Unidad II: Calidad del Software

La calidad del software es un concepto complejo que no es directamente comparable con la calidad de la manufactura de productos. En la manufacturación, la noción de calidad viene dada por la similitud entre el producto desarrollado y su especificación. En un mundo ideal, esta definición debería aplicarse a todos los productos, pero, para sistemas de software, existen estos problemas:

1. La especificación se orienta hacia las características del producto que el consumidor quiere. Sin embargo, la organización desarrolladora también tiene requerimientos (como los de mantenimiento) que no se incluyen en la especificación.
2. No se sabe cómo especificar ciertas características de calidad (por ejemplo, mantenimiento) de una forma no ambigua.
3. Es muy difícil redactar especificaciones concretas de software. Por lo tanto, aunque un producto se ajuste a su especificación, los usuarios no lo consideran un producto de alta calidad debido a que no responde a sus expectativas.

Se deben reconocer estos problemas con la especificación del software y se tienen que diseñar procedimientos de calidad que no se basen en una especificación perfecta. En concreto, atributos del software como mantenibilidad, seguridad o eficiencia no pueden ser especificados explícitamente. Sin embargo, tienen un efecto importante en cómo es percibida la calidad del sistema.

Algunas personas piensan que la calidad puede lograrse definiendo estándares y procedimientos organizacionales de calidad que comprueban si estos estándares son seguidos por el equipo de desarrollo. Su argumento es que los estándares deben encapsular las buenas prácticas, las cuales nos llevan inevitablemente a productos de alta calidad. En la práctica, sin embargo, es más importante la

gestión de la calidad que los estándares y la burocracia asociada para asegurar el seguimiento de estos estándares.

Los buenos gestores aspiran a desarrollar una «cultura de la calidad» donde todos seamos responsables de que el desarrollo del producto sea llevado a cabo obteniendo un alto nivel de calidad en éste. Mientras estándares y procedimientos son las bases de la gestión de la calidad, los gestores de calidad experimentados reconocen que hay aspectos intangibles en la calidad del software (elegancia, legibilidad, etc.) que no puede ser incorporada en los estándares. Ellos ayudan a la gente interesada en estos aspectos intangibles de calidad y fomentan comportamientos profesionales en todos los miembros del equipo.

La gestión formal de la calidad es particularmente importante para equipos que desarrollan sistemas grandes y complejos. La documentación de la calidad es un registro de que es hecho por cada subgrupo en el proyecto.

Esto ayuda a la gente a ver qué tareas importantes no deben ser olvidadas o que una parte del equipo no haga suposiciones incorrectas acerca de lo que otros miembros han hecho. La documentación de calidad es también un medio de comunicación sobre el ciclo de vida de un sistema. Ésta permite al grupo responsabilizarse de la evolución del sistema para saber qué ha hecho el equipo de desarrollo.

Para sistemas pequeños, la gestión de calidad es importante todavía, pero se debe adoptar una aproximación más informal. No son tan necesarios los documentos porque el grupo puede comunicarse informalmente. La clave de la calidad en el desarrollo de sistemas pequeños es el establecimiento de cultura de calidad y asegurarse de que todos los miembros del equipo hacen una aproximación positiva a la calidad del software.

La gestión de calidad del software se estructura en tres actividades principales:

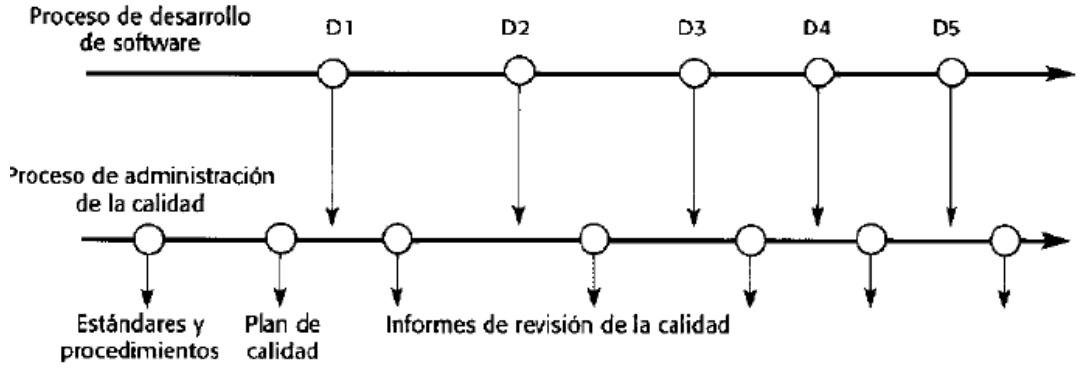
1. *Garantía de la calidad.* El establecimiento de un marco de trabajo de procedimientos y estándares organizacionales que conduce a software de alta calidad.

