¿Cuándo un software es suficientemente bueno?

Linda María Fernández Olvera
UPIICSA-IPN
Ingeniería de Diseño
lfernandezo1500@alumno.ipn.mx

Abstract – ¿Cómo se define que un software es suficientemente bueno? En el presente ensayo se definirán conceptos que facilitarán el entendimiento del concepto de calidad y todos los factores que influyen en ella. Además de los costos que traen consigo la falta de calidad.

Índice de Términos – calidad, riesgos, usabilidad, confiabilidad, seguridad, eficiencia.

I. INTRODUCCIÓN

La calidad de software comienza a considerarse en la industria cuando las corporaciones reconocieron que se desperdiciaban miles de millones de dólares en software que no cumplía con los requerimientos ni la funcionalidad que se había acordado. Tanto el gobierno como la industria comenzaron a darle más importancia a la posibilidad de que fallas en el software afecten la infraestructura y provoquen pérdidas económicas.

El código defectuoso es responsable del tiempo que están fuera de servicio los sistemas y produce pérdidas en miles de millones de dólares, además de pérdidas en reparación, productividad y deuda técnica.

Si un equipo de software hace énfasis en la calidad de todas las actividades de la ingeniería de software, se reduce el número de repeticiones que deben hacerse. Esto da como resultado menores costos y, lo que es más importante, mejora el tiempo de llegada al mercado.

II. DESARROLLO

"La calidad es un concepto de facetas múltiples"[1] que se describe desde cinco puntos de vista diferentes:

- Punto de vista trascendental: la calidad es algo que se reconoce de inmediato, pero que no es posible definir explícitamente.
- Punto de vista del usuario: concibe la calidad en términos de las metas específicas del usuario final. Si un producto las satisface, tiene calidad.
- Punto de vista del fabricante la define en términos de las especificaciones originales del producto.
- Punto de vista del producto sugiere que la calidad tiene que ver con las características inherentes (funciones y características) de un producto.
- Punto de vista basado en el valor: mide la calidad de acuerdo con lo que un cliente está dispuesto a pagar por un producto. En realidad, la calidad incluye todo esto y más.

La calidad del diseño se refiere a las características que los diseñadores especifican para un producto. En el desarrollo del software, la calidad del diseño incluye el grado en el que el diseño cumple las funciones y características especificadas en el modelo de requerimientos.

La calidad de la conformidad se centra en el grado en el que la implementación se apega al diseño y en el que el sistema resultante cumple sus metas de requerimientos y desempeño.

Si un producto de software beneficia mucho a los usuarios finales, éstos se mostrarán dispuestos a tolerar problemas ocasionales de confiabilidad o desempeño.

A. Calidad del software

Entonces, la calidad del software se define como el proceso eficaz de software que se aplica de manera que crea un producto útil que proporciona valor medible a quienes lo producen y a quienes lo utilizan.

Un proceso eficaz de software establece la infraestructura que da apoyo a cualquier esfuerzo de elaboración de un producto de software de alta calidad. Los aspectos de administración del proceso generan las verificaciones y equilibrios que ayudan a evitar que el proyecto caiga en el caos, contribuyente clave de la mala calidad.

Un producto útil entrega contenido, funciones y características que el usuario final desea; sin embargo, de igual importancia es que entrega estos activos en forma confiable y libre de errores.

Al agregar valor para el productor y para el usuario de un producto, el software de alta calidad proporciona beneficios a la organización que lo produce y a la comunidad de usuarios

[1] Garvin, D., "What Does 'Product Quality' Really Mean?" Sloan Management Review, otoño 1984, pp. 25-45.

finales. La organización que elabora el software obtiene valor agregado porque el software de alta calidad requiere un menor esfuerzo de mantenimiento, menos errores que corregir y poca asistencia al cliente.

Dimensiones de la calidad de Garvin

David Garvin menciona que la calidad se toma en cuenta al adoptar un punto de vista multidimensional a través de 8 dimensiones:

Calidad del desempeño. ¿El software entrega todo el contenido, las funciones y las características especificadas como parte del modelo de requerimientos, de manera que da valor al usuario final?

Calidad de las características. ¿El software tiene características que sorprenden y agradan la primera vez que lo emplean los usuarios finales?

