



MODELOS DE CALIDAD
EN EL DESARROLLO DE
SOFTWARE



FAVA - Formación en Ambientes Virtuales de Aprendizaje

SENA - Servicio Nacional de Aprendizaje.

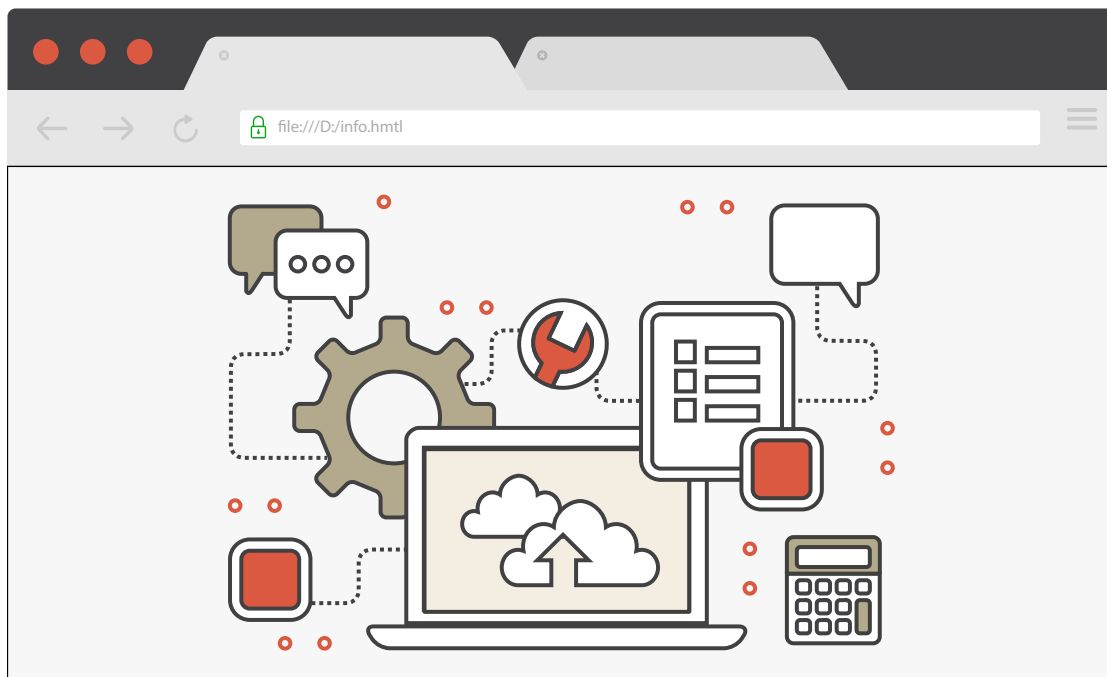
Estructura de contenidos

Pág.

Introducción	3
Mapa de contenido	4
Desarrollo de contenidos	5
1. Calidad de software	5
1. 1 Modelo de capacidad y madurez (CMM)	5
2. Modelos de calidad	7
2. 1 Modelo CMMI (Capability Maturity Model Integration)	7
2. 2 Modelo ISO/ IEC 15504 (SPICE)	8
2. 2. 1 Estructura de la norma	8
2. 3 Modelo ISO/IEC 25000	11
2. 3. 1 Funcionalidad	12
2. 3. 2 Confiabilidad o fiabilidad	13
2. 3. 3. Facilidad de uso (Usabilidad)	13
2. 3. 4. Eficiencia	13
2. 3. 5. Mantenibilidad	14
2. 3. 6. Portabilidad	14
3. Mejora del proceso del software a nivel de equipos y desarrolladores	15
3. 1 PSP (Process software Personal)	16
3. 1. 1 Características	16
3. 1. 2 Objetivos del PSP	16
3. 1. 3 Aplicación del PSP	16
3. 1. 4 Principios de planeación y de calidad en PSP	17
3. 2 TSP (Team Software Process)	17
3. 2. 1 Características	17
3. 2. 2. Principios del TSP	17
3. 2. 3. Objetivos del TSP	18
3. 2. 4 Los roles y responsabilidades	18
Glosario	19
Bibliografía	20
Control del documento	21

