Fundamentos y procesos en calidad de software

Breve descripción:

Este componente formativo aborda fundamentos teóricos y prácticos de la calidad de software, integrando Normas ISO/IEC 9126, 14598 y su transición a ISO/IEC 25000. Incluye métricas básicas, factores de calidad, ciclo de vida del software, aplicación de estándares (PSP, TSP, CMMI, GQM) y modelos de procesos para garantizar productos confiables, eficientes y alineados con las expectativas del cliente.

Mayo 2025

Tabla de contenido

Introducción

El componente formativo “Fundamentos y procesos en calidad de software” explora los principios esenciales para garantizar que el software cumpla con estándares de excelencia reconocidos globalmente. Incluyendo desde normas históricas como la ISO/IEC 9126, que define atributos clave como funcionalidad y usabilidad, hasta su evolución hacia la familia ISO/IEC 25000 (SQuaRE), este recorrido integra métricas prácticas y modelos de mejora continua. El objetivo es comprender cómo transformar códigos en productos confiables, eficientes y alineados con las demandas actuales de la industria. A través de un enfoque teórico-práctico, se analizará la transición de Normas ISO/IEC 9126 y 14598 al marco SQuaRE y la aplicación de calidad en el ciclo de vida del software mediante modelos como PSP, TSP, CMMI y GQM. Se identifican herramientas para evitar fallos costosos y transformar desarrollos en soluciones escalables, utilizando mapas conceptuales, casos reales y metodologías que convierten la calidad en un hábito estratégico.

Video 1. *Fundamentos y procesos en calidad de software*

Enlace de reproducción del video

Síntesis del video: fundamentos y procesos en calidad de software
Estimado aprendiz, le damos la bienvenida al componente formativo titulado fundamentos y procesos en calidad de software, en este espacio se explorarán los conceptos esenciales que permiten garantizar la calidad en cada etapa del desarrollo de software, comprendiendo la importancia de los estándares y modelos que respaldan estos procesos. Conocerá sobre la evolución de las normas de calidad en el desarrollo de software, destacando la transición de los estándares ISO y 9,126 y 14,598 hacia la familia ISO y EC25,000 permitiendo establecer un marco más estructurado para la gestión de la calidad optimizando su aplicación en la industria y asegurando el cumplimiento; además se describirán los factores simétricas básicas que permiten evaluar la calidad del software considerando aspectos clave como funcionalidad, confiabilidad, usabilidad y eficiencia. Estos elementos son fundamentales para garantizar que un producto cumpla con las expectativas del usuario y los requisitos del negocio; así mismo se abordará la aplicación de la calidad en el desarrollo de software y su relación con el ciclo de vida del producto asegurando que la gestión de la calidad esté presente desde las primeras fases hasta su implementación y mantenimiento. Para ello se presentará modelos y estándares de apoyo como PSP, TSP, CMMI y GQM, los cuales ofrecen metodologías y estrategias para optimizar los procesos y mejorar la eficiencia en la producción de software.

Le invitamos a apropiarse y aplicar los conceptos y métodos disponibles para llevar a cabo la aplicación de la calidad de software en el proceso de desarrollo. Inscríbase ahora en www.sena Sofía plus.edu.co.

1. Aplicación de la calidad en el desarrollo

La calidad en el desarrollo de software es un eje fundamental para garantizar productos que cumplan con las expectativas funcionales, técnicas y de experiencia del usuario, se entiende como un conjunto de prácticas y métodos que aseguran que el producto final cumpla con los requerimientos, funcione de forma confiable y se adapte al contexto de uso.

En un entorno tecnológico en constante evolución, las normas internacionales actúan como guías para estandarizar procesos, métricas y criterios de evaluación. Este componente explora modelos como ISO/IEC 9126, ISO/IEC 14598 y la familia ISO 25000, que brindan marcos de referencia para medir, gestionar y mejorar la calidad del software desde su diseño hasta su mantenimiento.

1.1. Factores clave: análisis, pruebas, código limpio y refactorización

El proceso de aplicación de la calidad parte de la realización de un análisis riguroso que permite identificar requerimientos y posibles fallos en etapas tempranas. La ejecución de pruebas sistemáticas garantiza que el software cumpla con sus funciones, minimizando errores en producción. Asimismo, la elaboración de un código limpio, organizado, legible y documentado facilita la identificación y solución de errores, mientras que la refactorización consiste en la reestructuración del código sin alterar su funcionalidad para mejorar su mantenimiento y rendimiento. A continuación, se presenta un ejemplo de tabla que resume estos factores:

Tabla 1. Factores clave en la aplicación de calidad en el desarrollo de software

Factor	Descripción
Análisis	Estudio y comprensión de los requerimientos y del contexto del desarrollo.
Pruebas	Ejecución de casos de prueba para validar el funcionamiento correcto en diversas condiciones.
Código limpio	Redacción de código claro, estructurado, bien documentado para facilitar su comprensión y mantenimiento.
Refactorización	Reorganización del código para mejorar su calidad sin modificar la funcionalidad existente.