2. *Planificación de la calidad.* La selección de procedimientos y estándares adecuados a partir de este marco de trabajo y la adaptación de éstos para un proyecto software específico.

3. *Control de la calidad.* La definición y fomento de los procesos que garanticen que los procedimientos y estándares para la calidad del proyecto son seguidos por el equipo de desarrollo de software.

2.1 La gestión de proyectos usando un marco de calidad

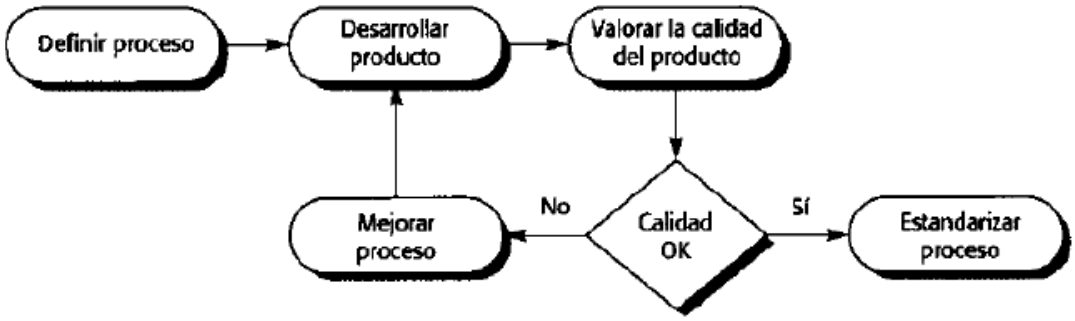
La gestión de la calidad provee una comprobación independiente de los procesos de desarrollo software. Los procesos de gestión de la calidad comprueban las entregas del proyecto para asegurarse que concuerdan con los estándares y metas organizacionales. El equipo de garantía de calidad debe ser independiente del equipo de desarrollo para que puedan tener una visión objetiva del software. Ellos transmitirán los problemas y las dificultades al gestor principal de la organización.



La gestión de la calidad y el desarrollo del software

Un equipo independiente de calidad garantiza que los objetivos organizacionales y la calidad no sean comprometidos por consideraciones de presupuesto o agenda.

Una suposición subyacente de la gestión de calidad es que la calidad del proceso de desarrollo afecta directamente a la calidad de los productos derivados.
La siguiente figura muestra una aproximación basada en proceso para conseguir la calidad del producto.



Calidad basada en procesos

Hay un vínculo claro entre la calidad del proceso y del producto en producción debido a que el proceso es relativamente fácil de estandarizar y monitorizar.
El software no se manufactura, sino que se diseña. El desarrollo de software es un proceso más creativo que mecánico. La calidad del producto, también se ve afectada por factores externos, como la novedad de una aplicación o la presión comercial para sacar un producto rápidamente.

En el desarrollo software, por lo tanto, la relación entre la calidad del proceso y la calidad del producto es muy compleja. Es difícil de medir los atributos de la calidad del software, en consecuencia, es difícil explicar cómo influyen las características del proceso en estos atributos. Además debido al papel del diseño y la creatividad en el proceso software, no podremos predecir la influencia de los cambios en el proceso en la calidad del producto.

La calidad del proceso tiene una influencia significativa en la calidad del software. La gestión y mejora de la calidad del proceso debe minimizar los defectos en el software entregado.

La gestión de la calidad del proceso implica:

1. Definir estándares de proceso.
2. Supervisar el proceso de desarrollo para asegurar que se sigan los estándares.
3. Hacer informes del proceso para el gestor del proyecto y para el comprador del software.

