Administración de Proyectos de Software

1.Introducción y conceptos básicos

Conceptos básicos: proyecto y su administración

Algunas preguntas guía

- ¿Qué es un proyecto?
- ¿Es diferente un proyecto de software?
- ¿Necesita administrarse un proyecto?
- ¿Qué se hace para administrar un proyecto?

Proyectos de software

I. ¿Qué se entiende por proyecto de software?

- Proyecto
  - Planta y disposición que se forma para un tratado o para la ejecución de una cosa de importancia
  - Designio de ejecutar algo
  - Conjunto de escritos, dibujos y cálculos hechos para dar idea de lo que ha de ser y costar una obra de ingeniería o arquitectura

Diccionario Léxico Hispano, W.M. Jackson, Inc. Editores

Características clave de un proyecto (1/2)

1. Involucra tareas no rutinarias
2. Requieren planeación
3. Se deben lograr objetivos o crear productos específicos
4. Tiene un lapso de tiempo específico predefinido

Hughes y Cotterell, Software Project Management, McGraw-Hill, Secciones 1.2

Características clave de un proyecto (2/2)

5. El trabajo se lleva a cabo por otras *personas*
6. Involucra varias *especializaciones*
7. Los trabajos se llevan a cabo en varias *fases*
8. Los *recursos* son limitados
9. Los proyectos son largos o *complejos*

Hughes, B. y Cotterell, M., Software Project Management, McGraw-Hill, Secciones 1.2 y 1.3

Ejercicio 1

- ¿Cuál de los siguientes puede considerarse proyecto?
  - Producir la edición de un periódico
  - Hacer un túnel bajo el mar
  - Realizar un matrimonio

Hughes, B. y Cotterell, M., Software Project Management, McGraw-Hill

Características de los proyectos de software

- Invisibilidad.
  - El avance dentro de un proyecto de construcción de una carretera o puente puede ser visto, mientras que en el software no es inmediatamente visible.
- Complejidad
  - Los proyectos de software contienen mayor complejidad que otros tipos de proyectos respecto al dinero gastado
- Flexibilidad
  - El software puede cambiarse más fácilmente que otros productos

Hughes, B. y Cotterell, M., Software Project Management, McGraw-Hill, Secciones 1.3

Proyectos de software versus otros tipos de proyectos

- Muchas técnicas aplicables a la administración de proyectos son aplicables a la administración de proyectos de software.
- Sin embargo:
  - los productos de proyectos de software tienen ciertas características que los hacen diferentes.
  - Una forma de percibir la administración de proyectos de software es haciendo visible lo que es invisible

Hughes, B. y Cotterell, M., Software Project Management, McGraw-Hill, Secciones 1.3

Proyecto

- No todos los proyectos son iguales
- "Un proyecto es como viajar en carretera. Algunos proyectos son simples y rutinarios, como conducir hacia la tienda a plena luz del día. Pero la mayoría de los proyectos que valen la pena realizar son más parecidos a conducir un camión, en la montaña, de noche"… *en Nepal*

  (Cem Kamer, James Bach y Bret Pettichord)

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill, Sección 21.1

Administración de proyectos

Administración

1. Planear: decidir qué se va a hacer
2. Organizar: hacer preparativos
3. Asignar personal: elegir personas adecuadas
4. Dirigir: dar órdenes
5. Monitorear: observar el progreso
6. Controlar: emprender acciones para corregir problemas de funcionamiento
7. Innovar: proponer soluciones novedosas
8. Representar: conectar con clientes y usuarios

Hughes, B. y Cotterell, M., Software Project Management, McGraw-Hill

Administración vista de administradores

- Enfrentar fechas límite
- Enfrentar limitaciones en recursos
- Comunicar efectivamente a los diversos grupos
- Conseguir que todos se comprometan
- Establecer hitos medibles
- Enfrentar cambios
- Lograr plan de acuerdo con desarrolladores
- Ganar compromiso de gerencia
- Enfrentar conflictos
- Negociar con vendedores y subcontratistas

Problemas comunes en proyectos según administradores

- Estimaciones y planes deficientes
- Falta de estándares y medidas de calidad
- Falta de guía sobre toma de decisiones en organización
- Falta de técnicas para hacer visible el progreso
- Papeles y responsabilidades mal definidos

Problemas comunes en proyectos según personal

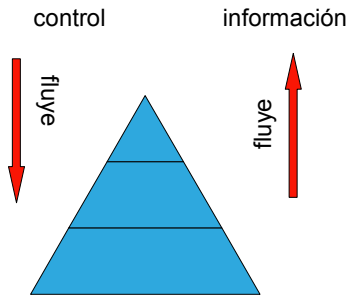
- Trabajo definido de modo inadecuado
- Administradores ignorantes de tecnología informática
- Falta de conocimiento del área de aplicación
- Falta de estándares
- Documentación inadecuada
- Retraso de actividades precedentes
- Falta de comunicación con usuarios
- Trabajo duplicado por mala comunicación

Problemas comunes en proyectos según personal

- Falta de compromiso
  - Hay un solo interesado y se va
- Conocimiento muy especializado
- Cambio de ambiente de software
- Cambio de requerimientos
- Presión de fecha límite
- Falta de control de calidad
- Falta de entrenamiento
- Administración lejana

