Medidas de Usabilidad de Componentes Software

Manuel F. Bertoa y Antonio Vallecillo, *Member IEEE*

***Resumen*— La última década ha marcado el primer intento real de convertir el desarrollo software en una ingeniería usando los conceptos de Desarrollo Software Basado en Componentes (DSBC) y de Componentes COTS (Commercial Off-The-Shelf) cuyo objetivo es crear elementos de alta calidad que se puedan ensamblar para construir un sistema funcional. Uno de los procesos críticos dentro del DSBC es la selección de los componentes que, cumpliendo los requisitos de funcionalidad definidos por el usuario, formarán parte del producto final. Sin embargo, existe una falta de modelos de calidad y de medidas que ayuden en la evaluación de las características de calidad de los componentes software durante este proceso de selección. Este trabajo presenta un conjunto de medidas para valorar la Usabilidad de los componentes COTS, y describe el método seguido para obtenerlas y validarlas empíricamente.**

***Palabras clave*— Software Metrics, Software Quality.**

I. INTRODUCCIÓN

El Desarrollo de Software Basado en Componentes (DSBC) se ha convertido en una importante alternativa para el desarrollo de aplicaciones software, en particular para sistemas distribuidos. Uno de los de pilares en este proceso de DSBC es la adecuada selección de los componentes COTS.

Actualmente, la mayoría de las cuestiones técnicas y tecnológicas del DSBC parecen haber sido abordadas satisfactoriamente. Es por tanto ahora cuando en mejores condiciones se encuentra la comunidad de Ingeniería del Software para tratar en profundidad los aspectos extra-funcionales del DSBC, y la evaluación de su calidad, temas que cada día cobran mayor importancia por el impacto industrial que comienza a tener el DSBC.

Hasta hace relativamente poco tiempo, existía una falta absoluta de métricas que pudieran ayudar a evaluar objetivamente lo atributos de calidad de los componentes software. Los estándares internacionales a cargo de estos temas (p.e. ISO/IEC 14598 [1] y la serie ISO/IEC 9126 [2]) están actualmente bajo revisión. El proyecto SQuaRE [3] se ha creado específicamente para hacerlos converger, intentando eliminar las lagunas, conflictos y ambigüedades que presentes actualmente en ellos. Un inconveniente de estos estándares internacionales es que proporcionan modelos y directrices de calidad muy generales, pero difíciles de aplicar en dominios concretos tales como el DSBC o los componentes COTS. En este sentido, han aparecido nuevas propuestas que tratan de definir modelos de calidad para componentes COTS o para sistemas basados en componentes [4, 5, 6, 7, 8, 9].

El objetivo del presente trabajo es definir un conjunto de medidas objetivas, realistas y fáciles de automatizar para componentes software, que se puedan usar para describir y valorar sus características de calidad y que puedan ayudar en el proceso de selección de componentes. Estas medidas podrían ofrecer una información similar a la que proporcionan actualmente los catálogos de componentes hardware, los cuales son muy útiles para seleccionarlos y para buscar sustitutos si se necesita una solución alternativa o de respaldo.

La valoración de la calidad de un componente software industrial es en general un objetivo demasiado amplio y ambicioso. Así, el Modelo de Calidad de ISO/IEC 9126 define la calidad de un producto software en términos de seis características principales (Funcionalidad, Fiabilidad, Usabilidad, Eficiencia, Mantenibilidad y Portabilidad), las cuales se refinan en 27 subcaracterísticas. Esto nos obliga a centrar nuestro esfuerzo en una característica concreta, la Usabilidad, debido a su importancia dentro del DSBC. La Usabilidad es inherente a la calidad del software porque expresa la relación entre el software y su dominio de aplicación. Por lo tanto, en este trabajo presentamos un conjunto de medidas para valorar la Usabilidad de los componentes software. Además, describimos el proceso seguido para obtenerlas y validarlas, proceso que puede reutilizarse para la definición y validación de medidas de otras características de calidad.

II. CONCEPTOS BÁSICOS

A. Desarrollo de Software Basado en Componentes (DSBC)

En primer lugar, es necesario definir que se entiende por componente software. En este trabajo vamos a utilizar la definición de Szyperski: Los componentes software son unidades binarias desarrolladas, adquiridas e incorporadas al sistema de forma independiente, y que interactúan para formar un sistema funcional [11].

El adjetivo "COTS" hará referencia a una clase especial de componentes: entidades comerciales (es decir, que se pueden vender o licenciar) que permiten el empaquetado, distribución, almacenamiento, recuperación y particularización por parte del usuario, que generalmente son de grano grueso, y que residen en repositorios software [6].

