Tema 1

Conceptos generales

Proyecto: Actividad planificada. Conjunto de actividades relacionadas y coordinadas para alcanzar unos objetivos dentro de unos límites de presupuesto, calidad y tiempo.

PMBOK (Project Management Body Of Knowledge): Proyecto es esfuerzo temporal que se lleva a cabo para crear un producto, servicio o resultado único con una elaboración gradual.

Características de los proyectos

- Tareas no rutinarias.
- Se precisa una planificación.
- Requiere objetivos específicos o un producto especificado a crear.
- Tiene una duración predeterminada.
- El trabajo se realiza por varias personas que pueden tener distintas especializaciones.
- Trabajo dividido en fases.
- Con restricciones sobre los recursos a usar.
- En general, un proyecto es largo o complejo.

Etapas del desarrollo de un proyecto

1. Estudio de viabilidad: Cuánto supone hacerlo.
2. Planificación: Cómo se va a hacer.
3. Ejecución del proyecto: Llevarlo a cabo.

Etapas del ciclo de vida de un proyecto

1. Fase de iniciación: Idea.
2. Fase de definición: ¿Qué?
3. Fase de diseño: ¿Cómo?
4. Fase de desarrollo: ¿Cómo se implementará?
5. Fase de implementación.
6. Fase de seguimiento: Mantenimiento.

Contenidos de un estudio de factibilidad

- Introducción: Qué es el documento.

- Descripción de la situación actual.
- Descripción del problema.
- Desarrollo propuesto:
  - Aspectos de negocios y financieros.
  - Aspectos técnicos.
  - Aspectos organizativos.
- Costes estimados:
  - Costes de desarrollos.
  - Costes operativos (de explotación).
- Beneficios previstos.
- Recomendación

Características de los proyectos de software

La mayor parte de las técnicas utilizadas en gestión de proyectos son aplicables a proyectos de software, pero tienen algunas características específicas:

- Invisibilidad: En otras ingenierías, el progreso es visible. En informática es más complicado.
- Complejidad: Suele ser más complicado por unidad de coste que otros proyectos.
- Conformidad: Otros proyectos se asientan sobre leyes físicas y constantes. Los proyectos de software deben satisfacer los requisitos de los clientes, que no son constantes.
- Flexibilidad: La facilidad de cambio del software es una de sus fortalezas, pero en general se considera que el software debe adaptarse a los componentes de los que forma parte y no a la inversa.

Qué es gestión de proyectos

Aplicación de conocimiento, habilidades, herramientas y técnicas a las actividades de los proyectos con el fin de conseguir o superar las necesidades y expectativas de un proyecto. Implica equilibrio entre alcance, tiempo, coste y calidad; usuarios con necesidades y expectativas diferentes; y requisitos identificados y no identificados.

Objetivos de la gestión

- Calidad
- Productividad

- Reducción del riesgo

Las funciones del gestor de un proyecto son planificar, organizar, gestionar personal, dirigir, controlar... Debe tratar con ideas, cosas materiales y personas.

La figura del gestor de proyectos

Es la persona responsable de alcanzar los objetivos del proyecto. La gestión de un proyecto incluye:

- Identificar los requisitos.
- Establecer objetivos claros y posibles.
- Equilibrar las demandas concurrentes de calidad, alcance, tiempo y costes.
- Adaptar las especificaciones, los planes y el enfoque a las diversas inquietudes y expectativas de los diferentes interesados.

A menudo se habla de una "triple restricción", equilibrio entre alcance, tiempo y costes del proyecto, que afecta a la calidad.

Un proyecto grande se puede descomponer en subproyectos con sus respectivos gestores.

Las funciones de un gestor de proyectos son:

- Planificar: decidir qué se va a hacer.
- Organizar: realizar los preparativos adecuados.
- Gestionar el personal: seleccionar la gente adecuada para cada trabajo.
- Dirigir: Dar las instrucciones adecuadas.
- Monitorizar: Comprobar el progreso.
- Controlar: Tomar las acciones para corregir los atrasos.
- Innovar: Abordando nuevas soluciones.
- Representar: Tratar con usuarios, etc.

Modelos de organización:

- Organización funcional: Un ejecutivo jefe del que dependen administradores funcionales que coordinan grupos de staff que realizan una función.
- Organización por proyectos: los administradores de proyectos que dependen del ejecutivo jefe coordinan a grupos de staff divididos por proyectos. Favorece la fidelidad de los miembros al proyecto y la autoridad del gestor de proyecto. Por contra, los miembros del proyecto dejan de serlo cuando éste termina, y se pueden producir ineficiencias en el uso de los recursos.

- Organización matricial fuerte: Del jefe ejecutivo dependen managers de funcionalidad y un manager de project managers. El proyecto se compone de un project manager y los staff necesarios de cada función.

En grandes organizaciones puede existir una unidad organizativa centralizada que supervisa la gestión de todos los proyectos de la organización. Esta unidad aporta la visión y objetivos de la organización a cada proyecto, y optimiza los recursos compartidos.

Los problemas más comunes en planificación y gestión de proyectos son:

- Estimaciones y planes no adecuados
- Carencia de estándares y medidas de calidad
- Carencia de orientación sobre la toma de decisiones organizativas
- Definición de roles
- Criterios de éxito incorrectos
- Ignorancia de la dirección en IT
- Falta de conocimiento en el área de aplicación
- Falta de documentación actualizada
- Tareas anteriores al proyecto con retraso (p. ej. entrega del equipamiento)
- Falta de comunicación y coordinación
- Falta de compromiso
- Demasiados cambios en requisitos
- Entorno de software cambiante
- Presión por los plazos de entrega
- Gestión remota
- Falta de entrenamiento

Un proceso define quién hace qué, cuando y cómo para alcanzar cierto objetivo.

Las áreas y procesos implicados en la gestión de proyectos de acuerdo con el PMBOK:

- Gestión de integración
- Gestión del alcance
- Gestión del tiempo
- Gestión del coste
- Gestión de la calidad
- Gestión de recursos humanos
- Gestión de comunicaciones
- Gestión de riesgos
- Gestión de aprovisionamiento

El objetivo de la gestión de proyectos es *finalizar un proyecto* en el tiempo estipulado, con el presupuesto definido, con la funcionalidad requerida, para satisfacción del cliente y sin agotar a los miembros del equipo. Además hay que proporcionar visibilidad sobre el progreso del proyecto.

Procesos

Los proyectos están compuestos de procesos. Un proceso es un conjunto de acciones que se realizan para obtener un resultado.

- **Procesos relacionados con la gestión del proyecto**: relacionados con la descripción y la organización del trabajo del proyecto.
- **Procesos orientados al producto**: relacionados con la especificación y creación del producto resultante del proyecto.

Los procesos de ambos tipos se solapan e interaccionan a lo largo del desarrollo del proyecto.

Los cinco grupos de procesos son:

- **Grupo de procesos de iniciación**: Define y autoriza el proyecto o una fase del mismo.
- **Grupo de procesos de planificación**: Define y refina los objetivos, y planifica el curso de acción requerido para lograr los objetivos y el alcance del proyecto.
- **Grupo de procesos de ejecución**: Integra a personas y otros recursos para llevar a cabo el plan de gestión para el proyecto.
- **Grupo de procesos de seguimiento y control**: Mide y supervisa regularmente el avance, a fin de identificar las variaciones respecto del plan de gestión del proyecto, de tal forma que se tomen medidas correctivas cuando sea necedsrio para cumplir con los objetivos del proyecto.
- **Grupo de procesos de cierre**: Formaliza la aceptación del producto, servicio o resultado, y termina ordenadamente el proyecto o una fase del mismo.

Tema 2

Planificación (PDF)

Gestión del tiempo de proyecto (Planificación): Incluye todos los procesos necesarios para que se pueda garantizar la finalización del proyecto en el tiempo estipulado:

- Definición de actividades: Identificar las actividades que se deben realizar para obtener los productos resultantes del proyecto.
- Secuenciación de actividades: Identificar y documentar las dependencias de interacción entre actividades.
- Estimación de recursos de las actividades: Estimar tipo y cantidades de recursos necesarios para cada actividad.
- Estimación de duración de actividades: Identificar el número de periodos de trabajo que se precisarán para finalizar cada actividad.
- Desarrollo de la planificación: Analizar las secuencias de actividades, duración de las mismas y requisitos de recursos para llevarlo a cabo.
- Control de la planificación: Controlar los cambios que se produzcan respecto a la planificación.

Definición de actividades

Implica la identificación y documentación de las actividades específicas que se deben realizar para producir todo tipo de productos identificados en la estructura de descomposición del trabajo. Las actividades deben estar de acuerdo con los objetivos que se pretenden conseguir y, por lo tanto, con el alcance del proyecto.

Entradas:

- Factores ambientales de la empresa
- Activos de los procesos de la organización: Existencia de pautas formales o informales en la empresa, así como información histórica. Hay que considerar que actividades se han precisado en proyectos anteriores y similares.
- Definición del alcance del proyecto: Los productos entregables del proyecto, las restricciones y las suposiciones documentadas en el enunciado del alcance del proyecto.
- Restricciones: Aquellos factores que limitarán las opciones del equipo de gestión del proyecto.
- Suposiciones: Factores que, a efectos de planificación, se consideran verdaderos. Suelen implicar un riesgo y suelen ser una salida de la identificación de los riesgos.
- Estructura de descomposición del trabajo
- Diccionario de la EDT (estructura de descomposición del trabajo)
- Plan de gestión del proyecto.

EDT, Estructura de Descomposición del Trabajo (WBS, Work Breakup Structure)

Se usa para crear la lista de trabajos (tareas o actividades) a realizar. Se realiza una descomposición en árbol de los trabajos del proyecto. EDT identifica los "elementos terminales". Es el punto de partida fundamental para la planificación de un proyecto. Puede estar orientada a la actividad o al entregable. Se recomienda utilizar el método de notas adhesivas desde el principio.

Recomendaciones para EDT (WBS):

- No más de 100-200 elementos terminales. Si hubiera más sería recomendable usar subproyectos.
- No más de 3-4 niveles de profundidad.
- No más de 5-9 trabajos por nivel.
- El conocimiento humano solo "trabaja" con 3 bits.
- La memoria a corto plazo de la mayoría es capaz de manejar 5-9 items.
- Si la descomposición es de grano muy fino, la planificación será más complicada pues se reduce la atención.
- Cuantas más tareas haya, más complicadas serán las dependencias entre ellas que hay que controlar.
- Los trabajos deben ser de un tamaño y complejidad similares.
- Con trozos manejables tendremos una sensación de progreso.
- El nivel de granularidad es complicado, depende de muchos factores.

