COSTUME: Un método para la combinación de modelos de calidad

Juan P. Carvallo, Xavier Franch, Carme Quer, Xavier Burgués, Gemma Grau

Universitat Politècnica de Catalunya (UPC)
c/ Jordi Girona 1-3 (Campus Nord, C6) E-08034 Barcelona (Catalunya, España)
{carvallo, franch, cquer, diafebus, ggrau}@lsi.upc.es

Resumen. El uso de modelos de calidad durante la selección de componentes COTS (*Comercial Off-The-Shelf*) proporciona un entorno adecuado para la descripción de los dominios a los que éstos pertenecen. En este artículo tratamos la construcción de modelos de calidad para Sistemas Software basados en Componentes COTS (SSCC), que definiremos como sistemas compuestos por varios componentes COTS interconectados. Los procesos de selección llevados a cabo para obtener un SSCC requieren la selección de varios productos COTS. Proponemos un método para la construcción de modelos de calidad para SSCC basado en la aplicación de cuatro actividades. Nuestro objetivo es conseguir que los factores de calidad que aparecen en los SSCC estén definidos en términos de los factores de calidad de los componentes COTS y, de esta forma, obtener eficientemente modelos de calidad y sean fáciles de entender, analizar, mantener y reutilizar.

1 Introducción

Los requisitos de calidad juegan un papel crucial en los métodos y procesos hasta ahora propuestos para seleccionar componentes COTS (*Commercial Off-The-Shelf*) [1, 2, 3]. Su tratamiento durante el proceso de selección exige un gran esfuerzo para obtener, de forma eficiente, descripciones fiables, exhaustivas y comprensibles de la calidad de los COTS.

Los modelos de calidad [4] son una forma especialmente atractiva de estructurar dichas descripciones. Un *modelo de calidad* proporciona una jerarquía de *factores de calidad* del software así como *métricas* para evaluarlas. Los modelos de calidad pueden usarse en muchos contextos, por ejemplo para evaluar la calidad de los sistemas desarrollados a medida [5, 6].

En este artículo estamos interesados en el uso de modelos de calidad para facilitar la selección de COTS. En dicho contexto, los modelos de calidad están relacionados con dominios de COTS. De esta manera, los modelos de calidad deben abarcar los factores de calidad que son comunes en todos los productos COTS del dominio. Una vez el modelo de calidad esta disponible, tanto las evaluaciones de los factores de calidad de los COTS, como los requisitos de calidad que serán usados durante un proceso de selección deben traducirse a conceptos definidos en el modelo (figura 1).

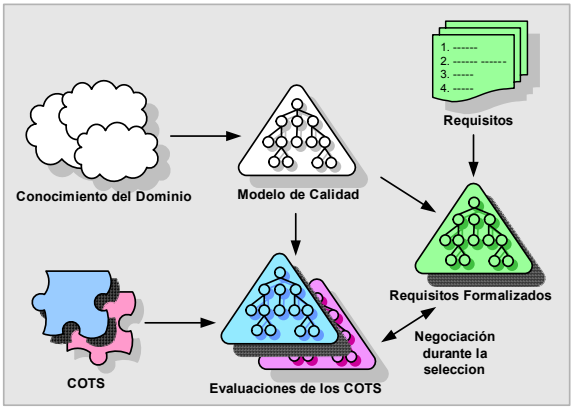

Figura 1. Uso de los modelos de calidad en la selección de COTS.

En este artículo abordamos la construcción de modelos de calidad para *Sistemas Software basados en Componentes* COTS (SSCC), sistemas compuestos por varios COTS interconectados. Nuestra propuesta puede ser aplicada sobre tipos particulares de SSCC como las *product lines* [7] y los *sistemas de información cooperativos* [8]. Nos proponemos definir las partes que componen los modelos de calidad para SSCC y presentar el método COSTUME de apoyo a su construcción.

2. Introducción a COSTUME

El punto de partida de esta investigación es el método para construir modelos de calidad explicado en [9, 10], basado en nuestras experiencias en el campo de la selección de COTS. En la aplicación de este método sobre componentes COTS de granularidad gruesa (p.e., el dominio de los servidores de correo [11]), descubrimos algunos defectos y comprendimos la necesidad de mejorar el método considerando de forma explícita el caso de los SSCC. El resultado es COSTUME (*CO*mposite *S*of*T*ware system q*U*ality *M*odel d*E*velopment), un método orientado a la definición de modelos de calidad para SSCC que consiste en las actividades siguientes, descritas detalladamente en las próximas secciones:

- *Actividad 1. Análisis del dominio del SSCC.* Identificación de los elementos organizativos que rodean el SSCC, así como de todos los sistemas externos con los que el SSCC deberá interaccionar.
- *Actividad 2. Descomposición del SSCC en subsistemas.* A partir de la actividad anterior el SSCC es descompuesto en elementos individuales, de manera que cada uno ofrece servicios bien definidos.
- *Actividad 3. Construcción de modelos de calidad individuales para el SSCC.* Aplicación de nuestro método [9, 10] para construir un modelo de calidad basado en el estándar de calidad ISO/IEC 9126-1 [12], para cada elemento del SSCC.
- *Actividad 4. Combinación de los modelos de calidad individuales.* Combinación de los modelos obtenidos para crear un único modelo ISO/IEC 9126-1 para el SSCC.