SM. F. Bertoa y A. Vallecillo son Profesores del Departamento de Lenguajes y Ciencias de la Computación de la Universidad de Málaga, España. (e-mail: bertoa@lcc.uma.es, av@lcc.uma.es).

Actualmente, grandes organizaciones como el ejército estadounidense ya han decidido utilizar los componentes COTS con objeto de mantener el coste y esfuerzo de sus desarrollos software bajo control [12]. Igualmente, están proliferando los vendedores de componentes software, con compañías como ComponentSource, que están empezando a distribuir, vender y licenciar componentes de terceras casas [10, 13]. Por último, también las grandes compañías de software tradicional están empaquetando sus productos como componentes (p.e., Sun, Microsoft, SAP, etc.).

B. Medición del Software

Al más alto nivel, el objetivo principal de un proceso de medición del software es satisfacer ciertas necesidades de información mediante la identificación de las entidades y de los atributos de estas entidades (los cuales serán el objeto del proceso de medición). Los atributos y las necesidades de información se relacionan a través de los conceptos mensurables (los cuales pertenecen a un Modelo de Calidad).

En este trabajo nos vamos a concentrar en una necesidad de información particular: "la valoración de la Usabilidad de un conjunto de componentes software que son candidatos a ser integrados en un sistema software, con el objetivo de seleccionar el mejor entre ellos". Como Modelo de Calidad hemos seleccionado una adaptación del modelo de calidad de ISO/IEC 9126. Como ese modelo de calidad es genérico para cualquier producto software, requiere cierta particularización para el caso concreto de los componentes software. El modelo particularizado, denominado COTS-QM, fue presentado en [4].

Las entidades de nuestro estudio serán los componentes software, sin importar si están desarrollados internamente o adquiridos como componentes COTS. Los atributos se evalúan utilizando medidas. Una medida relaciona una forma de medir y una escala de medición. Una forma de medir es la secuencia lógica de operaciones, descrita de forma general, usada para cuantificar un atributo con respecto a una escala especificada [14].

Se pueden distinguir tres clase de medidas: medidas base, medidas derivadas e indicadores. Las medidas base no dependen de ninguna otra medida (p.e., el número de figuras en el manual). Una medida derivada se obtiene de otras medidas base o derivadas (p.e., la ratio de métodos por interfaz). Un indicador es una medida que se obtiene a partir de otras medidas usando un modelo de análisis (basado en unos criterios de decisión) para obtener un resultado de la medición que satisfaga una necesidad de información.

III. USABILIDAD EN EL DSBC

Como mencionábamos en la introducción, en este trabajo nos vamos a centrar en la Usabilidad y adoptaremos la definición de ISO/IEC 9126, adaptándola al caso particular de los componentes software, de acuerdo a COTSQM [4]. Así, definimos Usabilidad como "la capacidad de un componente software para ser entendido, aprendido, usado y atractivo para los usuarios, cuando se usa bajo condiciones concretas."

Desde nuestro punto de vista, los usuarios de componentes serán "los miembros de los equipos software que desarrollan y mantienen sistemas basados en componentes". Los usuarios de componentes necesitan: identificarlos, localizarlos y seleccionarlos de acuerdo con las restricciones de la arquitectura del sistema y con los requisitos de los propietarios del sistema; integrarlos para construir el sistema; y, posteriormente, continuar con el mantenimiento del sistema cuando aparezcan nuevas versiones o cuando los componentes se actualizan para corregir errores. Por brevedad, en adelante para referirnos a esta clase de usuarios usaremos el término "desarrollador de sistemas", aunque este término abarque en nuestro caso a diseñadores, evaluadores de componentes, constructores e integradores del sistema y mantenedores de un sistema basado en componentes. Debemos recalcar que no hemos incluido ni a los usuarios finales del sistema basado en componentes, ni a los desarrolladores de los propios componentes individuales. En nuestra opinión, ninguno de ellos puede ser calificado como usuario de componente: los usuarios finales usan un sistema final, sin conocer (ni necesitarlo) si ha sido desarrollado utilizando componentes software u otra metodología; análogamente, los desarrolladores de componentes fabrican componentes individuales, pero no necesariamente usan componentes para construirlos. ISO/IEC 9126 y COTS-QM definen la Usabilidad en términos de cinco subcaracterísticas que en el caso de los componentes, se pueden definir como sigue:

- **Comprensibilidad**: capacidad del componente para permitir al desarrollador de sistemas comprender si el componente es adecuado, y cómo puede usarse en tareas y condiciones de uso particulares.
- **Aprendibilidad**: capacidad del componente para permitir al desarrollador de sistemas aprender su utilización. Una medida de Aprendibilidad debe ser capaz de valorar el esfuerzo que necesitarán los desarrolladores de sistemas para aprender como se usan las funciones particulares (interfaces, operaciones, etc.) o valorar la eficacia de la documentación suministrada.
- **Operabilidad**: capacidad del componente para permitir al desarrollador de sistemas operar con él y controlarlo.
- **Atracción**: capacidad del componente para resultar atractivo a los usuarios. Como los usuarios considerados no son usuarios finales del sistema, esta subcaracterística no parece ser relevante en este contexto.
- **Conformidad de Usabilidad**: capacidad de un componente para adherirse a estándares, convenciones, guías de estilo o regulaciones relacionadas con la Usabilidad. Actualmente no conocemos ningún estándar que afecte a la Usabilidad de los componentes y, por tanto, no vamos a considerar esta subcaracterística.