Información y control

- Decisiones:
  - estratégicas (objetivos),
  - tácticas (alcanzar metas)
  - operativas (día con día)
- Información:
  - medidas de actuación
  - medidas predictivas



Administración de Proyectos (1/2)

- Necesaria si se quiere obtener un producto en tiempo y forma
  - *Es el arte de dirigir y coordinar los recursos humanos y materiales a lo largo de la vida de un proyecto por medio de técnicas modernas de administración, para lograr los objetivos en: alcance, costo, tiempo, calidad y satisfacción.*

Administración de Proyectos (2/2)

- Aunque el administrador de software realiza lo mismo que cualquier administrador, pero resulta más difícil debido a:
  - El producto es intangible
  - No existen procesos de software estándar
  - A menudo los proyectos grandes de software son "únicos"

Ejercicio 2

- Explique por qué la intangibilidad de los sistemas de software plantea problemas para la administración de proyectos de software.

Actividades de la Administración de Proyectos de Software

- Redacción de la propuesta
- Planeación y calendarización del proyecto
- Costeo del proyecto
- Supervisión y revisión del proyecto
- Selección y evaluación del personal
- Redacción y presentación de información

1. Redacción de la propuesta

- La redacción de la propuesta deberá incluir:
  - Objetivos del proyecto
  - Cómo se llevará a cabo
  - Costos y calendarización
  - Justificación del por qué se entregará a un equipo u organización
- Resulta crítica pues la habilidad de realizar la redacción sólo se adquiere con el tiempo.

Sommerville, I., "Ingeniería de Software", 6ª ed., Addison Wesley, Sección 4.1

2. Planeación y calendarización del proyecto

- Se refiere a la identificación de:
  - Actividades
  - Hitos
  - Entregas producidas por un proyecto
- Por lo que se debe bosquejar un plan de desarrollo hacia las metas del proyecto

Sommerville, I., "Ingeniería de Software", 6ª ed., Addison Wesley, Sección 4.1

3. Costeo del proyecto

- Actividad que se refiere al estimado de recursos que se requieren para llevar a cabo el proyecto.

Sommerville, I., “Ingeniería de Software”, 6ª ed., Addison Wesley, Sección 4.1

4. Supervisión y revisión del proyecto

- Actividad continua
- Se lleva un control de los avances de los costos y progresos reales versus los planeados
- Pueden apoyarse en mecanismos formales
- El administrador debe detectar la realidad después de entrevistarse con el personal del proyecto

Sommerville, I., “Ingeniería de Software”, 6ª ed., Addison Wesley, Sección 4.1

5. Selección y evaluación del personal

- El administrador debe emplear a la gente ideal para el proyecto.
- Sin embargo en ocasiones se presentan problemas como:
  - Bajo presupuesto para contratar personal calificado
  - El personal adecuado no está disponible
  - La empresa desea que su personal novato obtenga experiencia

6. Redacción y presentación de información

- El administrador debe dar informes al cliente y a su organización
- Los reportes de avance deben ser concisos y coherentes.
- Debe saber comunicarse con el cliente

"Un administrador de proyectos de software debe tener la habilidad de comunicarse efectivamente de forma oral y escrita"

Sommerville, I., "Ingeniería de Software", 6ª ed., Addison Wesley, Sección 4.1

Las cuatro "P"

Elementos que intervienen en la Administración de Proyectos

- La efectividad de la administración de proyectos se enfoca en las cuatro "P"
  - Personal: ¿Quiénes?
  - Producto: ¿Qué?
  - Proceso: ¿Cómo?
  - Proyecto: ¿Para qué?

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill, Sección 21.1

Personal -1-

- El software lo hacen seres humanos
- Para seres humanos
- Y a veces afecta a otros seres humanos

- Así lo más importante en el software son los seres humanos (personas)

Producto
-2-

- Lo que desea obtenerse al final del proyecto
- A veces se detalla en entregables (partes del producto a lograr)

Proceso
-3-

- Un proceso de software proporciona un marco de trabajo para establecer el plan detallado para el desarrollo del software
  - Puede ser simplemente un método o una metodología
- Una forma de hacer las cosas

Pressman, R., “Ingeniería de Software”, 6ª ed., McGraw-Hill, Sección 21.1

Proyecto
-4-

- Lo que se desea emprender, para ciertas personas, empleando un equipo de personas específico, para obtener un producto deseado y usando un proceso adecuado.

Personal
-1-

Personal

- La componente más importante para el éxito de un proyecto es el factor humano
  - Estudio realizado por IEEE en 1988
- A pesar de la importancia del ser humano, los gerentes ignoran el desarrollo de su personal.
- Aspectos a considerar referentes al factor humano
  - Participantes
  - Líderes de equipo
  - Estructura organizacional del equipo de software
  - Conflictos de coordinación y comunicación

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill, Sección 21.1

Participantes (1/2)

- Gestores ejecutivos
  - Definen aspectos del negocio que influenciará al proyecto
- Gestores técnicos
  - Planifican, motivan, organizan y controlan a los profesionales

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill, Sección 21.1

Participantes (2/2)

- Profesionales
  - Proporcionan las habilidades técnicas para la ingeniería de un producto
- Clientes
  - Especifican los requerimientos para la IS
- Usuarios finales
  - Interactúan con el software una vez que se libera

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill, Sección 21.1

Líderes de equipo

- Los profesionales de computación, generalmente, no tiene las competencias para ser líderes, las adquieren por necesidad.