Fuente: SENA,2025.

1.2. Relación entre procesos, calidad interna/externa y contexto de uso

En la práctica, la calidad del software se evalúa a través de dos perspectivas principales: la calidad interna, que se centra en las características del código y su estructura (por ejemplo, la eficiencia y mantenibilidad), y la calidad externa, que considera la experiencia del usuario, el rendimiento y la seguridad en el entorno de uso. La integración de ambos enfoques en el proceso de desarrollo permite ajustar la aplicación de técnicas de revisión, auditoría y pruebas, garantizando que el producto se adecúe a las necesidades planteadas en su contexto operativo.

Calidad interna: mide atributos como modularidad o cobertura de pruebas.

Calidad externa: evalúa resultados tangibles, como tiempo de respuesta o tasa de errores.

Contexto de uso: considera factores como dispositivos móviles o conexiones lentas en zonas rurales.

Figura 1. Interrelación entre calidad interna, calidad externa y en uso

Fuente: SENA, 2025.

2. Ciclo de vida del software

El ciclo de vida del software comprende una serie de fases que estructuran el proceso de desarrollo, permitiendo una gestión ordenada y la aplicación de medidas de control y mejora continua.

2.1. Fases: requerimientos, diseño, codificación, pruebas, validación y mantenimiento

El proceso inicia con la definición de requerimientos, en donde se identifican las necesidades del cliente y se establecen los objetivos del producto. Posteriormente, en la fase de diseño se define la arquitectura y se planifica la estructura del software. La codificación consiste en la implementación del diseño a través del desarrollo del código, fase en la que se aplican buenas prácticas para garantizar un código limpio. Durante la etapa de pruebas se ejecutan evaluaciones para identificar errores y asegurar la funcionalidad. La validación permite comprobar que el producto cumple con los requerimientos establecidos, y el mantenimiento se encarga de las actualizaciones y correcciones posteriores a la entrega.

2.2. Herramientas de revisión y auditoría

Para asegurar la calidad durante cada fase, se emplean diversas herramientas de revisión y auditoría, tales como sistemas de control de versiones, software de análisis estático de código y listas de chequeo para la revisión de requisitos y documentación. Estas herramientas permiten detectar inconsistencias, mejorar la calidad del código y garantizar el cumplimiento de las normativas vigentes en la industria.

Revisión de código: herramientas como SonarQube detectan vulnerabilidades como SQL injection.

Auditorías de procesos: métodos como CMMI evalúan la madurez organizacional en niveles del 1 (ad-hoc) al 5 (optimizado).

Figura 2. Ciclo de vida del *software*

Fuente: SENA, 2025.

3. Transición de las normas ISO/IEC 9126 e ISO/IEC 14598 a ISO/IEC 25000

La evolución de las normas que regulan la calidad en el desarrollo de software ha permitido integrar nuevos criterios y metodologías que fortalecen la medición y evaluación del producto.

3.1. Evolución histórica

La Norma ISO/IEC 9126 se estableció como referencia para definir y medir la calidad de los productos de software, mientras que la ISO/IEC 14598 se orientó a la evaluación de dichos productos mediante la aplicación de métricas específicas. Con el tiempo, la necesidad de un marco más completo y actualizó surgió, lo que condujo a la integración de ambas normas en el conjunto ISO/IEC 25000. Este cambio responde a la evolución de las tecnologías y a la demanda de una evaluación que abarque tanto la calidad del producto como la eficiencia en el uso, pero presentaban limitaciones.

Fragmentación: métricas y requisitos dispersos en múltiples documentos.

Falta de adaptabilidad: no consideraban contextos modernos como DevOps o desarrollo ágil.

Enfoque estático: poca flexibilidad para integrar nuevas tecnologías (Garzás, 2012).

La familia ISO/IEC 25000 SQuaRE surge en 2014 como un marco unificado, integrando y actualizando estos estándares. Su objetivo es ofrecer un enfoque holístico que abarque.

Calidad del producto: características técnicas.

Calidad en uso: experiencia del usuario final.

Evaluación sistemática: procesos repetibles y adaptables (ISO/IEC, 2014).