Modelos de calidad en el desarrollo de software

Introducción

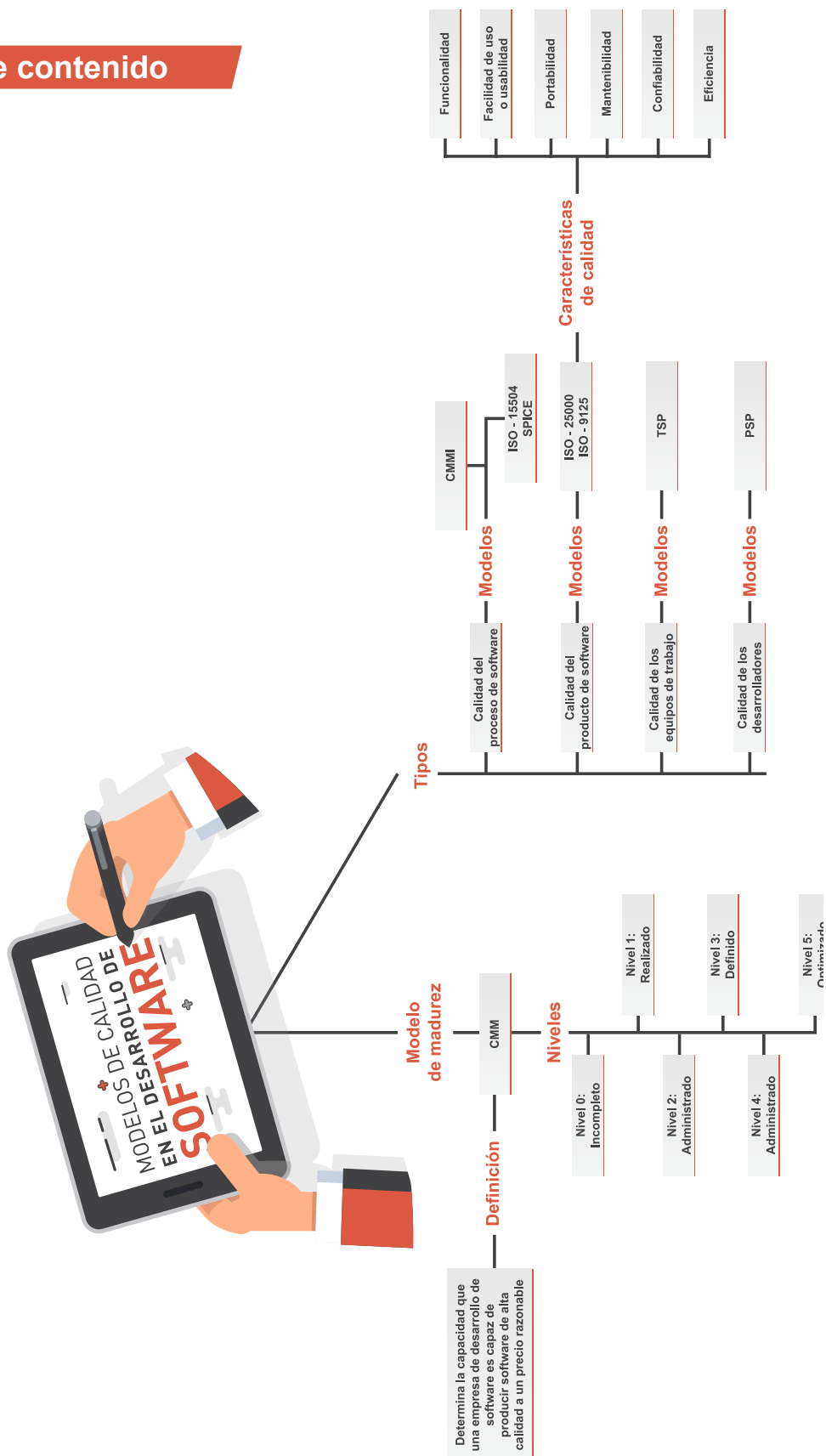


Los modelos de calidad son herramientas que guían a las organizaciones a la mejora continua y a la competitividad dándoles especificaciones de los requisitos que deben de cumplir para poder brindar productos y servicios de alto nivel. Un modelo de calidad de software es un conjunto de buenas prácticas para el ciclo de vida del software enfocadas tanto en el proceso como en el producto de software.

Construir un modelo de calidad es bastante complejo y es usual que estos modelos descompongan las características de calidad del producto software y que estas características se usen para determinar los ítems de una lista de comprobación de la misma.

A continuación se dan a conocer algunos de los modelos de gestión de la calidad en el desarrollo de software que se pueden implementar tanto a nivel de empresas desarrolladoras de software como a nivel de equipos de desarrollo y desarrolladores individuales.

Mapa de contenido



Desarrollo de contenidos

1. Calidad de software.

Actualmente la calidad tanto del proceso como de los productos de software se ha convertido en uno de los principales desafíos que tiene que afrontar la ingeniería del software. Tanto la industria del software como la academia y organismos internacionales han venido realizando una gran cantidad de esfuerzos sobre cómo lograr software de calidad y cómo evaluarlo.

La calidad de acuerdo a la Real Academia Española es: “propiedad o conjunto de propiedades inherentes a una cosa, que permiten apreciarla como igual, mejor o peor que las restantes de su especie”

La calidad del software según la IEEE es: “grado con el cual el cliente o usuario percibe que el software satisface sus expectativas”.

La calidad del software según la ISO es: “el conjunto de características de una entidad que le confieren su aptitud para satisfacer las necesidades expresadas y las implícitas”.

En síntesis la calidad del software es el conjunto de cualidades medibles y específicas que varía de un sistema a otro, dependiendo del tipo de software que se va a desarrollar, para determinar su utilidad e idoneidad. Este desarrollo debe ser confiable, mantenible y flexible para disminuir los costos de mantenimiento y perfeccionamiento durante el tiempo de utilización y durante las etapas del ciclo de vida del software.

1. 1 Modelo de capacidad y madurez (CMM)

En el proceso de desarrollo de software se buscan los siguientes tres atributos de calidad deseables (NAIK, 2008):

- a) Que los productos sean de la más alta calidad. Un producto ideal debe ser libre de defectos.
- b) Que los proyectos se lleven a cabo en el tiempo estimado.
- c) Que los proyectos se lleven a cabo en el presupuesto estimado.

Debido a que estos atributos deseables no siempre se cumplían en sus contratos el departamento de defensa de los Estados Unidos encargó al Instituto de Software de la universidad Carnegie Mellon la creación de un marco o framework para evaluar las capacidades de sus contratistas y de esta forma disminuir la problemática asociada a un proceso de desarrollo de software defectuoso (NAIK, 2008).

Es así como nace el Capability Maturity Model o CMM que determina la capacidad de una empresa de desarrollo de software es capaz de producir software de alta calidad a un precio razonable o bajo.

El CMM define los siguientes grados de madurez:

Nivel	Nombre	Características
0	Incompleto	El área de proceso involucrado no logra los objetivos definidos para el nivel 1.
1	Realizado	Se satisfacen todos los objetivos del área de procesos. Se hacen las tareas necesarias para producir productos operativos.
2	Administrado	Se satisfacen los criterios del nivel 1. Además los equipos están alineados con una política organizacional.
3	Definido	Se satisfacen los criterios del nivel 2. Además los procesos se hacen a la medida y se encuentran estandarizados.
4	Administrado cuantitativamente	Se satisfacen los criterios del nivel 3. Además los procesos son controlados y mejorados de manera cuantitativa.
5	Optimizado	Se satisfacen los criterios del nivel 4. Además el proceso se adapta y se optimiza para satisfacer las necesidades cambiantes del cliente.

