. y SALIETTI, R. (2008). "*An Experience on the Construction of Quality Models for some Context Management Software Domains*". En: Procs. 7th IEEE International Conference on COTS-Based Software Systems (ICCBSS'08).

GILB, T. (1988). "*Principles of Software Engineering Management*". Addison Wesley.

GRADY R.B. y CASWELL, D.L. (1987). "*Software Metrics: Establishing a Company-Wide Program*". Prentice-Hall.

GRAU, G., CARVALLO, J.P., FRANCH, X. y QUER, C. (2004). "*DesCOTS: A Software System for selecting COTS Components*". En: Procs. 30th IEEE Euromicro Conference.

HORGAN, G., KHADDAJ, S. y FORTE, P. (1999) "*An essential Views Model for Software Quality Assurance*". En: Project Control for Software Quality, Shaker Publishing.

INSTITUTE OF ELECTRIAL ELECTRONIC ENGINEERING (1998). IEEE Std 1061-1998 IEEE *Standard for a Software Quality Metrics Methodology – Description.*

INTERNATIONAL STANDARDS ORGANIZATION (1986). ISO International Standard 8402: Quality *Management and Quality Assurance-Vocabulary*.

INTERNATIONAL STANDARDS ORGANIZATION (2001-2004). ISO/IEC Standard 9126-1: Software Engineering – Product *Quality – Part 1: Quality Model; Part 2: External Metrics; Part 3: Internal Metrics; Part 4: Quality in Use Metrics.*

INTERNATIONAL STANDARDS ORGANIZATION (2005). ISO/IEC 25000: 2005 Software Engineering -- *Software product Quality Requirements and Evaluation* (SQuaRE) -- Guide to SQuaRE.

KELLER, S., KAHN, L. y PANARA, R. (1990) "*Specifying Software Quality Requirements with Metrics*". *Systems and Software Requirements Engineering* - IEEE Computer Society Press – Tutorial.

LI, J., CONRADI, R., SLYNGSTAD, O.P.N., TORCHIANO, M., MORISIO, M. y BUNSE, C. (2008). "*A State-of-the-Practice Survey of Risk Management in Development with Off-the-Shelf Software Components*". En: IEEE Transactions on Software Engineering, 34(2).

MADANMOHAN, T.R. y RAHUL, D. (2004). "*Open Source Reuse in Commercial Firms*". En: IEEE Software, 21(6).

MCCALL, J.A., RICHARDS, P.K. y WALTERS, G.F. (1977) "*Factors in Software Quality*". RADC TR-77-369, Vols I, II, III, US Rome Air Development Center Reports NTIS AD/A-049.

PAPAZOGLOU, M.P. (2008). "*Web Services: Principles and Technology*". Prentice-Hall.

SZYPERSI, C. (2002). "*Component Software: Beyond Object-Oriented Programming (2ª ed.)*". Addison-Wesley.