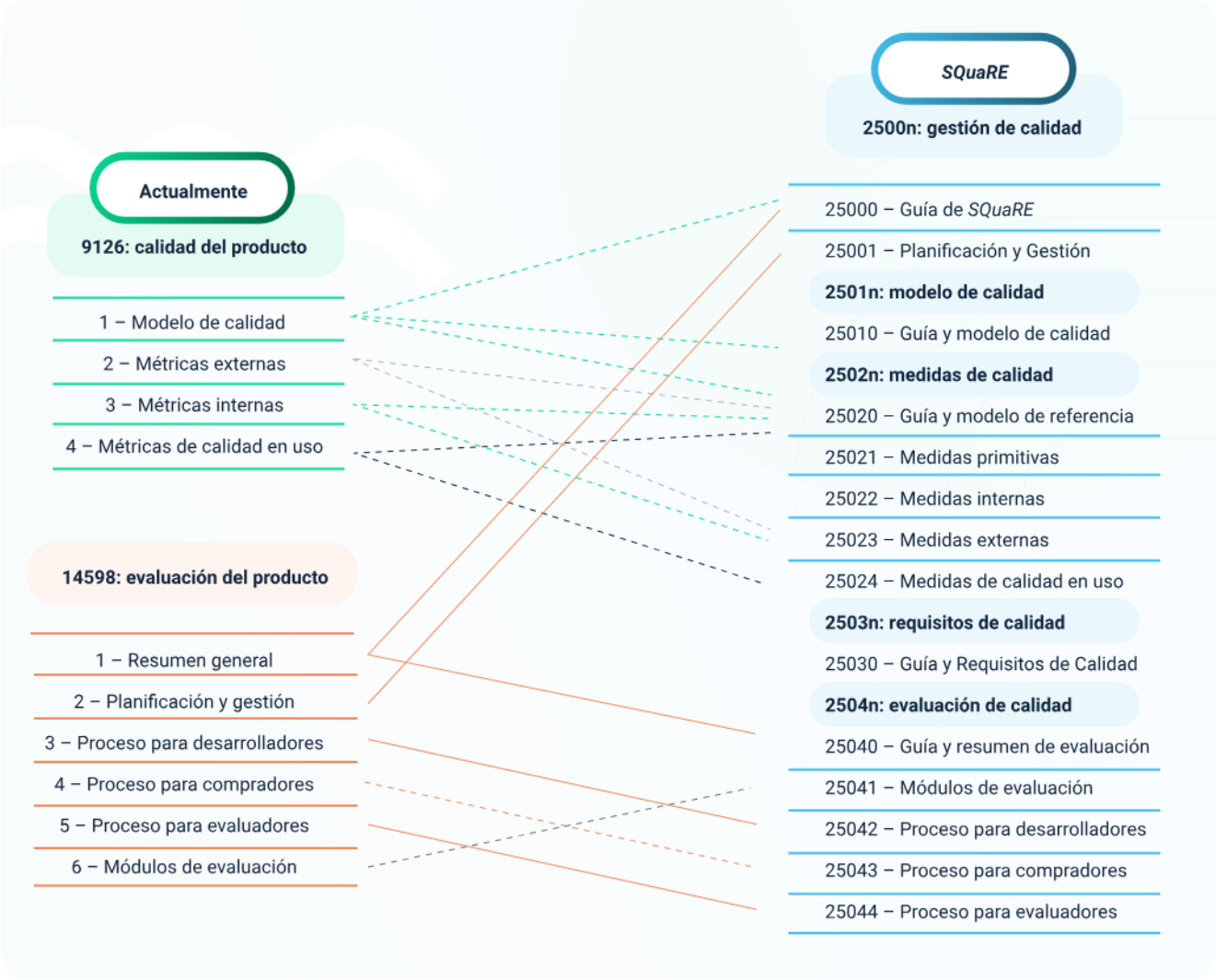
3.2. Integración en el marco SQuaRE

El modelo SQuaRE (System and Software Quality Requirements and Evaluation) consolida los elementos de calidad de las normas anteriores y establece directrices para definir, medir y evaluar la calidad del software en función de requisitos internos, externos y de uso. La integración en el marco SQuaRE permite disponer de un modelo integral que favorece la planificación, la ejecución de pruebas y la gestión de la calidad de manera sistemática. SQuaRE incorpora.

Modelos de calidad: define 8 características y 31 subcaracterísticas. Ejemplo: la eficiencia incluye subcaracterísticas como comportamiento temporal (tiempo de respuesta) y utilización de recursos (memoria consumida).

Métricas estandarizadas: ISO/IEC 25023 proporciona fórmulas para medir atributos cuantificables. Ejemplo: tiempo de respuesta promedio.

Figura 3. Evolución hacia SQuaRE



Fuente: SENA, 2025.

4. Medición de la calidad del software

La medición de la calidad en el software implica cuantificar aspectos que, en muchos casos, resultan intangibles, lo cual supone desafíos significativos en el proceso de evaluación.