2. Modelos de calidad.

Para estandarizar las buenas prácticas de calidad en la industria en general y en la industria del software en particular se ha propuesto los siguientes modelos de calidad.

2. 1 Modelo CMMI (Capability Maturity Model Integration).

El modelo CMM visto en el capítulo anterior fue aplicado inicialmente al proceso del software y específicamente su nombre es CMM-SW. Sin embargo en el desarrollo de software existen otros procesos importantes como el proceso de pruebas, el proceso de subcontratación, entre otros, que también requieren ser evaluados.

Es así como aparece el CMMI o Capability Maturity Model Integration que usa el mismo marco conceptual o framework CMM pero aplicado a las siguientes áreas:

- Desarrollo de software.
- Ingeniería de sistemas.
- Desarrollo integrado de productos.
- Alianza de la industria electrónica.
- Adquisición de software.
- Recursos humanos.

Es importante mencionar que igual que las normas ISO este modelo indica qué se debe hacer pero no cómo hay que hacerlo, es decir, es un marco conceptual pero no una metodología.

La evolución de los estándares CMM se puede apreciar en la figura 2. 1:

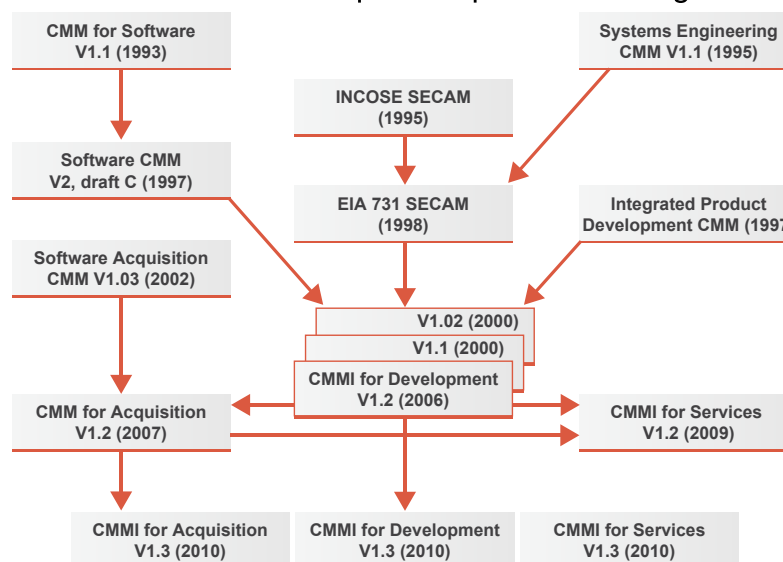


Figura 2. 1. Evolución de los modelos de capacidad basados en CMM Fuente: (CMMI, 2010)

2. 2 Modelo ISO/ IEC 15504 (SPICE)

El ISO/IEC 15504, también conocido como “Software Process Improvement Capability Determinación”, abreviado SPICE, en español, “Determinación de la Capacidad de Mejora del Proceso de Software” es un modelo para la mejora y evaluación de los procesos de desarrollo y mantenimiento de sistemas de información y productos de software.

La norma ISO 15504 SPICE es una norma abierta e internacional para evaluar y mejorar la capacidad y madurez de los procesos. Junto con la ISO-12207, la norma aplica a la evaluación y mejora de la calidad del proceso de desarrollo y mantenimiento de software.

A diferencia del CMMI que es una iniciativa de un sólo gobierno la ISO-15504 está avalada por un organismo de certificación internacional.

2. 2. 1 Estructura de la Norma.

La norma ISO/IEC 15504 proporciona un marco de trabajo para la evaluación de los procesos y establece los requisitos mínimos para realizar una evaluación de forma consistente. Actualmente esta norma está estructurada en siete partes (Alarcón, 2010).

Parte 1: conceptos y vocabulario.

Parte 2: realización de la evaluación (normativa).

Parte 3: guía para la realización de la evaluación.

Parte 4: guía sobre el uso para la mejora y determinación de la capacidad del proceso.

Parte 5: un ejemplo de modelo de evaluación de procesos.

Parte 6: un ejemplo de modelo de evaluación del ciclo de vida del sistema.

Parte 7: evaluación de la madurez de una organización.

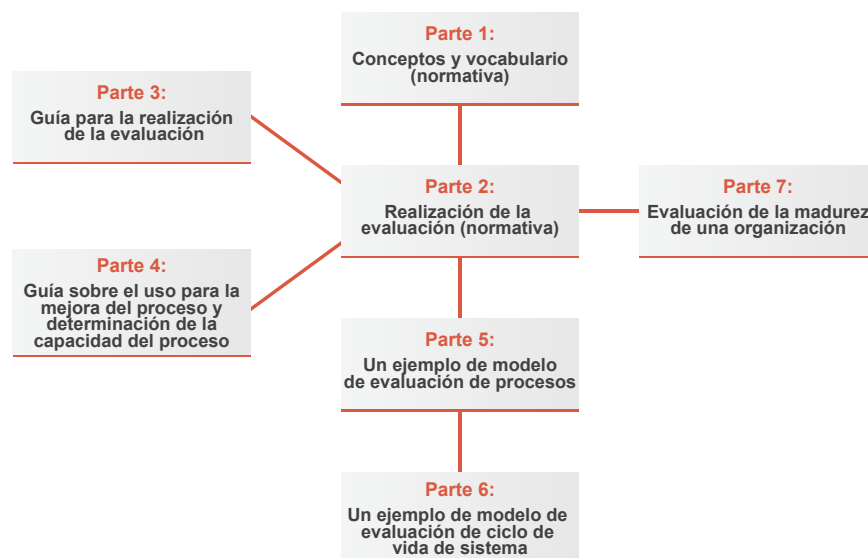


Figura 2. 2 Partes de la norma ISO/IEC 15504

La norma ISO/IEC 15504-7 define un marco de trabajo para determinar la madurez de la organización, de esta forma, se incorpora la posibilidad de evaluar a las organizaciones en ISO/IEC 15504 por niveles de madurez dando así una puntuación o calificación a la organización y no sólo a nivel de proceso.