3. Análisis del entorno

Los SSCC no funcionan de forma aislada, sino que se comunican con otros sistemas y personas que actúan conjuntamente como su *entorno*. La primera actividad en COSTUME es identificar los actores del entorno que interactúan con el SSCC.

A lo largo del artículo, usaremos un SSCC para servidores de correo como ejemplo. En la tabla 1 se presenta una propuesta de actores para su entorno, en la que identificamos el tipo del actor y describimos brevemente su objetivo principal.

Actor	Abb.	Tipo	Objetivo
Mail Server System	MSS	Software	Proveer infraestructura para comunicación
Mail Client System	MCS	Software	Proveer accesos a los mensajes
Mail Server User	MSU	Humano	Enviar y recibir mensajes
Mail Server Administrator	MSA	Humano	Configurar el servidor de mail para que trabaje en forma eficiente y precisa
Firewall	Fwll	Hardware	Filtrar solicitudes entrantes

Tabla 1. Actores del entorno de los servidores de correo.

Las relaciones entre estos actores pueden ser modeladas gráficamente utilizando modelos basados en actores. En nuestras experiencias hemos aplicado el enfoque *i** de Yu [13] y, en particular, el modelo de Dependencias Estratégicas (SD) para representar redes de relaciones de dependencia entre los actores de un sistema. Las oportunidades disponibles para los actores son exploradas emparejando un *depender*, que es un actor que "quiere", con un *dependee* que tiene la "habilidad" de dar lo que quiere el *depender*. Se puede desarrollar un modelo de dependencias a alto nivel del SSCC reconciliando las habilidades del *dependee* con las necesidades del *depender*.

La figura 2 muestra el *modelo del entorno* para el caso de los servidores de correo. Los objetivos (*goals*) y recursos (*resources*) están relacionados con las funcionalidades principales que ofrece el servidor de correo: mailing, facilidades de cooperación, administración de los listados de direcciones y tareas de administración. Los objetivos no funcionales (*soft goals*) son identificados siguiendo los conceptos del ISO/IEC. Encontramos, por ejemplo, objetivos no funcionales referentes a la seguridad y la eficiencia. Como sólo estamos interesados en la parte del entorno que involucra el SSCC que se analiza, en nuestros modelos de calidad no representamos las dependencias entre los actores y su entorno.

4. Descomposición en subsistemas

En COSTUME debe descomponerse adecuadamente el SSCC en componentes individuales para facilitar su análisis separado. A pesar de esto, no partimos de una visión basada en componentes sino de una basada en actores: identificamos los actores que participan en el SSCC y establecemos relaciones entre ellos. Como resultado, descomponemos el sistema en sus partes esenciales, destacando los servicios proporcionados por estos actores en vez de la distribución del sistema en los componentes.

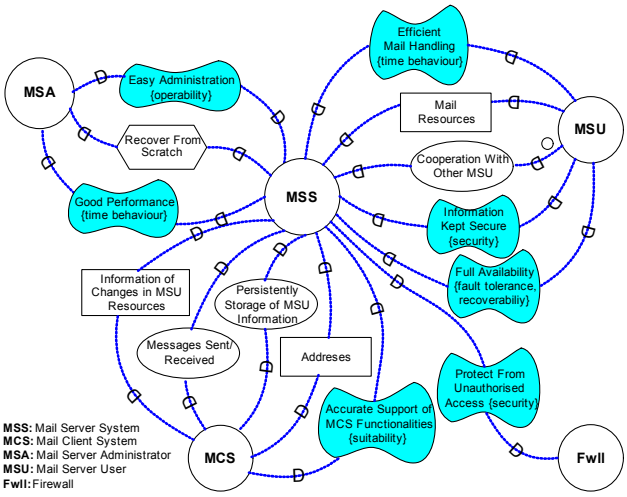

Figura 2. Modelo de entorno para los servidores de correo.

Esta actividad se realiza concurrentemente siguiendo dos caminos diferentes:

- Orientado al mercado. Es necesario tener un buen conocimiento de los tipos de herramientas actualmente disponibles en el mercado y las funcionalidades que proporcionan [14]. El continuo análisis de *white reports* y documentos técnicos de consultorías profesionales son un factor clave para el éxito.
- Orientado a objetivos. Es necesario tener la habilidad de identificar los actores del sistema y sus objetivos (cada actor tiene asignado un objetivo principal que debe conseguir). Se pueden adoptar técnicas existentes para el análisis e identificación de actores y objetivos [15] a nuestras necesidades especificas.

En COSTUME, las dependencias del modelo de entorno conducen a la identificación de algunas categorías de actores. Por ejemplo, la dependencia del objetivo *Cooperation with other MSU* hace surgir la necesidad de actores orientados al *groupware* y nuestro conocimiento del mercado revela la existencia de herramientas de *meeting scheduling*, y de *voice and video conference*. En la tabla 2 mostramos los actores identificados agrupados por categoría (omitiendo los objetivos de los actores por problemas de espacio). La tercera columna muestra las dependencias del modelo de entorno (ya mostradas en la figura 2) que justifican la necesidad del actor.

