**Unidad N°3: Calidad del Software**

* Calidad: concepto y visiones Garantía de calidad (SQA).
* Modelos de calidad: McCall, Boehm, ISO9126. Medición de la calidad. Mediciones de McCall.

Concepto de defecto. Medidas de aspectos de la calidad: Medición de densidad de defectos,

Mediciones de usabilidad, Mediciones de mantenimiento. Inspecciones de software.

* Introducción al CMMI. Nivel inicial. Niveles de madurez. Evolución del proceso. Estructura del CMMI.

Áreas clave del proceso por nivel. Áreas clave del proceso: estructura. Usando el CMMI. Ejemplo de área clave de proceso.

Introducción a la calidad del Software:

**¿Qué cosas ocurren frecuentemente en los proyectos de desarrollo de software?**

* Atrasos en las entregas
* Costos excedidos
* Falta cumplimiento de los compromisos
* No están claros los requerimientos
* El software no hace lo que tiene que hacer
* Trabajo fuera de hora
* ¿Dónde está ese componente?

Problemas Generales del Software:

* Carácter dinámico e iterativo a lo largo de todo el ciclo de vida
* Dificultades en conseguir productos totalmente depurados
* Dificultades en la gestión, estimación y medición.
* Alta dependencia de los proveedores
* Alto coste de los cambios

Causas:

* Ausencia de especificaciones completas, coherentes y precisas. Dificultad de obtenerlas.
* Ausencia de la aplicación sistemática de métodos, procedimientos y normas de ingeniería. Falta de organización.
* Escasez o ausencia de entornos integrados de producción
* Escasez de personal con formación y experiencia en nuevos métodos, normas y herramientas
* La propia cultura de las organizaciones. El software es considerado como gasto y no como inversión

La calidad es un concepto complejo y de facetas múltiples que puede describirse desde cinco diferentes puntos de vista:

* El trascendental dice que la calidad es algo que se reconoce de inmediato, pero que no es posible definir explícitamente.
* El del usuario concibe la calidad en términos de las metas específicas del usuario final. Si un producto las satisface, tiene calidad.
* El del fabricante la define en términos de las especificaciones originales del producto. Si éste las cumple, tiene calidad.
* El del producto sugiere que la calidad tiene que ver con las características inherentes de un producto.
* El basado en el valor la mide de acuerdo con lo que un cliente esta dispuesto a pagar por un producto

**Conceptos:**

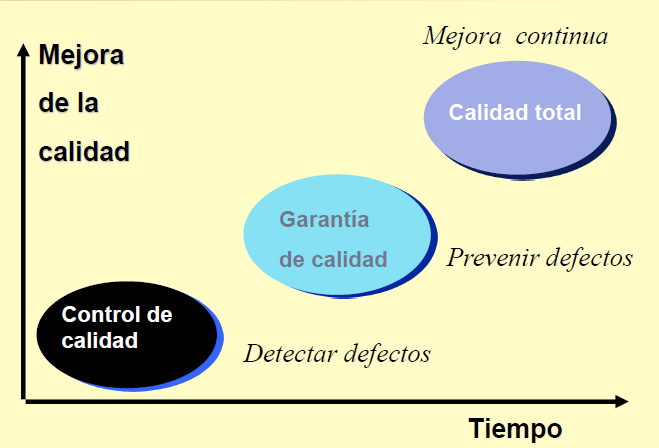
* Propiedad o conjunto de propiedades inherentes a una persona cosa que permiten diferenciarla con respecto a las que restan de su especie, como de mejor o peor calidad.
* Conjunto de características de un proceso o servicio que le confieren su aptitud para satisfacer las necesidades expresadas e implícitas.
* Conjunto de propiedades y de características de un producto o servicio, que le confieren aptitud para satisfacer las necesidades explicitas o implícitas.

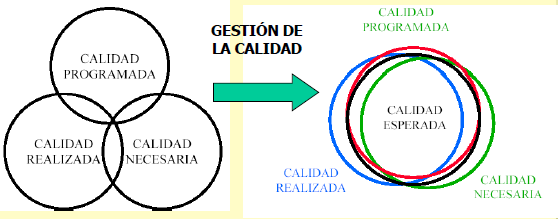
Conceptos de la calidad:

* No es absoluta
* Está sujeto a restricciones
* Trata de compromisos aceptables
* Los criterios de la calidad no son independientes
* La calidad está en permanente evolución
* Es multidimensional

Introducción a la calidad: La calidad puede incluir:

* Conformidad con las especificaciones
* Control del proceso
* Productos industriales “Idénticos”
* Conformidad con los requisitos y la confiablidad en el funcionamiento
* Cero defectos
* Adecuación del producto al uso
* Satisfacción de las expectativas del cliente:
* El mercado como eje de producto y servicio
* Dificultad para medir las expectativas
* Valor por dinero:
* Mercado basado en Precio y Calidad
* Preferencias del consumidor
* Excelencia:
* Aplicable a: Productos, Servicios, procesos, Empresas
* Lo mejor posible
* Un buen producto no es el que cumple con una determinada especificación, sino el que es bien percibido por el cliente





* Control de Calidad: “Conjunto de técnicas y actividades de carácter operativo, utilizadas para verificar los requerimientos relativos a la calidad del producto o servicio”
* Garantía de Calidad: “Conjunto de acciones planificadas y sistemáticas necesarias para proporcionar la confianza adecuada de que un producto o servicio satisfará los requerimientos dados sobre calidad”
* Calidad: “Conjunto de propiedades y características de un producto o servicio que le confieren su aptitud para satisfacer unas necesidades explícitas o implícitas”

Situación Actual:

* El producto (Software) es algo intangible y no regido por las leyes físicas.
* La disciplina, ingeniería de software, es relativamente reciente y muchos de sus conceptos importantes están aún inmaduros.
* Carencia de un corpus de conocimientos aceptado mayoritariamente que sirva como fundamentos

En una organización inmadura:

* Procesos software normalmente improvisados. Si se han especificado, no se siguen rigurosamente.
* Organización reactiva (resolver crisis inmediatas)
* Planes y presupuestos excedidos sistemáticamente
* Si hay plazos rígidos, se sacrifican funcionalidad y calidad del producto para satisfacer el plan
* No existen bases objetivas para juzgar la calidad del producto
* Cuando los proyectos están fuera de plan, las revisiones o pruebas se recortan o eliminan

**CALIDAD DE SOFTWARE:**

* “Conjunto de características de una entidad (producto o servicio) que le confiere su aptitud para satisfacer necesidades expresadas e implícitas”
* “la calidad del software es el grado con el que un sistema, componente o proceso cumple los requerimientos especificados y las necesidades o expectativas del cliente o usuario”
* “Concordancia con los requisitos funcionales y de rendimiento explícitamente establecidos con los estándares de desarrollo, explícitamente documentados y con las características implícitas que se espera de todo software desarrollado profesionalmente”
* “Concordancia del software producido con los requerimientos explícitamente establecidos, con los estándares de desarrollo prefijados y con los requerimientos implícitos no establecidos formalmente, que desea el usuario”
* “La totalidad de rasgos y atributos de un producto de software que le apoyan en su capacidad de satisfacer sus necesidades explícitas o implícitas”
* “Es el grado en el cual el software posee una combinación deseada de factores”

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. Existen tres puntos importantes:

1. Un proceso eficaz de software establece la infraestructura que da apoyo a cualquier esfuerzo de elaboración de un producto de software de alta calidad.
2. 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 el conjunto de requerimientos con los que se espera que cuente el software de alta calidad.
3. Al agregar valor para el producto 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.