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill, Sección 21.1

Líderes de equipo

- Jerry Weinberg, 1986:
  - Sugiere que los líderes de proyecto exitosos aplican un estilo de gestión de resolución de problemas.
    - Entender el problema, gestionar el flujo de ideas y, al mismo tiempo, hacer que el equipo se comprometa con la calidad.
  - Propone que para lograr un buen liderazgo se use el modelo MOI

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill, Sección 21.1

Modelo MOI para el liderazgo

- Motivación.
  - Habilidad para alentar al personal a producir según su mejor capacidad.
- Organización.
  - Habilidad para adecuar procesos existentes o inventar nuevos.
- Ideas e innovación.
  - Habilidad para alentar a la gente a ser creativo y trabajar dentro de los límites establecidos para la aplicación.

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill, Sección 21.1

Estructura organizacional del equipo de software

- En una organización existen prácticas y políticas que no son responsabilidad del administrador de proyectos de software,
  - pero sí es su responsabilidad la organización de los equipos desarrolladores de software.
- La forma de organización de los equipos depende de varios factores.

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill, Sección 21.1

Factores relevantes en la elección de organización de un equipo de software

Factores de Mantei

1. Dificultad del problema que se resolverá
2. Tamaño del problema resultante (en líneas de código o puntos de función).
3. Tiempo en que el equipo estará junto (vida del equipo)
4. Grado de modularización
5. Calidad y confiabilidad requeridos para el sistema
6. Rigidez de la fecha de entrega
7. Grado de sociabilidad (comunicación) que requiere el sistema

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill, Sección 21.1

Paradigmas organizacionales para los equipos de Ingeniería de Software (1/2)

- Constantine sugiere cuatro, en 1993:

1. Paradigma cerrado. Tipo jerárquico, funcionan bien en proyectos similares a los que ya se han hecho antes.
2. Paradigma aleatorio. Se organizan como ellos se acomoden y funcionan bien en proyectos innovadores o de adelantos tecnológicos.

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill, Sección 21.1

Paradigmas organizacionales para los equipos de Ingeniería de Software (2/2)

1. Paradigma abierto. Combinación de los anteriores, funcionan bien para problemas complejos, pero no eficientemente.
2. Paradigma sincrónico. Se apoya en la compartimentalización natural de un problema y organiza a los miembros del equipo para trabajar en partes del problema con poca comunicación activa entre ellos.

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill, Sección 21.1

Equipos Ágiles

- Los métodos ágiles subrayan la competencia individual en conjunción con la colaboración del grupo
- Para aprovechar la competencia de cada miembro del equipo y fomentar la colaboración eficaz a lo largo del proyecto, los equipos ágiles son auto organizados
- La planificación se mantiene al mínimo
- Se realizan reuniones periódicas de equipos para coordinar el trabajo que se debe hacer

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill, Sección 21.1

Toxicidad de equipo

- No todos los equipos funcionan bien (no cuajan)
- Jackman en 1998, propone cinco factores que fomentan un ambiente de equipo tóxico:
  - Atmósfera de trabajo frenética.
  - Alta frustración que provoca fricciones
  - Proceso de software "fragmentado o pobremente coordinado"
  - Poca definición de los papeles dentro del equipo
  - Continuas y repetidas exposiciones al fracaso

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill, Sección 21.1

Cómo evitar la Toxicidad de equipo

Toxina	Prevención
1. Atmósfera de trabajo frenética.	1. Toda la información del proyecto debe estar a la mano y no debe cambiarse.
2. Alta frustración que provoca fricciones	2. Dar a los miembros del equipo la misma responsabilidad.
3. Proceso de software "fragmentado o pobremente coordinado"	3. Comprender el problema y que el equipo elija el proceso.
4. Poca definición de los papeles dentro del equipo	4. Establecer la forma de revisar los avances y su calidad (RTF)
5. Continuas y repetidas exposiciones al fracaso	5. Establecer técnicas de equipo para realimentación y resolución de problemas.

Conflictos de coordinación y comunicación

- La **escala** de muchos esfuerzos de desarrollo es muy grande
- La **incertidumbre** es común
- La **interoperabilidad** con otros sistemas
- Para minimizar los conflictos
  - Se deben establecer mecanismos para la comunicación formal e informal entre los miembros del equipo y entre múltiples equipos.

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill, Sección 21.1

Ejercicio 3

- Proponga 10 lineamientos para que el ingeniero de Software ejerza su potencial completo en su trabajo.