Definición de actividades

Herramientas y técnicas:

- Descomposición: Implica dividir los elementos del proyecto en componentes más pequeños y, por lo tanto, más manejables con el objetivo de proporcionar un mejor control de la gestión. La salida de esto son actividades (acciones a realizar), no elementos resultantes de las mismas.
- Plantillas: Una lista de actividades o una parte de dicha lista de un proyecto anterior se utiliza, a menudo, como plantilla para el nuevo proyecto. Del mismo modo, la lista de actividades de una estructura de descomposición del trabajo se puede utilizar como plantilla para otras estructuras de descomposición semejantes.
- Planificación gradual.

Salidas:

- Lista de actividades: No es solo una lista. Debe incluir una descripción más o menos detallada de cada actividad para que los miembros del equipo del proyecto entiendan correctamente cómo se debe realizar el trabajo.

- Atributos de la actividad: Los atributos de la actividad para cada actividad del cronograma incluyen el identificador de la actividad, los códigos de la actividad, la descripción, las actividades predecesoras y sucesoras, las relaciones lógicas, adelantos y retrasos, requisitos de recursos, fechas impuestas, restricciones y suposiciones.
- Lista de hitos: Identifica todos los hitos e indica si el hito es obligatorio (exigido por contrato) u opcional.
- Cambios solicitados: El proceso "Definición de las Actividades" puede generar cambios solicitados que afecten al alcance del proyecto y la EDT.

Factores que afectan a la estimación de tareas (actividades):

- Tamaño de la tarea.
- Complejidad.
- Familiaridad con la tarea y el sistema completo a llevar a cabo.
- Familiaridad con la tecnología a usar.
- Habilidades y experiencias del equipo.
- Cantidad de "proceso" que se pide.
- Necesidades especiales de velocidad, fiabilidad, reutilización, etc.

Secuenciación de actividades

Implica la identificación y documentación de las dependencias de interacción entre actividades. Las actividades deben estar secuenciadas de forma precisa para poder permitir el desarrollo posterior de una planificación temporal alcanzable. Se puede realizar de modo manual o automático con herramientas adecuadas.

Entradas:

- Enunciado del alcance del proyecto: Incluye las características del producto que con frecuencia pueden afectar al establecimiento de la secuencia de actividades, tal como el plano físico de una planta que se va a construir o las interfaces del subsistema de un proyecto de software.
- Lista de actividades.
- Atributos de actividad: Identificador, actividades predecesoras, etc.
- Lista de hitos.
- Solicitudes de cambio aprobadas.

Dependencias fundamentales: Inherentes al trabajo a realizar. Por ejemplo, construir un prototipo antes de probarlo.

Dependencias discrecionales: Definidas por el equipo del proyecto. Por ejemplo, relacionadas con prácticas anteriores con buen resultado.

Dependencias externas: Cuando se precisa de materiales o colaboración con agentes externos al equipo del proyecto para realizar determinadas actividades.

Herramientas y técnicas:

- Método de diagrama de precedencia: Tareas en los nodos.
- Método de diagrama de flecha: Tareas en las flechas.
- Métodos de diagramación condicionales.
- Plantillas de red: Puede haber etapas completas de otros proyectos que se pueden incluir en el desarrollo planificado actualmente.

Salidas:

- Diagrama de la red del proyecto.
- Actualizaciones de la lista de actividades.

Estimación de los recursos de las actividades

Implica la determinación del tipo de los recursos (personas, equipos, software) y las cantidades de cada uno de ellos se precisan para realizar las actividades del proyecto. Se debe coordinar con la estimación de costes.

Entradas:

- Factores ambientales de la empresa: Disponibilidad de sistemas de información de la gestión de proyectos y herramientas de software para la elaboración de cronogramas.
- Activos de los procesos de la organización: Proporcionan las políticas de la organización ejecutante en lo que respecta al personal y al alquiler o compra de suministros y equipos que se evalúan durante la estimación de recursos de las actividades. Si estuviera disponible, se revisa la información histórica relacionada con los tipos de recursos que fueron necesarios para un trabajo similar en proyectos anteriores.
- Lista de las actividades.
- Atributos de la actividad.
- Disponibilidad de los recursos: Necesario conocer qué recursos están disponibles durante el periodo de tiempo planificado.
- Plan de gestión del proyecto: El plan de gestión del cronograma es un componente del plan de gestión del proyecto, que se utiliza para la Estimación de Recursos de las actividades.

Herramientas y técnicas:

- Juicio de expertos, ya sean de otros departamentos, de consultoras, asociaciones profesionales y técnicas o grupos industriales.

- Análisis de alternativas: Algunas actividades pueden tener diversos métodos de realización, por ejemplo desarrollo propio o subcontratación.
- Datos de estimación publicados.
- Software de gestión de proyectos.
- Estimación ascendente.

Salidas:

- Requisitos de recursos de las actividades: Descripción de la lista de recursos que se precisan junto con las cantidades necesarias para cada tipo de recursos de acuerdo con la estructura de descomposición del trabajo.
- Atributos de la actividad (actualizaciones).
- Estructura de descomposición de los recursos.
- Calendario de recursos (actualizaciones).
- Cambios solicitados.

Estimación de la duración de las actividades

Supone la valoración de las unidades de tiempo precisas para la realización de cada actividad considerada. Esta valoración debería realizarla, o al menos aprobarla, la persona o grupo con más conocimiento de las características de dicha actividad.

Entradas:

- Factores ambientales de la empresa.
- Activos de los procesos de la organización.
- Enunciado del alcance del proyecto.
- Lista de actividades.
- Atributos de la actividad.
- Requisitos de recursos de las actividades.
- Calendario de los recursos.
- Plan de gestión del proyecto: Registro de riesgos, estimaciones de costes de las actividades.

Herramientas y técnicas:

- Juicio de expertos.
- Estimación análoga.
- Estimación paramétrica.
- Estimación por tres valores: Optimista, más probable y pesimista.
- Análisis de reserva.

Salidas:

- Estimación de la duración de todas las actividades.
- Bases de las estimaciones.
- Actualización de los atributos de las actividades.

Resumen referente a las actividades en un proyecto.

Un proyecto está compuesto de un número de actividades interrelacionadas.

Un proyecto puede comenzar cuando al menos una de sus actividades está dispuesta para empezar, y finalizará cuando todas las actividades implicadas en el mismo finalicen.

Una actividad debe tener un comienzo y un fin definidos de forma clara y precisa. Normalmente el fin coincide con la realización de algo tangible.

Si una actividad precisa de un recurso, éste debe estar previsto para su comienzo y se supone que dicho recurso se utilizará de forma constante durante la duración de dicha actividad.

La duración de una actividad debe ser predecible, suponiendo condiciones normales y disponibilidad de los recursos requeridos.

Algunas actividades precisan la finalización de otras para poder comenzar.

Consideraciones respecto a los modelos de redes de un proyecto

Una red tiene que tener un único nodo de comienzo y un único nodo de fin.

Un nodo tiene duración, pues representa una actividad que consume tiempo. Pero un arco no tiene duración pues representa una relación entre actividades.

Se llaman **precedentes** a las actividades inmediatamente precedentes.

El tiempo en una red se mueve de izquierda a derecha.

No puede haber bucles ni "columpios" (dependencias a mitad de una actividad).

Es posible representar retardos entre actividades.

Desarrollo de la planificación (cronograma)

Significa determinar la fecha de comienzo y finalización de todas las actividades del proyecto.

Entradas:

- Diagrama de red del proyecto.

- Estimaciones de la duración de la actividad.
- Requisitos de recursos.
- Descripción de las disponibilidades de recursos.
- Calendarios.
- Restricciones: fechas impuestas, sucesos fundamentales o hitos principales.
- Suposiciones.
- Retrasos e intervalos.

Herramientas y técnicas:

- Análisis matemáticos: Método del camino crítico (CPM), evaluación gráfica y técnica de revisión (GERT), y evaluación de programa y técnica de revisión (PERT).
- Reducción de la duración.
- Simulación.
- Heurísticas de nivelado de recursos.
- Software de gestión de proyectos.

Salidas:

- Planificación del proyecto.
- Detalles de apoyo: P. ej, la lista de todas las suposiciones y restricciones identificadas.
- Plan de gestión de la planificación.
- Actualización de las necesidades de recursos.

Resumen de los aspectos que debe contemplar un plan de proyecto

1. Definición clara del producto a elaborar.
2. Estimación de coste y esfuerzo requeridos para la elaboración.
3. Estudio de problemas que se pueden presentar durante la ejecución del proyecto y cómo solventarlos o minimizarlos.
4. Determinación de las tareas que engloba el proyecto.
5. Recursos requeridos y su asignación a distintas tareas.
6. Elaboración del calendario de ejecución.

Control de la planificación

Significa considerar los cambios que se producen sobre lo planificado e intentar evitar situaciones críticas como consecuencia de los mismos

Entradas:

- Planificación del proyecto.
- Informes de seguimiento: Qué actividades se han desarrollado en plazo y cuáles no.
- Solicitudes de cambios.
- Plan de gestión de la planificación.

Herramientas y técnicas:

- Sistema de control de cambios en la planificación.
- Medidas de seguimiento: A veces hay que decidir si una variación en lo planificado requiere una acción correctiva.
- Planificación adicional.
- Software de gestión de proyectos.

Salidas:

- Actualizaciones del plan: A veces se precisan revisiones, que suponen cambios en las fechas de comienzo y final de las actividades planificadas.
- Acción correctiva. Se intenta dirigir la planificación de las actividades futuras hacia lo planificado en el plan.
- Lecciones aprendidas.

Gestión de los costes del proyecto

Incluye el proceso seguido para garantizar que el proyecto es finalizado con el presupuesto aprobado. Consta de las siguientes etapas:

1. Estimación de los costes: Desarrollar una estimación de los costes de los recursos necesarios para finalizar las actividades.
2. Presupuesto de los costes: Conseguir el coste total estimado para los elementos del trabajo.
3. Control de los costes: Controlar los cambios sobre el presupuesto del proyecto.