2.2 Estándares y Métricas de calidad en la ingeniería de SW

Un problema de la garantía de la calidad basada en el proceso es que el equipo de garantía de la calidad (QA) insista en unos estándares de proceso independientemente del tipo de software a desarrollar. El gestor principal debe intervenir para asegurar que el proceso de calidad ayude al desarrollo del producto en lugar de impedirlo.

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

1. Estándares de producto. Se aplican sobre el producto software que se comienza a desarrollar. Incluyen estándares de documentación, como cabecera de comentarios estándar para definición de clases, y estándares de codificación.
2. Estándares de proceso. Definen los procesos que deben seguirse durante el desarrollo del software. Pueden incluir definiciones de procesos de especificación, diseño y validación, así como una descripción de los documentos que deben escribirse en el curso de estos procesos.

Existe una relación muy cercana entre los estándares de producto y los estándares de proceso. Los estándares de producto se aplican a las salidas del proceso software y. en muchos casos, los estándares de proceso incluyen actividades de proceso específicas que garantizan que se sigan los estándares de producto.

Los estándares de software son importantes por varias razones:

1. Están basadas en el conocimiento de la mejor o más apropiada práctica de la empresa, evita la repetición de errores anteriores.
2. Proveen un marco de trabajo alrededor del cual se implementa el proceso de garantía de la calidad. El control de la calidad sencillamente asegura que los estándares se siguen adecuadamente.
3. Ayudan a la continuidad cuando una persona continúa el trabajo que llevaba a cabo otra. Se reduce el esfuerzo de aprendizaje cuando se comienza un nuevo trabajo.

Utilizando estándares como punto de partida, el equipo de garantía de la calidad debe crear un «manual» de estándares. Éste define los estándares que son apropiados para la organización.

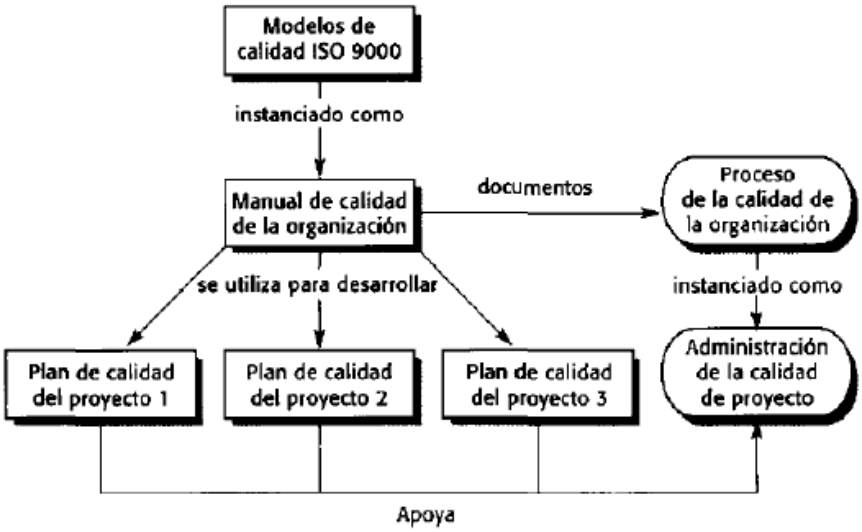
Algunas veces, los ingenieros de software consideran a los estándares como burocráticos e irrelevantes para las actividades técnicas de desarrollo de software. Para evitar estos problemas, los gestores de la calidad que fijan los estándares necesitan estar informados y tomar en consideración los siguientes pasos:

1. Involucrar a los ingenieros de software en el desarrollo de los estándares del proyecto.
2. Revisar y modificar los estándares de forma regular con el fin de reflejar los cambios en la tecnología.

3. Proveer herramientas de software para apoyar los estándares donde sea necesario.

El gestor del proyecto y el gestor de calidad pueden evitarse los problemas de estándares inapropiados si planean cuidadosamente la calidad. Deben decidir cuáles son los estándares del manual que utilizarán sin cambio alguno, cuáles se modificarán y cuáles se dejarán de lado.