"Los 10 mandamientos del Ingeniero de Software"

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill

Manejo de personal (I. Sommerville)

Notas por Juan Manuel Fernández Peña

2011

Factores críticos

- Objetividad
  - Trato equitativo, transparente
- Respeto
  - Aceptar habilidades diferentes, mientras haya aporte
- Incorporación
  - Escuchar y tomar en cuenta propuestas
- Honestidad
  - Sobre lo que va bien y lo que va mal; sus conocimientos

Selección de personal

- Fuentes de información
  - Los interesados
  - Entrevistas: útiles en aspectos de comunicación y habilidades sociales; fallan en aspectos técnicos
  - Recomendaciones de personas conocidas que han trabajado con los interesados

Selección de personal

- Algunos factores que influyen:
  - Experiencia en dominio de aplicación (algunos)
  - Experiencia en plataforma (si hay programación de bajo nivel)
  - Experiencia en lenguaje (en proyectos cortos)
  - Habilidad para resolver problemas (difícil; por trabajos)
  - Soporte educativo (poco relevante)
  - Habilidad de comunicación (oral, escrita, idiomas)
  - Adaptabilidad (según trabajos realizados)
  - Actitud (positiva respecto al trabajo; deseo de aprender)
  - Personalidad (compatibilidad)

Selección de personal

- Pasos para seleccionar
  - Crear especificación del trabajo
  - Crear perfil del empleado
  - Conseguir solicitantes
  - Examinar currícula
  - Entrevistas
    - Aptitudes
    - Personalidad
    - Muestras de trabajo
    - Entrevista personal

Motivación

- Las personas se motivan por satisfacción de necesidades, en orden creciente (según Maslov):
  - Fisiológicas (comer, dormir)
  - Seguridad (entorno protector)
  - Sociales (parte de grupo)
  - Estima (sentirse respetado)
  - Autorrealización (desarrollo personal)

Enfocarse en éstas

Motivación

- Sociales:
  - Dar tiempo a conocerse, lugar de intercambio, mecanismos informales
- Estima:
  - Mostrar que tienen valor, reconocer logros
- Autorrealización:
  - Asignar tareas demandantes, pero no imposibles, programas de capacitación

Motivación

- Orientación de personas:
  - A tareas: se motivan por su trabajo, reto intelectual, técnico (el software como reto)
  - A sí mismos: su éxito y reconocimiento (software como medio)
  - A interacción: presencia y acciones de compañeros; importantes en enfoque orientado a usuarios
- Puede ser variable, pero una domina

Gestión de grupos

- Factores para equipos
  - Composición: balance de habilidades, experiencia y personalidades
  - Cohesión: verdadero equipo, no colección de individuos
  - Comunicación: que exista y sea efectiva
  - Organización: todos se sienten valorados y satisfechos de su papel

Gestión de equipos

- En cuanto a balance de personalidades:
  - Los orientados a tareas son buenos para los aspectos técnicos; deben comprender conjunto, no aislarse. Unos ayudan a comunicar y los orientados a sí mismos son persistentes, seguirán hasta el final

Gestión de equipos

- Sobre cohesión
  - Grupo más importante que individuos, metas de grupo, lealtad, protección
  - Se forman estándares de calidad del grupo
  - Depende de la cultura organizacional
  - Problemas: resistencia irracional a cambio de liderazgo; pensamiento de grupo (no se piensa críticamente)

Gestión de equipos

- Comunicación:
  - Influye tamaño de grupo; al crecer crecen demasiado los canales de comunicación, distracción
  - La estructura afecta: los grupos menos formales se comunican mejor
  - La composición: personas parecidas chocan; balance de sexos
  - El entorno afecta (área de trabajo, cierta privacidad)

Personal

- El "factor humano" es tan importante que el SEI desarrollo el *Modelo de Madurez de Gestión Capacidad Personal* (MMCGP).
- Existen varias áreas clave prácticas para el personal de software:
  - Reclutamiento
  - Selección
  - Gestión del desempeño
  - Entrenamiento
  - Retribución
  - Desarrollo de carrera
  - Diseño de organización y trabajo
  - Desarrollo de cultura de equipo

Modelo de madurez de capacidades del personal PCMM

- A partir del modelo CMM se han desarrollado modelos específicos.
- Para el personal, siendo muy importante, se creó el PCMM
- Ahora se ha integrado en CMMI

People Capacity Maturity Model

Modelo de madurez de capacidad del personal

Antecedentes

- Las compañías globales y de alta tecnología requieren
  - Buenos productos y servicios
  - Desarrollar y retener empleados con talento y habilidades
- Se busca que no sólo sigan órdenes sino que sean centros inteligentes de acción hacia fines comunes

Antecedentes

- Principios para atraer, desarrollar y retener personal valioso (J. Pfeffer, Stanford):
  - Seguridad en el empleo
  - Contratación selectiva
  - Equipos autoadministrados y descentralización de toma de decisiones
  - Compensación alta según rendimiento de la organización
  - Entrenamiento
  - Reducir barreras y distinciones de estatus
  - Compartir información financiera y de rendimiento

Niveles del modelo

Nivel 1 Inicial

- Se hace ad-hoc, se reinventa cada vez
- No hay manera confiable de estimar esfuerzo
- El resultado depende del personal
- Se dice que el personal es valioso
- Pero no se hace nada por mejorar su valor

Nivel 2 Gestionado

- La base para mejorar, que sea repetible
- Establecer procesos bien definidos
- Control de compromisos y líneas base
- No estar presionando y correteando al personal innecesariamente

Nivel 3 Definido

- Se identifican las mejores prácticas
- Se documentan e integran a procesos
- Se definen métricas
- Se estandariza para toda la organización
- Surge cultura común