Estimación de costes Supone el desarrollo de una estimación de los costes de los recursos precisos para llevar a cabo las actividades del proyecto. Cuando el proyecto se realiza en el marco de un contrato hay que diferenciar coste de precio. Mientras que el coste implica una evolución cuantitativa siguiendo un método más o menos técnico, el precio es una decisión de negocio de la empresa.

Costes a considerar en un proyecto:

- Costes del hardware y software usados.

- Coste de dietas, viajes y aprendizaje.
- Coste del esfuerzo (principal factor de coste): - Salarios del personal involucrado en el proyecto. - Costes sociales y de seguros individuales.
- Costes indirectos aplicados al personal del proyecto: - Costes de mantenimiento del edificio, luz, calefacción, etc. - Costes administrativos y de comunicaciones. - Costes sociales colectivos.

Entradas:

- Estructura de descomposición del trabajo.
- Requisitos de recursos: descritos en planificación de recursos.
- Tasas de los recursos: En caso de recursos humanos, se debe conocer el coste por unidad de tiempo de cada tipo de recurso (analista, diseñador, etc).
- Estimación de la duración de las actividades.
- Información histórica: Ya sea de ficheros de proyectos anteriores, bases de datos de estimación de costes comerciales o conocimiento del equipo del proyecto.
- Gráfico de cuentas (contabilidad): A nivel de empresa, la realización de un proyecto está enmarcada en el contexto de una actividad económica, por lo que se debe conocer perfectamente su imputación contable.

Herramientas y técnicas:

- Estimaciones análogas.
- Modelado paramétrico.
- Estimaciones bottom-up.
- Herramientas automatizadas.

Salidas:

- Estimación de los costes: Valoraciones cuantitativas de los costes de los recursos precisos para finalizar el proyecto.
- Detalles de apoyo: Documentación básica sobre cómo se realizó la estimación. Documentación de cualquier suposición considerada al realizar la estimación. Rangos de los posibles resultados.
- Plan de gestión del coste: Describe cómo se gestionarán las posibles variaciones del coste.

Presupuestado de costes Implica la asignación de las estimaciones de coste totales a los elementos de trabajo individuales para establecer una línea base de coste que sirva para medir las prestaciones del proyecto.

Entradas:

- Estimación de costes.
- Estructura de descomposición de tareas.
- Planificación temporal del proyecto.

Herramientas y técnicas:

- Herramientas y técnicas específicas de estimación de costes.

Salidas:

- Línea base de costes: Valoraciones cuantitativas de los costes de los recursos precisos para finalizar el proyecto.

Control de costes Entradas:

- Línea base de costes.
- Informes de seguimiento.
- Solicitudes de cambio.
- Plan de gestión del coste.

Herramientas y técnicas:

- Sistema de control de cambios en los costes.
- Medidas de prestaciones (seguimiento).
- Planificación adicional.
- Herramientas informatizadas.

Salidas:

- Estimaciones de costes revisadas.

Requisitos del proyecto

Se dividen en:

- Limitaciones de recursos: Lista de cuántos.
- Objetivos de calidad: Lista de cómo.
- Requisitos funcionales: Lista de qué.

Todos los requisitos de calidad pueden y deben ser establecidos de forma no ambigua. La dificultad para definir la calidad es traducir las necesidades futuras en características medibles, de forma que un producto pueda ser diseñado y construido para proporcionar satisfacción al precio que el usuario va a pagar.

Si los requisitos son oscuros, incompletos o erróneos, la arquitectura del sistema también lo será.

Si la arquitectura no es correcta, las estimaciones de coste serán erróneas.

Si las estimaciones son erróneas, el proyecto estará mal gestionado.

Si los requisitos y la arquitectura son incorrectos, el diseño detallado también lo será y asimismo la implementación.

Las soluciones solo son válidas si satisfacen los requisitos establecidos, pero no más.

Aspectos comunes de la gestión de proyectos

Las actividades de gestión no son peculiares de los proyectos de software. La mayoría de las técnicas de gestión de proyectos son aplicables a proyectos informáticos. Los problemas de ingeniería complejos e innovadores tienden a sufrir los mismos problemas que los proyectos de software.

Aspectos clave:

- Una buena gestión es esencial para el éxito de un proyecto.
- La naturaleza intangible del software causa problemas de gestión.
- Las principales actividades de los gestores se centran en la planificación, estimación y control.
- La planificación y la estimación son procesos iterativos que se realizan a lo largo de todo el proyecto.

Señales de peligro para un proyecto de sistemas de información

- El personal no entiende perfectamente las necesidades del cliente.
- El ámbito del producto está mal definido.
- Mala gestión de los cambios.
- La tecnología adecuada cambia.
- Las necesidades comerciales cambian o están mal definidas.
- Los plazos de entrega no son realistas.
- Los usuarios se resisten.
- Se pierde el soporte de la dirección, o no se obtuvo adecuadamente.
- El equipo del proyecto carece de personal con las habilidades adecuadas.
- Los gestores no utilizan las mejores prácticas y las lecciones aprendidas.

Cómo debe actuar un gestor de proyecto para evitar estos problemas:

- Comience con el pie derecho: Hay que entender el problema a resolver correctamente y establecer objetivos y expectativas realistas para los que están implicados en el proyecto. Se precisa conseguir el equipo adecuado y darle autonomía, autoridad y tecnología para llevar a buen término el proyecto.
- Mantenga el ímpetu: No dejar que el proyecto se desintegre lentamente tras un buen comienzo.
- Rastree el progreso: El progreso se sigue conforme se elaboran los productos a obtener del mismo y se aprueban mediante los mecanismos de garantía de calidad establecidos. Se pueden tomar medidas de proyecto para realizar la valoración del proyecto.
- Tome decisiones inteligentes: Lo más sencillo es lo que mejor funciona. Si se tiene información de un software comercial o componentes existentes que satisfacen las necesidades de algo, no hay por qué plantearse abordar ese desarrollo.
- Realice un análisis de los resultados: Hay que aprender de las experiencias en proyectos anteriores.

El coste del cambio en desarrollo es entre 1.5 y 6 veces mayor que en la definición del proyecto, y después de la entrega es de 60 a 100 veces mayor que en la definición.

Competencias que necesita conocer cada gestor de proyectos de software

Producto:

- Valorar procesos.
- Conocimiento de procesos tipo.
- Definir el producto.
- Evaluar modelos de proceso alternativos.
- Gestionar los requisitos.
- Gestionar a los subcontratistas.
- Realizar una valoración inicial.
- Seleccionar métodos y herramientas.
- Elaborar procesos.
- Seguir la calidad del producto.
- Comprender las actividades de desarrollo.

Proyecto:

- Construir estructuras de descomposición del trabajo.
- Documentar los planes.
- Estimar costes.
- Estimar esfuerzo.
- Gestionar riesgos.
- Monitorizar el desarrollo.
- Planificar.
- Seleccionar medidas.
- Seleccionar herramientas de gestión de proceso.
- Seguimiento del proceso.
- Seguimiento del progreso del proyecto.

Personal:

- Evaluar equipos para mejorar las prestaciones.
- Considerar la propiedad intelectual.
- Realizar reuniones efectivas.
- Interacción y documentación.
- Liderazgo.
- Gestionar los cambios.
- Negociar con éxito.
- Planificar la progresión del personal.
- Presentar el proyecto y los resultados de forma efectiva.
- Seleccionar y entrevistar personas.
- Seleccionar equipo.
- Construir un equipo formando, dirigiendo y manteniéndolo.

Funciones del gestor de proyectos

- Arranque y tareas administrativas.
- Crea un plan de proyecto y una planificación.
  - Definir los objetivos del proyecto.
  - Identificar un proceso estándar adecuado para la ejecución del proyecto.
  - Ajustar el proceso estándar para que coincida con los requisitos.
  - Definir un proceso para gestionar los cambios en los requisitos.
  - Estimar el esfuerzo.
  - Planificar los recursos humanos y la organización del equipo.
  - Definir los hitos y crear una planificación temporal.
  - Realizar un plan de prevención de defectos.
  - Identificar o mitigar su efecto.

  - Definir un plan de medidas para el proyecto.
  - Definir un plan de formación para el proyecto.
  - Definir procedimientos de seguimiento del proyecto.

- Realizar una revisión del plan del proyecto y su planificación temporal.
- Obtener autorización del director del departamento.
- Definir y revisar un plan de configuraciones.
- Orientar al equipo del proyecto para el plan de gestión del proyecto.
- Ejecutar el proyecto como está establecido en el plan.
- Monitorizar el estado del proyecto.
- Revisar el estado del proyecto con un gestor senior de proyectos.
- Monitorizar el cumplimiento con el proceso del proyecto definido.
- Analizar los defectos y realizar actividades de prevención de defectos.
- Monitorizar las prestaciones a nivel de programa.
- Realizar revisiones de hitos y replanificar si es necesario.

La gestión de proyectos tradicional:

- No reconoce la existencia de iteraciones.
- Es inflexible, cambiar el plan se considera un fallo.
- No piensa en los proyectos en un sentido probabilístico.
- Es rehén del software existente de gestión de proyectos.
- Funciona en modo reactivo, no considera los avisos tempranos en el sistema.

Ideas sobre PERT y CPM (PDF)

- PERT: Program Evaluator and Review Technique.
- CPM: Critical Path Method.

Facilitan la planificación de proyectos en los que estén implicadas muchas actividades. Ambas fuerzan al gestor del proyecto a planificar con un cierto nivel de detalle. CPM añade el tema de gestión de recursos y costes. Ambas técnicas facilitan la respuesta a preguntas como:

- ¿Cuándo finalizará el proyecto?
- ¿Cuáles son las actividades críticas (las que retrasarían el proyecto si alguna de ellas se retrasa)?

- ¿Cuáles son las actividades no críticas (las que se pueden retrasar algo sin que haya incidencia en el proyecto)?
- ¿Cuánto se pueden retrasar las actividades sin retrasar significativamente el proyecto?
- ¿Cómo se pueden concentrar los recursos para acelerar la finalización del proyecto?

Por tanto, permiten:

- detectar cuellos de botella,
- cumplir plazos de entrega con mayor probabilidad,
- evaluar los efectos de los cambios en el desarrollo del proyecto.

Pasos a realizar

1. Definir el proyecto y todas sus actividades significativas.
2. Determinar las relaciones de precedencia entre dichas actividades.
3. Estimar el tiempo necesario para finalizar cada actividad.
4. Dibujar una red que conecte todas las actividades y las etiquete con las estimaciones de tiempo.
5. Calcular el camino crítico a través de la red.
6. Utilizar la red para planificar, temporizar, monitorizar y controlar el proyecto.