Un conjunto de estándares internacionales que se puede utilizar en el desarrollo de un sistema de gestión de calidad en todas las industrias es ISO 9000. Los estándares ISO 9000 pueden aplicarse a un amplio abanico de organizaciones desde las de manufactura hasta las de servicios. ISO 9001 es el más general de estos estándares y se aplica en organizaciones interesadas en el proceso de calidad de diseño, desarrollo y mantenimiento de productos. ISO 9001 no es un estándar específico para desarrollo de software, pero define principios generales que pueden aplicarse al software. Aquí podemos observar las relaciones entre ISO 9000, el manual de calidad y los planes de calidad de proyectos particulares.



ISO 9000 y la gestión de la calidad

Los procesos de garantía de calidad en una organización se documentan en un manual de calidad que define el proceso de calidad.

El estándar ISO 9000 se refiere simplemente a la definición de los procedimientos que son utilizados en la compañía y la documentación asociada que muestre que los procesos han sido seguidos. Éste no se ocupa de asegurar que estos procesos sean la mejor práctica, ni asegura la calidad del producto.

Los estándares de documentación en un proyecto de software son documentos muy importantes ya que son la única forma tangible de representar al software y su proceso. Los documentos estandarizados tienen una apariencia, estructura y calidad consistentes y. por lo tanto, son más fáciles de leer y de comprender.

Existen tres tipos de estándares de documentación:

1. Estándares del proceso de documentación. Definen el proceso a seguir para la producción del documento, esto implica definir los procedimientos involucrados en el desarrollo del documento y las herramientas de software utilizadas. También definen procedimientos de comprobación y refinamiento que aseguren que se produzcan documentos de alta calidad.

2. Estándares del documento. Gobiernan la estructura y presentación de los documentos.

3. Estándares para el intercambio de documentos. Aseguran que todas las copias electrónicas de los documentos sean compatibles.

Los estándares de calidad del proceso de documentos deben ser flexibles y les debe ser posible ajustarse a todos los tipos de documentos. Uno de los modelos posibles de proceso de documentación incluye: Crear un borrador, comprobarlo, revisarlo y rehacerlo es un proceso iterativo. Éste continúa hasta que se produce

un documento de calidad aceptable. El nivel de calidad aceptable depende del tipo de documento y de los lectores potenciales de éste.

Aunque los estándares del documento se adapten a las necesidades de un proyecto específico, una buena práctica es que se utilice el mismo estilo en todos documentos producidos por una organización.

Algunos ejemplos de estándares de documentos a desarrollar son:

1. Estándares de identificación de documentos. Puesto que los proyectos de sistemas grandes producen cientos de documentos, cada documento debe identificarse de forma única. Para los documentos formales, este identificador es el identificador formal definido por el gestor de configuraciones. Para documentos informales, el estilo del identificador del documento es definido por el gestor del proyecto.
2. Estándares de la estructura del documento. Cada clase de documentos producida durante un proyecto de software debe seguir alguna estructura estándar.
3. Estándares de presentación de documentos. Estos estándares definen un «estilo propio » para los documentos y contribuyen notablemente a la consistencia de éstos.
4. Estándares para actualizar los documentos. Conforme el documento evoluciona y refleja los cambios en el sistema, se debe utilizar una forma consistente para indicar los cambios en el documento.

Los estándares de intercambio de documentos son importantes debido a que se pueden intercambiar copias electrónicas de los documentos. La utilización de estándares de intercambio permite que los documentos se transfieran electrónicamente y puedan reconstruirse en su forma original.

La planificación de la calidad es el proceso en el cual se desarrolla un plan de calidad para un proyecto. El plan de calidad define la calidad del software deseado y describe cómo valorar ésta. Por lo tanto, define lo que es software de «alta calidad». Sin esta definición, los diferentes ingenieros pueden trabajar en direcciones opuestas para optimizar los atributos de proyecto.

El plan de calidad selecciona los estándares organizacionales apropiados para un producto y un proceso de desarrollo particulares. Esta estructura comprende:

1. Introducción del producto. Descripción del producto, el mercado al que se dirige y las expectativas de calidad.
2. Planes de producto. Contiene las fechas de terminación del producto y las responsabilidades importantes junto con los planes para la distribución y el servicio.
3. Descripciones del proceso. Contiene los procesos de desarrollo y de servicio a utilizar para el desarrollo y administración del producto.
4. Metas de calidad. Contiene las metas y planes de calidad para el producto. Incluyendo la identificación y justificación de los atributos de calidad importantes del producto.
5. Riesgos y gestión de riesgos. Contiene los riesgos clave que podrían afectar a la calidad del producto y las acciones para abordar estos riesgos.