El modelo de procesos de referencia que utiliza ISO/IEC 15504-7, propio de la industria del software, es la norma ISO/IEC 12207.

La norma ISO/IEC 15504-7 establece 6 niveles de madurez para clasificar a las organizaciones, tal y como se muestra en la Figura 2. 3.

Nivel de madurez	Características
0	La organización no tiene una implementación efectiva del proceso.
1	Se alcanza el propósito del proceso en términos generales. Los procesos se realizan cuando es necesario, pero no se hace de una forma planificada ni se realiza ningún seguimiento.
2	Se obtienen los productos del proceso, pero esta vez de acuerdo con una planificación y realizándose un seguimiento. Estos productos se ajustan a unos estándares y a unas especificaciones prefijadas. También se tienen definidos plazos y recursos.
3	Los procesos se realizan y se gestionan utilizando procesos definidos. Cada implementación de un proceso se hace utilizando procedimientos creados según un estándar y documentados.
4	Se recogen medidas detalladas de nivel de realización de los procesos y se analizan. Esto permite mantener el proceso dentro de unos límites predefinidos, así como disponer de una mejor posición para poder cuantificar la capacidad del proceso y predecir su comportamiento.
5	<p>La realización de los procesos se optimiza de forma continuada, de cara a su contribución a alcanzar los objetivos de negocio de la organización. Se establecen objetivos cuantitativos de eficacia y eficiencia en la realización de los procesos, basados en los objetivos de negocio de la organización.</p> <p>Se lleva a cabo un monitoreo continuo de los procesos y se analizan los datos obtenidos. Esto permite que los procesos estándar definidos dentro de la organización cambien dinámicamente para adaptarse de forma efectiva a los actuales y futuros objetivos de la empresa.</p>

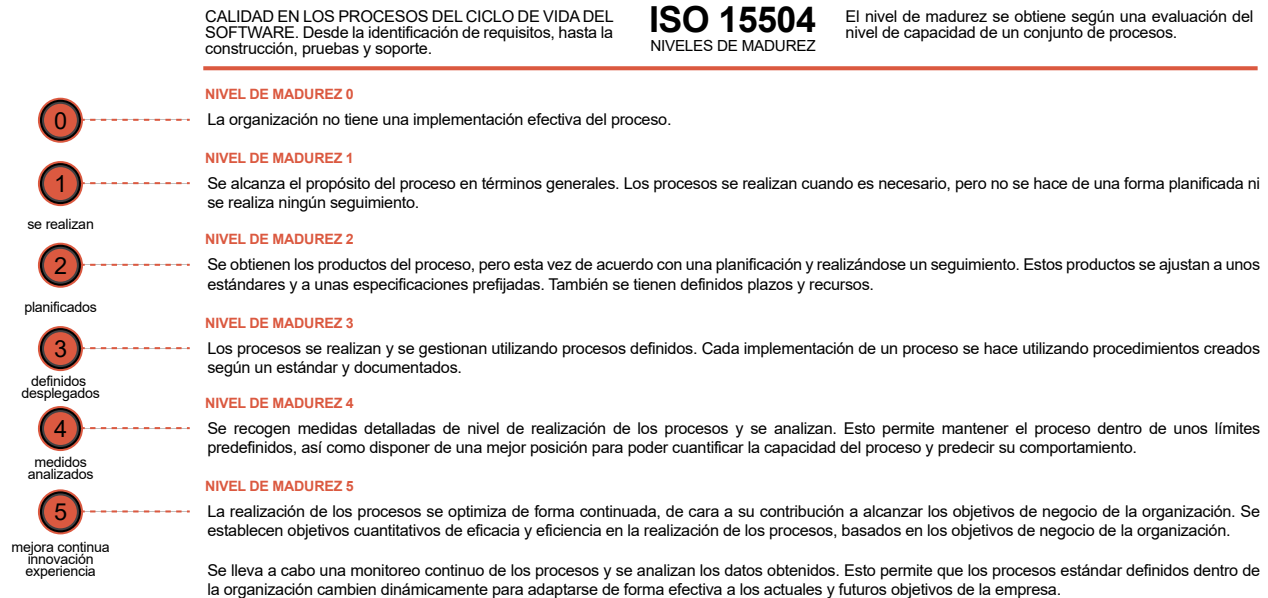


Figura 2. 3. Niveles de madurez de las organizaciones según ISO/IEC 15504

Características:

- Establece un marco y los requisitos para cualquier proceso de evaluación de procesos
- Provee requisitos para los modelos de evaluación de los procesos y para modelos de evaluación de organizaciones.
- Suministra guías para la definición de las competencias de un evaluador de procesos.
- Alcance: mejora y evaluación de procesos, y determinación de capacidad.

El modelo de procesos ISO/IEC 15504 – ISO/IEC 12207 – 2008: Normalmente, en la mejora de la calidad de los procesos participan dos tipos de modelos, el modelo de procesos y el modelo de evaluación. El modelo de procesos define un catálogo o colección estructurada de buenas prácticas que describen las características de un proceso efectivo mientras que el modelo de evaluación proporciona los principios requeridos para realizar una evaluación de la calidad e implantación de dicho modelo de procesos en una organización.

