MODELOS INICIALES DE CALIDAD

Se podría decir que la calida , al igual que la belleza, está en el ojo de quien lo mira sin embargo, desde el punto de vista de medición, se debe tener una definición precisa en términos de atributos del software que sean del interés al usuario. En general, éstos son atributos externos, sin embargo, muchas propuestas miden y analizan atributos internos porque los consideran predictores de aquellos externos.
Los atributos internos tienen dos ventajas:

- Están disponibles para medición más temprano.
- Son más fáciles de medir.

Desde el principio de la ingeniería de software, se observó que la calidad está compuesta por una composición de muchas características. Un modelo de calidad describe entonces estas características y sus relaciones, muchos modelos hacen difusa la distinción entre atributos internos y externos, lo que dificulta la comprensión del concepto de calidad.
Los modelos que se presentarán a continuación son los que han ganado mayor popularidad en la comunidad, pero no tienen sustento científico. Extrayendo los factores comunes a todos ellos, es posible derivar modelos propios adaptados a usos específicos.

MODELO DE McCall

El modelo de McCall fue el primero en ser presentado en 1977, y se originó motivado por US Air Force y DoD. Este modelo se focaliza en el producto final, identificando atributos claves desde el punto de vista del usuario.

Estos atributos se denominan factores de calidad y son normalmente atributos externos, pero también se incluyen algunos atributos posiblemente internos.

los factores de calidad son demasiados abstractos para ser medidos directamente, por lo que por cada uno de ellos se introduce atributos de bajo nivel denominados criterios de calidad.

Algunos criterios de calidad son atributos internos, reflejando la creencia de McCall que el atributo interno tiene un efecto directo en el atributo externo correspondiente.

un nivel más de descomposición es necesario, mapeando cada criterio de calidad en un conjunto de métricas de calidad que son atributos (tanto del producto como del proceso) de muy bajo nivel, medibles directamente.

Este modelo calidad propone tres perspectivas para agrupar los factores de calidad:

- Revisión del producto habilidad para ser cambiado. Incluye los siguientes factores de calidad:
    - Mantenibilidad esfuerzo requerido para localizar y corregir fallas.
    - Flexibilidad facilidad de realizar cambios.
    - Testeabilidad facilidad para realizar el testing, para asegurarse que el producto no tiene errores y cumple con la especificación.
- Transición del producto adaptabilidad al nuevo ambiente. Incluye los siguientes factores de calidad:
    - Portabilidad esfuerzo requerido para transferir entre distintos ambientes de operación.
    - Reusabilidad facilidad de reusar el software en diferentes contextos.
    - Interoperabilidad esfuerzo requerido para acoplar el producto con otros sistemas.

- Operación del producto características de operación. Incluye los siguientes factores:
    - Correctitud el grado en el que el producto cumple con su especificación.
    - Confiabilidad la habilidad del producto de responder ante situaciones no esperadas.
    - Eficiencia el uso de los recursos tales como tiempo de ejecución y memoria de ejecución.
    - Integridad protección del programa y sus datos de accesos no autorizados.
    - Usabilidad facilidad de operación del producto por parte de los usuarios.

El factor Mantenibilidad incluye los siguientes criterios:

- Consistencia
- Simplicidad
- Concisidad
- auto-descripción
- modularidad

La mantenibilidad está muy influenciada por el uso de buenas prácticas a lo largo de todo el ciclo de desarrollo. Algunas de estas buenas prácticas son:

- Seguir una metodología bien definida
- Usar buenas técnicas de diseño, tanto de procedimientos como de datos, para aumentar cohesión y reducir acoplamiento.
- Observar la documentación interna.
- Usar buenas prácticas de programación: nombres significativos, código legible, etc.

El factor Flexibilidad incluye los siguientes criterios:

- Expandibilidad
- Generalidad
- Auto-descripción
- Modularidad

El factor Testeabilidad incluye los siguientes criterios:

- Simplicidad
- Instrumentación

Dado su ubicación en tradicionales modelos de ciclo de vida de software, la facilidad de testing se define claramente como un criterio de calidad.
El testeo interactúa con otros criterios de calidad, por ejemplo correctitud y eficiencia, debe ser llevado a cabo siguiendo planes pre-definidos, con datos conocidos y cuyos resultados sean predeterminados.
La testeabilidad puede ser maximizada usando herramientas automáticas, buenas estrategias de cohesión y de diseño, y buenas prácticas de programación.
McCall definió originalmente métricas para testeabilidad consistentes en una matriz de com plejidad que involucra número y tamaño de módulos, tamaño de procedimientos, profundidad de anidamiento, número de errores por unidad de tiempo, etc.