Principios básicos del concepto de Calidad de Software:

* Debe ser construida durante análisis y diseño, no únicamente mediante la realización de revisiones y pruebas.
* Solo se alcanza con la contribución de todas las personas involucradas.
* Debe ser planificada y gestionada con eficacia.
* Dirigir esfuerzos a prevención de defectos.
* Reforzar los sistemas de detección y eliminación de defectos durante las primeras fases.
* Es un parámetro importante del proyecto al mismo nivel que los plazos de entrega, costos y productividad.
* Es esencial la participación de la dirección, que ha de propiciar la calidad.

Estándares y modelos de evaluación y mejora de los procesos software:

La garantía de calidad es el proceso que define cómo lograr calidad de software y cómo la organización de desarrollo conoce el nivel de calidad requerido en el software. El proceso de QA (**Quality Assurance**, o aseguramiento de la calidad) se ocupa ante todo de definir o seleccionar los estándares que deben ser aplicados al proceso de desarrollo de software o al producto de software.

Podemos definir dos tipos de estándares como parte del proceso de garantía de calidad:

* Estándares de producto: se aplican sobre el producto de software a desarrollar. Ej: de documentación, de definición de clases, de codificación.
* Estándares de proceso: definen los procesos que deben seguirse durante el desarrollo de software. Ej.: definición en los procesos de especificación, diseño y validación.

Los estándares son importantes por varias razones:

* Están basados en el conocimiento de la mejor o mas apropiada práctica de la empresa.
* Proveen un marco de trabajo alrededor del cual se implementa el proceso de garantía de calidad.
* Ayudan a la continuidad cuando una persona continua el trabajo que llevaba a cabo otra.

Algunos estándares son:

* ISO 9000 (ISO 9001:2000)
* (SPICE) ISO/IEC 15504
* CMM - CMMI
* Certificación. Organismos

**CALIDAD EN EL PRODUCTO:**

* El objetivo no es necesariamente alcanzar una calidad perfecta, sino la necesaria y suficiente para cada contexto de uso a la hora de la entrega y del uso por parte de los usuarios.
* Es necesario comprender las necesidades reales de los usuarios con tanto detalle como sea posible.
* Calidad del software como producto:
* ¿Hace lo que el usuario necesita?
* ¿Es lo que el usuario quiere?
* ¿Le soluciona el problema?
* ¿Lo hace como él quiere?
* ¿Se puede construir?
* ¿Es fácil de modificar, de corregir, de extender?
* ¿Se lo puede hacer en tiempo y forma, con costos bajos?
* ¿Me gusta? ¿Le gusta al usuario?

Hay un vinculo entre la calidad del proceso y del producto en producción debido a que el proceso es relativamente fácil de estandarizar y monitorizar. El desarrollo de software es un proceso mas creativo que mecánico donde la experiencia y habilidades individuales son importantes. La calidad del producto también se ve afectada por factores externos (novedad, presión comercial). En el desarrollo del software la relación entre la calidad del proceso y del producto es muy complejo.

**ATRIBUTOS DE CALIDAD:**

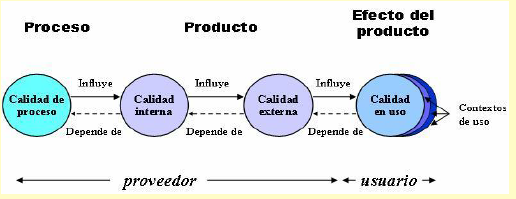
Los atributos de calidad son propiedades o características del sistema, que pueden afectar el grado de satisfacción de los interesados. Es decir, es el grado de concordancia entre las necesidades y el producto final.

Bass el al. (1998) establece una clasificación de los atributos de calidad en dos categorías:

* Observables vía ejecución (Externos): aquellos atributos que se determinan del comportamiento del sistema en tiempo de ejecución. Aquí se encuentra la: Disponibilidad, Confidencialidad, Funcionalidad, Desempeño, Confiablidad, Seguridad Externa (Safety), Seguridad interna (Security).
* No Observables vía ejecución (Internos): aquellos atributos que se establecen durante el desarrollo del sistema. Aquí se encuentran la: Configurabilidad, Integralidad, Integridad, Interoperabilidad, Modificabilidad, Mantenibilidad, Portabilidad, Reusabilidad, Escalabilidad, Capacidad de Prueba (Testability).

Existen diferentes aspectos de calidad:

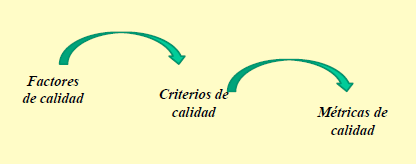
* Interna: medible a partir de las características intrínsecas, como el código fuente.
* Externa: medible en el comportamiento del producto, como en una prueba.
* En uso: durante la utilización efectiva por parte del usuario.



* Calidad del producto: Correctitud, usabilidad, mantenibilidad, confiablidad, rendimiento, disponibilidad, robustez, performance, amigabilidad, reusabilidad, portabilidad, etc.
* Calidad del proceso: El proceso debe estar definido, documentado y debe ser practicado y medido.
* Criterios de calidad: es necesario establecer criterios para medir y evaluar la calidad del producto y del proceso.

**MODELOS DE CALIDAD:**

Los modelos de calidad tratan de poner en práctica el concepto de calidad. La calidad del software se puede describir de manera jerárquica.



**Modelo de McCall:**

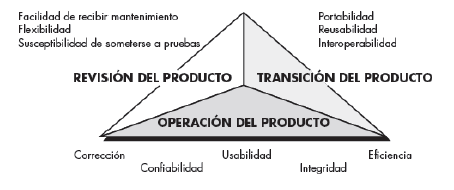
El modelo de McCall fue el primero en ser presentado en 1977, se originó motivado por US Air Force y DoD.

Este 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 demasiado abstractos para ser medidos directamente, por lo que por cada uno de ellos se introducen atributos de bajo nivel denominados criterios de calidad.