5. Construcción de modelos de calidad individuales

En [9, 10] propusimos un método para construir modelos de calidad para dominios independientes de COTS que cumplan la ISO/IEC. Dicho método consta de una serie de pasos para adaptar un modelo de calidad inicial basado en el estándar ISO/IEC 9126-1 [11] al dominio en cuestión. Partiendo de las seis características de calidad y su descomposición en subcaracterísticas:

- Añadimos nuevas subcaracterísticas específicas del dominio, refinamos la definición de algunas existentes o bien eliminamos algunas.

Categoría	Actores	Contexto de aplicación
Communication Support	Mail Servers	• Messages Sent/Received
	Routing Tools	• Efficient Mail Handling
Groupware Support	Meeting Scheduler Tools	• Co-operation With Other MSU
	Voice and Video-Conference Tools	
	Chatting Tools	
	Instant Messaging Tools	
	News Servers	
	Lists Servers	
Resources	Directory Services	• Mail Resources • Addresses • Persistent Storage of MSU information
	Compression Tools	• Mail Resources
Security Support	Anti-Spam Filter Managers	• Information Kept Secure
	Anti-virus Tools	
Administrative Support	Backup and Recovery Tools	• Full Availability
	Message Tracking Tools	• Efficient Mail Handling • Good Performance • Easy administration
	Configuration and Administration Tools	

Tabla 2. Actores para el caso de los servidores de correo

- Creamos una jerarquía de subcaracterísticas de calidad.
- Descomponemos las subcaracterísticas en atributos de calidad, de manera que representen factores observables en los COTS pertenecientes al dominio.
- Descomponemos atributos complejos en atributos simples directamente mesurables
- Determinamos las métricas para los atributos.

El resultado de este método es un modelo de calidad individual.

Definición 1. Modelo de calidad individual.

Un *modelo de calidad individual* para un dominio software *D*, es un modelo de calidad QM_D que cumple el estándar ISO/IEC 9126-1.

6. Composición de los Modelos de Calidad Individuales

Los modelos de calidad individuales dan una visión individual, aislada e incompleta de las partes del SSCC. Para conseguir una visión integrada de la calidad de todo el SSCC, la siguiente actividad consistirá en componer estos modelos de calidad. La composición se basa en combinar de forma apropiada los factores de calidad de los modelos de calidad. De manera formal:

Definición 2. Modelo de calidad de SSCC.

Sea *S* un SSCC y *A* el conjunto de actores en que se descompone. El *modelo de calidad de SSCC* para *S* se define como una tupla $QM_{SSCC} = (\{QM_D\}_{D \in A}, QM_S, Map_S)$:

- $\{QM_D\}_{D \in A}$ son modelos individuales tomando los actores de *A* como dominios. Los llamaremos *modelos de calidad componentes*.
- QM_S es un modelo de calidad que cumple la ISO/IEC 9126-1 y que llamaremos *modelo de calidad compuesto*.
- $Map_S = (MapSubcars_S, MapAttrs_S)$ es una familia de 2 funciones que asigna algunas de las subcaracterísticas y atributos de QM_S a aquéllos pertenecientes a $\{QM_D\}$.

Las características no son tenidas en cuenta de forma explícita porque se considera que todo modelo de calidad compuesto incluye las definidas en la ISO/IEC 9126-1.

El factor clave de la composición se basa en que la mayoría de los elementos del modelo de calidad están definidos como resultado de relacionar los nuevos elementos con los ya existentes, de manera que sólo aparecen unos pocos elementos (por ejemplo, aquéllos que se refieren a conceptos arquitectónicos). Por otro lado, la relación se define mediante la aplicación repetida de ciertos patrones de combinación de subcaracterísticas y atributos que serán definidos en el resto de esta sección.

6.1 Patrones de combinación para subcaracterísticas

Sea *S* un SSCC, *QMS* su modelo de calidad compuesto y QM_1, $QM_2 \in \{QMD\}$ dos modelos de calidad componentes. Podemos identificar 4 patrones para combinar subcaracterísticas (ver figura 3).

Para simplificar, las definiciones son establecidas sobre pares de elementos, fácilmente generalizables a *n* elementos. Cuando es necesario se usan prefijos representados por "::". Utilizamos la función *predecesor* para obtener el predecesor del factor de calidad en la jerarquía. La combinación de patrones debe ser aplicada de arriba a abajo (*top-down*), de manera que primero se aplica para construir los niveles superiores de la jerarquía y luego se completan los niveles inferiores.

Patrón de identificación

- Precondiciones:
  1) x e y son dos subcaracterísticas, $x \in QM_1$, $y \in QM_2$, $QM_1 \neq QM_2$
  2) existe una subcaracterística $p \in QM_S$ tal que $MapSubcars_S(p)$ = {predecesor(x), predecesor(y)}.
  3) $z \notin QM_S$.
- Patrón de aplicación: Identificación(x, y, z)
- Postcondiciones:
  1) $z \in QM_S$ tal que predecesor(z) = p.
  2) $MapSubcars_S(z) = \{QM_1::x, QM_2::y\}$.