El factor Portabilidad incluye los siguientes criterios:

- Auto-descripción
- Modularidad
- Independencia de la máquina
- Independencia del sistema operativo

Algunos autores (Sommerville) lo consideran parte de la Reusabilidad.
Se favorece mediante el seguimiento de estándares, tanto de procedimientos (X Windows) como de datos (XML)

El factor Reusabilidad incluye los siguientes criterios:

- Generalidad
- Modularidad
- Auto-descripción
- Independencia de la máquina
- Independencia del sistema operativo

Se puede favorecer la reusabilidad usando librerías de software, y técnicas de programación orientada a objetos.

Hay que tener en cuenta que el desarrollo de código reusable cuesta más tiempo y dinero.

Existe un factor económico difícil de medir: el costo de código reusable y la ganancia por reusar código ya desarrollado.

El factor Interoperabilidad incluye los siguientes criterios:

- Modularidad
- Interoperabilidad en comunicación
- Interoperabilidad en datos

La interoperabilidad está relacionada con la reusabilidad.
En la actualidad su importancia ha crecido debido al creciente interés de conectarse con sistemas legacy.
Se favorece mediante la adopción de estándares

El factor Correctitud incluye los siguientes criterios:

- Trazabilidad
- Completitud
- Consistencia

Correctitud es un factor muy difícil de identificar debido a la falta de terminología estándar.
Se lo pueden confundir con otros factores, tales como confiabilidad e integridad.
Para medirlo es necesario tener disponible una especificación formal de los requerimientos, cosa muy rara salvo en proyecto de alto presupuesto y sistemas críticos.
Las técnicas para verificarlo pueden ser: inspecciones de código, verificación matemática y analizadores estáticos de programas.

El factor Confiabilidad incluye los siguientes criterios:

- Tolerancia a errores
- Consistencia
- Simplicidad
- Exactitud

Combina la tolerancia tanto a errores de hardware como de software
Técnica de programación tales como tolerancia a las fallas, manejo de excepciones y programación defensiva ayudan.
Puede ser medido con medidas como:

a) Tiempo medio entre fallas
b) Tiempo medio antes de mantenimiento
c) Tiempo medio antes de recuperación
d) Probabilidad de falla

El factor Eficiencia incluye los siguientes criterios:

- Eficiencia en tiempo
- Eficiencia en espacio

Muchas técnicas favorecen este factor: el lenguaje de programación, el sistema operativo, optimización de algoritmos, normalización de datos.

El factor Integridad incluye los siguientes criterios:

- Control de acceso
- Auditoría de acceso

Involucra tanto evitar el acceso malintencionado, así como los daños causados por errores involuntarios de usuarios autorizados.

El factor Usabilidad incluye los siguientes criterios:

- Operabilidad
- Entrenamiento
- Comunicación
- Volumen de e/s
- Tasa de e/s

La usabilidad ha cambiado mucho desde la época de McCall incluye aspectos tales como adaptabilidad, aprendizaje, adecuación al contexto.
Algunos autores consideran por ejemplo que facilidad de aprendizaje es un factor de calidad independiente.
Se puede subdividir en:

a) Ergonomía general el equipo es adecuado para el uso previsto
b) Ergonomía de software estilos de diálogoes, metáforas, diseño de pantallas, etc.

Métricas De Calidad

La medición de cualquiera de estos factores está definida en este modelo en base a 41 métricas.
Para cada criterio existe una lista de condiciones que se deben cumplir en distintas etapas: requerimientos (R), diseño (D), implementación (I).
Se cuentan las condiciones que se satisfacen en cada una de las etapas, sobre el total posible, eso da una medida del criterio, que se pondera en partes iguales para medir el factor con los otros criterios asociados al factor.