Los factores que afectan la calidad del software se centran en tres aspectos importantes del producto de software: sus características operativas, su capacidad de ser modificado y su adaptabilidad a nuevos ambientes, es decir se centran en **tres aspectos importantes** de un producto de software (McCall):

* **Revisión del producto**: habilidad para ser cambiado.
* **Transición del producto**: adaptabilidad al nuevo ambiente.
* **Operación del producto**: características de operación.



* Características operativas:
* Corrección. ¿Hace lo que quiero?
* Fiabilidad. ¿Lo hace de forma fiable todo el tiempo?
* Eficiencia. ¿Se ejecutará en mi hardware lo mejor que pueda?
* Seguridad (integridad). ¿Es seguro?
* Facilidad de Uso. ¿Está diseñado para ser usado?
* Capacidad de soportar los cambios:
* Facilidad de mantenimiento. ¿Puedo corregirlo?
* Flexibilidad. ¿Puedo cambiarlo?
* Facilidad de Prueba. ¿Puedo probarlo?
* Adaptabilidad a nuevos entornos:
* Portabilidad. ¿Podré usarlo en otra máquina?
* Reusabilidad. ¿Podré reutilizar alguna parte del software?
* Interoperabilidad. ¿Podré hacerlo interactuar con otro sistema?

La revisión del producto incluye los siguientes factores de calidad:

* Mantenibilidad: esfuerzo requerido para localizar y corregir fallas. El factor mantenibilidad incluye los siguientes criterios: consistencia, simplicidad, concisidad, autodescripción y modularidad.

Ghezzi la divide en tres categorías:

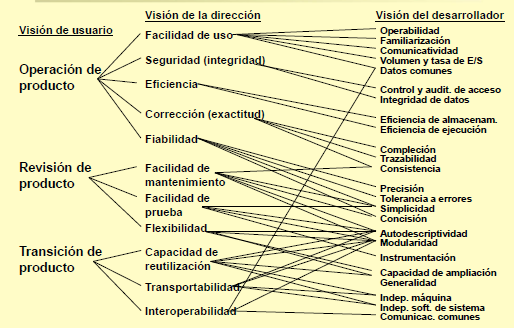
* Correctiva: concerniente a remover pequeñas fallas remanentes después del testeo.
* Adaptativa: concerniente al cambio del producto necesario por el cambio de sus requerimientos.
* Perfectiva: busca solo mejorar los algoritmos usados para hacerlos mas eficientes.
* Flexibilidad: facilidad de realizar cambios. Esfuerzo necesario para modificar un programa que ya opera.
* Testeabilidad: facilidad para realizar el testing, para asegurarse que el producto no tiene errores y cumple con las especificaciones. Esfuerzo que se requiere para probar un programa a fin de garantizar que realiza la función que se pretende.

La transición del producto 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.

La operación del producto incluye los siguientes factores de calidad:

* **Correctitud:** el grado en el que el producto cumple con sus especificaciones.
* **Confiabilidad**: la habilidad del producto de responder ante situaciones no esperadas, el factor de confiabilidad incluye los siguientes criterios: tolerancia a errores, consistencia, simplicidad y exactitud. Combina la tolerancia tanto a errores de hardware como de software. Puede ser medido con medidas como:
* Tiempo medio entre fallas
* Tiempo medio antes de mantenimiento
* Tiempo medio antes de recuperación
* Probabilidad de falla
* **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 usabilidad incluye los siguientes criterios: operabilidad, entrenamiento, comunicación, volumen de E/S y tasa de E/S. la usabilidad se puede subdividir en:
* Ergonomía general: el equipo es adecuado para el uso previsto.
* Ergonomía de Software: estilos de diálogos, metáforas, diseño de pantallas, etc.



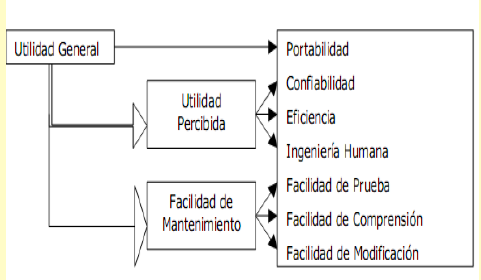
**Modelo de Boehm:**

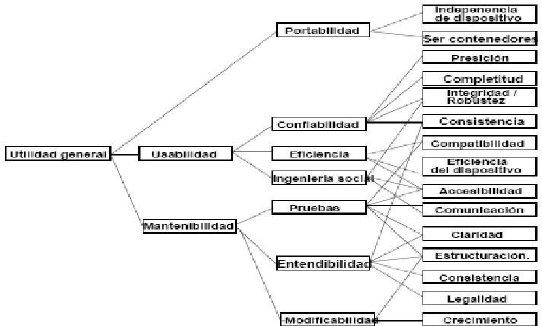
El modelo que presenta Boehm presenta una jerarquía de características donde cada una de ellas contribuye a la calidad global. Se centra en:

* Sus características operativas.
* Su capacidad para soportar los cambios.
* Su adaptabilidad a nuevos entornos.
* La evaluación del desempeño del hardware.

El modelo comienza con la utilidad general del software, afirmando que el software es útil, evitando pérdida de tiempo y dinero. La utilidad puede considerarse en correspondencia a los tipos de usuario que quedan involucrados.

* El primer tipo de usuario queda satisfecho si el sistema hace que el pretende que haga
* El segundo tipo es aquel que utiliza el sistema luego de una actuación.
* El tercero, es el programador que mantiene el sistema.





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:

* Portabilidad
* Confiabilidad
* Eficiencia
* Usabilidad
* Testeabilidad
* Facilidad de entendimiento
* Modificabilidad o flexibilidad

Características Primitivas: el nivel mas bajo corresponde a características directamente asociadas a una o dos métricas de calidad:

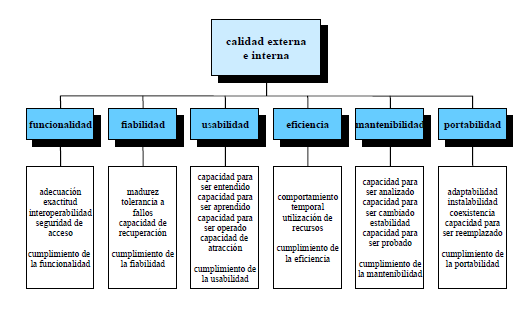
* De Portabilidad: independencia de dispositivos.
* De confiabilidad: exactitud, completitud, consistencia, robustez/integridad.

ISO/IEC 9126: Tecnologías de la Información – calidad de los productos Software

* Parte 1: Modelo de Calidad
* Parte 2: Métricas Externas
* Parte 3: Métricas Internas
* Parte 4: Métricas de Calidad en Uso

**Ejemplos de Uso:**

* Validar la compleción de una definición de requerimientos.
* Identificar requerimientos software.
* Identificar objetivos para el diseño software.
* Identificar requerimientos para las pruebas del software.
* Identificar requerimientos para el aseguramiento de la calidad
* Identificar criterios de aceptación para un producto software terminado.