Contexto de Aplicación

Hay muchas subcaracterísticas que aparecen de forma repetida en un gran número de modelos de calidad. En este caso, simplemente se añade la subcaracterística al modelo compuesto. De manera general, el patrón de identificación permite diferentes nombres en los diferentes modelos, aunque el caso habitual en este patrón es que tengan el mismo nombre.

Ejemplo

Un caso típico son las subcaracterísticas de la ISO/IEC 9126-1. Además, en nuestro caso de estudio, tanto los modelos de calidad de los dominios de *Voice and Video Conference* como los de *Chat*, descomponen su subcaracterística *Security* en *Local Security* y *External Security*. Hemos aplicado el patrón de identificación para obtener las mismas subcaracterísticas en el modelo compuesto.

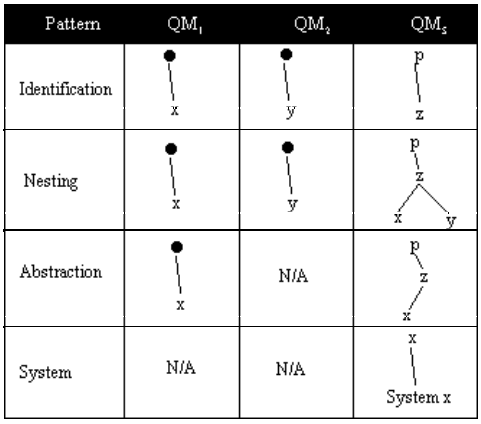

Figura 3. Patrones para las subcaracterísticas

Patrón de anidamiento

- Precondiciones:
  1) x e y son dos subcaracterísticas, $x \in QM_1$, $y \in QM_2$. En este caso, QM_1 y QM_2 pueden ser iguales.
  2) Existe una subcaracterística $p \in QM_S$ tal que $MapSubcars_S(p)$ = {predecesor(x), predecesor(y)}.
  3) $x, y, z \notin QM_S$.
- Patrón de aplicación: Anidamiento(x, y, p)
- Postcondiciones:
  1) $x, y, z \in QM_S$ tal que predecesor(z) = p, predecesor(x) = predecesor(y) = z.
  2) $MapSubcars_S(x) = \{QM_1::x\}$, $MapSubcars_S(y) = \{QM_2::y\}$.
  3) $z \notin$ dom($MapSubcars_S$)

Contexto de Aplicación

Al contrario de lo que sucede con la identificación, muchas subcaracterísticas deben mantenerse separadas en el modelo compuesto porque representan distintos factores de calidad, pero no es suficiente añadirlas tal cuál porque se empobrecería el modelo resultante desde el punto de vista de la comprensión. Por eso identificamos una nueva subcaracterística (z en la definición precedente) que ayudará a estructurar el modelo compuesto y que puede crearse siguiendo dos criterios diferentes:

- Orientado a dominio. La nueva subcaraterística z junta dos subcaracterísticas x e y del mismo dominio (i.e., QM_1 y QM_2 son el mismo).
- Orientado a funcionalidad. La nueva subcaracterística z junta dos subcaracterísticas x e y de dos dominios diferentes (i.e., QM_1 y QM_2 son diferentes) que hacen referencia a una misma funcionalidad.

Ejemplo

Una aplicación del patrón orientada al dominio aparece cuando se consideran dos subcaracterísticas de *Accuracy*, *Accurate Scanning&Repair* y *Actualization of Lists*, del dominio de los antivirus. En el modelo compuesto se agruparán introduciendo una nueva subcaracterística, sucesora de *Accuracy*, que llamaremos *Antivirus Accuracy*.

Un caso de orientación a la funcionalidad aparece con las subcaracterísticas de *Security*, *User Privileges* del dominio de los servicios de directorio y *Password Management* de los dominios de copia y recuperación. Las hemos combinado en el modelo compuesto definiendo la subcaracterística predecesora *Internal Security*.

Patrón de abstracción

- Precondiciones:
  1) x es una subcaracterística, $x \in QM_1$.
  2) existe una subcaracterística $p \in QM_S$ tal que $MapSubcars_S(p) = \{\text{predecesor}(x)\}$.
  3) $x, z \notin QM_S$.
- Patrón de aplicación: Abstracción(x, p)
- Poscondiciones:
  1) $x, z \in QM_S$ tal que predecesor(z) = p, predecesor(x) = z.
  2) $MapSubcars_S(x) = \{QM_1::x\}$.
  3) $z \notin \text{dom}(MapSubcars_S)$

Contexto de Aplicación

Caso especial del patrón de anidamiento que aparece en dominios pequeños en los que las subcaracterísticas de alto nivel se descomponen directamente en atributos. Al incorporar estas subcaracterísticas en el modelo compuesto, como éste ya tiene una descomposición en subcaracterísticas en el mismo nivel, es conveniente crear una subcaracterística que agrupe estos atributos y así obtener un modelo más equilibrado.