4.1. Desafíos en la medición de atributos intangibles

Entre los principales desafíos se encuentra la dificultad para cuantificar atributos como la usabilidad, la fiabilidad o la satisfacción del usuario. Estas características, al no ser físicas, requieren la adopción de métodos de medición que combinen análisis cualitativos y cuantitativos, basados en indicadores y métricas específicas. La complejidad reside en traducir percepciones y comportamientos en datos objetivos que permitan evaluar la calidad del producto. La calidad del software es intangible; por ejemplo, la "usabilidad" no se mide con unidades físicas. Se requieren métodos como encuestas de satisfacción con una escala del 1 al 5 para evaluar facilidad de uso o métricas de interacción con número de clics para completar una tarea.

4.2. Enfoque cualitativo frente al cuantitativo

El enfoque cualitativo se centra en describir y analizar características mediante observaciones, encuestas y estudios de caso, aportando una visión integral de la experiencia del usuario. Por otro lado, el enfoque cuantitativo utiliza datos numéricos obtenidos a través de métricas y pruebas automatizadas. La combinación de ambos enfoques permite una evaluación más completa, en la que se contraponen los resultados medibles con la interpretación del comportamiento y satisfacción del usuario.

Enfoque cualitativo: se centra en describir y analizar características mediante observaciones, encuestas y estudios de caso. Aporta una visión integral de la experiencia del usuario.

Objetivo: comprender el comportamiento, opiniones y percepciones de los usuarios.

Ejemplo: Feedback de usuarios en pruebas beta.

Herramientas:

- Entrevistas y encuestas abiertas.
- Evaluaciones heurísticas según Nielsen.
- Estudios de caso.
- Grupos focales.

Enfoque cuantitativo: utiliza datos numéricos obtenidos a través de métricas y pruebas automatizadas. Permite realizar mediciones objetivas y comparaciones precisas.

Objetivo: medir variables concretas como tiempo, frecuencia o cantidad.

Ejemplo: tiempo de carga de una página web en milisegundos.

Herramientas:

- Google Analytics.
- Tests A/B.
- Métricas de rendimiento.
- Software de monitoreo de usabilidad.

5. Factores de calidad según ISO/IEC 9126

La Norma ISO/IEC 9126 define una serie de características fundamentales que deben cumplir los productos de software para ser considerados de calidad, estructuradas en diversas dimensiones.

5.1. Caracteristicas de la norma

Cada característica posee subatributos que permiten un análisis detallado:

Funcionalidad: se refiere a la capacidad del software para satisfacer las necesidades especificadas, abarcando aspectos como adecuación, exactitud, interoperabilidad, seguridad de acceso y cumplimiento de la funcionalidad.

Fiabilidad: evalúa la capacidad del software para mantener su desempeño bajo condiciones específicas, considerando la madurez, tolerancia a fallos, capacidad de recuperación y cumplimiento de la fiabilidad.

Usabilidad: se relaciona con el esfuerzo requerido para el uso del software, abarcando la facilidad para entender, aprender, operar y la capacidad de atracción.

Eficiencia: mide la relación entre el rendimiento del software y los recursos empleados, incluyendo el comportamiento temporal y la utilización de recursos.

Mantenibilidad: valora el grado de facilidad para modificar el software, considerando aspectos como el análisis, cambios, estabilidad y pruebas.

Portabilidad: hace énfasis en la capacidad del software para ser transferido entre diferentes entornos, evaluando la adaptabilidad, instalabilidad, coexistencia y reemplazo.

A modo de apoyo, se puede consultar la siguiente tabla resumen:

Tabla 2. Características y subatributos de calidad del software según el modelo ISO/IEC 9126

Características	Subatributos principales
Funcionalidad	Adecuación, exactitud, interoperabilidad, seguridad, cumplimiento.
Fiabilidad	Madurez, tolerancia a fallos, capacidad de recuperación, cumplimiento.
Usabilidad	Facilidad de comprensión, aprendizaje, operación y atracción.
Eficiencia	Comportamiento temporal, utilización de recursos.
Mantenibilidad	Capacidad de análisis, cambio, estabilidad, pruebas.
Portabilidad	Adaptabilidad, instalabilidad, coexistencia, reemplazo.

Fuente: Sena (2025).

5.2. Métricas externas, internas y de calidad en uso

Las métricas se dividen en tres categorías:

Métricas internas: se obtienen mediante análisis del código y estructuras internas del software sin necesidad de su ejecución. Ejemplo: complejidad ciclomática (número de caminos en el código).

Métricas externas: se aplican durante la ejecución y permiten evaluar el comportamiento en condiciones reales de uso. Ejemplo: tasa de errores en producción (1 error por cada 1000 transacciones).

Métricas de calidad en uso: se centran en la satisfacción del usuario y la efectividad del software en el entorno operativo, considerando aspectos como eficiencia, eficacia y satisfacción. Ejemplo: tiempo promedio para completar una compra en una app.

Cada tipo de métrica complementa la evaluación global del software, posibilitando una revisión integral de sus atributos de calidad.