Ficha de especificación de tarea

Contendrá los siguientes datos:

- Número.
- Nombre.
- Descripción.
- Esfuerzo estimado.
- Personas.
- Recursos.
- Duración.
- Entregables.
- Predecesoras.

Características de la red:

Los **nodos** representan eventos (comienzo o fin de una o varias actividades). Los **arcos** representan actividades. Un **camino** es una secuencia de actividades conectadas desde el nodo de comienzo hasta el de finalización.

Dos nodos no pueden estar directamente conectados por más de un arco. Cada actividad se representa con un único arco.

Se pueden usar **actividades ficticias** para establecer una relación de precedencia. Es una actividad sin duración. Debe usarse para no violar las propiedades definidas anteriormente.

También se pueden representar las tareas en el nodo (el nodo es un cuadrado) o en la flecha que une los nodos (el nodo es un círculo).

La duración del proyecto será la longitud del camino más largo o **camino crítico**. Las **actividades críticas** son las que forman el camino crítico.

Tiempo más pronto posible para un evento.

E_j es el instante en el que se producirá el evento (Evento, no tarea) j si todas las actividades precedentes han comenzado tan pronto como sea posible.

- Pasada hacia adelante: Se calcula E_j para cada evento j comenzando en el primer nodo y avanzando hasta el evento final.
- Regla del tiempo más pronto posible: E_j es el **máximo** de las sumas $E_i + t_{ij}$ para cada evento inmediatamente precedente i e interviniendo en la actividad ij.

Tiempo más tarde permisible.

L_i es el tiempo más tarde en el que se puede comenzar el evento i sin que se produzca un retardo en el proyecto.

- Pasada hacia atrás: Se calcula L_i para cada evento i comenzando por el último nodo y moviéndose hacia atrás.
- Regla del tiempo más tarde permisible: L_i es el **mínimo** de las diferencias $L_j - t_{ij}$ entre cada evento inmediatamente siguiente y la actividad ij

Holgura para una actividad

Ventana de tiempo en la que puede comenzar dicha actividad.

ES: Tiempo más pronto posible para la actividad. Longitud del recorrido más largo desde el nodo de inicio hasta el comienzo de la actividad.

LS: Tiempo más tarde que puede empezar una actividad sin retrasar el proyecto.

Holgura de la actividad $ij = LS - ES = (L_j - t_{ij}) - E_i$.

Una **Actividad crítica** es la que tiene holgura 0. El camino crítico está formado por todas las actividades críticas. Si una actividad crítica se retrasa, se retrasará todo el proyecto.

Datos usados en PERT y CPM

- Early Start
- Early Finish
- Latest Start
- Latest Finish

La duración representa el tiempo necesario para llevar a cabo la actividad en el momento de planificar o el tiempo restante cuando se hacen replanificaciones.

Posibles dependencias entre tareas

- La tarea B no puede empezar hasta que la tarea A termine: Finish-to-start (FS).
- La tarea B no puede empezar hasta que la tarea A empiece: Start-to-start (Ss).
- La tarea B no puede terminar hasta que la tarea B termine: Finish-to-finish (FF).
- La tarea B no puede terminar hasta que la tarea A termine: Start-to-finish (SF).

Ventajas de PERT

- Fuerza a gestionar el plan.
- Permite observar las interrelaciones entre actividades e identifica el camino crítico.
- Expone el paralelismo que se puede dar en las actividades, lo que ayuda a la hora de exponer recursos.
- Permite la planificación temporal y simulación de planificaciones alternativas.
- Permite al gestor del proyecto realizar el seguimiento y control del mismo.

Holguras en CPM y PERT (PDF)

Tanto PERT como CPM son métodos que incorporan incertidumbre en la gestión de proyectos y dan las probabilidades de finalizar las etapas del proyecto en tiempos límite especificados y una estimación de la finalización del proyecto.

Sin embargo, CPM no incorpora incertidumbre en los tiempos de las tareas, y en su lugar supone que los tiempos de las actividades son propocionales a los recursos reservados para ellas, y que cambiando los recursos asignados cambia la duración de las tareas; lo que requiere más experiencia sobre la relación entre los recursos y tiempos.

Holguras en CPM

En CPM las holguras se llaman **flotantes** (float) e indican el grado de libertad de la planificación bajo una variedad de condiciones, por ejemplo la disponibilidad de los recursos. Hay cuatro holguras:

1. Holgura total.
2. Holgura libre.
3. Holgura independiente.
4. Holgura de seguridad o programada.

Para definir estas holguras, usamos las siguientes definiciones:

- Tiempo más pronto posible (ES_{ij}) de una actividad: instante más pronto posible que puede comenzar dicha actividad suponiendo que todas las anteriores han finalizado lo más pronto posible.
- Tiempo más tarde permisible ($LFij$) de una actividad: instante más tarde que puede finalizar una actividad sin retrasar el tiempo de finalización de todo el proyecto.

También hay otras dos definiciones para el tiempo más pronto posible de finalización (EF_{ij}) o más tarde permisible de comienzo (LS_{ij})

El **tiempo flotante total** coincide con la holgura en PERT y es el tiempo que una actividad puede retrasarse sin que dicho retraso afecte a la duración total del proyecto. Se puede obtener de dos formas:

- $Ft_{ij} = LS_{ij} - ES_{ij}$
- $Ft_{ij} = LF_{ij} - EF_{ij}$

Gestión de recursos (PDF)

Introducción

A lo largo del desarrollo de un proyecto se precisan distintos tipos de recursos. La idea es proporcionar pautas para la selección y adscripción de los recursos al proyecto a lo largo del tiempo. La forma de reservar los recursos puede implicar la existencia de restricciones sobre las tareas programadas y por lo tanto puede incidir en la planificación temporal considerada. Por tanto, una de las tareas del responsable del proyecto será buscar la concordancia entre las tareas planificadas y los recursos disponibles en cada momento.

Productos resultantes: calendarización de actividades, de recursos y del coste.

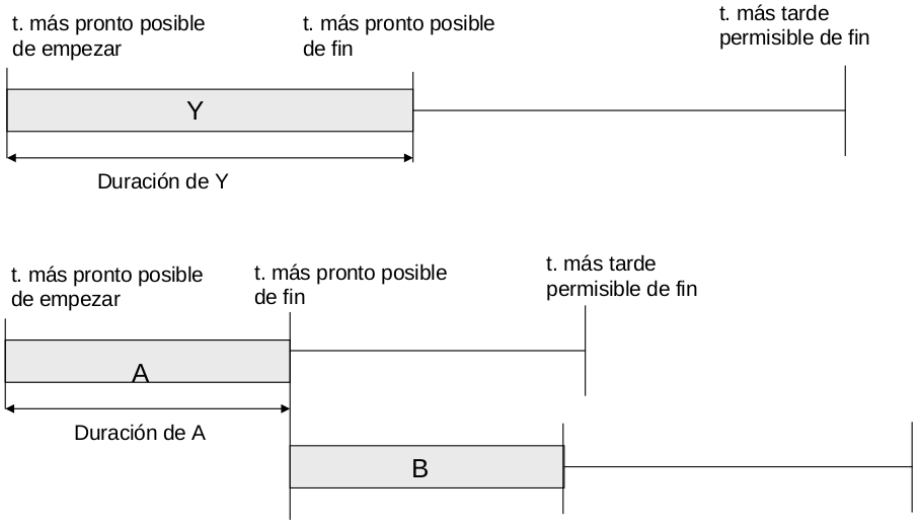

Figure 1: Cálculo de la holgura

Tipos de recursos

Según Pressman, los recursos pueden dividirse en:

- Personal (número, capacidad, ubicación)
- Software (componentes OTS, nuevo componentes, componentes probados, componentes de experiencia parcial)
- Entorno (herramienta software, hardware, recursos de red)

Cada recurso debe especificar: descripción, informe de disponibilidad, fecha de necesidad y tiempo durante el que se necesitará. En caso de RRHH, especificar las habilidades técnicas que requiere el recurso.

Hughes y Cotterell dividen los recursos en:

- Trabajo: Miembros del equipo de desarrollo, garantía de calidad, etc.
- Equipamiento: Material informático e infraestructura física para su funcionamiento, así como mobiliario.
- Materiales: Consumibles de informática, papel, etc.
- Espacio: En una organización existente, ya está disponible. Hay que contar con posible personal adicional.
- Servicios: Algunos proyectos requieren servicios especiales, como un sistema WAN u otros.
- Tiempo: Uno de los recursos principales que, a veces, está preestablecido.
- Dinero: Recurso secundario usado para comprar otros recursos.

Identificación de las necesidades de recursos

Aunque muchos recursos están ligados a una actividad, hay otros que no pero son parte de la infraestructura del proyecto, ya sea física, como espacio para nuevo personal, o de personal, como el esfuerzo de planificación y seguimiento que hace el responsable del proyecto, y por tanto tienen que reflejarse en el plan. Lo más indicado es hacer una lista de necesidades de recursos lo antes posible.

La estimación de recursos debe darse en días/persona, y para tareas cortas se pueden usar horas/personas aunque sería más uniforme usar fracciones de días/persona.

Rol de gestor de proyecto

El gestor de proyecto es quien decide quién hace qué y quién decide qué a lo largo del proyecto. Para ello, hay que considerar el balance entre posibilidades de aprendizaje frente a restricciones temporales, considerado las habilidades específicas necesarias para cada actividad. Este trabajo se facilita disponiendo de una especificación clara de los roles involucrados y un conocimiento del equipo.

Para cada rol, el gestor de proyecto debe definir tres aspectos: su responsabilidad, especificando qué está obligado a hacer; su autoridad, que es el derecho que tiene a realizar, mandar y tomar decisiones; y su exigencia, que dice en qué medida este rol asumirá la responsabilidad del éxito o fracaso de la actividad.

Fichas de recurso

Se debe crear una ficha para cada recurso en que se especifiquen al menos las siguientes características:

- Descripción del recurso.
- Informe sobre su disponibilidad.
- Fecha de comienzo en la que se precisa el recurso.
- Tiempo durante el que se precisa el recurso.
- En caso de recursos humanos, habilidades técnicas que precisa el recurso para la tarea.