Los planes de calidad obviamente difieren dependiendo del tamaño y del tipo de sistema que se desarrolle. Existe una amplia variedad de atributos de calidad del software potenciales a considerar en el proceso de planificación de la calidad. Ver atributos de calidad en la figura siguiente.

Seguridad	Comprensión	Portabilidad
Protección	Experimentación	Usabilidad
Fiabilidad	Adaptabilidad	Reutilización
Flexibilidad	Modularidad	Eficacia
Robustez	Complejidad	Aprendizaje

Atributos de la calidad del software

Puede ser que la eficacia sea primordial, por lo que será necesario sacrificar otros factores para alcanzarla. Estoce establece en el plan, y los ingenieros que trabajan en el desarrollo deben cooperar para lograrlo. El plan también define el proceso de evaluación de la calidad.

El control de la calidad implica vigilar el proceso de desarrollo de software para asegurar que se siguen los procedimientos y los estándares de garantía de calidad. En el proceso de control de calidad del proyecto se comprueba que las entregas cumplan los estándares definidos.

Existen dos enfoques complementarios que se utilizan para comprobar la calidad de las entregas de un proyecto:

1. Revisiones de la calidad donde el software, su documentación y los procesos utilizados en su desarrollo son revisados por un grupo de personas. Se encargan de comprobar que se han seguido los estándares del proyecto y el software y los documentos concuerdan con estos estándares. Se toma nota de las desviaciones de los estándares y se comunican al gestor del proyecto.
2. Valoración automática del software en la que el software y los documentos producidos se procesan por algún programa y se comparan con los estándares que se aplican a ese proyecto de desarrollo en particular.

Esta valoración automática comprende una medida cuantitativa de algunos atributos del software.

2.2.1 PSP y TSP

Personal Software Process (PSP)

El personal Software Process, conocido por sus siglas como PSP, es una metodología de reciente creación, proveniente del Instituto de Ingeniería del Software. PSP es una alternativa dirigida a los ingenieros de sistemas, que les permite mejorar la forma en la que construyen software. Considerando aspectos como la planeación, calidad, estimación de costos y productividad. PSP es una metodología que vale la pena revisar cuando el ingeniero de software está interesado en aumentar la calidad de los productos de software que desarrolla dentro de un contexto de trabajo individual.

Características

En PSP todas las tareas y actividades que el ingeniero de software debe realizar durante el proceso de desarrollo de un producto de software, están puntualmente definidas en un conjunto de documentos conocidos como scripts. Los scripts son el punto medular en PSP, por lo que hace mucho énfasis en que deban ser guidados en forma disciplinada, ya que de ello dependerá el éxito de la mejora que se busca. Gran parte de las tareas y actividades definidas en los scripts generará en su realización un conjunto de datos, fundamentalmente de carácter estadístico. La aplicación de PSP en varios procesos de desarrollo, y el análisis de la información estadística generada e cada uno de éstos, permitirán al ingeniero de software identificar, tanto sus fortalezas como sus debilidades, y crecer a través de un proceso de autoaprendizaje y auto mejora.

La calidad de PSP, es u aspecto fuertemente relacionado con la cantidad de defectos que el producto de software contiene.

En este nivel se introducen algunos métodos aplicables al proceso de desarrollo de software, dentro de un enfoque de proyectos a gran escala pero sin lidiar con problemas de comunicación y coordinación de los equipos de trabajo.

Pasos a seguir

Los scripts se organizan en cuatro niveles, identificados del 0 al 3, atendiéndose en cada nivel un conjunto de aspectos a mejorar del proceso de desarrollo de software. Al primer nivel se le conoce como 0 o medición personal, al segundo como nivel1 o de planeación personal, al tercero, como nivel2 o de calidad personal y al cuarto como nivel3 o cíclico personal.

Cada uno de estos niveles, con excepción del 3, tiene una versión que los extiende, introduciendo tareas y actividades para un mejor manejo de los aspectos de interés en nivel, o bien para incluir nuevos aspectos.