Ejemplo:

Para medir el criterio completitud del factor correctitud McCall sugiere las siguientes condiciones:

a) Referencias no ambiguas [R,D,I]
b) Referencias a datos bien definidas, o externas [R,D,I]
c) Todas las funciones definidas son usadas [R,D,I]
d) Todas las condiciones y procesamientos están definidos para cada punto de decisión [R,D,I]
e) Todos los parámetros formales y actuales coinciden [D,I]
f) Todos los reportes de problemas han sido resueltos [R,D,I]
g) El diseño concuerda con los requerimientos [D]
h) El código concuerda con el diseño [I]

Entonces se cuentan la cantidad de sí en cada etapa, resultando en la métrica de completitud:

$$\left(\frac{\text{Si en R}}{5} + \frac{\text{Si en D}}{7} + \frac{\text{Si en I}}{7}\right) / 3$$

Luego la correctitud se mide como la media entre las medidas de sus criterios:

$$\frac{\text{(Completitud +Trazabilidad +Consistencia)}}{3}$$

MODELO DE BOEHM

Es el segundo modelo de calidad más conocido es el presentado por Barry Boehm en 1978.
Este modelo introduce características de alto nivel, características de nivel intermedio y características primitivas, cada una de las cuales contribuye al nivel general de calidad.

Características De Alto Nivel representan requerimientos generales de uso pueden ser:

- Utilidad per-se cuan (usable, confiable, eficiente) es el producto en sí mismo.
- Mantenibilidad cuan fácil es modificarlo, entenderlos y retestearlo.
- Utilidad general si puede seguir usándose si se cambia el ambiente

Características De Nivel Intermedio representan los factores de calidad de Boehm:

- Portabilidad (utilidad general)
- Confiabilidad (utilidad per-se)
- Eficiencia (utilidad per-se)
- Usabilidad (utilidad per-se)
- Testeabilidad (mantenibilidad)
- Facilidad de entendimiento (mantenibilidad)
- Modificabilidad o flexibilidad (mantenibilidad)

Caracteristicas Primitivas corresponde a características directamente asociadas a una o dos métricas de calidad:

- De portabilidad
    - independencia de dispositivos
    - auto-contención
- De confiabilidad
    - auto-contención
    - exactitud
    - completitud
    - consistencia
    - robustez/integridad
- De eficiencia
    - accesibilidad
    - eficiencia de uso de dispositivos
- De usabilidad
    - robustez/integridad
    - accesibilidad
    - comunicación

- De testeabilidad
    - comunicación
    - auto descripción
    - estructuración
- De entendibilidad
    - consistencia
    - estructuración
    - concisidad
    - legibilidad
- De modificabilidad
    - estructuración
    - aumentabilidad

Comparación de los Modelos de McCall-Boehm

Aunque parezcan similares, la diferencia está en que McCall focaliza en medidas precisas de alto nivel, mientras que Boehm presenta un rango más amplio de características primarias, la mantenibilidad está más desarrollada en Boehm.

CRITERIOS	McCall	BOEHM
Correctitud	+	+
Integridad	+	+
Eficiencia	+	+
Testeabilidad	+	
Flexibilidad	+	+

Portabilidad	+	+
Modificab.		+
Entendib.		+
Confiabilidad	+	+
Usabilidad	+	+
Mantenim.	+	+
Interoperab	+	
Reusabilidad	+	+
Claridad		+
Document.		+
Validez		+

Estos modelos tienen sus límites:

Es difícil que las características y subcaracterísticas sean siempre perfectamente independientes.
Falta una asociación explícita entre los modelos y el proceso de software, el cómo realizar software de calidad.
Las características son en general propiedades abstractas medibles mediante métricas. No siempre existe una relación perfectamente lineal entre los valores de las métricas y las características que deben estimar.

La Calidad En El Ciclo De Vida Del Software

El foco en la calidad cambia durante el ciclo de vida:

Al principio, durante la recopilación de requerimientos y análisis, la calidad es especificada por los requisitos del usuario, sobre todo desde el punto de vista externo.

En la fase de diseño e implementación, la calidad externa se traduce en un diseño técnico, confrontándose con el punto de vista de los desarrolladores sobre la calidad interna y complementándose con los requisitos implícitos que el software debe cumplir.

La calidad final (la del uso) debe ser apropiada para los usuarios y el contexto de uso.

No existe una calidad perfecta o absoluta. Existe solamente una calidad necesaria y suficiente para un contexto dado.