6. Factores de evaluación según ISO/IEC 14598

La Norma ISO/IEC 14598 establece criterios y procedimientos para evaluar de forma objetiva el producto de software, complementando el modelo de calidad de la ISO/IEC 9126.

6.1. Características de la norma

La evaluación se fundamenta en criterios que aseguran que los procesos sean repetibles, es decir, que puedan ser ejecutados de manera consistente; reproducibles, garantizando que diferentes evaluadores obtengan resultados similares e imparciales, para que la valoración se realice sin sesgos. Estos elementos permiten que la evaluación del software se base en medidas objetivas y confiables.

- **Repetitividad**: resultados consistentes en múltiples evaluaciones (ejemplo: misma métrica en diferentes equipos).
- **Reproducibilidad**: métodos aplicables en diferentes contextos (ejemplo: evaluación en desarrollo y producción).
- **Imparcialidad**: neutralidad en la evaluación (ejemplo: uso de herramientas automatizadas) (ISO/IEC 14598, 1999).

6.2. Procesos para desarrolladores, usuarios finales y avaladores

La norma define distintos procesos de evaluación que se aplican según el rol.

- **Desarrolladores**: se establecen pautas que permitan integrar la evaluación durante el proceso de codificación, facilitando la detección temprana de defectos. Ejemplo: TDD - Test-Driven Development.
- **Usuarios finales**: se especifican procedimientos que aseguren que el software cumpla con las expectativas en su entorno de uso. Validando

funcionalidades en escenarios reales. Ejemplo: pruebas A/B en una plataforma de e-learning.

- **Avaladores o evaluadores externos**: se plantean criterios independientes que permiten realizar una valoración objetiva y fundamentada del producto, certificando cumplimiento de estándares. Ejemplo: auditorías ISO 9001.

Esta división de procesos favorece la mejora continua y la identificación oportuna de áreas de oportunidad en el desarrollo del software.

7. Modelos y estándares de apoyo

Diversos modelos y estándares complementan las normas ISO, proporcionando herramientas y metodologías para la mejora de la calidad en el desarrollo de software.

7.1. PSP (Personal Software Process): fases y registro de tiempos

El PSP es una metodología que permite a cada desarrollador gestionar de forma personal su proceso de trabajo. Se estructura en fases que van desde un proceso básico de identificación y registro de actividades (PSP 0) hasta un proceso cíclico de desarrollo de programas de mayor escala (PSP 3). El método enfatiza la planeación, la medición del tiempo y la identificación de defectos, facilitando la mejora continua a nivel individual. Se recomienda llevar un registro de tiempos mediante tablas que incluyan fecha, hora de inicio y fin, interrupciones, descripción de la actividad y unidades ejecutadas.

7.2. TSP (Team Software Process): trabajo en equipo y planificación

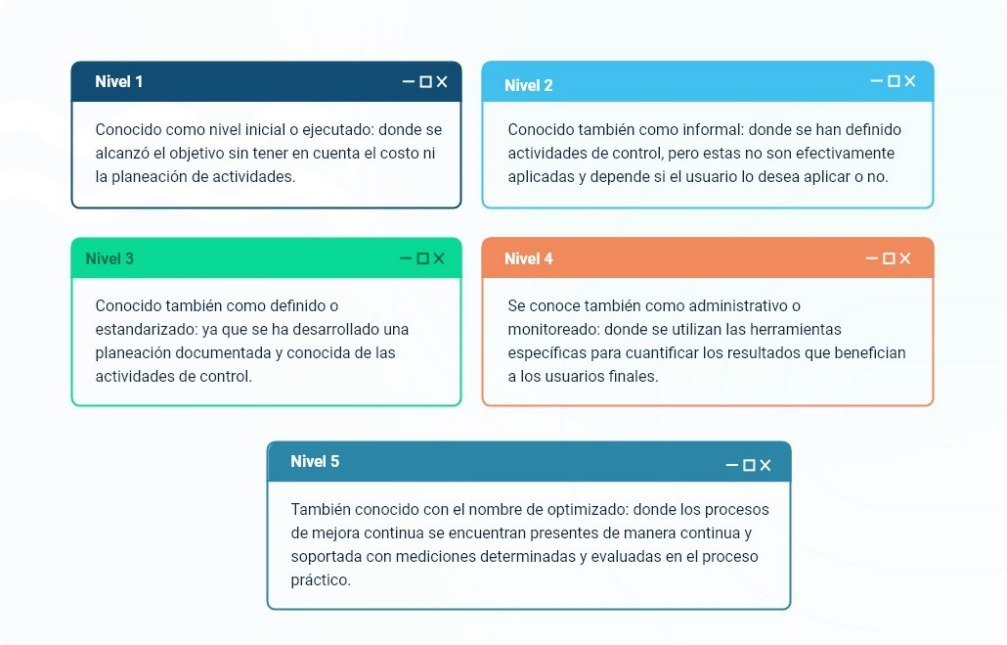
El TSP se orienta al trabajo colaborativo, promoviendo la coordinación y planificación en equipos de desarrollo. Bajo la guía de un coach, cada miembro conoce su rol y se establece un plan de actividades con tiempos definidos, lo que favorece la adaptación ante cambios y la optimización de recursos. La metodología se basa en la comunicación constante, la coordinación y la revisión periódica de los avances del proyecto.