Cada uno de los niveles extiende los aspectos considerados en el nivel inmediato anterior. Una de las razones de esta clasificación puede ser que el PSP es una metodología de mejora basada en datos estadísticos, los cuales deben ser cuidadosamente recabados por el ingeniero de software; el aumento gradual de la cantidad de datos que debe recolectar el ingeniero introduce, por consiguiente, el cambio en su manera de trabajo de una manera paulatina. Se recomienda un uso incremental de PSP, iniciando con el nivel más bajo durante un primer proyecto de desarrollo y, en proyectos siguientes, ascendiendo a niveles superiores. Los scripts no pueden utilizarse en forma separada o desordenada.

Ventajas y desventajas

PSP es una alternativa, de las muchas que han surgido recientemente, para mejorar el proceso de desarrollo de software. Más que clasificar un conjunto de sentencias como ventajas y desventajas, a continuación se citan algunas:

PSP es una metodología basada en estimación. La estimación permite saber cuándo y cómo se desarrollan las tareas de un proceso, por lo que podría citarse como un aspecto importante de esta metodología el estar basada en métricas y estimaciones.
La información de las métricas y estimaciones se utiliza para evaluar y mejorar procesos frutos. PSP parte de la premisa que, si el ingeniero de software conoce sus fortalezas y debilidades, puede establecer las acciones necesarias para erradicar o explotar los aspectos identificados en la forma en que desarrolla software.

Aunque lo mencionado en el punto anterior podría sonar bastante atractivo, la forma de llegar a ese auto conocimiento puede resultar tediosa, y en el peor de los casos, una pesadilla para el desarrollador. Salvo muy pocas excepciones, los ingenieros de software nunca realizan procedimientos formales.

La importancia del entrenamiento y el marco de trabajo personal del PSP es que provee a los ingenieros un proceso disciplinado, métricas de desempeño, habilidades de planeación y estimación y habilidades de administración de la calidad.

El PSP se desarrolló considerando los siguientes principios de calidad y planeación:

- Cada ingeniero debe planear su trabajo con base en sus datos históricos personales.
- Los ingenieros deben medir su trabajo y analizar los resultados para mejorar su desempeño.
- Los ingenieros deben sentirse personalmente responsables de la calidad de sus productos buscando decididamente hacer trabajo de calidad.
- Es más eficiente prevenir los defectos que encontrarlos y corregirlos.

- La manera correcta es siempre la manera más rápida y económica de hacer el trabajo.

Team Software Process (TSP)

En combinación con el Personal Software Process (PSP), el llamado Team Software Process (TSP) proporciona un marco de trabajo de procesos definidos que está diseñado para ayudarle a equipos de gerentes e ingenieros a organizar y producir proyectos de software de gran escala, que tengan tamaños mayores a varios miles de líneas de código. El objetivo del TSP es mejorar los niveles de calidad y productividad de un proyecto de desarrollo de software de un equipo, con el fin de ayudarlos a alcanzar los acuerdos de costos y tiempos en dicho desarrollo.

La versión inicial del TSP fue desarrollada por Watts Humphrey en 1996, y el primer Reporte Técnico para TSP fue publicado en el año 2000, patrocinado por el Departamento de Defensa de los Estados Unidos. El libro de Watts Humphrey llamado "Introduction to the Team Software Process" (Addison Wesley Professional, Massachusetts, 1999), presenta el TSP en detalle y se enfoca en el proceso de la construcción de un equipo productor de software, estableciendo objetivos del equipo, distribuyendo los roles, y otras actividades de trabajo en equipo.