Confiabilidad. ¿El software proporciona todas las características y capacidades sin fallar? ¿Está disponible cuando se necesita? ¿Entrega funcionalidad libre de errores?

Conformidad. ¿El software concuerda con los estándares locales y externos que son relevantes para la aplicación? ¿Concuerda con el diseño de facto y las convenciones de código?

Durabilidad. ¿El software puede recibir mantenimiento (cambiar) o corregirse (depurarse) sin la generación inadvertida de eventos colaterales?

Servicio. ¿Existe la posibilidad de que el software reciba mantenimiento (cambios) o correcciones (depuración) en un periodo de tiempo aceptablemente breve?

Estética. No hay duda de que todos tenemos una visión diferente y muy subjetiva de lo que es estético.

Percepción. En ciertas situaciones, existen prejuicios que influirán en la percepción de la calidad por parte del usuario.

Muchas de estas dimensiones (aunque no todas) sólo pueden considerarse de manera subjetiva.

Factores de la calidad de McCall

McCall, Richards y Walters[2] proponen una clasificación útil de los factores que afectan la calidad del software y se centran en tres aspectos importantes del producto de software: sus características operativas, su capacidad de ser modificado y su adaptabilidad a nuevos ambientes.

- Corrección. Grado en el que un programa satisface sus especificaciones y en el que cumple con los objetivos de la misión del cliente.
- Confiabilidad. Grado en el que se espera que un programa cumpla con su función y con la precisión requerida.
- Eficiencia. Cantidad de recursos de cómputo y de código requeridos por un programa para llevar a cabo su función.
- Integridad. Grado en el que es posible controlar el acceso de personas no autorizadas al software o a los datos.
- Usabilidad. Esfuerzo que se requiere para aprender, operar, preparar las entradas e interpretar las salidas de un programa.
- Facilidad de recibir mantenimiento. Esfuerzo requerido para detectar y corregir un error en un programa.
- Flexibilidad. Esfuerzo necesario para modificar un programa que ya opera.
- Susceptibilidad de someterse a pruebas. Esfuerzo que se requiere para probar un programa a fin de garantizar que realiza la función que se pretende.
- Portabilidad. Esfuerzo que se necesita para transferir el programa de un ambiente de sistema de hardware o software a otro.
- Reusabilidad. Grado en el que un programa (o partes de uno) pueden volverse a utilizar en otras aplicaciones.
- Interoperabilidad. Esfuerzo requerido para acoplar un sistema con otro.

Factores de la calidad ISO 9126

El estándar ISO 9126 se desarrolló con la intención de identificar los atributos clave del software de cómputo. Este sistema identifica seis atributos clave de la calidad:

- Funcionalidad. Grado en el que el software satisface las necesidades planteadas según las establecen los atributos siguientes: adaptabilidad, exactitud, interoperabilidad, cumplimiento y seguridad.
- Confiabilidad. Cantidad de tiempo que el software se encuentra disponible para su uso, según lo indican los siguientes atributos: madurez, tolerancia a fallas y recuperación.
- Usabilidad. Grado en el que el software es fácil de usar, según lo indican los siguientes subatributos: entendible, aprendible y operable.
- Eficiencia. Grado en el que el software emplea óptimamente los recursos del sistema, según lo indican los subatributos siguientes: comportamiento del tiempo y de los recursos.
- Facilidad de recibir mantenimiento. Facilidad con la que pueden efectuarse reparaciones al software,

[2] McCall, J., P. Richards y G. Walters, "Factors in Software Quality", tres volúmenes, NTIS AD-A049- 014, 015, 055, noviembre 1977.

según lo indican los atributos que siguen: analizable, cambiable, estable, susceptible de someterse a pruebas.

- Portabilidad. Facilidad con la que el software puede llevarse de un ambiente a otro según lo indican los siguientes atributos: adaptable, instalable, conformidad y sustituible.

B. El dilema de la calidad del software

El software suficientemente bueno contiene las funciones y características de alta calidad que desean los usuarios, pero al mismo tiempo tiene otras más oscuras y especializadas que contienen errores conocidos.