El estándar ISO 9126 se desarrolló con la intensión 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:
* Adecuación: capacidad del producto software para proporcionar un conjunto apropiado de funciones para tareas y objetivos de usuarios especificados.
* Exactitud: capacidad del producto software para proporcionar los resultados o efectos correctos o acordados, con el grado necesario de precisión.
* Interoperabilidad: Capacidad del producto software para proteger información y datos de manera que las personas o sistemas no autorizadas no puedan leerlos o modificarlos, al tiempo que no se deniega el acceso a las personas o sistemas autorizados.
* Cumplimiento funcional: capacidad del producto software para adherirse a normas, convenciones o regulaciones en leyes y prescripciones similares relacionadas con funcionalidad.
* Confiabilidad: Cantidad de tiempo que el software se encuentra disponible para su uso, según lo indican los siguientes atributos:
* Madurez: Capacidad del producto de software para mantener un nivel especificado de prestaciones en caso de fallos software o de infringir sus interfaces especificadas.
* Tolerancia a fallos: capacidad del software para mantener un nivel especificado de prestaciones en caso de fallos software o infringir sus interfaces especificados.
* Capacidad de recuperación: Capacidad del producto software para reestablecer un nivel de prestaciones especificado y de recuperar los datos directamente afectados en caso de fallos.
* Cumplimiento de la fiabilidad: Capacidad del producto software para adherirse a normas, convenciones o regulaciones relacionadas con la fiabilidad.
* Usabilidad: Grado en el que el software es fácil de usar, según lo indican los siguientes sub atributos:
* Capacidad para ser entendido: capacidad del producto software que permite al usuario entender si el software es adecuado y cómo puede ser usado para unas tareas o condiciones de uso particulares.
* Capacidad para ser aprendido: capacidad del producto software que permite al usuario aprender sobre su aplicación.
* Capacidad para ser operado: Capacidad del producto, software que permite al usuario operarlo y controlarlo.
* Capacidad de atracción: Capacidad del producto software para ser atractivo al usuario.
* Cumplimiento de la usabilidad: capacidad del producto software para adherirse a normas, convenciones, guías de estilo o regulaciones relacionadas con la usabilidad.
* Eficiencia: grado en el que el software emplea óptimamente los recursos del sistema, según los indican los sub atributos siguientes:
* Comportamiento temporal: capacidad del producto software para proporcionar tiempos de respuesta, tiempos de proceso y potencia apropiados, bajo condiciones determinadas.
* Utilización de recursos: capacidad del producto software para usar las cantidades y tipos de recursos adecuados cuando el software lleva a cabo su función bajo condiciones determinadas.
* Cumplimiento de la eficiencia: capacidad del producto software para adherirse a normas o convenciones relacionadas con la eficiencia.
* Mantenibilidad: Facilidad con la que pueden efectuarse reparaciones al software, según lo indican los atributos siguientes:
* Capacidad para ser analizado: es la capacidad del producto software para serle diagnosticados deficiencias o causas de los fallos en el software, o para identificar las partes que han de ser modificadas.
* Capacidad para ser cambiado: capacidad del producto software que permite que una determinada modificación sea implementada.
* Estabilidad: capacidad del producto software para evitar efectos inesperados debidos a modificaciones del software.
* Capacidad para ser probado: capacidad del producto software que permite que el software modificado sea validado.
* Cumplimiento de la mantenibilidad: Capacidad del producto software para adherirse a normas o convenciones relacionadas con la mantenibilidad.
* Portabilidad: facilidad con la que el software puede llevarse de un ambiente a otro según la indican los siguientes atributos: adaptable, instalable, conformidad y sustituible.

**ASEGURAMIENTO DE LA CALIDAD DEL SOFTWARE SQA**

Es el conjunto de actividades planificadas y sistemáticas necesarias para adoptar la confianza en que el producto (software) satisfará los requerimientos dados de calidad. El aseguramiento de calidad del software se diseña para cada aplicación antes de comenzar a desarrollarla y no después.

Es una serie de actividades que tienen que ser implementadas a través del proceso de desarrollo de software que incluyen actividades para asegurar alta calidad de producto, pruebas de aseguramiento de calidad y métricas para desarrollar estrategias que mejorarán el proceso de software.

**Función de SQA:**

Mejorar la calidad de los procesos de desarrollo y mantenimiento del software, monitoreando, durante el transcurso del ciclo de vida de los diferentes proyectos, el cumplimiento de estándares y procesos establecidos, antes de su puesta en productivo.

**Actividades:**

* Establecer planes, estándares y procesos que satisfagan las políticas de la organización y se ajusten a las necesidades de cada proyecto en particular.
* Revisar y auditar los productos y actividades desarrolladas para verificar que ellos satisfacen los procesos y estándares definidos.
* Proveer las herramientas necesarias que den soporte al proceso definido, facilitando su ejecución, visualización, administración y seguimiento.
* Proveer al equipo de proyecto y a otros interesados, los resultados sobre las revisiones, auditorias y actividades.
* Escalar problemas no resueltos dentro del equipo de un proyecto hacia un nivel apropiado de administración para su resolución.

También incluye: métricas de software para el control del proyecto, verificación y validación del software a lo largo del ciclo de vida lo cual incluye las pruebas y los procesos de revisión e inspección y Gestión de la configuración del software.

**Plan de Calidad:**

Mapa para institucionalizar la garantía de calidad del software. Es una planilla para definir las actividades de SQA aplicables a cada proyecto de software. El plan incluye:

* Sección Gestión: Tareas y actividades de SQA dentro del proceso de software y los roles y responsabilidades relativas a la calidad del producto.
* Sección Documentación: Detalle de los productos de trabajo del proceso de software que podrán ser revisados.
* Sección Estándares, Prácticas y Convenciones: Detalle de lo que está acordado y establecido para el proceso y los productos a obtener.
* Sección Revisiones y Auditorías: Revisiones que se llevarán a cabo durante el proceso y los responsables de cada una de ellas. (Ejemplo: revisiones de documentación, revisiones técnico formales, etc.)
* Sección de Pruebas: Plan y procedimiento de pruebas del software y de gestionar los defectos detectados.
* Sección Métodos y Herramientas que soportan las actividades de SQA.