En resumen, la Usabilidad de un componente Software será valorada en términos de su *Comprensibilidad*, *Aprendibilidad* y *Operabilidad*.

IV. MEDIDAS DE USABILIDAD

Al buscar en la literatura medidas relacionadas con el DSBC, hemos descubierto que la relación entre las medidas y las características de calidad que tratan de evaluar no está bien definida. Esto es un problema habitual en la mayoría de las propuestas y estándares internacionales que definen medidas para componentes software.

Nuestro objetivo en este trabajo es resolver este problema, intentando definir adecuadamente un conjunto de medidas basadas en los atributos que evalúan, y determinando de qué forma estas medidas valoran las características de calidad de un componente software, de acuerdo a un modelo de calidad.

Además, vamos a exigir algunos requisitos a las medidas usadas para que sean prácticas en un entorno industrial: primero, que sean objetivas y reproducibles; segundo, que sean fáciles de calcular (mejor aún si se pueden calcular de forma automática); y, por último, que permitan identificar un conjunto mínimo de medidas que proporcionen la máxima cantidad de información sobre los conceptos mensurables que estamos intentando evaluar (i.e, optimizar el número de medidas representativas).

Como ya hemos mencionado, de los tres conceptos mensurables relacionados con la Usabilidad de los componentes software (Calidad de la Documentación, Complejidad de la Solución y Complejidad del Problema), solo vamos a considerar los dos primeros, puesto que al comparar componentes que proporcionan una misma o muy parecida funcionalidad, la Complejidad del Problema será similar en todos ellos. Estos conceptos mensurables y sus atributos relacionados se muestran en el Cuadro 1. La forma en la cual los conceptos mensurables influyen en la Usabilidad de un componente y en las subcaracterísticas que la componen se trata posteriormente en la sección 5.

Generalmente, cada atributo puede tener una o más medidas (indicadores, medidas derivadas o medidas base) que lo evalúen. Un resumen de las medidas propuestas para los atributos de la Calidad de la Documentación se muestra en el Cuadro 2, y las medidas propuestas para los atributos relacionados con la Complejidad del Diseño en el Cuadro 3. En ambas tablas, debería existir una última columna con las medidas base a partir de las cuales se calculan las medidas derivadas. Esta columna se ha omitido por razones de espacio y porque la mayoría de las medidas base son evidentes. En total se han definido 34 medidas derivadas y 35 medidas base.

Sin embargo, proponer un conjunto de medidas, que es justo lo que actualmente aparece en la mayoría de las propuestas e incluso de los estándares [4, 5, 6, 7, 8, 9], no nos parece suficiente. En primer lugar, no todas las medidas propuestas en los Cuadros 2 y 3 son fáciles de calcular. Por ejemplo, la medida "Porcentaje de elementos funcionales correctamente usados tras leer el manual" es muy difícil de calcular. Estas medidas se muestran en itálica en estas tablas. Por contra, otras medidas que son difíciles de computar en productos software generales se pueden obtener fácilmente, y hasta automatizar, en el caso de los componentes software: usando técnicas de reflexión computacional se puede interrogar los componentes y obtener información sobre sus elementos funcionales (interfaces, operaciones, parámetros configurable, etc.). Análogamente, podemos utilizar el hecho de que la documentación suele estar en formato electrónico en la mayoría de los componentes y, por tanto, la podremos analizar detalladamente. Así, de nuestro conjunto de medidas iniciales, únicamente las que se pueden obtener de forma objetiva y reproducible aparecen en la columna izquierda de los Cuadros 4 y 5. En total hemos trabajado y evaluado 19 medidas base y 21 medidas derivadas.