Nivel 4 Predecible

- Se administra a partir de datos que describen el rendimiento de la empresa; no solo hitos
- Se caracterizan estadísticamente los procesos críticos
- Al volverse predecibles y cuantitativos, se adquiere conocimiento que permite mejorar las prácticas

Nivel 5 Optimizado

- Del conocimiento que se tiene se van proponiendo acciones de mejoramiento:
- Ajustar procesos
- Usar nuevas tecnologías
- Se administra el cambio
- Se identifican defectos persistentes

Resumen

- Se crea un ambiente donde
  - Las prácticas son repetibles
  - Las mejores prácticas se transfieren entre grupos
  - Se reducen las variaciones en rendimiento
  - Las prácticas se mejoran de continuo

Problemas en organizaciones inmaduras

- Sobrecarga de trabajo
- Ambiente de distracción
- Metas de rendimiento y retroalimentación poco claras
- Carencia de conocimiento relevante y habilidad
- Mala comunicación
- Baja moral

Áreas de proceso

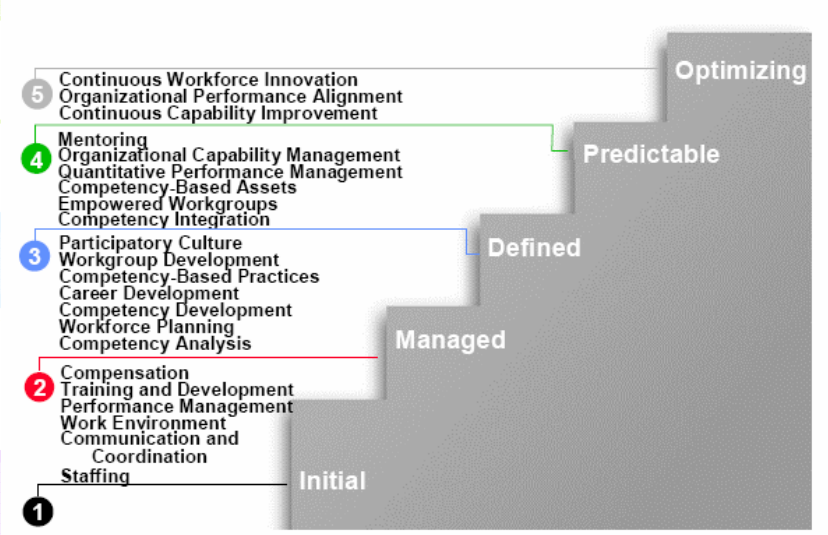


Figure 3.1 — Process areas of the People CMM

Arquitectura

Figure 4.1 — Structure of the People CMM

PROCESO
-2-

Proceso

- Un proceso dice quién está haciendo qué y cuándo y cómo lograr la meta

Proceso

- Un proceso de software proporciona el marco de trabajo para establecer el plan detallado para el desarrollo del software
- Dos tipos de actividades son generales en cualquier proceso:
  - Establecer: tareas, hitos, productos de trabajo y puntos de control de calidad
  - Controlar la calidad, gestionar la configuración de software y medir resultados.

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill, Sección 21.1

Actividades

- Un proceso tiene actividades genéricas para cualquier proyecto
  - Se detallan según necesidades de cada proyecto
- También tiene actividades sombrilla o de protección

Actividades genéricas

- Comunicación (requerimientos)
- Planeación
- Modelado (análisis, diseño)
- Construcción (codificación y prueba)
- Despliegue

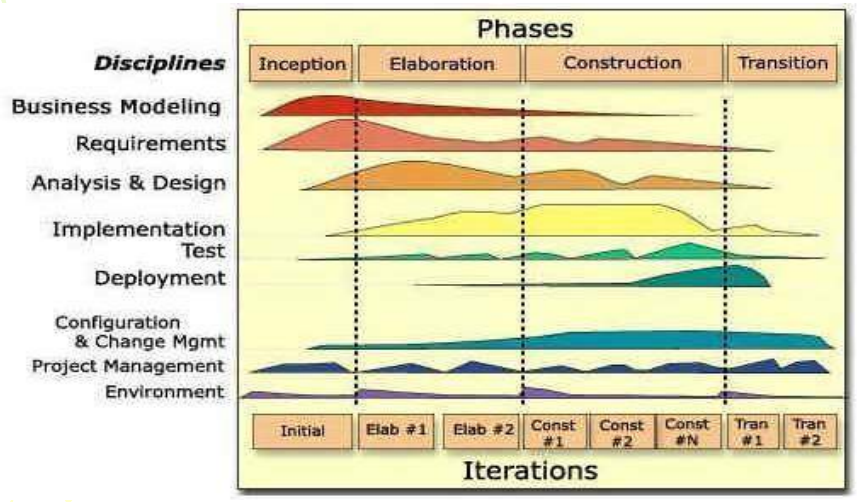
Actividades sombrilla

- Gestión del riesgo
- Seguimiento y control
- Revisiones
- Medición
- Gestión de la configuración
- Gestión de reutilización
- Preparación y producción de productos del trabajo

Procesos

- Personal Software Process
  - proyectos pequeños de una persona
- Team Software Process
  - proyectos para equipos pequeños
- Modelos reconocidos generales:
  - cascada, espiral, iterativos
- Modelos con nombre propio:
  - Proceso Unificado, Ciclo de vida estructurado de Yourdon
- Métodos Ágiles
  - Scrum, Crystal Clear, XP

