FACTORES DE ÉXITO EN PROYECTOS DE SOFTWARE

FACTOR	DEFINICIÓN	ESFUERZO
Patrocinio ejecutivo	Compromiso serio de DIRECTIVOS/EJECUTIVOS a proporcionar apoyo financiero y emocional. Animarán y ayudarán a llevar a buen término el proyecto.	15%
Madurez emocional	Conjunto de comportamientos básicos de cómo las personas trabajan juntas. En cualquier grupo, organización o empresa puede representar la suma de sus habilidades (BUEN MANEJO) o convertirse en el punto débil y la raíz del fracaso.	15%
Implicación de los usuarios	Los usuarios participan en el proceso de toma de decisiones y recopilación de información del proyecto. Esto incluye también las opiniones de los usuarios, la revisión de requisitos, la investigación básica, la creación de prototipos y otras herramientas de consenso.	15%
Optimización	Incorporación de mejoras en la eficacia empresarial. La optimización comienza con la gestión del alcance (del proyecto) en función del valor empresarial.	15%
Recursos cualificados	Son personas que entienden tanto el negocio como la tecnología.	10%
Arquitectura estándar	Conjunto coherente de prácticas, servicios y productos integrados para desarrollar, implantar y explotar aplicaciones informáticas.	8%
Proceso ágil	Significa que el equipo ágil y el propietario del producto son competentes en el proceso ágil.	7%
Ejecución modesta	Es tener un proceso con pocas partes móviles (presupuestos, plazos, recursos, objetivos y medidas), y que esas partes estén automatizadas y racionalizadas. Una ejecución modesta también significa utilizar las herramientas de gestión de proyectos con moderación y solo unas pocas funciones.	6%
Experiencia en gestión de proyectos	Es la aplicación de conocimientos, habilidades y técnicas a las actividades del proyecto con el fin de satisfacer o superar las expectativas de las partes interesadas y producir valor para la organización.	5%
Objetivos empresariales claros	Es la comprensión por parte de todos los interesados y participantes de la finalidad empresarial de la ejecución del proyecto. También podrían significar que el proyecto se ajusta a los objetivos y la estrategia de la organización.	4%

Esta es la recomendación de Standish Group sobre la cantidad de esfuerzo e inversión que debe considerarse para aumentar las probabilidades de éxito del proyecto.

FUENTES:

https://www.standishgroup.com/sample_research_files/CHAOSReport2015-Final.pdf

https://www.infoq.com/articles/standish-chaos-2015/

AHORA SÍ: ¿QUÉ ES CALIDAD DE SOFTWARE?

En el sentido más general se define como: **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.
- Las prácticas de ingeniería de software permiten al desarrollador analizar el problema y diseñar una solución sólida, ambas actividades críticas de la construcción de software de alta 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. Un producto útil siempre satisface los requerimientos establecidos en forma explícita por los participantes. Además, satisface requerimientos NO funcionales.

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 finales.

LOS SEIS PRINCIPIOS DE LA CALIDAD SOFTWARE

Watts Humphrey, el considerado padre de la calidad de los procesos software, estableció los 6 principios:

PRINCIPIO 1	Si un cliente no demanda calidad, probablemente no la conseguirá.
PRINCIPIO 2	Para obtener calidad de manera constante los desarrolladores deben gestionarla en su trabajo (o que la calidad en el software no la vamos a conseguir si el desarrollador no se involucra en ella).
PRINCIPIO 3	Para gestionar la calidad los desarrolladores deben medirla.
PRINCIPIO 4	La calidad de un producto la determina el proceso usado para desarrollarlo.
PRINCIPIO 5	Ya que las pruebas solucionan solo una fracción de los defectos, debes tener pruebas de calidad.
PRINCIPIO 6	La calidad solo la producen profesionales motivados orgullosos de su trabajo.

ESTÁNDARES Y MODELOS SOBRE CALIDAD DE SOFTWARE.

SOFTWARE: Dualidad Producto & Proceso.

Para el PRODUCTO:	Para el PROCESO:
Modelos de Calidad: McCall, Gilb, GQM, FURPS, Boehm Estándares de Calidad: ISO 25000 (SQuaRE)	Modelos: CMMI 2.0, PSP, TSP Estándares: ISO/IEC 12207, ISO/IEC 33000

Modelo. Propuesta, normalmente de carácter teórico-práctico, que tiene una serie de características que se consideran dignas de emular.

Estándar. Es una especificación o referencia predeterminada por alguna agencia, organismo acreditador o institución en el que se establece un patrón cuantitativo o cualitativo con valores que deben alcanzarse para demostrar el logro de un conjunto de objetivos.