Estándares de Calidad Serie ISO 9000

* ISO/IEC 9000-3: Lineamientos para la aplicación de la Norma ISO 9001 en el desarrollo, suministro y mantenimiento del Software
* ISO/IEC 9000-4: Guía para la gestión de un programa de seguridad de funcionamiento
* ISO/IEC 10007: Directrices para la gestión de la configuración
* ISO/IEC 9126-1: Software Quality Characteristics and Metrics
* ISO/IEC 12207: Software Life Cycle Processes
* ISO/IEC 14102: Information Technology -Guidelines for the evaluation and selection of CASE tools
* ISO/IEC 15026: System and Software Integrity Levels
* ISO/IEC 15271: Guide to ISO/IEC Software Life Cycle Processes
* ISO/IEC 15504: Software Process Assessment
* ISO/IEC 15846: Software Configuration Management
* ISO/IEC 17799: Seguridad Informática

**DEFECTO:**

En el contexto del proceso de software, los términos defecto y falla son sinónimos. Los dos implican un problema de calidad descubierto después de haberse liberado el software a los usuarios finales (o a otra actividad estructural del proceso del software)

* Error: problema de calidad que se detecta antes de que el software se entregue a los usuarios finales.
* Defecto: problema de calidad que se encuentra después de haber entregado el software a los usuarios finales.
* Falla: es la discrepancia visible que se produce al ejecutar un programa con un defecto, el cual es incapaz de funcionar correctamente.

Inspecciones de software:

Proceso de Verificación y Validación estático (es diferente a las pruebas) en el que el sistema de software se revisa para encontrar errores, omisiones y anomalías. Generalmente, se centran en el código fuente, pero puede inspeccionarse también modelos de diseños, o especificaciones de requerimientos.

Métricas para la Inspección:

* Esfuerzo de preparación, Ep: esfuerzo (en horas-hombre) requerido para revisar un producto del trabajo antes de la reunión de revisión real.
* Esfuerzo de evaluación, Ea: esfuerzo requerido (en horas-hombre) que se dedica a la revisión real.
* Esfuerzo de la repetición, Er: esfuerzo (horas-hombre) que se dedica a la corrección de los errores descubiertos durante la revisión.
* Tamaño del producto del trabajo, TPT: medición del tamaño del producto del trabajo que se ha revisado (por ejemplo, números de modelos UML o número de páginas de documento o de líneas de código)
* Errores menores detectados, Err menores: número de errores detectados que pueden clasificarse como mayores.

El esfuerzo total de inspección y el número total de errores descubiertos se definen como sigue:

E revisión: Ep + Ea + Er

Err total: Err menores + Err mayores

La densidad del error representa los errores encontrados por unidad del producto del trabajo revisada:

Densidad del error: Err total / TPT

MEDICIONES DE ATRIBUTOS DE CALIDAD:

Es imposible medir los atributos de calidad de software directamente. Los atributos de calidad como la mantenibilidad, usabilidad son atributos externos que nos dicen cómo ven el software los desarrolladores y los usuarios. Sin embargo, existe una relación entre los atributos internos y externos del software.

Por ejemplo:

* Mantenibilidad esta relacionado con el número de parámetros del procedimiento, la complejidad ciclomática, el tamaño del programa en líneas de código y la extensión del manual de usuario.
* Usabilidad: está relacionado con número de mensajes de error y la extensión del manual de usuario.

Las métricas de producto se refieren a las características del software. Éstas están relacionadas con diversos atributos de calidad. Las métricas dinámicas (recogidas de mediciones hechas en el programa) ayudan a valorar la eficiencia y fiabilidad de un programa.

Las métricas estáticas (recogidas de mediciones hechas en las representaciones del sistema como diseño, el programa o la documentación) ayudan a valorar la complejidad, comprensión y mantenibilidad de un sistema de software. Ejemplo de estas métricas: Longitud de código, complejidad ciclomática.

**MEJORA DEL PROCESO SOFTWARE:**

Frente a la “Crisis del Software”, donde los proyectos se caracterizan por:

* Terminar fuera de tiempo.
* Con mas costos de los programadores
* Con menos funcionalidad que la deseada
* Con dificultad para cerrar proyectos
* Y con dudosa calidad

Surge la necesidad de la creación de modelos de calidad para empresas de software.

La mejora de procesos significa entender los procesos existentes y cambiarlos para mejorar la calidad del producto y/o reducir los costos y el tiempo de desarrollo.

Los procesos también tienen atributos ej: comprensión (¿hasta qué punto se define completamente el proceso y como de fácil es comprender su definición?), aceptación (¿le proceso definido es aceptable y utilizable por los ingenieros responsables de producir software?), robustez (¿puede continuar el proceso a pesar de los problemas inesperados?), no es posible hacer la mejora de procesos para todos sus atributos simultáneamente.

La mejora de procesos es una actividad cíclica y tiene 3 estados principales:

1. Proceso de medición de los atributos del proyecto actual o del producto. El objetivo es mejorar las mediciones de acuerdo con las metas de la organización involucrada.
2. Proceso de análisis, el proceso es valorado y se identifican puntos flacos y cuellos de botella.
3. Introducción de lo cambios.

La mejora de procesos es una actividad a largo plazo y continúa ya que el entorno del negocio cambia y los procesos evolucionan para tener en cuenta los cambios.

El termino mejoramiento del proceso de software (MPS) implica:

* Que los elementos de un proceso de software efectivo pueden definirse en forma efectiva.
* Que un enfoque organizacional existente sobre el desarrollo del software puede valorarse en contraste con dichos elementos.
* Que es posible definir una estrategia de mejoramiento significativa.

**Modelo:**

Un modelo es una representación del mundo real, los cuales ayudan a entender y aplicar ideas de desarrollo de procesos. Un modelo se usa:

* Para determinar objetivos y prioridades de mejoramiento
* Para asegurar procesos estables, maduros y capaces.
* Como una guía para mejoramiento organizacional
* Como una guía de asesoramiento, para diagnosticar el estado de los esfuerzos de mejoramiento.

**MODELO DE MADURACIÓN DE CAPACIDADES:**

Un modelo de madurez se aplica dentro del contexto de un marco conceptual MPS. La intensión del modelo de madurez es proporcionar un indicio global de la “madurez del proceso” que muestra una organización de software, es decir, un indicio de la calidad del proceso de software, el grado en el que los profesionales entienden y aplican el proceso, y el estado general de la práctica de ingeniería de software.

**MODELO DE MADUREZ DE CAPACIDADES INTEGRADO (CMMI):**

El SEI se embarcó en un nuevo programa para desarrollar un modelo de capacidades integrado que integra otros modelos de ingeniería, utiliza las mejores prácticas de Ingeniería de Software (IS), proporciona guías para mejorar los procesos de la organización y la capacidad para gestionar el desarrollo, adquisición, mantenimiento de productos y servicios.

CMMI es un metamodelo de proceso exhaustivo que se impulsa en un conjunto de sistemas y capacidades de ingeniería del software que deben presentarse conforme las organizaciones alcanzan diferentes niveles de capacidad y madurez del proceso.