V. VALIDACIÓN DE LAS MEDIDAS

A. Validación Empírica

El objetivo de la validación es demostrar que una medida proporciona realmente una información útil para evaluar una característica de calidad. Con objeto de validar empíricamente nuestras medidas hemos realizado una familia de experimentos. Estos experimentos nos proporcionaron algunos datos (valores numéricos) sobre la Comprensibilidad, Aprendibilidad y Operabilidad de un conjunto de componentes:

- como *percibida* por un conjunto de usuarios conforme a las definiciones de Comprensibilidad, Aprendibilidad y Operabilidad, que les fueron dadas;
- como *percibida* por un conjunto de usuarios cuando se les pregunta "indirectamente" sobre la Comprensibilidad, Aprendibilidad y Operabilidad de los componentes que han analizado;
- *objetivamente*, como el resultado de un cuestionario que pedía a los sujetos que realizaran una serie de tareas con los componentes, para posteriormente valorar las respuestas de acuerdo a la corrección y al tiempo necesario para contestar las preguntas.

Entre Junio y Diciembre de 2004 se realizaron cinco experimentos. La idea era simular el proceso de selección de un componente software de entre un conjunto de potenciales candidatos. Se seleccionaron 12 componentes de un dominio de aplicación (*Print and Preview*). Todos los componentes eran productos COTS disponibles en vendedores comerciales, y anunciados en repositorios de componentes como ComponentSource. Estos componentes usaban diferentes modelos y tecnologías de componentes: Java, ActiveX y .NET.

El primer experimento se enfocó más hacia la valoración subjetiva de de la Usabilidad percibida de los componentes, con el objetivo de obtener los primeros datos sobre la Calidad de la Documentación y la Complejidad de la Solución de cada uno de ellos. Básicamente, los sujetos tenían que asignar un valor dentro de una escala semántica de siete valores (posteriormente estos valores se convertían en un entero entre (-3 y +3) a un conjunto de preguntas según su valoración subjetiva. Los dos últimos experimentos se enfocaron hacia las cuestiones objetivas (aunque se mantuvieron unas pocas preguntas de valoración subjetiva) sobre si el componente ofrecía determinados servicios o no, y sobre cómo se implementaban esas funcionalidades con cierto detalle.

Entidad	Necesidad de Información	Concepto Mensurable		Atributo
Software Componente	Evaluar la Usabilidad	Calidad de la Documentación	Calidad de los Manuales	Contenido de los Manuales
				Tamaño de los Manuales
				Efectividad de los Manuales
			Calidad de las Demos	Contenido de las Demos
			Calidad del Mktg Info	Contents del Mktg Info
		Complejidad de Diseño		Legibilidad del Diseño
				Comprensibilidad de las Interfaces
				Facilidad de Aprendizaje
				Complejidad de las API
				Configurabilidad

Cuadro 1: Conceptos medibles y atributos de calidad relativos a la Usabilidad.

Atributo	Indicador	Medida Derivada
Contenido de los Manuales	Cobertura de los Manuales	Porcentaje de FE descritos en los Manuales
		Porcentaje de Interfaces descritas en los Manuales
		Porcentaje de Métodos descritos en los Manuales
		Porcentaje de Parámetros Configurables descritos en los Manuales
	Consistencia de los Manuales	*Porcentaje de EF incorrectamente descritos en los Manuales*
		Diferencia entre Versiones del Componente
	Legibilidad de los Manuales	Ratio de ficheros HTML de Manuales por EF
		Ratio de figuras por kilo-palabra
Tamaño de los Manuales	Adecuación de los Manuales	Ratio de palabras por EF
		Ratio de palabras por Interfaz
		Ratio de palabras por Método
		Ratio de palabras por Parámetro Configurable
Efectividad de los Manuales	Ratio de Efectividad	*Porcentaje de EF correctamente usados después de leer los Manuales*
Contenido de las Demos	Cobertura de las Demos	*Porcentaje de EF incluidos en las demos*
	Consistencia de las Demos	*Diferencia con Versiones de las Demos*
Contenido del Mktg Info	Cobertura del Mktg Info	*Número de EF en Mktg Info*
	Consistencia del Mktg Info	*Diferencia con Versiones del Mktg Info*

Cuadro 2: Medidas relativas a la calidad de la documentación

Atributo	Indicador	Medida Derivada
Legibilidad del Diseño	Nombres significativos	Porcentaje de EF con nombres largos (> 20 carac.)
		Longitud media del nombre de los EF
Comprensibilidad de Interfaces	Comprensibilidad de EF	*Porcentaje de EF usados sin error*
Facilidad de Aprendizaje	Tiempo para Usar	*Tiempo Medio para usar correctamente el componente*
	Tiempo para Maestría	*Tiempo medio para ser un experto en el componente*
Complejidad de API	Densidad de Interfaces	Ratio de Interfaces por Interfaz Requerida
		Ratio de Valores de Retorno por Método
		Ratio de Argumentos de Métodos por Método
		Porcentaje de Métodos sin Argumentos
		Ratio de Métodos por Interfaz
		Ratio de Constructores por Clase
		Ratio de Constructores por Método
Configurabilidad	Ratio de Configurabilidad	Ratio de Parámetros Configurables por Interfaz
		Ratio de Parámetros Configurables por Método

Cuadro 3: Medidas relativas a la complejidad del diseño.