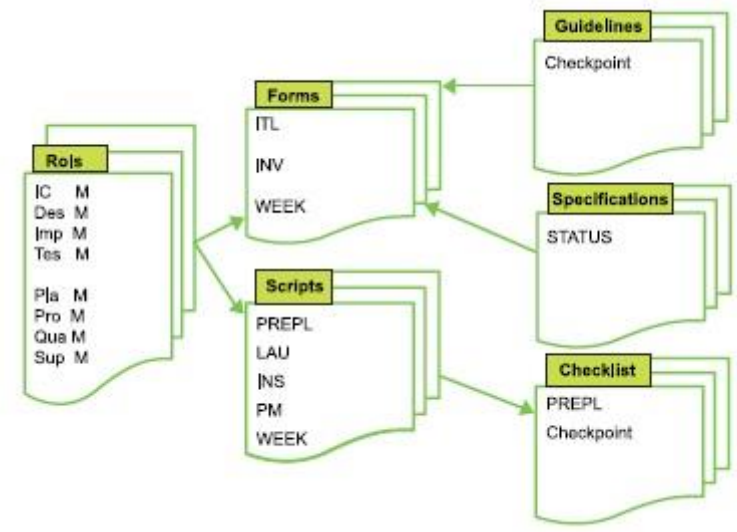


Figura 1. Elementos de proceso de TSP.

Funcionamiento

Antes que los ingenieros de software puedan participar en el TSP, se requiere que ya hayan aprendido sobre el Personal Software Process (Personal Software Process), de manera tal que el TSP pueda funcionar de manera adecuada. El TSP comienza con un proceso de cuatro días llamado *despegue*. El despegue está diseñado para comenzar el proceso de construcción de los equipos y durante éste tiempo, los equipos y sus administradores establecen metas, definen roles, evalúan riesgos y producen un plan de equipo. El despegue generalmente se hace con un coach específicamente entrenado, o con un líder que ya ha gerenciado varios proyectos que han usado TSP para su desarrollo.

El TSP es un proceso diseñado para equipos de software auto-dirigidos y de alto desempeño, ayudándolos a planear su trabajo, negociar compromisos con la gerencia, dar seguimiento cabal a sus compromisos y producir productos de calidad mientras mejoran su rendimiento. El marco de trabajo de TSP incluye roles, plantillas, procesos, guías, especificaciones y listas de chequeo. La Figura 1 muestra el marco de trabajo con algunos ejemplos de los elementos de proceso.

Características TSP

- Usa equipos auto-dirigidos con base en el estilo de administración de Peter Drucker (administración del conocimiento) junto con un coach que ayuda a desarrollar las habilidades de trabajo en equipo en los individuos.
- Tiene procesos operacionales flexibles que permiten a los equipos adaptar los procesos, contando además con un marco de trabajo de métricas que soporta a su proceso e incluye técnicas para la administración de la calidad usando revisiones personales, inspecciones e índices de desempeño de la calidad.
- Usa planes detallados con actividades no mayores a 10 hrs en periodos de 3-6 meses y establece juntas de cierre (postmortems) para finales de ciclo o de proyecto.
- Utiliza lanzamientos de proyectos de 3.5 días para planear las actividades y para integrar a los miembros del equipo.
- Cada miembro tienen asignado roles, metas y riesgos del proyecto.
- Los calendarios del equipo son desglosados en calendarios personales que son ajustados con base en datos personales.

Figura 2. Características de PSP y TSP.

2.2.2 CMM

CMM es una aplicación del sentido común para el gerenciamiento de procesos y conceptos de mejora de la calidad del desarrollo y mantenimiento del software, es además una guía desarrollada por y para la comunidad profesional del software.

- Un modelo para la mejora organizacional.
- Una estructura confiable y consistente para evaluar y mejorar las capacidades de una organización.

¿Porque surge CMM?

Crisis del software de principios de los 80, debido a una falta de eficiencia en los procesos de desarrollo de programas.

¿Qué puedo hacer con CMM?

- Ayudar a la comunicación, al establecer un lenguaje común en el ámbito organizacional
- Facilitar poner el foco de atención en cuestiones críticas
- Proveer recomendaciones generales
- Ayudar a priorizar acciones de mejora

El CMM ha demostrado su impacto en la calidad del software producido bajo sus directrices. Las limitaciones más determinantes en CMM es que sólo es un modelo de referencia, es decir, sólo describe los procesos clave que una organización debe tener, y no explica cómo se deben implementar estos (Montes, 2001).

Otro aspecto limitante en CMM es el tiempo para escalar de un nivel a otro: de 23 a 30 meses lo cual genera una problemática en obtención de resultados inmediatos (SEI, 2000).

Los resultados observados demuestran que los proyectos generados bajo alguna de las técnicas: TSP y PSP permite una disminución en tiempo y un control en el proceso de desarrollo (McAndrews, 2000).