Un software de mala calidad nadie lo comprará y el software perfecto tomará tanto tiempo en realizarse y será tan caro de producir que quedará fuera del modelo de negocio.

E l costo de la calidad incluye todos los costos en los que se incurre al buscar la calidad o al realizar actividades relacionadas con ella y los costos posteriores de la falta de calidad. Puede dividirse en los costos que están asociados con la prevención, la evaluación y la falla.

Los costos de prevención incluyen lo siguiente:

1) el costo de las actividades de administración requeridas para planear y coordinar todas las actividades de control y aseguramiento de la calidad
2) el costo de las actividades técnicas agregadas para desarrollar modelos completos de los requerimientos y del diseño
3) los costos de planear las pruebas
4) el costo de toda la capacitación asociada con estas actividades.

Los costos de evaluación incluyen las actividades de investigación de la condición del producto la “primera vez” que pasa por cada proceso.

Los costos de falla son aquellos que se eliminarían si no hubiera errores antes o después de enviar el producto a los consumidores. Los costos de falla se subdividen en internos y externos.

Riesgos. La implicación es que el software de mala calidad aumenta los riesgos tanto para el desarrollador como para el usuario final.

Negligencia. En la mayor parte de los casos, el cliente afirma que el desarrollador ha sido negligente (en cuanto a la manera en la que aplicó las prácticas del software), por lo que no merece el pago. Es frecuente que el desarrollador diga que el cliente ha cambiado repetidamente sus requerimientos y trastornado de diversas maneras los acuerdos para el trabajo. En cualquier caso, es la calidad del sistema lo que está en entredicho.

Calidad y seguridad. La seguridad del software se relaciona por completo con la calidad. Debe pensarse en seguridad, confiabilidad, disponibilidad y dependencia, en la fase inicial, en la de diseño, en la de arquitectura, pruebas y codificación, durante todo el ciclo de vida del software.

El efecto de las acciones de la administración

Es frecuente que la calidad del software reciba influencia tanto de las decisiones administrativas como de las tecnológicas. Incluso las mejores prácticas de la ingeniería de software pueden ser arruinadas por malas decisiones gerenciales y por acciones cuestionables de la administración del proyecto.

Control de calidad

El control de calidad incluye un conjunto de acciones de ingeniería de software que ayudan a asegurar que todo producto del trabajo cumpla sus metas de calidad. Los modelos se revisan para garantizar que están completos y que son consistentes. El código se inspecciona con objeto de descubrir y corregir errores antes de que comiencen las pruebas. Se aplica una serie de etapas de prueba para detectar los errores en procesamiento lógico, manipulación de datos y comunicación con la interfaz. La combinación de mediciones con retroalimentación permite que el equipo del software sintonice el proceso cuando cualquiera de estos productos del trabajo falla en el cumplimiento de las metas de calidad.

Aseguramiento de calidad

El aseguramiento de la calidad establece la infraestructura de apoyo a los métodos sólidos de la ingeniería de software, la administración racional de proyectos y las acciones de control de calidad, todo de importancia crucial si se trata de elaborar software de alta calidad. Además, el aseguramiento de la calidad consiste en un conjunto de funciones de auditoría y reportes para evaluar la eficacia y completitud de las acciones de control de calidad.

III. CONCLUSIÓN

La calidad del software no sólo se ve. Es el resultado de la buena administración del proyecto y de una correcta práctica de la ingeniería de software.

Si espera construir software de alta calidad, debe entender el problema que se quiere resolver. También debe ser capaz de crear un diseño que esté de acuerdo con el problema y que al mismo tiempo tenga características que lleven al software a las dimensiones y factores de calidad.

La calidad tiene un costo que puede estudiarse en términos de prevención, evaluación y falla. Los costos de prevención incluyen todas las acciones de la ingeniería de software diseñadas para prevenir los defectos. Los costos de evaluación están asociados con aquellas acciones que evalúan los productos del trabajo de software para determinar su calidad. Los costos de falla incluyen el precio interno de fallar y los efectos externos que precipitan la mala calidad.

REFERENCIAS

[1] Roger S. Pressman, "Ingeniería del Software, un enfoque práctico" 7ª ed.Connecticut University, New York: McGraw-Hill, 2010, pp. 338 – 351.