Medida Derivada	Usab_D	Usab_I	Usab_O	Comp_O	Apren_O	Oper_O
Porcentaje de EF descritos en los Manuales	0.09	0.62	0.54	-0.13	0.62	0.59
Porcentaje de Interfaces descritos en los Manuales	-0.21	0.31	0.55	0.11	0.60	0.57
Porcentaje de Métodos descritos en los Manuales	0.06	0.58	0.39	-0.31	0.47	0.45
Porcentaje de Parámetros Configurables descritos en los Manuales	0.39	0.83	0.78	0.19	0.84	0.82
Diferencia entre Versiones del Componente	0.58	0.93	0.87	0.34	0.91	0.90
Ratio de ficheros HTML de los Manuales por EF	0.63	0.63	0.92	**0.95**	0.88	0.89
Ratio de figuras por palabra	-0.57	-0.47	0.05	0.28	0.03	0.02
Ratio de palabras por EF	0.64	0.71	**0.96**	0.90	0.94	**0.95**
Ratio de palabras por Interfaz	0.75	0.80	**0.97**	0.88	**0.95**	**0.96**
Ratio de palabras por Método	0.66	0.69	**0.95**	0.92	0.92	0.93
Ratio de palabras por Parámetro Configurable	-0.06	0.44	0.66	0.21	0.71	0.68
Porcentaje de EF con nombres largos (> 20 carac.)	-0.17	0.39	0.25	-0.42	0.34	0.31
Ratio de Interfaces por Interfaz Requerida	-0.75	-0.28	-0.30	-0.72	-0.22	-0.26
Ratio de Valores de Retorno por Método	-0.29	-0.71	-0.87	-0.43	-0.91	-0.89
Ratio de Argumentos de Métodos por Método	-0.65	-0.94	-0.68	-0.11	-0.73	-0.72
Porcentaje de Métodos sin Argumentos	-0.09	0.43	0.22	-0.47	0.30	0.27
Ratio de Parámetros Configurables por Interfaz	0.70	0.20	0.00	0.34	-0.06	-0.03
Ratio de Parámetros Configurables por Método	0.85	0.45	0.47	0.79	0.40	0.44
Ratio de Métodos por Interfaz	0.44	0.20	-0.30	-0.35	-0.29	-0.28
Ratio de Constructores por Clase	0.31	0.78	0.62	-0.05	0.69	0.67
Ratio de Constructores por Método	-0.43	-0.10	0.36	0.28	0.37	0.35

Cuadro 4: Coeficientes de correlación (R) de las medidas derivadas.

Base Medida	Usab_D	Usab_I	Usab_O	Comp_O	Apren_O	Oper_O
Número de palabras en los Manuales	0.42	0.82	0.89	0.39	0.93	0.91
Ficheros HTML de los Manuales	0.56	0.67	**0.96**	0.88	0.93	0.94
Número de Interfaces	-0.91	-0.53	-0.37	-0.58	-0.32	-0.35
Número de Métodos	-0.45	-0.03	-0.32	-0.86	-0.23	-0.26
Número de Parámetros Configurables	0.66	0.25	-0.11	0.06	-0.14	-0.11
Número de Constructores	-0.69	-0.18	-0.17	-0.63	-0.10	-0.13
Número de Argumentos de Métodos	-0.87	-0.61	-0.72	-0.91	-0.66	-0.69
Número de Valores de Retorno	-0.52	-0.19	-0.51	**-0.95**	-0.43	-0.45
Número de EF	-0.43	-0.05	-0.37	-0.89	-0.28	-0.31
Número de Interfaces en los Manuales	-0.84	-0.40	-0.23	-0.51	-0.18	-0.21
Número de Métodos en los Manuales	-0.27	0.20	-0.10	-0.73	-0.01	-0.04
Número de Parámetros Configurables en los Manuales	0.89	0.94	0.63	0.28	0.65	0.66
Número de EF en los Manuales	-0.26	0.22	-0.06	-0.70	0.03	0.00
Número de Figuras en los Manuales	0.03	0.12	0.62	0.76	0.59	0.59
Longitud media del nombre de los EF	-0.22	0.26	-0.04	-0.68	0.05	0.02
Número de EF con nombres largos (>20 carac.)	-0.24	0.26	0.00	-0.65	0.09	0.06
Número de Métodos sin Valor de Retorno	-0.18	0.37	0.19	-0.48	0.28	0.25
Número de Métodos sin Argumentos	-0.37	0.08	-0.20	-0.79	-0.11	-0.14
Número de Métodos y Constructores sin Argumentos	-0.39	0.07	-0.20	-0.79	-0.11	-0.14

Cuadro 5: Coeficientes de correlación (R) para las medidas base.