7.3. CMMI (Capability Maturity Model Integration): niveles de madurez

El modelo CMMI proporciona un marco para la mejora de procesos en el desarrollo, mantenimiento y operación de software.

Se estructura en cinco niveles de madurez:

Figura 4. Niveles de CCMMI



Fuente: SENA, 2025.

La aplicación de CMMI permite identificar y fortalecer los procesos críticos de la organización, favoreciendo la calidad y eficiencia en el desarrollo.

7.4. GQM (Goal Question Metric): metas, preguntas y métricas

El modelo GQM se basa en la definición de metas claras que se descomponen en preguntas específicas y, a partir de estas, se derivan métricas cuantificables. Este enfoque permite estructurar la medición de la calidad en tres niveles:

Figura 5. Niveles de GQM

Fuente: SENA, 2025.

Esta metodología facilita la alineación de las actividades de evaluación con los objetivos estratégicos del desarrollo de software.

8. Familia de Normas ISO/IEC 25000

La familia ISO/IEC 25000, también conocida como SQuaRE, integra un conjunto de normas que ofrecen un marco integral para la gestión y evaluación de la calidad del software.

8.1. Divisiones del modelo SQuaRE

El modelo SQuaRE se compone de varias divisiones que abordan aspectos específicos de la calidad.

- **Gestión de calidad (ISO/IEC 2500n):** establece directrices para la administración de la calidad en el desarrollo. Ejemplo: ISO/IEC 25001 para planificación.
- **Modelo de calidad (ISO/IEC 2501n):** define las características y subcaracterísticas que debe cumplir el software en términos de calidad interna, externa y en uso. Ejemplo: ISO/IEC 25010 para usabilidad.
- **Medidas de calidad (ISO/IEC 2502n):** ofrece un marco de referencia para la definición de métricas y medidas primitivas que permiten evaluar de manera objetiva la calidad. Ejemplo: ISO/IEC 25023 para rendimiento.
- **Requisitos de calidad (ISO/IEC 2503n):** especifica los requerimientos que debe satisfacer el software para cumplir con las expectativas del cliente. Ejemplo: ISO/IEC 25030 para especificaciones.
- **Evaluación de calidad (ISO/IEC 2504n):** proporciona las pautas y procedimientos para llevar a cabo la valoración y auditoría del producto. Ejemplo: ISO/IEC 25044 para evaluadores; ISO/IEC 25000, 2014.

A modo de referencia, la siguiente tabla resume las divisiones incluidas en el modelo SQuaRE:

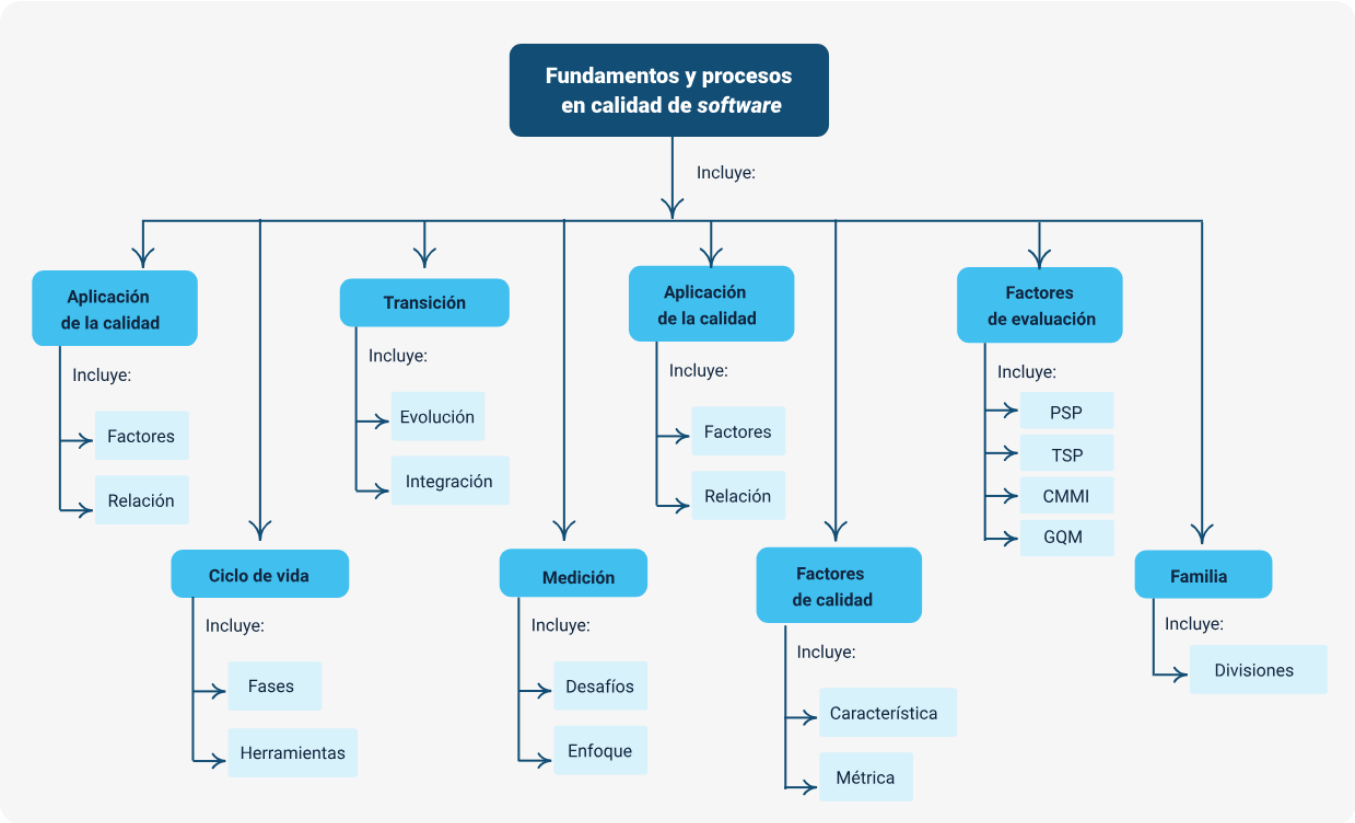
Tabla 3. Divisiones del modelo SQuaRE y sus contenidos principales

División	Contenidos principales
2500n (Gestión)	Directrices y parámetros de gestión de la calidad.
2501n (Modelo)	Definición de características y subcaracterísticas de calidad (interna, externa y en uso).
2502n (Medidas)	Modelo de referencia para la definición de métricas y medidas primitivas.
2503n (Requisitos)	Especificación de los requisitos de calidad a satisfacer.
2504n (Evaluación)	Pautas para la evaluación y auditoría del producto de software.

Fuente: SENA (2025).

Síntesis

A continuación, se presenta una visión general del componente formativo “Fundamentos y procesos en calidad de software”. En este módulo se ofrece un análisis profundo de los principios y metodologías que aseguran la calidad en el desarrollo de software. Se exploran aspectos esenciales como la aplicación de la calidad en el desarrollo, donde se destacan factores clave tales como el análisis, las pruebas, el código limpio y la refactorización, así como la relación entre procesos, calidad interna/externa y el contexto de uso. Además, se abordan temas fundamentales como el ciclo de vida del software, la transición de normas (ISO/IEC 9126 y ISO/IEC 14598 a ISO/IEC 25000), la medición de la calidad (incluyendo desafíos y enfoques cuantitativos y cualitativos), y los factores de calidad y evaluación según las normas internacionales. Asimismo, se incorporan modelos y estándares de apoyo como PSP, TSP, CMMI y GQM, y se describe la estructura de la familia de Normas ISO/IEC 25000, lo que proporciona a los aprendices herramientas para analizar, evaluar y mejorar la calidad de sus desarrollos de software. Se propone la elaboración de un mapa conceptual que integre estos temas, mostrando la interrelación entre cada uno de los elementos del proceso de calidad.


Fundamentos y procesos en calidad de *software*
Incluye:
Aplicación de la calidad
Incluye:
Factores
Relación
Ciclo de vida
Incluye:
Fases
Herramientas
Transición
Incluye:
Evolución
Integración
Medición
Incluye:
Desafíos
Enfoque
Aplicación de la calidad
Incluye:
Factores
Relación
Factores de calidad
Incluye:
Característica
Métrica
Factores de evaluación
Incluye:
PSP
TSP
CMMI
GQM
Familia
Incluye:
Divisiones

Glosario

Calidad interna: capacidad de un conjunto estático de atributos para satisfacer las necesidades declaradas e implícitas de un producto de software bajo ciertas condiciones especificadas.

Calidad externa: capacidad de un producto de software para desarrollar el comportamiento de un sistema de forma que satisfaga las necesidades del usuario en condiciones reales de uso.

Calidad en uso: grado en que un producto satisface objetivos con efectividad, eficiencia y satisfacción en un contexto específico.

CMMI (Capability Maturity Model Integration): modelo que evalúa y mejora procesos de desarrollo de software mediante niveles de madurez.