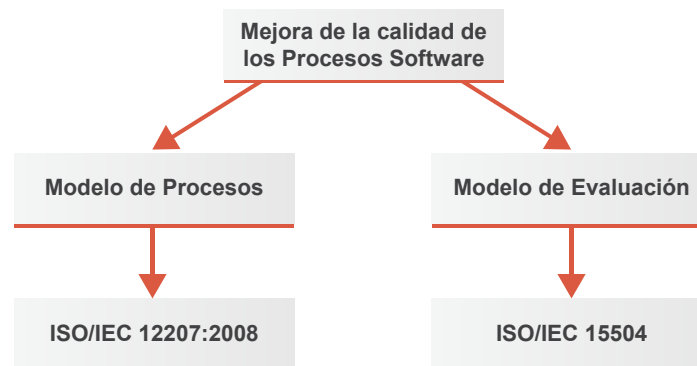


Figura 2. 4. Modelo de procesos ISO/IEC 15504 e ISO/IEC 12207

2.3 Modelo ISO/IEC 25000

La norma ISO/IEC-25000 también llamada “System and Software Quality Requirements and Evaluation (SQuaRE)” es la evolución de la norma ISO/IEC-9126: “Software Product Evaluación”, (Evaluación de los productos de Software) y crean un marco común para evaluar la calidad del software como producto.

El modelo CMMI y el estándar ISO-15504 están enfocados en la mejora del proceso de desarrollo de software. En cambio la norma ISO-25000 está enfocada en la calidad del software como producto.

La norma define 6 características de calidad y describe un modelo de procesos para la evaluación de productos de software (NAIK, 2008).

El estándar ISO/IEC 25000 define un marco conceptual que considera los siguientes factores:

- Calidad del proceso.
- Calidad del producto software (Calidad interna y externa).
- Calidad en el uso.

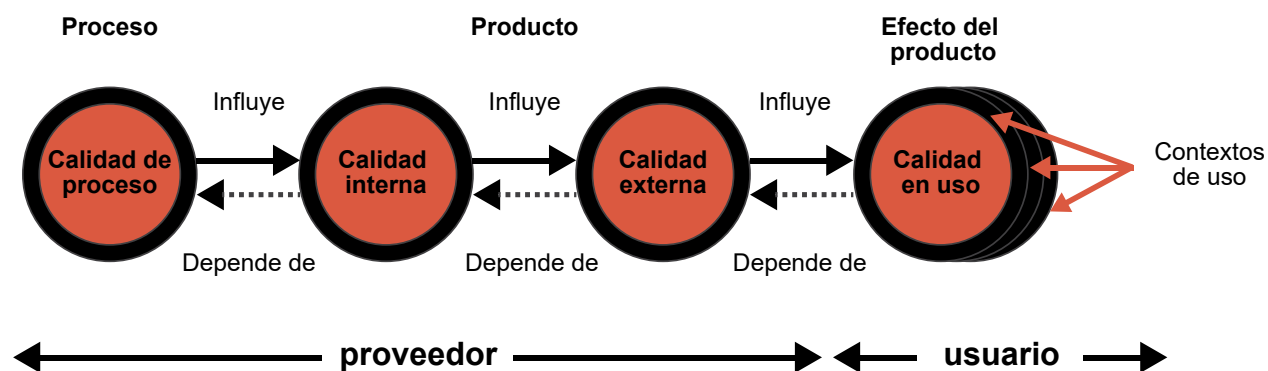


Figura 2. 5 Estructura de procesos de la norma ISO/IEC 25000 y 9126

Calidad Interna: Toma en cuenta todas las características del producto software a nivel interno, debe ser medido y evaluado tomando en cuenta los parámetros de calidad definidos, están sujetos a mejora en las etapas de pruebas e implementación sin perder la esencia definida para ello.

Calidad Externa: Desde una visión externa enfocadas a la ejecución del producto, se determinan las características de calidad que debe tener, es evaluada en un ambiente simulado y con métricas de la industria. Se espera eliminar la mayor cantidad de fallas posibles durante las pruebas, sin afectar la arquitectura del software.

Atributos de la norma para calidad externa e interna:

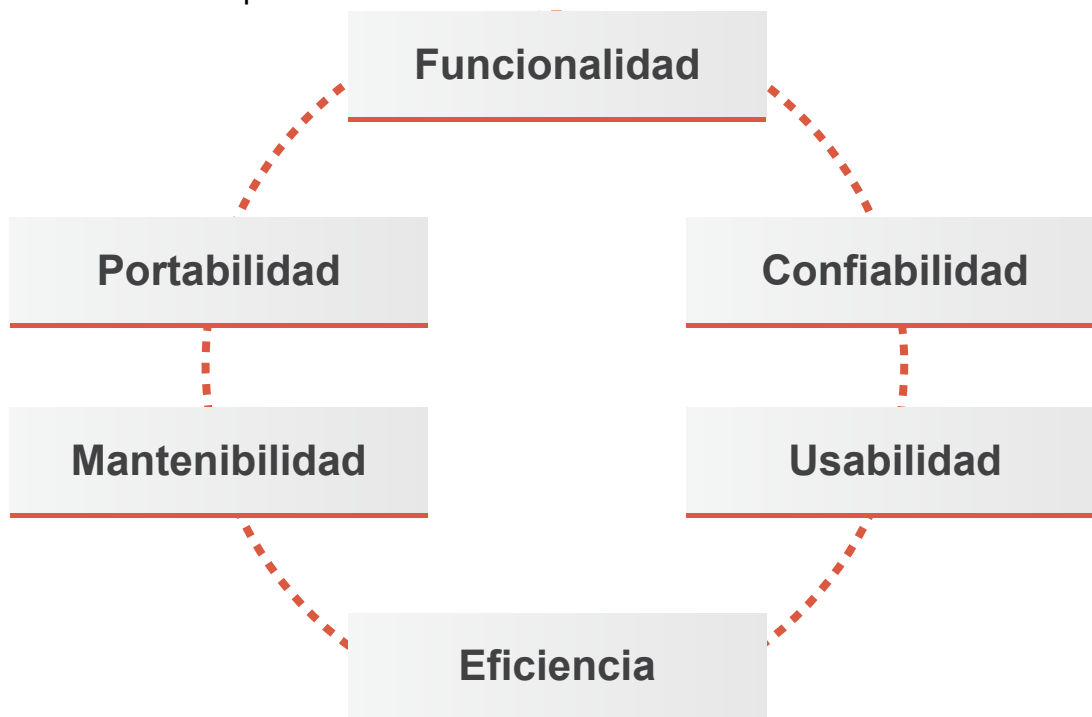


Figura 2. 6. Atributos de calidad de la norma ISO-25000 / 9126

2. 3. 1 Funcionalidad.

Conjunto de atributos que se relacionan con un conjunto de funciones que debe suministrar el producto y sus propiedades específicas. Las funciones son aquellas que satisfacen las necesidades explícitas o implícitas. Entre las sub-caraterísticas se tienen:

- a) **Adecuación:** capacidad del producto de software para proporcionar un conjunto apropiado de funciones para tareas y objetivos de usuario especificados.
- b) **Exactitud:** capacidad del producto de software para proporcionar los resultados o efectos correctos o acordados con el grado necesario de precisión.
- c) **Interoperatividad:** capacidad del producto de software para interactuar con uno o más sistemas especificados.
- d) **Seguridad de acceso:** capacidad del producto de software para proteger la información y datos de manera que las personas o sistemas no autorizados no puedan leerlos o modificarlos al tiempo que no deniega el acceso a las personas o sistemas autorizados.
- e) **Cumplimiento funcional:** capacidad del producto de software para adherirse a normas, convenciones o regulaciones en leyes y prescripciones similares relacionadas con funcionalidad.

2. 3. 2 Confiabilidad o fiabilidad.

Conjunto de atributos que se relacionan con la capacidad del software para mantener su nivel de desempeño bajo unas condiciones establecidas en un periodo de tiempo también establecido. Entre las sub-caraterísticas se tienen:

- a) **Madurez:** capacidad del producto de software para evitar fallar como resultado de fallos en el software.
- b) **Tolerancia a fallos:** capacidad del software de mantener un nivel especificado de prestaciones en caso de fallos de software o de infringir sus interfaces especificadas.
- c) **Capacidad de recuperación:** capacidad del producto de software para restablecer un nivel de prestaciones especificado y de recuperar los datos directamente afectados en caso de fallo.
- d) **Cumplimiento de la fiabilidad:** capacidad del producto de software para adherirse a normas, convenciones o regulaciones relacionadas con la fiabilidad.

2. 3. 3. Facilidad de uso (Usabilidad)

Conjunto de atributos que se relacionan con el esfuerzo necesario para usar y la valoración del usuario de este esfuerzo bajo condiciones establecidas. Entre las sub-caraterísticas se tienen:

- a) **Capacidad para ser entendido:** capacidad del producto de software que permite al usuario entender si el software es adecuado y cómo puede ser usado para unas tareas o condiciones particulares.
- b) **Capacidad para ser aprendido:** capacidad del producto de software que permite al usuario aprender sobre su aplicación.
- c) **Capacidad para ser operado:** capacidad del producto de software que permite al usuario operarlo y controlarlo.
- d) **Capacidad de atracción:** capacidad del producto de software para ser atractivo al usuario.
- e) **Cumplimiento de la usabilidad:** capacidad del producto de software para adherirse a normas, convenciones, guías de estilo o regulaciones relacionadas con la usabilidad.

2. 3. 4. Eficiencia.

Conjunto de atributos involucrados en la relación entre el desempeño del software y la cantidad de recursos usados bajo condiciones establecidas. Entre las sub-caraterísticas se tienen:

- a) **Comportamiento temporal:** capacidad del producto de software para proporcionar tiempos de respuesta, tiempos de proceso y potencia apropiados bajo condiciones determinadas.
- b) **Utilización de recursos:** capacidad del producto de software para usar las cantidades y tipos de recursos adecuados cuando el software lleva a cabo su función bajo condiciones determinadas.

- c) **Cumplimiento de la eficiencia:** capacidad del producto de software para adherirse a normas o convenciones relacionadas con la eficiencia.

2. 3. 5. Mantenibilidad

Conjunto de atributos relacionados con el esfuerzo necesario para hacer ciertas modificaciones. Entre las sub-características se tienen:

- a) **Capacidad para ser analizado:** es la capacidad o facilidad del producto de software para que le sean diagnosticadas deficiencias o causas de los fallos o para identificar las partes que han de ser modificadas.
- b) **Capacidad para ser cambiado:** capacidad del producto de software que permite que una determinada modificación sea implementada.
- c) **Estabilidad:** capacidad del producto de software para evitar efectos inesperados debidos a modificaciones del software.
- d) **Capacidad para ser probado:** capacidad del producto de software que permite que el software modificado sea validado.
- e) **Cumplimiento de la mantenibilidad:** capacidad del producto de software para adherirse a normas o convenciones relacionadas con la mantenibilidad.

2. 3. 6. Portabilidad.

Conjunto de atributos relacionados con la capacidad del software de ser transportado de un ambiente a otro. Entre las sub-características se tienen:

- a) **Adaptabilidad:** capacidad del producto de software para ser adaptado a diferentes entornos especificados sin aplicar acciones o mecanismos distintos de aquellos proporcionados para este propósito por el propio software considerado.
- b) **Instalabilidad:** capacidad del producto de software para ser instalado en un entorno especificado.
- c) **Coexistencia:** capacidad del producto de software para coexistir con otro software independiente en un entorno común compartiendo recursos comunes.
- d) **Capacidad para reemplazar:** capacidad del producto de software para ser usado en lugar de otro producto de software para el mismo propósito en el mismo entorno.
- e) **Cumplimiento de la portabilidad:** capacidad del producto de software para adherirse a normas o convenciones relacionadas con la portabilidad.