La evaluación de estas cuestiones objetivas se realizó teniendo en cuenta el tiempo que se tardaba en encontrar la respuesta y en la corrección de la propia respuesta. Un valor numérico entre -3 y +3 se le asignaba a cada una de las respuestas.

Un total de 68 sujetos participaron en los experimentos de valoración de la Usabilidad de la muestra de componentes. El número de sujetos en cada experimento varió entre 9 y 18 personas.

B. Resultado (directo) de los experimentos

El resultado de estos experimentos se resume en seis variables (cuyos valores están en el rango -3 a +3) para los componentes muestreados: Usabilidad Percibida Directa (Usab_D), Usabilidad Percibida Indirecta (Usab_I), Comprensibilidad Objetiva (Comp_O), Aprendibilidad Objetiva (Apren_O), Operabilidad Objetiva (Oper_O) y Usabilidad Objetiva (Usab_O). Las dos primeras corresponden a la Usabilidad percibida, medida directa e indirectamente; las siguientes tres se obtienen como resultado de la valoración de las preguntas objetivas en los últimos experimentos; y la Usabilidad Objetiva es el promedio de las otras tres variables objetivas.

Hemos encontrado una correlación (aunque no estadísticamente relevante) entre la Usabilidad percibida y la objetiva. Curiosamente, hemos visto que los valores de la usabilidad son similares para valores altos (≥ 1,3) pero difieren más cuanto peor es la Usabilidad percibida.

Una vez cuantificada la información sobre la Usabilidad de los componentes muestreados, el siguiente paso es medir los componentes usando las medidas definidas y buscar correlaciones entre los resultados de la medición obtenidos y los datos de Usabilidad de los cuestionarios. Una fuerte correlación (p.e., R > 0; 97) significaría que esa medida explica bien los datos de Usabilidad y, por tanto, demostraría que esa medida es representativa y válida para medir la correspondiente subcaracterística.

Los Cuadros 4 y 5 muestran, respectivamente, las matrices de correlación obtenidas para nuestras medidas base y derivadas. Observándolos podemos sacar algunas conclusiones interesantes:.

- El número de palabras por interfaz (análogamente, el número de palabras por elemento funcional (EF) o por método) explica bien la Usabilidad del componente (R^2 = 0,95). Este resultado es consistente con la idea intuitiva que la Usabilidad de un componente es mejor si su manual contiene abundante información sobre sus interfaces, operaciones y otros EF.
- La Comprensibilidad (Comp_O) se puede explicar adecuadamente (R^2 = 0,90) mediante el número de ficheros HTML por EF del manual. Básicamente, esto significa que la estructura, organización y navegabilidad de la páginas electrónicas del manual (cada página se almacena como un fichero HTML) influyen directamente sobre la Comprensibilidad que el usuario tiene del componente. En concreto, para mejorar la Comprensibilidad del componente, las páginas electrónicas (pantallas) no deben ser muy largas ni deben contener excesiva información.
- Muy pocas medidas base son representativas. Básicamente, el tamaño del manual (medido en numero de palabras o de archivos) tiene alguna influencia sobre la Usabilidad (R^2 = 0;92). Por otra parte, el número de Valores de Retorno influye sobre la Comprensibilidad (R = -0;95;R^2 = 0;90), lo cual parece indicar que tener muchos métodos que no devuelven ningún valor produce componentes más difíciles de comprender.
- Con un menor grado de influencia, el número de interfaces y de argumentos de los métodos influyen en la Usabilidad directamente percibida (mejor cuanto menos clases y argumentos) y que la ratio de argumentos por método tiene un impacto negativo sobre la Usabilidad percibida indirectamente.

También aparecen algunos resultados inesperados:

- Por ejemplo, el número de figuras y tablas en el manual no parece tener una influencia directa sobre la Usabilidad percibida de los componentes. Esto sugiere que los usuarios (técnicos) no aprecian los dibujos y tablas en los manuales cuando evalúan subjetivamente la Usabilidad (aunque este medida sí es influyente en la evaluación objetiva de la Comprensibilidad, como veremos más adelante).
- Originalmente, los autores pensamos que la completitud de los manuales iba a tener una importante influencia en la Usabilidad. Sin embargo, el porcentaje de clases, métodos y EF que aparecen descritos en los manuales no parece tener una fuerte influencia sobre ninguno de los valores de la Usabilidad (aún cuando alguno de los manuales solo describen el 50% de las clases y métodos públicos del componente.)
- Por último, pensábamos que la longitud de los nombres de las clases, métodos, campos, etc., podría influir en la Aprendibilidad y la Comprensibilidad del componente. Sin embargo, estas impresiones no han sido corroboradas por los datos obtenidos.