Gestión de recursos humanos en el PMBOK

El PMBOK dedica un capítulo a la gestión de recursos humanos, dividiéndolo en cuatro partes:

- 9.1 Planificación de los recursos humanos: identificar y documentar los roles de proyecto, responsabilidades y relaciones de informe así como crear el plan de gestión del personal.

- 9.2 Adquirir el equipo de proyecto: obtener los recursos humanos necesarios.
- 9.3 Desarrollar el equipo del proyecto: mejorar las competencias y la interacción de los miembros del equipo para lograr un mejor rendimiento.
- 9.4 Gestionar el equipo del proyecto: hacer un seguimiento del rendimiento de los miembros del equipo, proporcionar retroalimentación, resolver polémicas y coordinar cambios a fin de mejorar el rendimiento del proyecto.

En este caso el PMBOK se enfoca más en aspectos relacionados con la gestión de recursos humanos que con la planificación de los mismos.

Cuadro RACI

En el PMBOK se habla de una matriz de responsabilidades para el personal. Cada persona tiene una responsabilidad respecto a cada actividad. Estas responsabilidades pueden ser *Responsible* (Responsable), *Accountable* (Subordinado-responsable), *Consult* (Consultado) e *Inform* (Informado).

Diagramas de planificación de recursos

Es útil poder ver únicamente las tareas que hay asignadas a cada recurso para comunicar a los participantes el uso de un recurso compartido, verificar que se usan de forma equilibrada o verificar que no se pretende usar ningún recurso más de lo posible. Para esto se usan el diagrama de Gantt y el de cargas.

Asignación de recursos a tareas

A la hora de asignar recursos a tareas hay que tener en cuenta:

- Disponibilidad. Tiene que haber personas adecuadas para no alargar el proyecto.
- Criticidad. Asignar recursos con más experiencia a actividades del camino crítico puede ayudar a reducir la duración del proyecto o disminuir el riesgo de alargamiento.
- Riesgo. Es importante identificar tareas que plantean más riesgos y qué factores influyen en ello para asignar personal.
- Entrenamiento. La organización se beneficia asignando personas inexpertas en tareas no críticas donde, con la holgura, podrán ir formándose.
- Formación de equipo. La selección debe tener en cuenta las cualidades considerando las características del proyecto y que las personas deben trabajar conjuntamente.

Conflictos sobre recursos

Se producen conflictos sobre recursos cuando el mismo recurso se precisa en más de un lugar al mismo tiempo. Se pueden resolver retrasando una de las actividades (aprovechando la holgura o retrasando la finalización del proyecto), moviendo un recurso desde una actividad no crítica, proporcionando recursos adicionales (aumenta el coste), dividiendo la actividad en dos para facilitar la asignación o permitiendo una sobrecarga de trabajo (también supone mayor coste).

Es conveniente priorizar las tareas de forma que la asignación de los recursos se haga siguiendo un orden racional predeterminado. Existen varias formas de proporcionar prioridades a los recursos:

- Prioridad por holgura total: Las tareas con menor holgura tienen la mayor prioridad a la hora de asignar recursos.
- Prioridad por lista ordenada: Las actividades que se realizan al mismo tiempo se ordenan según un criterio sencillo. Según Burman sería: - Actividad crítica más corta. - Actividad crítica. - Actividad no crítica más corta. - Actividad no crítica con menor holgura. - Actividad no crítica.

Se necesita maximizar el porcentaje de utilización de los recursos, es decir, reducir periodos vacíos entre tareas. Hay que balancear costes frente a anticipar la fecha de finalización. También es necesario permitir contingencias.

La planificación de recursos puede crear nuevas dependencias entre actividades incidiendo en los caminos críticos. Por ello, es mejor no añadir dependencias a la red de actividades para reflejar las restricciones de recursos. La restricción puede desaparecer durante el proyecto pero el enlace permanece.

Gestión de personas y organización de grupos (PDF)

En general la gestión de personas y la organización de equipos de trabajo son temas que tienen incidencia en todas las etapas de la planifiación y ejecución de proyectos, pero hay algunas cuestiones que conviene tener en cuenta:

- Algunos proyectos pueden incidir sobre la salud y seguridad durante la realización del mismo.
- Aunque, en general, un responsable de proyecto tiene poco control sobre la estructura organizativa, debe conocer las implicaciones que la misma puede tener sobre el desarrollo del proyecto.
- El alcance y la naturaleza de las actividades pueden estar planteados de una forma que mejore la motivación del personal.
- Muchos riesgos para el éxito del proyecto están relacionados con la gestión del personal.

- Las cualidades de los individuos deben ser tenidas en cuenta a la hora de asignar personas a las actividades.

Gestión de personas y organización de grupos en el PMBOK

El desarrollo de un equipo de proyecto, según el PMBOK, mejora las competencias e interacciones de los miembros del equipo a fin de mejorar el rendimiento del proyecto. Se busca mejorar las habilidades de los miembros del equipo a fin de aumentar su capacidad de completar las actividades del proyecto, y mejorar la cohesión entre los miembros del equipo buscando aumento de la productivdad a través de mayor trabajo en equipo.

Un trabajo en equipo efectivo puede ser la ayuda mutua cuando la carga del trabajo no es equilibrada, una mejor comunicación interpersonal o compartir información y recursos. Los esfuerzos para el desarrollo del equipo son más beneficiosos cuando se realizan en las fases tempranas, pero deberían tener lugar durante todo el proyecto.

Comportamiento organizativo

Frederick Taylor intenta analizar la forma más productiva de realizar tareas manuales con el objetivo de seleccionar la persona más adecuada para realizar el trabajo, instruir a dicha persona en los mejores métodos y proporcionar incentivos salariales a los mejores trabajadores.

En desarrollo de software, el desarrollo de métodos estructurados o el uso de herramientas CASE son un ejemplo de énfasis en mejores prácticas.

Douglas McGregor plantea dos teorías contrapuestas sobre la relación de los trabajadores con el trabajo. La teoría X dice que los trabajadores solo actúan mediante amenazas y la teoría Y se basa en que la gente quiere y necesita trabajar.

Planificación de recursos en el PMBOK

El PMBOK considera el plan de gestión de personal como una de las salidas de la planificación de recursos. Este plan contempla aspectos como adquisición de personal, horarios, criterios de liberación, necesidades formativas, reconocimientos y recompensas, cumplimiento de regulaciones y otras normas y seguridad si fuera aplicable.

Trabajo en grupo

Aunque se suele considerar la ingeniería informática como un trabajo en solitario, la mayor parte trabajan en pequeños grupos, y una gran parte del tiempo de

trabajo se dedica a interacciones con otros compañeros, superiores, clientes, etc.

No es suficiente reunir a varias personas para formar un grupo. Un grupo se debe enfrentar a obstáculos y debe permitir el debate sano cuando hay conflictos. El grupo debe buscar la consecución de un objetivo común y debe tener relaciones armoniosas entre sus miembros. De esa forma, un grupo transforma las energías individuales en energía de grupo.

La estructura del grupo puede ser muy dispar pero es recomendable que sea flexible, aunque es necesario que haya una única persona que sea el contacto con el exterior.

Características de los grupos

Cinco etapas básicas del desarrollo de un equipo según Tuckman y Jensen:

1. Formación. Los miembros se llegan a conocer e intentan fijar las pautas de comportamiento.
2. Storming. Se producen conflictos cuando los miembros del equipo intentan ejercer el liderazgo y se están estableciendo los métodos de operación.
3. Norming. Conflictos determinados, empieza a emerger un setimiento de identidad de grupo.
4. Actuación (performing). Énfasis en la tarea a llevar a cabo.
5. Finalización (adjourning). El grupo se disgrega.

No suele funcionar juntar a la mejor gente. Se precisa un balance de roles con habilidades y roles de gestión.

Problemas habituales

- Liderazgo no efectivo.
- Fracaso a la hora de comprometese o cooperar.
- Falta de participación.
- Procrastinación.
- Baja calidad.
- Función de reptado, peloteo.
- Evaluación inefícaz entre colegas.

Motivación de las personas

La satisfacción en el trabajo se puede mejorar aumentando las posibilidades de logros, reconocimiento, responsabilidad y avance en la carrera. Mejorar en supervisión, salario y condiciones laborarles puede reducir la falta de satisfacción en el trabajo.

Se puede mantener la motivación proporcionando oportunidades para que los miembros estén al corriente, indicando la importancia y valor del grupo, haciendo que la gente se sienta importante, clarificando objetivos, identificando el progreso y reconociendo los logros.

Tareas

Hay que considerar qué tareas deben realizarse en grupo y cuáles pueden ser tareas individuales. Según Hughes y Cotterell, las tareas en grupo se pueden categorizar de esta manera:

- Tareas aditivas. El trabajo de cada participante se añade para conseguir el resultado final, siendo las personas intercambiables. Existe el riesgo de que unos contribuyan más que otros.
- Tareas compensatorias. Los esfuerzos se promedian. Por ejemplo, al hacer estimaciones.
- Tareas disyuntivas. Solo uno tiene la respuesta correcta y el resto la acepta como tal.
- Tareas conjuntivas. El progreso viene dado por la tasa del miembro más lento. Si cada uno desarrolla un módulo, el proyecto no está terminado hasta que todos han terminado.

A la hora de asignar tareas a las personas del equipo se puede establecer este orden como heurística:

1. La persona tiene la habilidad y la experiencia necesarias para hacer el trabajo y lo quiere hacer.
2. La persona tiene la habilidad y la experiencia necesarias para hacer el trabajo y está preparada para hacerlo.
3. La persona tiene la habilidad y la experiencia necesarias para hacer el trabajo pero no está preparada para hacerlo.
4. La persona puede ser entrenada y formada para hacer el trabajo.
5. La persona no puede hacer el trabajo.

Competencias clave (AEIPRO) por niveles

- A. Ha dirigido con éxito la gestión del trabajo en equipo para programas o carteras importantes de una organización o unidad de una organización.
- B. Ha dirigido con éxito situaciones de trabajo en equipo de proyectos complejos.
- C. Ha dirigido con éxito situaciones de trabajo en equipo de proyectos de complejidad limitada.
- D. Tiene el conocimiento requerido en relación con la dirección del trabajo en equipo de proyectos y puede aplicarlo.

Consideraciones a tener en cuenta

Para conseguir una participación y comunicación óptimas, cada miembro del grupo debe:

- Participar tanto como emisor como receptor.
- Dirigirse al grupo y no solo a un individuo.
- Participar de forma positiva.
- Participar en el liderazgo.
- Dar pruebas de atención a los demás miembros.
- Comunicar sentimientos e impresiones.