Eficiencia: relación entre el rendimiento del software y los recursos utilizados.

Fiabilidad: capacidad del software para mantener su nivel de funcionamiento bajo condiciones específicas durante un tiempo determinado.

Funcionalidad: capacidad del software para cumplir con las funciones especificadas, incluyendo adecuación, exactitud y seguridad.

GQM (Goal Question Metric): enfoque para definir métricas basadas en objetivos, preguntas y medidas cuantificables.

Mantenibilidad: grado de facilidad para modificar el software, incluyendo análisis, cambios y pruebas.

Portabilidad: capacidad del software para ser transferido entre diferentes entornos.

PSP (Personal Software Process): método para mejorar la productividad individual en el desarrollo de software mediante planificación y registro de tiempos.

Refactorización: reestructuración del código sin alterar su funcionalidad para mejorar su mantenibilidad o rendimiento.

SQuaRE (ISO/IEC 25000): familia de normas para gestionar requisitos y evaluación de calidad en sistemas y software.

TSP (Team Software Process): metodología para optimizar el trabajo en equipo en proyectos de software, enfocada en roles y planificación colaborativa.

Usabilidad: facilidad con la que los usuarios pueden aprender, operar y entender el software.

Material complementario

Tema	Referencia APA del material	Tipo	Enlace
Modelo CMMI	SEI. (2010). Modelo CMMI para Desarrollo v1.3. Carnegie Mellon University.	Guía técnica	https://es.slideshare.net/slideshow/spanish-technical-report-cmmi-v-1-3-26416661/26416661
PSP y TSP	Humphrey, W. S. (2017). Introducción al PSP.	Libro/PDF	https://www.uv.mx/personal/asumano/files/2010/07/psp.pdf
Ciclo de Vida del Software	SENA. (2017).	Video educativo	https://www.youtube.com/watch?v=XGSPIaLtJ-M

Referencias bibliográficas

Abud Figueroa, M. (2000). Calidad en la Industria del Software. La Norma ISO-9126. Recuperado de https://repositorio.utp.edu.co/server/api/core/bitstreams/1bb30bc9-250c-4764-8366-27b1e6ed2ef1/content

Garzás, J. (2012). Cómo estandarizar la evaluación de la calidad software... la ISO 9126 y la ISO 25000. Recuperado de: http://www.javiergarzas.com/2012/10/iso-9126-iso-25000-2.html

Humphrey, W. S. (2017). Introducción al PSP. Recuperado de https://www.uv.mx/personal/asumano/files/2010/07/psp.pdf.

ISO/IEC. (2014). ISO/IEC 25000:2014 – Ingeniería de sistemas y software – Requisitos y evaluación de la calidad de sistemas y software (SQuaRE). Ginebra: Organización Internacional de Normalización (ISO).

Instituto de Ingeniería del Software (SEI). (2010). Modelo CMMI para desarrollo v1.3: Guía para la integración de procesos y la mejora de productos [Presentación de diapositivas]. Carnegie Mellon University. Recuperado de https://es.slideshare.net/slideshow/spanish-technical-report-cmmi-v-1-3-26416661/26416661

Créditos

Nombre	Cargo	Centro de Formación y Regional
Milady Tatiana Villamil Castellanos	Responsable del Ecosistema de Recursos Educativos Digitales (RED)	Dirección General
Miguel de Jesús Paredes Maestre	Responsable de línea de producción	Centro de Comercio y Servicios - Regional Atlántico
Sandra Aydeé López Contador	Experta temática	Centro de Gestión de Mercados Logística y Tecnologías de la Información - Regional Distrito Capital
Heydy Cristina González García	Evaluadora instruccional	Centro de Comercio y Servicios - Regional Atlántico
Jair Coll	Evaluador instruccional	Centro de Comercio y Servicios - Regional Atlántico
Carmen Alicia Martínez Torres	Diseñador web	Centro de Comercio y Servicios - Regional Atlántico
Álvaro Guillermo Araújo Angarita	Desarrollador full stack	Centro de Comercio y Servicios - Regional Atlántico
Alexander Rafael Acosta Bedoya	Animador y productor audiovisual	Centro de Comercio y Servicios - Regional Atlántico
María Fernanda Morales Angulo	Evaluador de contenidos inclusivos y accesibles	Centro de Comercio y Servicios - Regional Atlántico
Luz Karime Amaya Cabra	Evaluador de contenidos inclusivos y accesibles	Centro de Comercio y Servicios - Regional Atlántico
Jonathan Adie Villafañe	Validador y vinculador de recursos digitales	Centro de Comercio y Servicios - Regional Atlántico
Jairo Luis Valencia Ebratt	Validador y vinculador de recursos digitales	Centro de Comercio y Servicios - Regional Atlántico