Ejemplo

La subcaracterística ISO/IEC *Fault Tolerance* no se descompone en el dominio de las herramientas de *Routing* de manera que atributos como *Time to Recalculate Path* son sucesores de esta subcaracterística. Sin embargo, al construir el modelo compuesto, es conveniente añadir una nueva subcaracterística a la que llamamos *Adaptable Routing,* sucesora de *Fault Tolerance* y que agrupe los atributos mencionados.

Patrón de sistema

- Precondiciones:
  1) x es una subcaracterística, $x \in QM_S$.
- Patrón de aplicación: Sistema(x)
- Poscondiciones:
  1) *"Sistema x"* $\in QM_S$ tal que predecesor(*"Sistema(x)"*) = x.
  2) *"Sistema x"* $\notin \text{dom}(MapSubcars_S)$

Contexto de Aplicación

Este patrón permite introducir una subcaracterística para agrupar todos aquellos atributos que no pueden ser definidos en términos de los factores de calidad de los componentes, y que acostumbran a surgir al considerar la arquitectura del sistema.

Ejemplo

En el modelo de calidad compuesto suele aparecer la nueva subcaracterística *System Interoperability* para agrupar los atributos que evalúan la capacidad de interoperabilidad que tienen los componentes de los dominios del SSCC entre ellos.

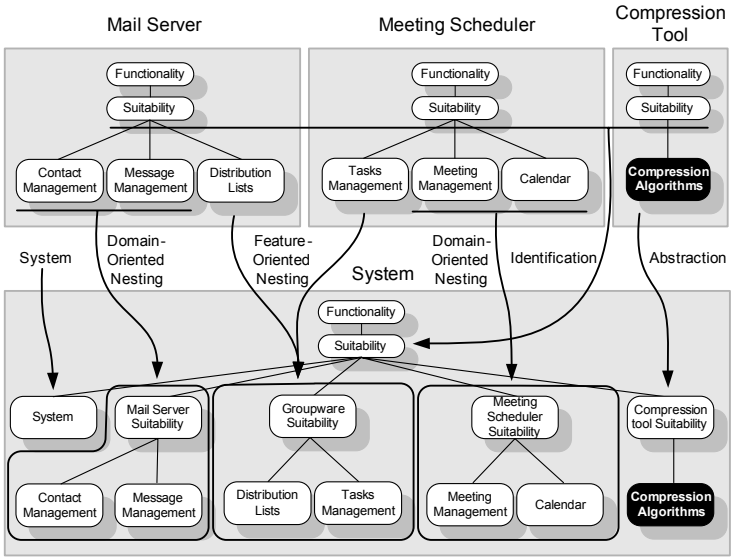

Figura 4. Ejemplo de aplicación de los patrones de subcaracterísticas (atributos en fondo negro).

La figura 4 muestra un ejemplo final de la parte de *Suitability* que usa estos cuatro patrones. El patrón de identificación es usado para identificar las subcaracterísticas de *Suitability* de los diferentes dominios. El patrón de anidamiento orientado a dominio es usado para conservar la parte de la S*uitability* inherente en los servidores de correo (p.e., *Contact Management* y *Message Management*) y los *Meeting Schedulers* (p.e., *Meeting Arrangement* y *Calendar*). El patrón de anidamiento orientado a funcionalidad se aplica para agrupar funcionalidades relacionadas como *Distribution Lists* y *Meeting Management* (en ambos casos, habrá atributos relativos a la gestión de grupos de ítems, ya sean listas de contactos o reuniones). El patrón de abstracción introduce nuevas subcaracterísticas para poner la *Suitability* de los algoritmos de compresión al mismo nivel que los otros. Finalmente, el patrón de sistema es utilizado para introducir una nueva subcaracterística que puede agrupar futuros atributos.

Los patrones definidos en esta sección pueden ser aplicados de diferentes maneras. Hemos encontrado que una estrategia con buenos resultados es la siguiente:

- El patrón de identificación es usado en los niveles superiores de la jerarquía.
- El patrón de anidamiento orientado a dominio suele aplicarse en los niveles inferiores de la jerarquía de subcaracterísticas. Ocasionalmente en este nivel también se aplican algunos patrones de identificación cuando se combinan atributos de dominios parecidos (e.g., *Antivirus* y *Antispam*).
- El patrón de abstracción es usado para nivelar la jerarquía.
- El patrón de sistema sólo se aplica en el primer nivel de las subcaracterísticas para evitar proliferación.
- El patrón de anidamiento orientado a funcionalidad sólo se usa cuando se identifica un nuevo concepto en el modelo compuesto.
- Si se sigue esta estrategia, la aplicación de patrones se convierte en un proceso mecánico a excepción del último caso que no es demasiado frecuente.

6.2 Patrones de combinación para atributos

Los atributos de calidad del modelo compuesto para el SSCC son básicamente de tres tipos: atributos heredados de los modelos individuales, atributos redefinidos a partir de los conceptos propios de los modelos y nuevos atributos que representan nuevas funcionalidades. Los patrones de atributos tienen en cuenta estos tres casos.