Proyectos según Proceso Unificado



Selección de proceso

- En cada proyecto se selecciona un proceso, entre los anteriores u otros o se forma específico para un proyecto

Descomposición del proceso

- Una vez escogido el proceso: Ejemplos:
  - Lineal secuencial, para sistemas muy pequeños y muy bien definidos
  - DRA, para restricciones de tiempo ceñido y se puede modularizar
  - Incremental, si no puede entregarse el producto completo por restricciones de tiempo o falta de claridad en los requerimientos

Descomposición del proceso (1/2)

- Para proyectos pequeños en menos de 48 horas se debe:
  - Desarrollar una lista de conflictos que deben clarificarse.
  - Reunirse con los clientes para abordar los conflictos que deben clarificarse.
  - Desarrollar en conjunto un enunciado del ámbito.
  - Revisar el enunciado del ámbito con todos los implicados.
  - Modificar el enunciado del ámbito según se requiera.

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill

Descomposición del proceso (2/2)

Para proyecto más complejo, *actividad de comunicación;* las actividades serían:

1. Revisar la petición del cliente.
2. Planificar y programar una reunión formal con el cliente.
3. Llevar a cabo investigaciones para especificar la solución propuesta y los enfoques existentes.
4. Preparar un "documento de trabajo" y una agenda para la reunión formal.
5. Celebrar la reunión.
6. Desarrollar en conjunto miniprospectos que reflejen los datos, función y características de comportamiento del software.
7. Revisar cada miniprospecto o para valorar su corrección, consistencia y eliminar la ambigüedad.
8. Ensamblar los miniprospectos en un documento más amplio.
9. Revisar el documento más amplio o colección de casos de uso con todos los implicados.
10. Modificar el documento más amplio según se requiera.

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill7

Producto
-3-

Producto

- Antes de crear un proyecto se deben establecer:
  - Objetivos y ámbito del producto
  - Soluciones alternativas
  - Identificación de restricciones técnicas y de gestión
  - El desarrollador de software y cliente deben reunirse para definir objetivos y ámbito del producto
  - En ese momento empieza la Ingeniería de procesos del negocio y continua con Ingeniería de Requerimientos.

Pressman, R., “Ingeniería de Software”, 6ª ed., McGraw-Hill, Sección 21.1

Producto

- Para empezar un proyecto se necesitan estimaciones cuantitativas y un plan organizado.
- Para obtener lo anterior se necesita información sólida del producto a construir.
- Un análisis de requerimientos sería lo deseable, pero:
  - Éste se lleva semanas y hasta meses
  - Aunque se agilice, los requerimientos siempre cambian durante el proyecto.
- Dada la problemática se opta por establecer y acotar el ámbito del producto.

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill

Ámbito de software (1/3)

- El ámbito se define al responder las siguientes preguntas:
  - Contexto:
    - ¿Cómo encaja el software que se desarrollará en un sistema más grande, producto o contexto de negocio?
    - ¿Qué restricciones se imponen como resultado del contexto?

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill

Ámbito de software (2/3)

- Objetos de información:
  - ¿Qué objetos de datos visibles al usuario se producen como resultado del software?
  - ¿Qué objetos de datos se requieren de entrada?
- Función y desempeño
  - ¿Qué funciones realizará el software para transformar los datos de entrada?
  - ¿Habrá características de desempeño especial a considerar?

Ámbito de software (3/3)

- Características del ámbito del software
  - No debe ser ambiguo ni incompresible a niveles de gestión y técnico
  - Los enunciados deben acotarse, es decir, debe llevar datos cuantitativos. Ejemplos:
    - Número de usuarios simultáneos, tamaño de la lista de correos, tiempo de respuesta máximo permitido.
  - Contener las restricciones o limitaciones. Ejemplos:
    - El costo del producto restringe el tamaño de la memoria
  - Se describen los factores para reducir riesgos. Ejemplo:
    - Los algoritmos deseados se comprenden bien y están disponibles en C++

Descomposición del problema

- Se aplica en dos áreas:
  - La funcionalidad a entregar
  - El proceso a aplicar para entregar
- Para entender el problema se aplica la descomposición del mismo.
- Antes de comenzar la estimación se evalúa y refina la redacción del ámbito
- Generalmente basados en la funcionalidad

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill

Ejemplo de refinamiento del ámbito (1/2)

- Construcción de un nuevo procesador de texto con lo que habitualmente se tiene y varias funciones novedosas:
  - entrada continua mediante voz, edición automática de copia, capacidad de diseño de página, índice y contenido automáticos.
- El administrador del proyecto incluirá la lista de funciones y luego agregará preguntas respecto de a las funcionalidades novedosas.