Tiene dos instancias en etapas que es compatible con CMM de software y permite un desarrollo del sistema de la organización, la gestión de los procesos a valorar y su asignación a un nivel de Madurez entre 1 y 5, y su versión continua permite una clasificación mas detallada y considera 24 área de procesos.

CMMI es un modelo que contiene cinco niveles de madurez los cuales describen un camino evolutivo que va desde una organización caótica a una organización con procesos de software maduros y disciplinarios. Está basado en conceptos de Calidad Total y de mejoramiento continuo y fue creado por el SEI en el año 2000 basado en modelos prexistentes.

**Beneficios:**

* Reducción de costos por:
* Estimaciones basadas en hechos
* Reducción de reprocesos
* Acuerdos claros sobre el servicio y la funcionalidad del producto a entregar
* Aumento en la confiabilidad por:
* Reducción consistente de errores
* Cumplimiento consistente de fechas
* Mayor efectividad por:
* Visibilidad sobre el proceso y sobre el producto
* Operar con estándares documentados
* Personal entregado

**Formas de Implementar:**

CMMI provee dos representaciones:

**Continua**

* Permite entre otras cosas, seleccionar el orden de implementación que mejor reúna los objetivos de la organización
* Los niveles indican el grado de capacidad del proceso.
* Los niveles se definen para cada área de proceso. Existen 6 niveles (0 – 5)

Esta representación continua favorece la flexibilidad en el orden hacia el cual se dirigen las mejoras. A medida que una organización del SW adquiere madurez en su proceso, esta lo institucionaliza.

**En etapas**

* Provee una probada secuencia de mejoras, comenzando con prácticas básicas de gestión y progresando por medio de un predefinido y probado camino de sucesivos niveles. Cada uno de estos niveles es el fundamento para el siguiente.
* Los niveles indican el grado de madurez organizacional
* Los niveles se basan en agrupamientos de área de proceso. Existen 5 niveles (1 – 5)

La principal diferencia entre el modelo continuo y el en etapas, es que el primero define niveles de capacidad mientras que en el segundo define niveles de madurez. Los niveles de madurez son:

* Nivel 1: Inicial
* Nivel 2: Administrado
* Nivel 3: Definido
* Nivel 4: Administrado cuantitativamente
* Nivel 5: Optimizado

Las principales ventajas de la representación en etapas son:

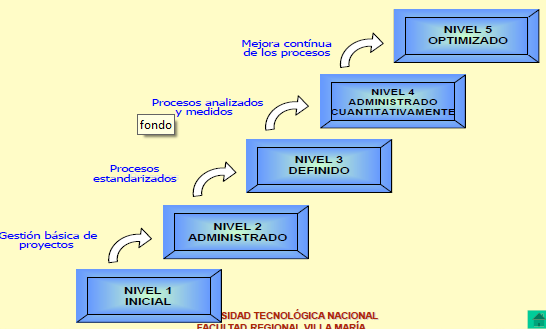
* Secuencia contrastada de mejoras, inicia con la práctica de gestión básica de proyectos
* Permite comparaciones dentro de la organización y entre diferentes organizaciones
* Fácil migración desde el modelo SW-CMM al CMMI
* Permite efectivizar una mejora no solo de productos sino también de procesos.

En cuanto a las desventajas podemos mencionar que la adopción de este modelo puede resultar costosa para organizaciones pequeñas.

**Disciplinas del modelo:**

* Ingeniería de Sistemas: desarrollo total de sistemas, software o no software
* Gestión de Proveedores: adquisición de productos de trabajos externos
* Desarrollo integrado de productos y procesos: acercamiento sistemático, colaboración entre partes, satisfacer requerimientos de clientes.
* Ingeniería de software: desarrollo de sw, aplicación sistemática, disciplina, cuantitativa, desarrollo, operación, mantenimiento.

**Niveles de Madurez:**



1. **INICIAL:** la empresa no dispone de procesos y controles definidos. Se trabaja con procedimientos que no están normalizados, es decir, procedimientos tanto del propio desarrollo de software como de su planificación y control, que no están establecidos explícitamente antes de su uso.

La característica de las empresas que se encuentran en este nivel es que no hay un control de la gestión de proyectos de software efectivo, porque puede suceder que la empresa disponga de procedimientos y técnicas formales, tanto de gestión como del proceso, y de herramientas, pero no se utilizan de manera estándar en todos los proyectos.

1. **REPETIBLE**: La empresa tiene métodos estandarizados facilitando procesos repetibles.

Las empresas que se encuentran en este nivel son las que disponen de un control básico de la gestión de proyectos, gestión de calidad y gestión de la configuración.

El problema en este tipo de organización es que introducir cualquier cambio tiene un alto grado de riesgo de fracaso

1. **DEFINIDO:** los procesos están bien caracterizados o comprendidos. Ya se cuenta con estándares, procedimientos, herramientas, y métodos. Las empresas que se encuentran en este nivel se caracterizan por disponer de:

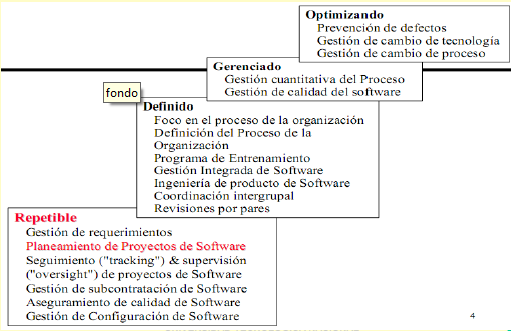
* Un grupo de procesos, cuyo objetivo es el de mejorar el proceso software
* Una metodología de desarrollo software que describa las actividades técnicas y de gestión requeridas para la adecuada ejecución del proceso.

1. **ADMINISTRADO CUANTITATIVAMENTE:** la empresa posee controles avanzados, métricas y retroalimentación.

Las empresas que han alcanzado este nivel disponen de un control de los costos y calidad de las principales etapas del proceso. Es pre requisito que exista una metodología de desarrollo software para realizar una medición efectiva.

1. **OPTIMIZACION:** la empresa emplea métricas con propósitos de optimización.

En este nivel, las organizaciones se encuentran en un proceso de mejora continua. Se usan todos los procesos y técnicas modernas, lo mismo que la administración cuantitativa. Las organizaciones se enfocan en la mejora a través de técnicas y procesos de prevención de defectos, cambios de tecnología y cambios en procesos.