Una de las conclusiones más interesantes que se puede sacar de las tablas de correlaciones es que no hay una medida individual que proporcione una explicación realmente buena ($R^2 \approx 1$) de la Comprensibilidad, Aprendibilidad u Operabilidad de un componente, ni tampoco de la Usabilidad globalmente. Sólo hay una medida que por sí misma explica mas de un 95% de una característica de calidad (la ratio de palabras por clase, con R = 0,97; R^2 = 0,95). El resto de las medidas están por debajo de esos valores.

C. Análisis de Regresión Lineal

En este punto debemos recordar lo expuesto en la introducción del trabajo como un defecto común en la mayoría de las propuestas de medición software: las medidas definidas por la mayoría de los autores y estándares internacionales aparecen generalmente asignadas a una única subcaracterística de calidad, aunque en teoría podrían evaluar más de una característica. De hecho, no creemos que exista una única relación directa entre una medida y una subcaracterística, si no

que hay diferentes grados de relación entre cada medida y cada subcaracterística.

Nótese también que estos resultados son totalmente consistentes con las definiciones de las tres subcaracterísticas de Usabilidad y, además, tienen sentido según nuestras

Subcaracterística	Medidas de las que depende		R^2	Relación
Comprensibilidad (Comp_O)	Ratio Ficheros HTML por EF ($Fich$)	Porc. Métodos sin Argumentos ($MsinA$)	0,99996	$Comp\_O = 0{,}2797Fich - 1{,}0737\overline{M}sinA + 1{,}1984$
Aprendibilidad (Apren_O)	Kilo-Palabras por Interfaz (PpI)	Valores de Retorno por Método ($VRpM$)	0,98891	$Apren\_O = 0{,}145PpI - 3{,}362\overline{V}RpM + 2{,}636$
Operabilidad (Oper_O)	Param. Config. por Método ($PCpM$)	Valores de Retorno por Método ($VRpM$)	0,99001	$Oper\_O = 2{,}642PCpM - 8{,}180\overline{V}RpM + 6{,}165$

Cuadro 6. Explicación de las subcaracterísticas por parte de las medidas.

Partiendo de esa premisa y usando los datos obtenidos en nuestros experimentos y las medidas observadas en los componentes, hemos utilizado un análisis de regresión lineal para buscar combinaciones de medidas que suministren mejores explicaciones de las tres subcaracterística. Nuestros resultados son buenos: todas las subcaracterísticas de Usabilidad pueden ser explicadas correctamente mediante la combinaciones lineales de dos medidas, obteniéndose valores para R^2 cercanos al 0,99.

Estas combinaciones se muestran en el Cuadro 6 y, entre otras cosas, proporciona una información muy interesante sobre la existencia de relaciones entre los atributos de los componentes y el Modelo de Calidad, es decir, la conexión entre la Calidad de la Documentación y la Complejidad del Diseño por un lado y, por el otro, la Comprensibilidad, Aprendibilidad y Operabilidad de un componente software:

- La Comprensibilidad tiene una fuerte dependencia tanto de la Ratio de ficheros HTML por EF como del Porcentaje de métodos sin argumentos. Esto significa que tanto la Calidad de los Manuales como la Complejidad del Diseño influyen en la Comprensibilidad del componente.
- La Aprendibilidad depende tanto de la Ratio de palabras por Interfaz (o palabras por clase, en el caso de componentes Java) como del Porcentaje de métodos sin valores de retorno. Ambas, la Calidad de los Manuales y la Complejidad del Diseño influyen en esta subcaracterística.
- Finalmente, la Operabilidad depende de la Ratio de parámetros configurables por método (o campos por clase, en el caso de componentes Java) y del Porcentaje de métodos sin valores de retorno. En este caso, ambas medidas están relacionadas con la Complejidad del Diseño (y por tanto el concepto mensurable aparentemente con más influencia sobre la Operabilidad).

Es destacable que algunas ecuaciones del Cuadro 6 no contienen medidas que sean muy relevantes por sí solas. Esto significa que una combinación de medidas menos representativas puede llegar a ser más representativa que las propias medidas individuales y que cualquier otra medida individual.

intuiciones. Por ejemplo, la controlabilidad de un componente y su adecuación para ser particularizado fueron dos de los requisitos de la Operabilidad (Sección 3). Para un componente software, normalmente esto se consigue mediante los parámetros configurables. Precisamente, lo que hemos encontrado es que la Ratio de de parámetros configurables por método tiene una fuerte influencia sobre esta subcaracterística (cuando se combina con la densidad de valores de retorno).