Ejemplo de refinamiento del ámbito (2/2)

- Preguntas de refinamiento realizadas por el administrador del proyecto
  - ¿la entrada continua de voz requiere que el usuario del producto lo "entrene"?
  - ¿qué capacidades proporcionará la característica de edición de copia?
  - ¿cuán sofisticada será la capacidad de diseño de página?
- Conforme surjan las preguntas irán apareciendo las particiones …

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill

Selección de Proceso

- Características importantes que hay que considerar para escoger el proceso
  - Los clientes solicitaron el producto y el personal
  - Las características del producto
  - El ambiente de trabajo en que trabaja el equipo de software
- Una vez seleccionado el proceso se define el plan preliminar del proyecto con base en las actividades del marco de trabajo

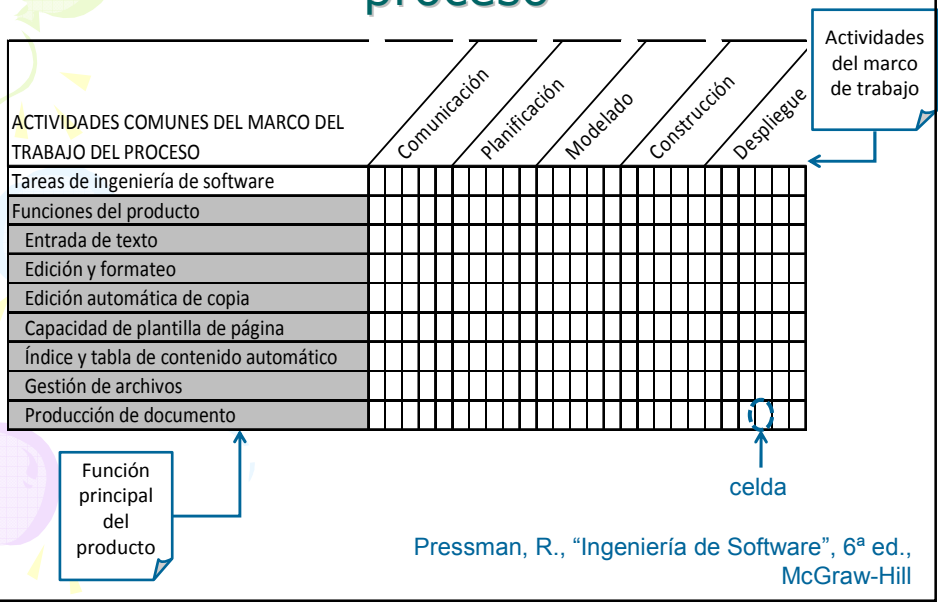
Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill

Combinación de producto y proceso

- Una vez definido el marco de trabajo se deberá aplicar a cada una de las funciones establecidas para el producto (figura).
  - Además deben incluirse las actividades de ingeniería para cada actividad del marco de trabajo
- El trabajo del AP consiste en estimar:
  - Requisitos de recursos para cada celda de la matriz
  - Fecha de inicio y final
  - Productos de trabajo de cada tarea

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill

Combinación de producto y proceso



Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill

Proyecto
-4-

Elementos importantes de un proyecto

- Hay tres procesos sucesivos que dan inicio a un proyecto:

1. Estudio de Factibilidad.
   - Investigación que se realiza para decidir las perspectivas de un proyecto y si conviene empezarlo.
2. Planeación.
   - Debe escribirse un plan que abarque todos las etapas en general y detalle la primera etapa, dejando las últimas etapas pendientes para cuando se tenga más información
3. Ejecución del proyecto.
   - Donde se realizan las diversas etapas del ciclo de vida de un proyecto de software.

Hughes, B. y Cotterell, M., Software Project Management, McGraw-Hill, Secciones 1.4

Proyecto como sistema

- A un proyecto le interesa crear un sistema o transformar un sistema viejo y es en sí un sistema
- Sistemas, subsistemas y medio ambiente
  - Sistema. Conjunto de partes interrelacionadas, forma parte de otros sistemas y está compuesto por subsistemas
  - Medio ambiente. Los sistemas existen dentro de un medio ambiente que puede afectarlos y el sistema no puede controlarlo

Hughes, B. y Cotterell, M., Software Project Management, McGraw-Hill, Secciones 1.6

Gestión de Proyectos (1/7)

- La gestión de un proyecto de software exitoso requiere entender qué puede salir mal. John Reel en 1999 define 10 señales que indican que un proyecto de está en peligro:

1. El personal de software no entiende las necesidades de sus clientes.
2. El ámbito del producto está mal definido.
3. Los cambios se gestionan mal.
4. La tecnología elegida cambia.
5. Las necesidades comerciales cambian.
6. Los plazos de entrega no son realistas.
7. Los usuarios se resisten.
8. Se pierde el patrocinio
9. El equipo de proyecto carece de personal con las habilidades apropiadas.
10. Los gestores evitan las mejores prácticas y las lecciones aprendidas.

Pressman, R., “Ingeniería de Software”, 6ª ed., McGraw-Hill

Gestión de Proyectos (2/7)

- Debido a las causas anteriores, algunos profesionales muy experimentados con frecuencia se refieren a la regla 90-90 cuando estudian proyectos de software .
  - El primer 90 por ciento de un sistema absorbe el 90 por ciento del esfuerzo y el tiempo asignados. El último 10 por ciento toma el otro 90 por ciento del esfuerzo y el tiempo asignados [ZAH94].

Pressman, R., "Ingeniería de Software", 6ª ed., McGraw-Hill

Gestión de Proyectos (3/7)

- John Reel en 1999 sugiere un enfoque de cinco partes para proyectos de software:
  1. Comience con el pie derecho. Esto se logra trabajando duro para entender el problema que será resuelto y entonces establecer objetivos y expectativas realistas para todos los que estarán involucrados en el proyecto.