Sea QM_S el modelo de calidad compuesto y QM_1, $QM_2 \in \{QM_D\}$ dos modelos de calidad individuales del SSCC. Identificamos 5 patrones de combinación de atributos. Las definiciones se formulan respecto a pares de atributos pero pueden extenderse al caso general. Para mayor brevedad, suponemos que los atributos descienden directamente de subcaracterísticas (y no de otros atributos).

Patrón de delegación

- Precondiciones:
  1) x es un atributo, $x \in QM_1$, $x \in QM_2$.
  2) existe una subcaracterística $p \in QM_S$ tal que predecesor(x)$\in$ M$apSubcars_S(p)$.
- Patrón de aplicación: Delegación(x, p)
- Postcondiciones:
  1) $x \in QM_S$ de manera que predecesor(x) = p.
  2) $MapAttrs_S(x) = \{QM_1::x, QM_2::x\}$.
  3) La métrica de $QM_S::x$ se define como $x = QM_1::x$.

Contexto de aplicación

Dado un atributo en particular, se da el caso que en uno de los componentes este atributo predomina sobre el resto. Por lo tanto, el atributo del modelo compuesto se definirá sólo en términos del predominante.

Ejemplo

El atributo *Log Files* puede estar en varios componentes del SSCC, pero podemos estar interesados únicamente en saber si está presente en el servidor de correo. Así, consideramos que el SSCC genera *Log Files* si el servidor de correo los genera.

Patrón de redefinición

- Precondiciones:
  1) x es un atributo, $x \in QM_1$, $x \in QM_2$.
  2) existe una subcaracterística $p \in QM_S$ tal que predecesor(x)$\in$ M$apSubcars_S(p)$.
- Patrón de aplicación: Redefinición(x, z)
- Postcondiciones:
  1) $x \in QM_S$ de manera que predecesor(x) = p.
  2) $MapAttrs_S(x) = \{QM_1::x, QM_2::x\}$.
  3) La métrica de x se define como $x = K$.

Contexto de aplicación

Este patrón se aplica cuando un concepto que aparece en algún modelo componente es válido en el modelo compuesto, pero la definición del modelo compuesto no depende de los modelos componentes.

Ejemplo

Time in Market es un atributo de la subcaracterística *Maturity*. Todos los modelos componentes tienen este atributo y el modelo compuesto también deberá tenerlo, pero su valor no será calculado a partir de los otros.

Patrón de combinación

- Precondiciones:
  1) x es un atributo, $x \in QM_1$, $x \in QM_2$.
  2) existe una subcaracterística $p \in QM_S$ tal que predecesor(x)$\in$ M$apSubcars_S(p)$.
- Patrón de aplicación: Combinación(x, p)
- Postcondiciones:
  1) $x \in QM_S$ de manera que predecesor(x) = p.
  2) $MapAttrs_S(x) = \{QM_1::x, QM_2::x\}$.
  3) Las métricas de x se definen como x = f($QM_1::x$, $QM_2::x$).

Contexto de aplicación

El caso más habitual de aplicación ocurre cuando se tienen en cuenta todos los atributos de los modelos componentes y se definen métricas para combinarlos.

Ejemplos

Existen varios ejemplos con diferentes funciones de combinación:

- Aditivas. El atributo *Total Average Time to Send a Message* perteneciente a la subcaracterística *Time Behaviour*, puede definirse como la suma de cinco atributos: A*verage Sending Time* (servidores de correo), *Virus Scanning Time* (antivirus), M*ail Compression* (compresión de datos), *Time to Find Destination Node* (herramientas de routing), and *Time to Update Log File* (servidores de correo).
- Valor mínimo o máximo. El atributo *Mean Time Between Failures* puede ser definido cómo el mínimo de los *Mean Time Between Failures* de los componentes.
- Unión e intersección de conjuntos. El atributo *Languages Of Documentation* puede definirse cómo la intersección de los valores de los modelos individuales.
- Otras funciones complejas. La subcaracterística *Communication Protocol* está definida en los modelos individuales cómo una función respecto a una variable. En el modelo compuesto estará definida respecto a dos variables definidas como *CommProt*(x, y) = x::*CommProt*(y), siendo x e y dos modelos individuales de componentes (p.e., servidores de correo y servicio de directorio), y siendo x::*CommProt*(y) el protocolo de comunicación que usa x para comunicarse con y.

Patrón de nuevo atributo combinado

- Precondiciones:
  1) x es un atributo, $x \notin QM_1$, $x \notin QM_2$.
  2) existe una subcaracterística $p \in QM_S$.
- Patrón de aplicación: NuevoAtributoCombinado(x, p)
- Postcondiciones:
  1) $x \in QM_S$ de manera que predecesor(x) = p.
  2) $x \notin MapAttrs_S(x)$.
  3) La métrica de x se define como $x = f(QM_1::x, QM_2::x)$.

Contexto de aplicación

En el modelo compuesto, algunas veces es necesario definir nuevos atributos derivados en función de los existentes. En este caso, el nuevo atributo no se relación con los otros pero se usa una métrica para indicar su dependencia.