Hay que tener encuenta el tiempo perdido por cambios de contexto. En general, trabajar en más de un proyecto genera pérdida de tiempo y favorece la disminución de la productividad. Se considera que un programador necesita entre 10 y 15 minutos para volver a un estado de concentración tras una distracción. Para evitarlas, algunos sugieren periodos de 25 minutos totalmente exentos de interrupciones.

La productividad de un equipo desciende con el número de miembros. No por meter más miembros se reduce necesariamente el tiempo del proyecto. Saturar a los miembros del equipo tampoco proporciona mejores resultados. Programar con música puede hacerte menos productivo porque reduce la creatividad.

Equipos dispersos y virtuales

Tener equipos dispersos reduce costes de habilitación de los centros de trabajo, también evitando ruidos y distracciones. Estudios afirman que el trabajo remoto puede ser más productivo. Incluso se puede contratar a gente en otros países aprovechando menores costes, contratos de corta duración, aprovechamiento de especialistas, e incluso aprovechar la diferencia horaria para que unos trabajen y otros hagan las pruebas después.

Sin embargo, en los equipos dispersos hay que especificar cuidadosamente y documentar formalmente los requisitos de trabajo y los procedimientos, pues la coordinación puede ser difícil. Se puede generar falta de confianza por la falta de contacto. También hay que modificar los métodos de pago.

Liderazgo

Hay varios estilos. Puede ser **orientado a tareas**, con foco en el trabajo que está siendo realizado, u **orientado a personas**, con foco en las relaciones.

Ante la incertidumbre sobre la forma en que se va a realizar el trabajo o si el personal no tiene experiencia, es preferible la orientación a tareas, mientras que si esta incertidumbre se reduce, es mejor la orientación a personas.

Reuniones

Las reuniones deben ser adecuadas para no generar pérdida de tiempo, frustraciones y descenso de compromiso. El trabajo no se hace en las reuniones, sino entre ellas. Una reunión constituye una etapa, sirve para coordinar, repartir tareas, informar, etc. Una gestión eficaz de las reuniones implica prepararlas, conducirlas y cerrarlas.

No se deben realizar reuniones improvisadas. Se debe elegir un "animador" que es quien prepara el orden del día y estima la duración de cada punto. Cada persona tiene que preparar la reunión individualmente leyendo previamente la documentación, realizando las tareas previstas y anotando proposiciones y preguntas.

Al conducir la reunión hay que recibir a los participantes reactivando la solidaridad y estrechando lazos. El moderador no decide los contenidos. Se encarga de los procesos, abre la reunión, presenta el orden del día y la duración prevista de cada punto, se asegura de que el objetivo de la reunión está claro, permite expresarse a todos, hace que el grupo tome decisiones, deja las conclusiones por escrito.

Al final de la reunión hay que evitar las salidas precipitadas y un final tipo "pescadilla que se muerde la cola". Hay que resumir el trabajo realizado yendo a lo esencial (las decisiones tomadas), siendo breve, fiel y objetivo. Hay que evaluar periódicamente el funcionamiento del grupo, teniendo en cuenta la calidad de la participación, la comunicación, el desarrollo de las reuniones y de la animación.

Es tan importante el contenido (de qué se habla) como la forma (cómo se habla).

Participación en el grupo

Contribuir a la formación del grupo significa buscar, definir, perseguir y alcanzar el objetivo común. Hay que crear un clima de eficacia, evitando la apatía y la febrilidad.

La participación no adecuada se produce cuando existe interacción insuficiente entre un miembro y el objetivo común, ya sea porque internamente decide no decir nada o no implicarse, es indiferente hacia el tema y no participa, tiene preocupaciones personales o somnolencia, está descontento porque se siente obligado, se opone abiertamente a las reuniones o trabajo en grupo por considerarlo ineficaz, tiene pocas habilidades sociales o no tiene suficiente competencia o información sobre el contenido de los debates.

Posiciones de participación

En función del grado de compromiso de un miembro con relación al objetivo común del grupo, la posición puede ser:

- Centro: Orienta al grupo en la búsqueda, definición o consecución del objetivo común.
- Emisor: Aporta una contribución personal directamente relacionada con el objetivo común, dando su opinión personal sin generalizar, aportando sus comentarios en forma de acuerdo o desacuerdo, respondiendo a las preguntas y planteando preguntas relativas al tema en cuestión.
- Receptor: Está atento y receptivo respecto a lo que pasa en el grupo según el objetivo común.
- Satélite: No dirige su atención al objetivo común.
- Ausente: No va a las reuniones, o llega tarde.

Autoevaluación

Cada miembro del grupo puede preguntarse si participa, si la comunicación fue buena (buena atmósfera, sin agresividad, todo el mundo pudo hablar), si las reuniones estaban bien preparadas y organizadas, las intervenciones han sido coordinadas, se ha fomentado la comunicación entre miembors, se han alcanzado los objetivos, se han respetado las duraciones previstas, se ha cerrado la reunión eficazmente... Además, puede hacer un balance personal expresado a través de proposiciones y comentarios.

Gestión de riesgos en desarrollo de software (PDF)

Introducción

Los proyectos de software pueden ir mal por múltiples razones como conocimiento poco adecuado de las necesidades del usuario, documentos de requisitos poco elaborados, gestión pobre de requisitos, arquitectura o diseño pobre o inexistente, codificar antes de plantear las preguntas, comprensión pobre del diseño o código legado, falta de revisión entre pares para detectar problemas en fases iniciales, personal sin experiencia o incompetente, pruebas poco efectivas...

Un **riesgo** es, según el PRINCE2, una "casualidad de exposición a las consecuencias adversas de eventos futuros" y, según el PMBOK, es "un evento o una condición de inciertos que, si ocurren, tienen un efecto positivo o negativo sobre los objetivos del proyecto".

Los riesgos se relacionan con posibles problemas futuros, no con los actuales. Implican por tanto una posible causa y su efecto. Un riesgo es una medida de la probabilidad y pérdida de que se produzca un resultado inadecuado que afecte al producto, proceso o proyecto software.

Por ejemplo, en un viaje en coche podríamos tener los siguientes riesgos:

Situación imprevista	Problema	Plan de acción
Congestión inusual de tráfico	Podría llegar tarde a una entrevista	Escuchar informes de tráfico p
Pinchazo de un neumático	Podría no llegar a la entrevista o llegar tarde	Asegurarse que el neumático d
Rotura del coche o accidente	Probablemente no llegar a la entrevista	No hay

Un riesgo está caracterizado por su probabilidad de ocurrencia y por sus consecuencias.

Tipos y fuentes de riesgo

Una clasificación de tipos de riesgos es:

- De proyecto.
  - Operativo.
  - Organizativo.
  - Contractual.
- De proceso.
  - De gestión.
  - Técnico.
- De producto.
  - Conocidos.
  - Predecibles.
  - Impredecibles.

Los tipos de riesgos según Hughes & Cotterell son:

- Debidos a dificultades de estimación.
- Debidos a suposiciones hechas en el proceso de planificación.
- Imprevisibles.

Algunas causas de los riesgos de proyecto son las restricciones de recursos, las interfaces externas, las relaciones con los proveedores, las políticas internas, los problemas de coordinación interna del equipo o del grupo o la financiación no adecuada. Las causas de los riesgos de proceso pueden ser un proceso de software no documentado, la falta de revisiones efectivas de colegas, ausencia de prevención de defectos, un proceso de diseño pobre, una gestión pobre de requisitos o una planificación ineficaz. En cuanto a los riesgos de producto, sus causas pueden ser la falta de experiencia en el dominio, un diseño complejo,

interfaces definidas deficientemente, sistemas de legado poco comprendidos o requisitos vagos o incompletos.

Los **factores de riesgo** pueden ser:

- debidos a la aplicación,
- debidos al personal,
- debidos al proyecto,
- métodos del proyecto,
- de hardware / software,
- debidos a cambios en el personal,
- debidos a los proveedores,
- debidos al entorno,
- factores ligados a la salud y a la seguridad.

Esta relación no implica que todos vayan a existir en todos los proyectos, pero sí que se deben considerar.

Paradigma de la gestión de riesgos

Se denomina **gestión de riesgos** a la práctica de valorar y controlar los riesgos que afectan a un producto, proceso o proyecto software. Su propósito es identificar problemas potenciales antes de que ocurran. En general, la idea es describir inicialmente los objetivos y después describir los riesgos en términos de incertidumbre, pérdidas y tiempo.

Algunos conceptos básicos de la gestión de riesgos son: objetivos, incertidumbre, pérdidas (oportunidad y coste de oportunidad), tiempo, elección, tomar decisiones inteligentes, resolver el riesgo, prevenir problemas.

La gestión del riesgo es necesaria porque el desarrollo de software tiene riesgos inherentes, el riesgo aumenta a medida que aumenta la complejidad del sistema, y el riesgo podría impedir alcanzar los objetivos si éste no se considera.

Entre las características de la gestión de riesgos se encuentran: considerar las oportunidades y amenazas, analizar la incertidumbre incluyendo ambigüedades, o preocuparse de las raíces de la incertidumbre en términos de las "seis Ws": quién (who), por qué (why), qué (what), de qué forma (which way), con qué (wherewithal), cuándo (when).

Por qué se deben gestionar los riesgos Todos los proyectos conllevan riesgos, y es muy probable que alguno ocurra. La gestión de riesgos es, por tanto, una inversión de futuro, pues muchas veces es más barato evitar un problema potencial que corregir uno que se ha producido. Si se corrigen los problemas a medida que se presentan, el flujo de problemas que se vayan produciendo podría mantenernos ocupados.

Es importante conocer dónde están los riesgos para enfocar sobre áreas esenciales de riesgos. La gestión intuitiva de los riesgos rara vez es suficiente en el caso de proyectos grandes, complejos. La gestión mejora la predictibilidad y control de los proyectos.

Gestionar los riesgos permite conseguir un conocimiento consistente de los riesgos en toda la organización. Además permite aprender de riesgos que se hayan producido.

Proceso de gestión de riesgos Este proceso es continuo, orientado hacia el futuro, y es una parte importante de la gestión de proyectos. A pesar de su importancia, en muchos casos no se realiza debido a la dificultad de medir su éxito, a la novedad de la disciplina, a la dificultad de la comprensión del concepto de riesgo, a conflictos con la cultura interna de ciertas empresas que desaconseja la aproximación analítica a la gestión de riesgos, a que no se le da la importancia que merece, o a procesos de proyecto caóticos.