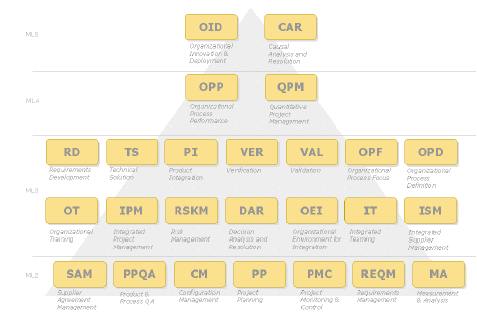


**Estructura:**

CMMI plantea que las organizaciones pueden ubicarse en alguno de cinco posibles niveles de madurez, dependiendo del grado de sofisticación de sus procesos.

1. INICIAL
2. ADMINISTRADO: Administración básica de proyectos
3. DEFINIDO: Proceso estandarizado
4. ADMINISTRADO CUANTITATIVAMENTE: Proceso administrado cuantitativamente
5. OPTIMIZADO: Proceso en mejora continua

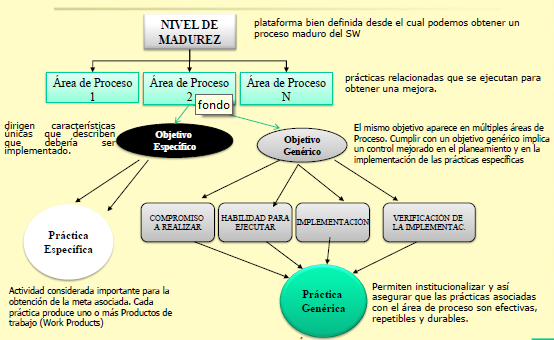
A su vez, cada nivel de madurez, con excepción del inicial, queda caracterizado por un conjunto de áreas de proceso que agrupan prácticas que, al ser ejecutadas colectivamente, permiten cumplir con algún objetivo que es considerado importante para el modelo. Cabe destacar que antes de introducir prácticas de un nivel determinado deben estabilizarse las prácticas del nivel anterior. En la siguiente imagen podemos ver la representación por niveles (áreas de proceso por nivel de madurez):



Además de poder ver las áreas de proceso por nivel de madurez, el modelo propone una vista alternativa (llamada continua) en donde las áreas están agrupadas por categoría, donde las mismas son: Gestión de Proyecto, Soporte, Ingeniería y Gestión de Procesos. En la siguiente imagen podemos ver la representación continua (áreas del proceso por categoría):



En la representación por niveles, cada nivel de madurez contiene varias áreas de proceso, las que a su vez quedan definidas por uno o varios objetivos específicos y un objetivo genérico. Cada uno de ellos tiene vinculado un conjunto de prácticas, llamadas específicas y genéricas respectivamente.



**Objetivos y prácticas Genéricas:**

Los objetivos y practicas genéricas tienen que ver con el grado de institucionalización de los procesos (compromiso con la ejecución, capacidad para ejecutar, dirección de la ejecución, verificación de la ejecución). Son llamados así porque son los mismos en todas las áreas de proceso (aunque hay aspectos específicos para cada una de ellas).

Objetivos genéricos:

* Compromiso para la ejecución: Acciones que deben tomarse para asegurar que el proyecto se establezca y perdure.
* Capacidad para ejecutar: precondiciones que deben existir (en el proyecto u organización) para implementar el proceso.
* Actividades ejecutadas: Roles y procedimientos necesarios para implementar un área clave de proceso.
* Mediciones y análisis: Medir el proceso y analizar las mediciones.
* Verificación de implementación: pasos para asegurar que las actividades se ejecutan de acuerdo con el proceso que se ha implementado.

Objetivos y Prácticas Específicas:

Los objetivos y prácticas especificas están vinculadas a un área de proceso determinada. Son considerados elementos que deben ser satisfechos para implementar exitosamente los procesos relacionados con un área de proceso en particular.

**Áreas del nivel Gestionado – Nivel II:**

* Gestión de Requerimientos (REQM): gestiona los requerimientos del proyecto y de sus componentes e identifica inconsistencias.
* Planificación del Proyecto (PP): establece y mantiene los planes que definen las actividades del proyecto.
* Seguimiento y control del proyecto (PMC): comprensión del progreso del proyecto y la aplicación de medidas correctivas cuando el plan del proyecto se desvía.
* Acuerdos con los proveedores (SAM): adquisición de productos y servicios de proveedores externos con los cuales existen acuerdos.
* Análisis y mediciones (MA): desarrolla y mantiene una capacidad de medición que se utiliza en el soporte de gestión de la información.
* Garantía de la calidad del proceso y del producto (PPQA): provee personal y gestión con comprensión objetiva del proceso y los productos del trabajo asociados.
* Gestión de las configuraciones (CM): establece y mantiene la integridad de los productos de trabajo usando identificación, control y estado de configuraciones.

Lo que permite a un proceso ser gestionado es:

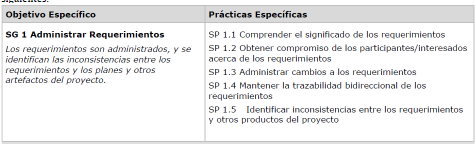
* Establecer y mantener una política organizacional de planificación y mejora del proceso de planificación.
* Proveer los recursos adecuados para mejorar el proceso de gestión del proyecto, desarrollando las herramientas y proveyendo los servicios del proceso.
* Seguimiento y control del proyecto y aplicación de las medidas correctivas adecuadas cuando sea necesario.
* Revisión de actividades, estado y resultados del proceso de planificación del proyecto con una gestión a alto nivel y resolución de problemas.

Para el nivel 2 se define el siguiente objetivo genérico (GG), y las siguientes prácticas genéricas (GP):

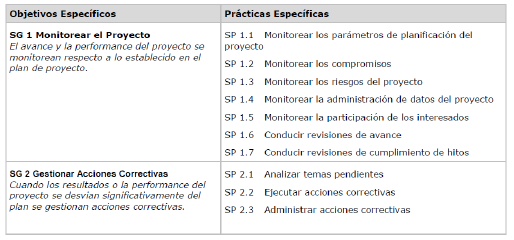
GG 2: Institucionalizar un proceso administrativo.