Ejemplo

El atributo *Operating Systems Supported* de la subcaracterística *Portability* puede tener como valor un conjunto de etiquetas. Usar una intersección de estos conjuntos es una definición muy restrictiva debido a que el SSCC del servidor de correo puede instalarse en plataformas distintas. Por esto definimos el atributo *Feasible Operating Systems Combination* que será un conjunto de conjuntos de etiquetas que registrarán las combinaciones válidas de sistemas operativos para los componentes.

Patrón de nuevo atributo de sistema

- Precondiciones:
  1) x es un atributo, $x \notin QM_1$, $x \notin QM_2$.
  2) existe una subcaracterística $p \in QM_S$.
- Patrón de aplicación: NuevoAtributoSistema(x, p)
- Postcondiciones:
  1) $x \in QM_S$ de manera que predecesor(x) = p.
  2) $x \notin MapAttrs_S(x)$.
  3) Las métricas de x se definen como $x = K$.

Contexto de aplicación

Este último caso aparece cuando un atributo no está relacionado con todos los factores de calidad de los modelos individuales. La mayoría de las veces el patrón puede aplicarse para capturar las propiedades arquitectónicas del SSCC pues, tal y como hemos mencionado anteriormente, algunas de las nuevas subcaracterísticas que se crean en el modelo compuesto provienen de la aplicación del patrón de sistema.

Ejemplo

El atributo *Cohesion* de la subcaracterística *Maintainability* no depende de los factores de calidad sino de la forma en que los COTS interoperan en el SSCC.

7. Análisis de COSTUME

A primera vista, la construcción de tantos modelos de calidad individuales puede parecer una actividad costosa en tiempo y recursos, pero en esta sección argumentamos que dicho coste se amortiza a lo largo del tiempo.

7.1 Construcción dirigida por los requisitos

Debido a las presiones de tiempo a que se someten los proyectos reales, es inviable aspirar a realizar un desarrollo completo de los modelos de calidad individuales. En vez de esto, proponemos su construcción dirigida por los requisitos que concentra el

esfuerzo en los factores de calidad directamente relacionados con el proyecto de selección para el que se construye el modelo. Consecuentemente, consideramos un proceso de selección iterativo [3, 16], en el que el proceso de establecimiento de requisitos y su evaluación en el modelo de calidad pueden ser intercalados.

No sólo el tipo de requisito, sino también su nivel de detalle debe ser considerado. Siguiendo esta pauta, los requisitos de los COTS que no proporcionan las funcionalidades principales del SSCC no deben ser tan detallados. Por ejemplo, en el SSCC para los servidores de correo, los requisitos sobre la detección de virus serán tan vagos como: "Los mensajes enviados por el sistema no deberan ser infectados por virus" sin especificar, p.e. que acciones deben realizarse cuando un virus es detectado. Consecuentemente, el modelo de calidad para el dominio de los antivirus puede verse reducido a unos pocos atributos.

7.2 Reutilización de Modelos de Calidad

Sin embargo, la visión orientada a requisitos puede ser perjudicial para otros propósitos, pues no estamos pensando en modelos de calidad de usar y tirar, únicamente construidos para el proyecto en el que surgen. Al contrario, pensamos en modelos de calidad que, de manera parcial o total, puedan ser usados en muchos proyectos. De forma específica, la reutilización puede aplicarse en las situaciones siguientes:

- Reutilización en diferentes procesos de selección del mismo tipo de SSCC. A pesar de que los requisitos pueden ser diferentes, es posible ampliar el modelo de calidad del SSCC con nuevos factores de calidad que los cubran.
- Reutilización en la construcción de nuevos modelos para SSCC. Este caso es especialmente cierto en el caso de los modelos de calidad de dominios de propósito general como los antivirus y las herramientas de copia y recuperación: una vez construido el modelo individual, puede ser reutilizado en un gran número de modelos de calidad de SSCC.
- Finalmente, algunos resultados del estudio del entorno pueden ser reutilizados para la construcción de otros modelos de calidad compuestos en los que aparece el mismo grupo de actores. Por ejemplo, en caso de construir un nuevo modelo compuesto para un sistema de clientes de mail, sus dependencias respecto a los sistemas servidores de correo pueden ser reutilizadas.

  Queda implícito que asumimos la existencia de un repositorio de modelos de calidad que pueden ser usados en los proyectos de selección y como punto de partida para la construcción de nuevos modelos de calidad. También asumimos que este repositorio debe ser actualizado con nuevos modelos de calidad. Proponemos en [17] la organización de este repositorio como una taxonomía en la que dominios relacionados pueden compartir partes comunes de sus modelos de calidad.

7.3 Mantenimiento de modelos de calidad SSCC

El mercado de COTS se ve continuamente afectado por los cambios. Cada pocos meses aparecen nuevos productos y versiones de los productos existentes, incorpo-

rando nuevas funcionalidades y mejorando las existentes. Los factores de calidad que aparecen en cada versión deben ser incorporados en los modelos de calidad.