Una buena gestión de riesgos es proactiva, integrada, sistemática y disciplinada (personal, proceso, infraestructura e implementación)

Principios de la gestión de riesgos del software

- Mantenimiento de una perspectiva global.
- Tener una visión previsora, pensando en los riesgos que puedan aparecer en el futuro.
- Alentar la comunicación abierta, admitiendo sugerencias de riesgos potenciales de cualquier usuario o miembro del equipo.
- Integración del proceso de desarrollo de software con la consideración permanente de los riesgos.
- Enfatizar un proceso continuo.
- Desarrollar una visión conjunta del producto implicando a los usuarios.
- Alentar el trabajo en equipo.

Paradigma de la gestión de riesgos de software en SEI 92

- Identificar: Buscar y localizar riesgos antes de que se presenten los problemas.
- Analizar: Transformar los datos de riesgo en información para la toma de decisiones. Evaluar el impacto, probabilidad y establecer el tiempo, clasificar y priorizar los riesgos.
- Planificar: Trasladar la información sobre riesgos en decisiones y acciones de disminución de su efecto, tanto en el presente como en el futuro, e implementar dichas acciones.

- Seguir: Monitorizar los indicadores de riesgo y las acciones para mitigar su efecto.
- Controlar: Corregir las desviaciones sobre los planes de disminución de su efecto.
- Comunicar: Proporcionar información y realimentación

En cualquier caso, debe existir la comunicación de todas estas funciones de gestión de riesgo. Todas estas actividades pueden ser concurrentes entre los diferentes riesgos.

Hughes & Cotterell plantean el siguiente marco de trabajo para riesgos:

- Identificación del riesgo. ¿Qué riesgos puede haber?
- Análisis y priorización de los riesgos. ¿Cuáles son los riesgos más serios?
- Planificación de los riesgos. ¿Qué haremos si se presentan?
- Monitorización del riesgo. ¿Cuál es el estado actual del riesgo?

El proceso de gestión de los riesgos La gestión de los riesgos es un ciclo que parte de una base de conocimientos de riesgos que se usa para **identificar** riesgos (lista de riesgos potenciales), después se **analizan** (lista priorizada), se **planifican** (evasión de riesgos y planes de contingencia), se **siguen** (valoración de los riesgos), se **resuelven** y se **aprende** del proceso, llevando el nuevo conocimiento a la base de conocimientos de riesgos.

La identificación de riesgos Algunas tareas relacionadas con la identificación de riesgos son: la realización de una valoración de los riesgos a través de entrevistas, identificación sistemática de riesgos con listas de comprobación o tormentas de ideas, definición de los atributos de los riesgos, documentación de los riesgos identificados, o comunicación de riesgos identificados.

Algunas pautas para la identificación de riesgos son las siguientes:

- Comenzar con una tormenta de ideas abierta, que permite aprender y utilizar una técnica efectiva.
- Realizar una tormenta de ideas focalizada por área, stakeholder, objetivo, área técnica, etc.
- Utilizar una lista de comprobación para garantizar una cobertura suficiente, utilizando los elementos de la lista como puntos de discusión. Las listas se pueden personalizar a través de la experiencia.

La cuantificación de un riesgo está relacionada con su probabilidad y su impacto.

Los riesgos se ordenan según lo críticos que sean en función de su cuantificación.

La cuantificación de los riesgos parece conveniente puesto que permite una evaluación más crítica de las propuestas de soluciones, alienta el diseño con la

percepción de riesgo, facilita la realimentación sobre riesgos que se han olvidado, permite la realimentación sobre el impacto de los riesgos que se han anticipado, facilita la reserva de recursos para tratar los riesgos, y nos permite determinar si un riesgo es aceptable.

El PMBOK define los siguientes puntos sobre la cuantificación de riesgos:

Entradas:

- Tolerancias del cliente al riesgo.
- Fuentes de riesgo.
- Eventos de riesgo potenciales.
- Estimaciones de coste.
- Estmiaciones de la duración de la actividad.

Herramientas y técnicas:

- Valor monetario esperado.
- Sumas estadísticas.
- Simulaciones usando por ejemplo PERT o CPM.
- Árboles de decisión.
- Juicio de expertos.

Salidas:

- Lista de amenazas a considerar o de oportunidades a seguir.
- Lista de amenazas a aceptar y de oportunidades a ignorar.

La documentación de los riesgos extraída de la identificación incluye relacionarlos con un proyecto, fecha, nombre, categoría, probabilidad, consecuencias, creador, fase y elemento de WBS.

Un método para identificar riesgos es crear una lista de comprobación de los elementos de riesgo:

- Tamaño del producto: riesgos asociados con el tamaño general del software a construir o modificar.
- Impacto en el negocio: riesgos asociados con las limitaciones de gestión o el mercado.
- Características del cliente: riesgos asociados con las características de los clientes y la forma de comunicación del desarrollador con el cliente.
- Definición del proceso: riesgos asociados con el grado de definición del proceso y su seguimiento por la organización.
- Entorno de desarrollo: riesgos asociados con la disponibilidad y calidad de las herramientas que se van a emplear en la construcción del producto.

- Tecnología a construir: riesgos asociados con la complejidad del sistema a construir y la tecnología que contiene el mismo.
- Tamaño y experiencia de la plantilla: riesgos asociados con la experiencia técnica y de proyectos de los ingenieros de software.

Dentro de la identificación de los riesgos se encuentra la tarea de la comunicación de estos. Hay que notificar a todas las partes afectadas, ya sean usuarios, responsables, equipo de gestión, márketing, ventas, soporte, finanzas, garantía de calidad, etc.

El análisis de los riesgos Las actividades que comprende el análisis de riesgos son:

- Agrupamiento: eliminar riesgos redundantes, combinar riesgos relacionados, enlazar riesgos dependientes.
- Determinación de los conductores de riesgos (risk drivers): factores subyacentes que afectan a la gravedad de las consecuencias. Pueden afectar a la estimación de la probabilidad, consecuencia y exposición al riesgo. Aumentan el conocimiento de cómo se pueden mitigar los efectos de los riesgos.
- Ordenación de la probabilidad, consecuencia, exposición, marco temporal.
- Determinación de las causas raíz (fuentes del riesgo): análisis anterior de las causas, identificando causas comunes.

Tras este análisis se puede mantener una lista Top-n de los n riesgos más importantes a tener en cuenta para observar la evolución periódica de éstos.

La documentación de los riesgos extraída del análisis aporta un enunciado que describe brevemente el riesgo, el contexto del mismo (cuándo, cómo, dónde, por qué...) y un análisis de su impacto en el proyecto.

La planificación de los riesgos Existen varias estrategias para resolver los riesgos:

- Evitación del riesgo: prevenir su ocurrencia reduciendo a cero su probabilidad.
- Protección del riesgo: reduce la probabilidad y/o consecuencias del riesgo antes de que ocurra.
- Reducción del riesgo: reduce la probabilidad y/o consecuencias del riesgo una vez ocurre.
- Investigación del riesgo: obtener más información para eliminar o reducir la incertidumbre sobre el riesgo.
- Reservar el riesgo: utilizar la planificación reservada previamente o la holgura del presupuesto.

- Transferencia del riesgo: reorganización que desplace el riesgo a otra parte.
- Aceptación del riesgo: si el coste de la evitación del riesgo puede ser mayor que el coste de asumirlo si se produce.

Actividades de la planificación de riesgos:

- Especificar escenarios: ¿cómo seríamos capaces de decir qué está ocurriendo?
- Definir umbrales cuantificados para los avisos iniciales: qué monitorizar, cuándo consideramos que el riesgo va a ocurrir.
- Desarrollar alternativas de resolución: formas de eliminar, mitigar o manejar el riesgo.
- Seleccionar la mejor aproximación a la resolución: ¿cuál es la que tiene mejor retorno de la inversión (ROI)?
- Especificar el plan de acción de riesgos, documentando las decisiones.

El seguimiento de los riesgos Las actividades del seguimiento de riesgos abarcan:

- Monitorización de escenarios de riesgo, vigilando la aparición de signos de ocurrencia de un escenario de riesgo.
- Comparar los indicadores con las condiciones de disparo.
- Notificar a los stakeholders y ejecutar el plan de acción.
- Recoger estadísticas y actualizar la base de datos de riesgos.

Por cada item de riesgo se deben preparar las respuestas apropiadas a preguntas como quién es el responsable de la acción, cuándo se debe realizar, qué medida hay que vigilar y cuál es el valor de disparo de la medida.

La planificación y el seguimiento de riesgos añaden a la documentación del riesgo el escenario que describe qué sucedería si el riesgo ocurre, el indicador del riesgo (métrica que hay que seguir), la condición de disparo (valor que indica que se ha producido el riesgo), el punto de comprobación del indicador, la estrategia de resolución y el plan de acción.

La resolución de riesgos Sus objetivos son, entre otros, asignar responsabilidad y autoridad al nivel de detalle más bajo posible, seguir un plan de acción perfectamente documentado, informar del esfuerzo dedicado a la resolución de riesgos, tomar acciones correctivas cuando sea necesario, estar preparado para adaptarse a circunstancias cambiantes, mejorar la comunicación dentro del equipo y controlar de forma sistemática los riesgos.

Las actividades abarcan:

- Reconocimiento con acuse de recibo de la notificación: permitir a los usuarios conocer que estás informado, indicar tiempo de respuesta, determinar la responsabilidad / pertenencia.
- Ejecutar el plan de acción: improvisar, adaptar y superar el riesgo usando el sentido común.
- Proporcionar actualizaciones continuas, permitiendo a los usuarios conocer el progreso realizado resolviendo el riesgo.
- Recoger estadísticas actualizando la base de datos de riesgos.

La resolución de riesgos aporta a la documentación la firma de todos los responsables: el ingeniero de software, de calidad, el gestor de proyecto y el responsable de márketing.

Capacidad para la gestión de riesgos (modelo CMM del SEI)

Según la capacidad para gestionar riesgos en un proyecto, se pueden dar 5 niveles:

1. Riesgos ignorados o solo seguidos de forma ad-hoc.
2. Los riesgos se suelen registrar, son seguidos y se manejan en el momento en que se descubren.
3. Los riesgos son cuantificados, analizados, planificados, seguidos y resueltos de forma sistemática.
4. Los análisis cuantificados se utilizan para determinar la resolución coste / beneficio del proyecto.
5. Se usan las estadísticas de riesgos para realizar mejoras organizativas o de proceso.