Gestión de Proyectos (4/7)

2. Mantenga el Ímpetu. Muchos proyectos tienen un buen comienzo y luego lentamente se desintegran:
   - el gestor del proyecto debe proporcionar incentivos para conservar los reveses del personal en un mínimo absoluto;
   - el equipo debe resaltar la calidad en cada tarea que realiza y
   - los gestores ejecutivos debe hacer todo lo posible por mantenerse fuera del camino del equipo.

Gestión de Proyectos (5/7)

3. Rastree el progreso. En un proyecto de software el progreso:
   - Se rastrea conforme se elaboran los productos de trabajo y se aprueban como parte de una actividad de aseguramiento de la calidad.
   - Se pueden recopilar y aplicar procesos del software y medidas del proyecto para valorar el progreso contra los promedios establecidos por la organización que desarrolla software.

Gestión de Proyectos (6/7)

4. Tome decisiones inteligentes. El gestor del proyecto y del equipo de software deben encaminarse a:
   - emplear software comercial o componentes de software existentes,
   - evitar interfaces personalizadas cuando estén disponibles enfoques estándar,
   - identificar y evitar riesgos obvios, y
   - asignar más tiempo que el que considere necesario a las tareas complejas o riesgosas

Gestión de Proyectos (7/7)

5. Realice un análisis de resultados:
   - establezca un mecanismo consistente para extraer lecciones aprendidas por cada proyecto,
   - evalúe la planificación real y la prevista,
   - recolecte y analice métricas de proyecto de software,
   - obtenga realimentación de los miembros del equipo y de los clientes, y
   - registre los hallazgos en forma escrita.

El principio W5HH (1/4)

- Barry Boehm en 1996, establece:
  - "Usted necesita un principio organizador que escale hacia abajo para proporcionar planes simples para proyectos simples".
  - sugiere un enfoque que aborde los objetivos del proyecto, los hitos y planificación, responsabilidades, gestión, enfoques técnicos y recursos requeridos.
- En el principio W5HH, se realizan una serie de preguntas que conducen a una definición de las características claves del proyecto y al plan de proyecto resultante

El principio W5HH (2/4)

- ***(Why)****¿Por qué se desarrolla el sistema?*
  - La respuesta a esta pregunta permite a todas las partes evaluar la validez de las razones del negocio para el trabajo de software. ¿el propósito del negocio justifica el gasto de personal, tiempo y dinero?
- ***(What)****¿Qué se hará?*
  - La respuesta a esta pregunta establece el conjunto de tareas que se requerirá para el proyecto.
- ***(When)****¿Cuándo se hará?*
  - La respuesta a esta pregunta ayuda al equipo a establecer una planificación del proyecto al identificar cuándo se realizarán las tareas del proyecto y cuándo se alcanzarán los objetivos.

El principio W^{5}HH (3/4)

- ***(Who)*** *¿Quién es el responsable de una función?*
  - Establecer la responsabilidad de cada miembro del equipo. La respuesta a esta pregunta ayuda a lograrlo.
- ***(Where)****¿Dónde están ubicados en la organización?*
  - No todos los papeles y responsabilidades residen en el equipo de software. El cliente, los usuarios y otros participantes también tienen responsabilidades.

El principio W^{5}HH (4/4)

- ***(How)*** *¿Cómo se hará el trabajo desde los puntos de vista técnico y de gestión?*
  - Una vez establecido el ámbito del producto se debe definir una estrategia de gestión y técnica para el proyecto.
- ***(How much)*** *¿Cuánto de cada recurso se necesita?*
  - La respuesta a esta pregunta se deriva al desarrollar estimaciones con base en las respuestas a las preguntas anteriores.

Prácticas Críticas

- El Airlie Council en 1999 ha elaborado una lista de "prácticas críticas de software para la gestión basada en el desempeño".
  - "empleadas consistentemente por, y consideradas críticas por, proyectos de software muy exitosos y por organizaciones cuya 'línea base' de desempeño es mucho mejor que los promedios de la industria".
- Las prácticas críticas incluyen para la integridad del proyecto:
  - gestión de proyecto basado en métricas,
  - costo empírico y estimación de la planificación,
  - seguimiento del valor ganado,
  - gestión del riesgo formal,
  - seguimiento de defectos frente a objetivos de calidad y
  - gestión al tanto del personal.

Ejercicio 4

- *Usted ha sido nombrado administrador de proyecto de software para una compañía que atiende el mundo de la ingeniería genética. Su labor es gestionar el desarrollo de un nuevo producto de software que acelera el ritmo de la clasificación de genes. El trabajo está orientado a Investigación y Desarrollo, pero la meta es elaborar un producto dentro del siguiente año.*
  - ¿qué estructura de equipo elegiría y por qué?
  - ¿qué modelos de procesos de software elegiría y por qué?

Glosario

- Hitos: puntos finales de una actividad del proceso de software. Son puntos finales de una etapa lógica del proceso de desarrollo.
- Producto a entregar (entregable). Resultado del proyecto que se entrega al cliente, generalmente al finalizar el proyecto.
  - Los productos a entregar son hitos, pero éstos no necesariamente son entregables.