3. Mejora del proceso del software a nivel de equipos y desarrolladores.

Los modelos CMMI, SPICE e ISO 25000 vistos en el capítulo dos están dirigidos a las organizaciones, sin embargo la calidad tanto del producto como del proceso del software se basan también en la madurez de los equipos de trabajo y de los desarrolladores de software.

Para este fin se desarrollaron modelos que se enfocan en mejorar el trabajo realizado tanto por los equipos de desarrollo como los desarrolladores de manera individual.

En este contexto aparecieron los modelos TPS (Team Process Software) dirigido a equipos de trabajo y el PSP (Personal Software Process) dirigido a desarrolladores individuales. Se pueden relacionar los modelos CMM, TPS y PSP así (Scalone, 2006):

- CMM – mejora la capacidad de la organización y el enfoque de la Dirección
- TSP – mejora el rendimiento del equipo. Existe un enfoque respecto del proceso o producto
- PSP – mejora las habilidades individuales. Tiene un enfoque respecto del personal.

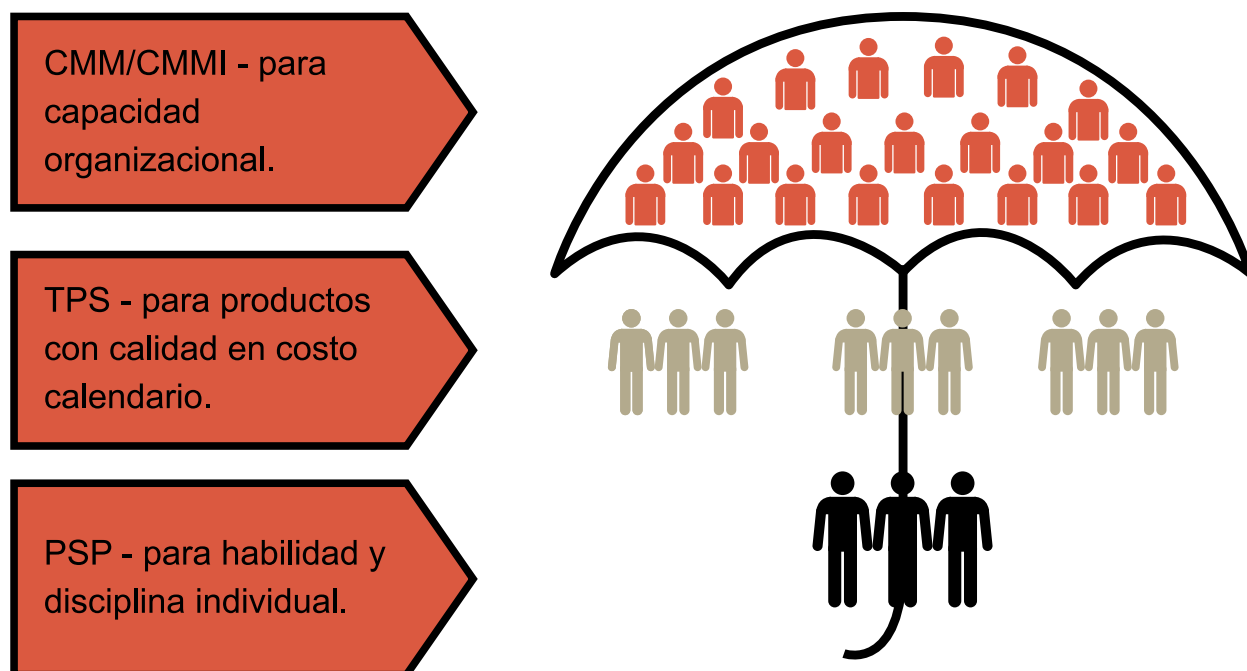


Figura 3. 1. Relación CMM, TPS y PSP.

3. 1 PSP (Process software Personal)

En el año de 1995 el PSP fue propuesto por Watts Humphrey. Inicialmente estaba dirigido para estudiantes pero en 1997 con el lanzamiento del libro “An Introduction to the Personal Software Process” el PSP ya estaba destinado a los desarrolladores profesionales (Pomeroy-Huff, 2009).

Este proceso está dirigido a mejorar la calidad del software desarrollado por una persona en particular.

3. 1. 1 Características:

- El PSP busca proporcionar un marco de trabajo para el personal involucrado en el proceso de desarrollo de software.
- El PSP se centra en la administración del tiempo y en la administración de la calidad a través de la eliminación temprana de defectos.
- PSP demuestra cómo manejar la calidad desde el principio del trabajo.

3. 1. 2 Objetivos del PSP:

- Planificar, estimar, medir, seguir y controlar todo el proceso de desarrollo de software.
- Lograr una disciplina de mejora continua en el proceso de desarrollo.
- Mejorar los niveles de calidad del proceso de desarrollo de software
- En general, PSP provee calidad y productividad a todo el proceso de desarrollo de software.

3. 1. 3 Aplicación del PSP.

PSP puede ser aplicado en:

- Desarrollo de programas.
- Definición de requerimientos.
- Documentación.
- Pruebas de sistemas.
- Mantenimiento de sistemas.

3. 1. 4 Principios de planeación y de calidad en PSP.