VI. Conclusiones

Este trabajo ha presentado un conjunto de medidas base y derivadas que se pueden utilizar para evaluar la Usabilidad de los componentes software. Además, se ha mostrado cómo validar estas medidas, calculando cómo influyen individualmente sobre las subcaracterísticas de la Usabilidad y cuáles son las combinaciones de medidas más representativas para evaluara estas subcaracterísticas.

Se ha visto como la combinación apropiada de medidas puede evaluar mejor la Usabilidad de un componente que cualquier medida individual. Básicamente, esto ocurre porque las características de calidad no dependen de un concepto mensurable determinado, sino de una combinación de ellos.

Nuestro planes son ahora realizar más experimentos con más componentes (p.e., con otros dominios de aplicación) y en el entorno de la industria, con el objeto de recopilar más datos que nos ayuden a corroborar nuestros resultados y a refinar nuestras ecuaciones. Basándonos en estas ecuaciones, tenemos pensado trabajar en modelos de análisis que nos puedan ayudar a definir los indicadores de calidad apropiados para la Usabilidad. Finalmente, estamos integrando las herramientas que hemos desarrollado para automatizar las medidas de los componentes (es decir, los programas que analizan los manuales y los programas que "interrogan" los componentes usando reflexión) de tal forma que en corto plazo podamos ofrecer un servicio de evaluación de componentes para la industria software que ayude en la selección de los componentes que se adecúan mejor a sus aplicaciones con un esfuerzo menor y con mayor precisión.

REFERENCIAS

[1] ISO/IEC 14598:2001. Software Engineering – Product Evaluation. June 2001.

[2] ISO/IEC 9126-1:2001. Software Engineering – Product Quality – Part 1:Quality model. June 2001.

[3] ISO/IEC FDIS 25000:2005. Software Engineering – Software Product Quality Requirements and Evaluation (SQuaRE) – Guide to SQuaRE. Jan 2005.

[4] Manuel F. Bertoa and Antonio Vallecillo. Quality attributes for COTS components. I+D Computación, 1(2):128–144, Nov 2002.

[5] Pere Botella, Xavi Burgués, et al. Using quality models for assessing COTS selection. In Proc. of WER'02, pp. 263–277, Nov 2002.

[6] Allan W. Brown and Kurt Wallnau. The current state of CBSE. IEEE Software, 15(5):37–46, Sep-Oct 1999.

[7] S. Sedigh Ali, A. Ghafoor y R.A. Paul. Software engineering metrics for COTS based systems. IEEE Computer, 34(5):44–50, 2001.

[8] R. Simao and A. Belchior. Quality characteristics for software components: Hierarchy and quality guides. In Component-Based Software Quality: Methods and Techniques, LNCS 2963, pp. 188–211, 2003.

[9] H. Washizaki, H. Yamamoto y Y. Fukazawa. A metrics suite for measuring reusability of software components. In Proc. of METRICS' 03, pp. 221–225, Sep 2003.

[10] M. F. Bertoa, J. M. Troya y A. Vallecillo. A survey on the quality information provided by software component vendors. In Proc. of QAOOSE 2003, pp. 25–30, July 2003.

[11] C. Szyperski. Component Software. Beyond Object-Oriented Programming. Addison-Wesley, 2 ed, 2002.

[12] L.E. Sweeney, E.V. Kortright y R.J. Buckley. Developing an RM-ODP based architecture for the defense integrated military human resources system. In Proc. of WOODPECKER' 01, pp. 110–124, July 2001.

[13] V. Seppanen and N. Helander. Original software component manufacturing: Survey of the state of the practice. In Proc. of the 27th Euromicro Conference, pp. 138–145, Sep 2001. IEEE CS Press.

[14] ISO/IEC 15939:2002. Software engineering – Software measurement process. Jan 2002.

[15] Barry Boehm, J. R. Brown, J.R. Kaspar, et al. Characteristics of Software Quality. North Holland, 1978.

BIOGRAFÍAS

Manuel F. Bertoa es Profesor Colaborador del Departamento de Lenguajes y Ciencias de la Computación de la Universidad de Málaga, España. Su investigación se centra en el Desarrollo de software basado en componentes y en la Medición del software. Es Ingeniero de Telecomunicación por la Universidad Politécnica de Madrid.

.

Antonio Vallecillo es Profesor Titular de Universidad del Departamento de Lenguajes y Ciencias de la Computación de la Universidad de Málaga, España. Sus principales áreas de investigación se centran en el desarrollo de software dirigido por modelos, Software basado en Componente, Procesamiento distribuido abierto, y el uso industrial de los métodos formales. Obtuvo su título de Licenciado en Matemáticas y de Doctor en Informática en la Universidad de Málaga. Es el representante de la Universidad de Málaga en ISO y en OMG, asimismo, es miembro de ACM, IEEE e IEEE Computer Society.