2.2.3 MOPROSOFT

Modelo de Procesos para la Industria del Software

Modelo para la mejora y evaluación de los procesos de desarrollo y mantenimiento de sistemas y productos de software. Desarrollado por la Asociación Mexicana para la Calidad en Ingeniería de Software a través de la Facultad de Ciencias de la Universidad Nacional Autónoma de México (UNAM) y a solicitud de la Secretaría de Economía para obtener una norma mexicana que resulte apropiada a las características de tamaño de la gran mayoría de empresas mexicanas de desarrollo y mantenimiento de software. Moprosoft es el nombre del modelo en la comunidad universitaria y profesional, y la norma técnica a la que da contenido es la NMX-059/01-NYCE-2005 que fue declarada Norma Mexicana el 15 de agosto de 2005 con la publicación de su declaratoria en el Diario oficial de la Federación.

Moprosoft considera que los modelos de evaluación y mejora CMMI eI SO/IEC 15504 no resultan apropiados para empresas pequeñas y medianas de desarrollo y mantenimiento de software. Sobre las áreas de procesos de los niveles 2 y 3 del modelo SW-CMM e inspirándose en el marco de ISO/IE 15504 se ha desarrollado este modelo.

2.3. Impacto de la calidad en tiempo, costo y alcance del proyecto

El alcance de un proyecto —llamado también alcance del trabajo— es el trabajo que debe hacerse para que el cliente se convenza de que las entregas (las cosas por hacer), es decir el producto u objetos tangibles que han de suministrarse) *cumplan con los requisitos o criterios de aceptación acordados al comenzar el*

proyecto. Por ejemplo, el alcance podría ser el trabajo de limpiar el suelo, de construir una casa y de poner la jardinería ornamental según las especificaciones hechas por el cliente y aceptadas por el contratista.

Por estructuración se entiende la facilidad con que las funciones pueden ser compartimentalizadas y la naturaleza jerárquica de la información a tratar. A medida que el grado de estructuración aumenta, la posibilidad de estimar con precisión mejora y, por consiguiente, el riesgo disminuye.
Bajo el concepto de la administración de proyectos, se asignan representantes de cada uno de los departamentos funcionales de las divisiones al equipo asignado al proyecto. Cada miembro del equipo deriva una guía funcional experta y control administrativo del gerente de departamento. El equipo incluye al siguiente personal clave:

- Gerente de Proyectos
- Ingeniero de Proyectos
- Gerente de Construcción del proyecto
- Coordinador de construcción del proyecto
- Ingeniero de puesta en marcha del proyecto
- Ingeniero de aseguramiento de la calidad del proyecto
- Supervisor de costo y programas del proyecto
- Administrador del proyecto
- Gerente de aprovisionamiento del proyecto
- Asistente del controlador del proyecto

La estimación de costos de una actividad es una evaluación cuantitativa de los costes probables de los recursos necesarios para completar las actividades del cronograma del proyecto. Este tipo de estimación puede presentarse en forma de resumen o en detalle.
Los costos se estiman para todos los recursos que se aplican a la estimación de costos de la actividad. Esto incluye, entre otros, la mano de obra, los materiales,

los equipos, los servicios, las instalaciones, la tecnología de la información, y categorías especiales como una asignación por inflación o una reserva para contingencias de costo.

- Estimación por analogía
- Determinación de Tarifas de Costes de Recursos
- Estimación Ascendente
- Estimación Paramétrica
- Software de Gestión de Proyectos
- Análisis de Propuestas para Licitaciones

La estimación de recursos y costes es una actividad importante que debe llevarse a cabo con el mayor detalle posible, porque permite al comprador establecer una aproximación al coste total y plazos del desarrollo del sistema.
Para ello se requiere experiencia, acceso a una buena información histórica y determinación para confiar en medidas cuantitativas cuando todo lo que existe son datos cualitativos.

Factores que afectan a esta estimación:

- La complejidad del proyecto, cuantificando la misma en función de:
- Número de módulos y nivel de interrelación entre los mismos.
- Número y tipo de las interfaces externas con otros sistemas, programas o datos.
- Grado de distribución y heterogeneidad del entorno de implantación.
- Grado de sofisticación de las herramientas de desarrollo.
- Naturaleza de los algoritmos que se deben diseñar y programar.