La estructura interna de los modelos de calidad facilita la identificación y clasificación de nuevos factores de calidad. Adicionalmente a esta propiedad, común a cualquier enfoque basado en modelos de calidad, los modelos de calidad para los SSCC que proponemos son especialmente adecuados para añadir nuevos grupos de funcionalidades mediante la definición de actores y la actualización de sus correspondencias. Si analizamos la historia de los servidores de correo, puede comprobarse que los primeros sistemas no proporcionaban algunas funcionalidades como el *meeting scheduling* o los antivirus. Para un futuro próximo, pueden preverse nuevos servicios como el reconocimiento de voz (para dictar los correos) y *domain-ontology alignment* (p.e., para organizar las carpetas o determinar el tema del correo). Los nuevos actores generarán nuevos modelos de calidad individuales que no interferirán con los actuales. Los SSCC se actualizarán teniendo en cuenta estos nuevos componentes.

8. Conclusiones y trabajo futuro

En este artículo hemos propuesto un método para crear modelos de calidad para SSCC. El método COSTUME está organizado en cuatro actividades que incluyen: análisis del entorno, descomposición en subsistemas, construcción de modelos de calidad individuales y composición de modelos de calidad individuales. Los factores de calidad del modelo de calidad compuesto son definidos en términos de los modelos individuales relacionados con los actores del SSCC.

Pensamos que las contribuciones principales de nuestro trabajo son:

- La definición de un método más preciso para la construcción de modelos de calidad complejos utilizando una colección de patrones. A pesar de que ya hay unos cuantos métodos para construir modelos de calidad [5, 6], incluyendo el nuestro [9, 10], éstos no están especialmente orientados al tipo de sistemas compuestos que hemos tratado en este artículo.
- El proceso para construir modelos de calidad es eficiente en el sentido que facilita la reutilización. En la sección 7 hemos presentado 3 tipos de reutilización que hacen que nuestro método sea menos costoso que otros, considerado a largo plazo.
- La evolución del mercado requiere cambios en los modelos de calidad, pero la separación de los conceptos propuesta en COSTUME evita modificaciones innecesarias de los modelos de calidad existentes. Por otro lado, los modelos SSCC son altamente trazables: su definición mantiene constancia de los factores de calidad del los modelos componentes que afectan los factores de calidad del SSCC.
- Nuestra definición de los modelos de calidad para los SSCC es dual porque su forma interna es muy estructurada (permite identificar los modelos individuales) y, a la vez, cumple el estándar ISO/IEC 9126-1, hecho que puede posibilitar su mayor aceptación.

Actualmente estamos desarrollando una herramienta de apoyo a COSTUME. Disponemos de un primer prototipo para la construcción de modelos de calidad individuales que tiene como objetivo a corto plazo dar soporte a la aplicación de patrones.

Como trabajo futuro, planeamos integrar el concepto de ontología en el método y proponer una ontología propia.

References

[1] D. Carney, F. Long. "What Do You Mean by COTS? Finally a Useful Answer". *IEEE Software*, 17(2), March 2000.

[2] J. Kontyo. "A Case Study in Applying a Systematic Method for COTS Selection". *Procs. 18th IEEE ICSE*, 1996.

[3] N. Maiden, C. Ncube. "Acquiring Requirements for COTS Selection". *IEEE Software,* 15(2), 1998.

[4] *ISO Standard 8402: Quality management and quality assurance-Vocabulary*, 1986.

[5] R.G. Dromey. "Cornering the Chimera". *IEEE Software*, 13(1), 1996.

[6] B. Kitchenham, S.L. Pfleeger. "Software Quality: the Elusive Target". *IEEE Software*, 13(1), 1996.

[7] P. Clements, L. Northrop. *Software Product Lines: Practices and Patterns*. SEI series in SE, Addison-Wesley 2002.

[8] M. Papazoglou, G. Schlageter (eds.). *Cooperative Information Systems: Trends & Directions*. Academic Press, 1998.

[9] X. Franch, J.P. Carvallo. "A Quality-Model-Based Approach for Describing and Evaluating Software Packages". *Procs. 10th IEEE RE*, 2002.

[10] X. Franch, J.P. Carvallo. "Using Quality Models in Software Package Selection". *IEEE Software*, 20(1), 2003.

[11] *ISO/IEC Standard 9126-1 Software Engineering – Product Quality – Part 1: Quality Model*, 2001.

[12] J.P. Carvallo, X. Franch, C. Quer. "Defining a Quality Model for Mail Servers". *Procs. 2nd ICCBSS*, LNCS 2580, 2003.

[13] E. Yu. "Towards Modeling and Reasoning Support for Early-Phase Requirements Engineering". *Procs. 3rd IEEE ISRE*, 1997.

[14] M. Bertoa, J.M. Troya, A. Vallecillo. "A Survey on the Quality Information Provided by Software Component Vendors". 7th ECOOP QAOOSE workshop, 2002.

[15] Annie I. Antón. "Goal-Based Requirements Analysis". In *Procs.* 2nd IEEE ICRE 1996.

[16] X. Burgués, C. Estay, X. Franch, J.A. Pastor, C. Quer. "Combined Selection of COTS Components". *Procs. 1st ICCBSS*, LNCS 2255, 2002.

[17] J.P. Carvallo, X. Franch, C. Quer, M. Torchiano. "Characterization of a Taxonomy for Business Applications and the Relationships among them". In *Procs. 3rd ICCBSS*, LNCS, 2004 (to appear).