FACTORES DE CALIDAD.

La calidad del software es una compleja mezcla de factores que variarán a través de diferentes aplicaciones y según los clientes que las pidan. En las siguientes secciones, se identifican los factores de la calidad del software y se describen las actividades humanas necesarias para conseguirlos.

Factores de calidad de McCall. Los factores que afectan a la calidad del software se pueden categorizar:

(1) Factores que se pueden medir directamente (por ejemplo, defectos por punto de función).
(2) Factores que se pueden medir sólo indirectamente (por ejemplo, facilidad de uso o de mantenimiento).

En todos los casos debe aparecer la medición. Debemos comparar el software (documentos, programas, datos) con una referencia y llegar a una conclusión sobre la calidad.

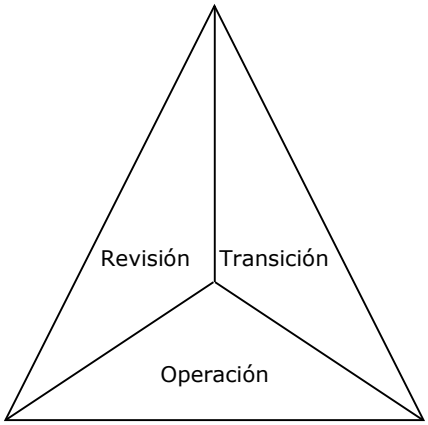
McCall y sus colegas propusieron una útil clasificación de factores que afectan a la calidad del software. Estos factores de calidad del software, mostrados en la Figura, se concentran en tres aspectos importantes de un producto software: sus características operativas, su capacidad de cambios y su adaptabilidad a nuevos entornos.

F. Facilidad de mantenimiento.	¿Puedo corregirlo?
G. Flexibilidad.	¿Puedo cambiarlo?
H. Facilidad de prueba.	¿Puedo probarlo?

Capacidad de soportar los cambios (Revisión)

I. Portabilidad.	¿Podré usarlo en otra máquina?
J. Reusabilidad.	¿Podré reutilizar alguna parte del software?
K. Interoperabilidad.	¿Podré hacerlo interactuar con otro sistema?

Adaptabilidad a nuevos entornos (Transición)



A. Corrección.	¿Hace lo que quiero?
B. Fiabilidad.	¿Lo hace de forma fiable todo el tiempo?
C. Eficiencia.	¿Se ejecutará en mi hardware lo mejor que pueda?
D. Seguridad (Integridad).	¿Es seguro?
E. Usabilidad (Facilidad de uso).	¿Está diseñado para ser usado?

Características operativas (Operación)

Refiriéndose a los factores anotados en la Figura, McCall proporciona las siguientes descripciones:

Corrección.	Hasta dónde satisface un programa su especificación y logra los objetivos propuestos por el cliente.
Fiabilidad.	Hasta dónde se puede esperar que un programa lleve a cabo su función con la exactitud requerida.
Eficiencia.	La cantidad de recursos informáticos y de código necesarios para que un programa realice su función.
Seguridad (Integridad).	Hasta dónde se puede controlar el acceso al software o a los datos por personas no autorizadas.
Usabilidad (Facilidad de uso).	El esfuerzo necesario para aprender a operar con el sistema, preparar los datos de entrada e interpretar las salidas (resultados) de un programa.
Facilidad de mantenimiento.	El esfuerzo necesario para localizar y corregir un error en un programa. (Esta es una definición muy limitada).
Flexibilidad.	El esfuerzo necesario para modificar un programa que ya está en funcionamiento.
Facilidad de prueba.	El esfuerzo necesario para probar un programa y asegurarse de que realiza correctamente su función.
Portabilidad.	El esfuerzo necesario para transferir el programa de un entorno hardware/software a otro entorno diferente.
Reusabilidad.	Hasta dónde se puede volver a emplear un programa (o partes de un programa) en otras aplicaciones, en relación al "empaquetamiento" y alcance de las funciones que realiza el programa.
Interoperabilidad.	El esfuerzo necesario para acoplar un sistema con otro.

Relaciones entre los factores del Modelo de McCall

Gillies ofrece una tabla basada en el trabajo de Perry:

	C	R	E	I	U	M	T	F	P	R
Correctness										
Reliability	+									
Efficiency										
Integrity			0							
Usability	+	+	0	+						
Maintainability	+	+	0		+					
Testability	+	+	0		+	+				
Flexibility	+	+	0		+	+	+			
Portability			0			+	+			
Reusability		0	0	0		+	+	+	+	
Interoperability			0	0					+	

Donde
"+" indica una relación directa
"0" indica una relación inversa

---------- ***FIN DEL DOCUMENTO***