Los principios de planeación y de calidad en PSP

- Cada ingeniero es diferente, para ser más eficiente, debe planificar su trabajo basándose en su experiencia personal.
- Para mejorar constantemente su desempeño, los ingenieros deben utilizar personalmente procesos bien definidos y medidos.
- Los ingenieros deben asumir la responsabilidad personal de la calidad de sus productos.
- Cuanto antes se detecten y corrijan los errores menos esfuerzo será necesario
- Es más efectivo evitar los defectos que detectarlos y corregirlos.
- La manera correcta de hacer las cosas es siempre la manera más rápida y más económica de hacer un trabajo.

3. 2 TSP (Team Software Process)

TSP fue creado en 1996 por Watts Humphrey TSP busca suministrar un modelo operacional que ayude a los equipos de desarrollo a realizar trabajos de calidad. (Scalone, 2006).

Para el uso del TSP los desarrolladores primero deben ser entrenados en PSP.

3. 2. 1 Características:

- Se fundamenta en el trabajo en equipo al establecer directrices para la definición de objetivos, planificación y control de sus procesos, enfocados en prácticas de ingeniería avanzada. Se establecen actividades de apoyo para la conformación de equipos de trabajo y su administración.
- Se posibilita encontrar defectos en las etapas iniciales del ciclo de desarrollo del software, al ser reparados tempranamente, se reduce el tiempo de pruebas y por tanto el tiempo total.
- La aplicación de este modelo ayuda a mejorar el desempeño de los individuos y por tanto de los equipos haciéndolos más disciplinados y ágiles.

3. 2. 2. Principios del TSP:

- Es necesario incorporar a los técnicos en las actividades de planeación, lo cual ayuda a obtener planes más detallados e incentiva su compromiso con el mismo.
- La carga de trabajo del equipo debe estar equilibrada para mejorar los tiempos requeridos en el proyecto.
- Debe primar el interés por mantener los atributos de calidad definidos para el producto y así aumentar la productividad.

3. 2. 3. Objetivos del TSP.

Los objetivos del TSP son:

- a) Ayudar a los equipos de desarrollo a elaborar productos de calidad dentro de los costos y tiempos establecidos.
- b) Tener equipos rápidos y confiables.
- c) Optimizar el desempeño de los equipos durante todo el proyecto.

3. 2. 4 Los Roles y responsabilidades.

Los Roles y responsabilidades en los equipos en TSP son:

- a) **Líder del equipo:** dirige al equipo y se asegura que todos reporten sus datos de los procesos y completen su trabajo tal y como se planeó. Realiza los reportes semanales del avance del equipo.
- b) **Gestor de desarrollo:** guía al equipo en el diseño y desarrollo del producto.
- c) **Gestor de planificación:** apoya y guía al equipo en la planificación y seguimiento en el trabajo.
- d) **Gestor de calidad o proceso:** apoya al líder en definir sus necesidades acerca del proceso y a establecer y administrar el plan de calidad. Genera estándares para obtener un trabajo uniforme. Modera las inspecciones y revisa cada artefacto generado.
- e) **Administrador de requerimientos o soporte:** dirige el equipo en el desarrollo de requerimientos de software y ayuda a dar a conocer la tecnología y en las necesidades de apoyo administrativo. Administra el plan de configuración.

Glosario

CMM: acrónimo de Capability Maturity Model o Modelo de Madurez de capacidades.

CMMI: acrónimo de Capability Maturity Model Integration. Modelo para de madurez de capacidades aplicado al proceso del software planteado por el Software Engineering Institute.

Framework: sinónimo de marco conceptual.

ISO: acrónimo de International Organization for Standarization. Organización Internacional de Normalización.

PSP: acrónimo de Personal Software Process o proceso personal del software.

SPICE: acrónimo de Software Process Improvement Capability Determination. Modelo para el mejoramiento del software.

SquaRe: acrónimo de System and Software Quality Requirements and Evaluation. Calidad y evaluación de los requerimientos de calidad de software y sistemas. Modelo de calidad para evaluar el software como producto basado en la norma ISO-25000.

TSP: acrónimo de Team Software Process o proceso del software en equipo.

Bibliografía

Naik, K. , Priyadarsi, T. (2008). *Software Testing and Quality Assurance*. Theory and Practice. New Jersey: Wiley.

Pressman, R. (2010). *Ingeniería del software, un enfoque práctico*. México: McGraw-Hill.

Alarcón, A. , Gonzalez, J. , Rodriguez, S. (2010). *Guía para pymes desarrolladoras de software, basada en la norma ISO/IEC 15504*. Revista Virtual Universidad Católica del Norte. Recuperado de <http://revistavirtual.ucn.edu.co/index.php/RevistaUCN/article/viewFile/339/651>


Scalone, F. (2006). *Estudio comparativo de los modelos y estándares de calidad del software (Tesis de maestría)*. Buenos Aires: Universidad tecnológica nacional. Recuperado de <http://laboratorios.fi.uba.ar/lsi/scalone-tesis-maestria-ingenieria-en-calidad.pdf>

Control del documento

**CONSTRUCCIÓN
OBJETO DE
APRENDIZAJE**



MODELOS DE CALIDAD EN EL DESARROLLO DE SOFTWARE	
Centro Industrial de Mantenimiento Integral - CIMI Regional Santander	
Líder línea de producción:	Santiago Lozada Garcés
Asesores pedagógicos:	Rosa Elvia Quintero Guasca Claudia Milena Hernández Naranjo
Líder expertos temáticos:	Rita Rubiela Rincón Badillo
Experto temático:	César Marino Cuéllar Chacón - V1
Experto temático:	Nelson Mauricio Silva Maldonado - V2
Diseño multimedia:	Jesús Antonio Vecino Valero
Programador:	Francisco José Lizcano Reyes
Producción de audio:	Victor Hugo Tabares Carreño



Este material puede ser distribuido, copiado y exhibido por terceros si se muestra en los créditos. No se puede obtener ningún beneficio comercial y las obras derivadas tienen que estar bajo los mismos términos de la licencia que el trabajo original.