* GP 2.1: Establecer políticas organizacionales
* GP 2.2: Planificar el proceso
* GP 2.3: Proveer Recursos
* GP 2.4: Asignar Responsabilidades
* GP 2.5: Entrenar el personal
* GP 2.6: Administrar la configuración
* GP 2.7: Identificar e Involucrar a los interesados
* GP 2.8: Monitorear y controlar los procesos
* GP 2.9: Evaluar adhesión objetivamente
* GP 2.10: Revisar el estado con la alta gerencia

Por ejemplo, en el área Administración de Requerimientos (REQM), los objetivos y prácticas específicos son las siguientes:



Otro ejemplo para el área Seguimiento y Control de proyectos (PMC), los objetivos y prácticas específicos son las siguientes:



**Áreas del Nivel Definido - Nivel III:**

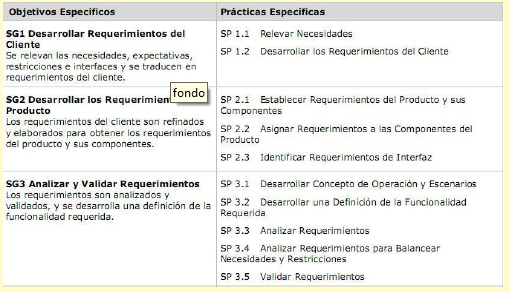
* Desarrollo de Requerimientos (RD)
* Solución Técnica (TS)
* Integración del Producto (PI)
* Verificación (VE)
* Validación (VA)
* Enfoque organizacional en el Proceso (OPF)
* Definición organizacional del Proceso (OPD)
* Entrenamiento Organizacional (OT)
* Gestión del Riesgo (RSKM)
* Análisis y toma de decisiones (DAR)
* Administración integrada del proyecto (IPM)
* Gestión integrada de Proveedores (SS)(ISM)
* Ambiente organizacional para la integración (IPPD)(OEI)
* Equipo Integrado (IPPD)(IT)

Para el nivel 3 se define el siguiente objetivo genérico (GG), y las siguientes prácticas genéricas (GP):

GG 3: Institucionalizar un proceso definido

* GP 3.1 establecer un proceso definido
* GP 3.2 Recolectar información para mejoras

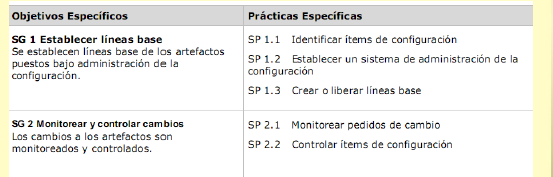
Por ejemplo, para el área Desarrollo de Requerimientos (RD), los objetivos y prácticas específicos son las siguientes:



Gestión de Riesgos:



Administración de las configuraciones:



Unidad N°5: Estimación para proyectos de Software

▪ Estimación de proyecto de software

▪ Técnicas de descomposición

▪ Modelos empíricos de estimación

▪ Estimación para proyectos orientados a objetos

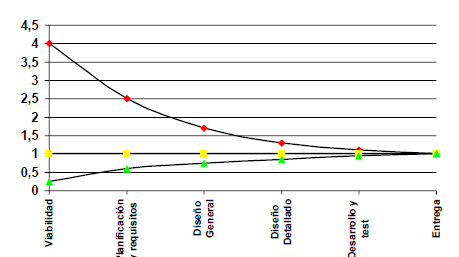
Estimación de proyectos de Software: actividad de la planificación del proyecto de software que intenta determinar cuánto dinero, esfuerzo, recursos y tiempo tomará construir un sistema o producto software.

La estimación es la aplicación continua de técnicas basadas en las medidas de los procesos de desarrollo del software y sus productos, para producir una información de gestión significativa y a tiempo. Esta información se utilizará para mejorar esos procesos los productos que se obtienen de ellos.

Factores que afectan al riesgo de estimación:

* La complejidad del proyecto tiene un fuerte efecto sobre la incertidumbre inherente a la planificación. Sin embargo, la complejidad es una medida relativa que es afectada por la familiaridad con el esfuerzo pasado.
* El tamaño del proyecto es otro factor importante que puede afectar la precisión y la eficacia de las estimaciones. Conforme aumenta el tamaño, la interdependencia entre varios elementos del software crece rápidamente.
* El grado de incertidumbre estructural también tiene un efecto sobre el riesgo de estimación. En este contexto, estructura se refiere al grado en el cual se solidificaron los requisitos, la facilidad con la que se dividieron las funciones y la naturaleza jerárquica de la información que debe procesarse.
* Disponibilidad de información histórica: métricas sobre proyectos pasados.

Precisión de las estimaciones en función de la fase del proyecto:



Mientras más conozca, mejor estimará. En consecuencia, actualice sus estimaciones conforme avance el proyecto.

Objetivos de la estimación:

* Predecir las variables involucradas en el proyecto con cierto grado de certeza.
* Trata de aportar una predicción de algún indicador importante para la gestión de proyectos de software tiempo, esfuerzo, cantidad de defectos esperados, entre otros.
* Es razonable conocer, antes de comenzar a desarrollar el SW, cuanto se va a invertir, qué tareas se deben realizar y cuánto tiempo se necesitará.

Necesidad de un estimador:

Se requiere un estimador temprano, independiente del lenguaje y fácil de calcular. El estimador debe ser un profesional que no tenga ningún interés, directo o indirecto, en los resultados del proceso de estimación y que esté únicamente guiado por su profesionalismo.

El principal objetivo del estimador es obtener estimaciones de calidad, las cuales no tienen siempre porqué coincidir con las expectativas de la empresa en términos de costo y tiempo.

Requisitos que debe tener un buen estimador:

Formación y experiencia profesional adecuada.

Una posición en la organización que le permita adoptar un juicio independiente.

Debe basarse en un método que pueda ser explicado, cuestionado, discutido y auditado.

Debe poder describir su experiencia en cada estimación.

Debe documentar su estimación, incluyendo los resultados obtenidos y cualquier información necesaria para hacer el proceso de estimación repetible y verificable.

¿Cuándo se debe llevar a cabo la estimación?

* La estimación es un proceso continuo. A medida que el proyecto avanza, más se conoce de él, y por lo tanto más parámetros están disponibles para introducir en un modelo de estimación.
* La estimación continua nos permite el uso de un único modelo coherente que pueda capturar y utilizar la información sobre el proyecto a medida que éste se conozca.
* El proceso de estimación comienza usando unas pocas variables claves para proveer las «macro características» de un proyecto, y evoluciona incorporando información de más bajo nivel para producir las «micro características» del proyecto.

Problemas de Estimación:

* Problemas políticos: cuando las estimaciones se convierten en objetivos, cuando se ajusta el precio por conveniencia.
* Problemas Técnicos: No existen datos históricos para estimar.

Planificación de proyectos

El objetivo de la planificación del proyecto de software es proporcionar un marco conceptual que permita al gerente hacer estimaciones razonables de recursos, costo y calendario. Además, las estimaciones deben intentar definir los escenarios de mejor caso y peor caso, de modo que los resultados del proyecto puedan acotarse. Aunque hay un grado inherente de incertidumbre, el equipo de software se embarca en un plan que se haya establecido como consecuencia de dichas tareas. Por tanto, el plan debe adaptarse y actualizarse conforme avanza el proyecto.

Ámbito del software:

El ámbito del software describe las funciones y características que se entregan a los usuarios finales; los datos que son entrada y salida; el contenido que se presenta a los usuarios como consecuencia de usar el software y el desempeño, las restricciones, las interfaces y la confiabilidad que se ligan al sistema.