También se puede clasificar esta capacidad según los siguientes niveles:

- Invisible: No hay evidencia sobre la realización de actividades de gestión de riesgos. La gestión de riesgos es intuitiva y se incluye implícitamente en la gestión de proyectos.
- Ad hoc: Los gestores de proyectos realizan de forma ocasional actividades de gestión de riesgos por iniciativa propia.
- Sugerida: Existen plantillas para documentar los resultados de las actividades de gestión de riesgos, como una selección de gestión de riesgos en el plan de proyecto o una lista de riesgos en el informe de progreso del proyecto, pero no son requeridas.
- Requerida: Los resultados de las actividades de gestión de riesgos son requeridos y comprobados formalmente, se solicita un plan de gestión de iresgos y se obtienen, actualizan y analizan las listas de riesgos con frecuencia.

- Apoyada: Existe un proceso definido para realizar la gestión de riesgos en la organización, incluyendo métodos, herramientas, pautas e infraestructuras de apoyo.
- Mejorada: Hay un proceso sistemático para capturar la experiencia en gestión de riesgos y mejorar las prácticas de dicha gestión basadas en las experiencias anteriores.

Aprendiendo de los riesgos

En el post mortem hay que analizar cuáles fueron los riesgos no anticipados, la gravedad real de la consecuencia, si las estrategias de resolución funcionaron como se esperaba, qué riesgos se podían haber podido prevenir, transferir, protegido frente, reducir o manejar únicamente reservándole recursos extra.

A partir de ahí hay que sacar en claro qué medidas preventivas se pueden tomar en el futuro, qué problemas significativos de prestaciones para el vendedor / partner existen, y qué se puede compartir con otros equipos de proyecto.

Los 10 riesgos más importantes según Boehm (1991)

- Fallos de personal.
- Planificaciones de tiempo y presupuesto no realistas.
- Desarrollo de funciones de software incorrectas.
- Desarrollo de interfaces incorrectas.
- "Chapado en oro"
- Secuencia continuada de cambios en los requisitos.
- Fallos en las tareas realizadas externamente.
- Fallos en los componentes externos.
- Fallos en las prestaciones de tiempo real.
- Capacidades informáticas forzadas.

Tema 3: Proceso unificado (RUP)

Introducción

El **Proceso Unificado (RUP)** es un proceso de desarrollo de software aunque según sus desarrolladores es todo un marco de trabajo genérico que puede especializarse para gran cantidad de sistemas software, diferentes áreas de aplicación, tipos de organizaciones, etc.

Está basado en componentes, lo que indica que el sistema a construir estará formado por componentes software interconectados a través de interfaces. Usa UML como lenguaje de modelado.

Los tres elementos clave que resumen el proceso unificado son:

- es dirigido por casos de uso,
- está centrado en la arquitectura,
- es iterativo e incremental.

Los objetivos de RUP son: proporcionar un marco de trabajo para gestionar proyectos software; proporcionar guías prácticas para planificar, gestionar el personal, ejecutar, realizar y seguir proyectos; y proporcionar un marco para gestionar riesgos. No cubre otras cosas como el reclutamiento, formación, etc. de personal; la gestión de presupuestos; o la gestión de contratos con proveedores y usuarios.

Está enfocado hacia un proceso de desarrollo iterativo, gestionando los riesgos, planificando el proyecto iterativo a través del ciclo de vida y planificando cada iteración particular, y monitorizando el progreso del proyecto iterativo obteniendo métricas.

El proceso de desarrollo de software debe estar adaptado al entorno, a las personas y a los objetivos de las organizaciones que lo utilizan. Ningún proceso puede ser adecuado a todas las organizaciones y por tanto un proceso debe ser adaptable y configurable.

Algunos factores que se deben tener en cuenta para la adaptación del proceso son el dominio de la aplicación, las personas y factores organizativos, los factores relacionados con el ciclo de vida del producto, o los factores de proceso como el cumplimiento de estándares.

El RUP consiste en una estructura estática de personas, actividades, artefactos y flujos de trabajo, y una estructra dinámica de ciclos, fases, iteraciones, hitos y relaciones entre ellos.

Una **actividad** es una unidad de trabajo que puede ejecutar uno o varios individuos en un rol específico. Tiene un propósito claro expresado en términos de actualizar un artefacto. La granularidad de una actividad es generalmente de horas o pocos días. Ejemplos son planificar una iteración, encontrar casos de uso y actores, o revisar el diseño.

Un **artefacto** es una pieza de información producida, modificada y utilizada en un proceso, un producto tangible del proyecto. Se utiliza por los roles como entrada para la realización de sus actividades, y también es el resultado de las actividades que realizan. En el metamodelo, la clase “rol” tiene como métodos las actividades y como parámetros los artefactos.

Un recurso humano puede desempeñar uno o más roles, que le permiten ejecutar las actividades asociadas a tales roles.

Flujos de trabajo fundamentales

El ciclo de desarrollo del proceso unificado se divide en cuatro fases en las que los distintos flujos de trabajo se desarrollan con distinta intensidad. Cada fase está dividida en iteraciones. Estas cuatro fases son:

- Inicio (inception): define el ámbito del proyecto y desarrolla los casos de negocio. Define los objetivos.
- Elaboración: se elabora el plan de proyecto, se especifican las características y el marco de referencia de la arquitectura. Define la arquitectura.
- Construcción: se construye el proyecto. Define la funcionalidad inicial.
- Transición: se pasa el producto a los usuarios. Define la versión final del producto.

Al final de cada iteración se dispone de una nueva versión de los entregables.

Los flujos de trabajo de ingeniería o de proceso son:

- modelado de negocio,
- requisitos,
- análisis y diseño,
- implementación,
- prueba,
- lanzamiento.

Los flujos de trabajo de apoyo o accesorios son:

- gestión de configuración,
- gestión del proyecto,
- entorno.

Los casos de uso van progresando a través de las fases. Al terminar la fase de inicio, la mayoría del modelo de negocio ha sido terminado y la mayoría de los casos de uso han sido identificados. Muy pocos se describen, se analizan, y mucho menos se implementan. Al terminar la fase de elaboración el modelo de negocio está terminado, la inmensa mayoría de casos de usos han sido identificados y muchos de ellos han sido descritos. Algunos de ellos han sido analizados y apenas ninguno ha sido diseñado, implementado y probado. Al terminar la fase de construcción, todos los casos de usos tienen que haber sido identificados, descritos, analizados, diseñados, implementados y probados.

Beneficios del desarrollo iterativo:

- Se produce una mitigación, en las capas iniciales, de los riesgos más críticos (técnicos, requisitos, objetivos, facilidad de uso, etc.).

- El progreso es visible desde muy pronto.
- La realimentación temprana, la implicación del usuario y la adaptación conducen a un sistema refinado que satisface más estrechamente las necesidades reales de los "usuarios".
- La complejidad se gestiona, el equipo no se abruma por la "parálisis del análisis" o por etapas muy grandes y complejas.
- El aprendizaje que se va obteniendo en cada iteración se puede utilizar metódicamente para mejorar el proceso de desarrollo, iteración a iteración.

A medida que se avanza en el desarrollo de la ingeniería de software se debe definir un modelo de proceso para soportar una evolución de los planes, requisitos y arquitectura junto con puntos de sincronización bien definidos; una gestión de requisitos y medidas objetivas de progreso y calidad; y una evolución de las capacidades del sistema a través de demostraciones de funcionalidad incremental.

El ciclo de vida se divide en dos etapas. La etapa de ingeniería, con equipos más pequeños, es donde se realizan actividades menos previsibles de diseño y síntesis. La etapa de producción es donde se realizan las actividades más previsibles relacionadas con construcción, prueba y lanzamiento, en un equipo más grande.

Aspecto del ciclo de vida	Énfasis durante etapa de ingeniería	Énfasis durante etapa de producción
Reducción del riesgo	Planificación, factibilidad técnica	Coste
Productos	Línea base de la arquitectura	Líneas base de entregas de producto
Actividades	Análisis, diseño, planificación	Implementación, pruebas
Valoración	Demostración, inspección, análisis	Pruebas
Economía	Resolución de la *diseconomía* de escala	Explotación de la economía de escala
Gestión	Planificación	Operaciones

La transición entre ingeniería y producción es un evento crucial para los "stakeholders".

Los diez puntos fundamentales del RUP

- Desarrollar una visión
- Gestionar el plan
- Identificar y mitigar los riesgos
- Asignar y seguir características
- Examinar los casos de negocio
- Diseñar una arquitectura de componentes
- Construir de forma iterativa e incremental y probar el producto

- Verificar y evaluar resultados
- Gestionar y controlar los cambios
- Proporcionar apoyo a los usuarios

Modelos

Un modelo es una simplificación de la realidad con el objetivo de representar un sistema grande y complejo que sería difícil de comprender al considerarlo completamente. Un modelo no es la realidad sino una representación de ella. Un modelo único no puede cubrir todos los aspectos del desarrollo de software. Se necesitan múltiples modelos para cubrir todos ellos, coordinados de forma adecuada.

Los modelos son representaciones completas y consistentes del sistema a construir.

Partimos de un **modelo de casos** de uso que modela los casos de uso y sus relaciones con los actores. El modelo de casos de uso es especificado por un **modelo de análisis** que refina los casos de uso y realiza una visión inical del comportamiento del conjunto de objetos. Es realizado por un **modelo de diseño** que define la estructura estática del sistema como subsistemas, clases e interfaces y define los casos de uso como colaboraciones de subsistemas, clases e interfaces. Es implementado por un **modelo de implementación** que incluye componentes representados por su código fuente y la correspondencia entre clases y componentes. Es distribuido por un **modelo de despliegue** que define los nodos físicos y establece la correspondencia entre componentes y nodos. Es verificado por un **modelo de pruebas** que describe los casos de prueba que verifican los casos de uso.

Fase de inicio

Objetivos principales:

- Establecer el ámbito y los límites del proyecto, incluyendo conceptos operativos y criterios de aceptación.
- Localizar los casos de uso críticos y los escenarios principales que dirigirán los objetivos fundamentales del diseño.
- Comprobar al menos una arquitectura candidata frente a algunos de los escenarios principales.
- Estimar el coste completo y la planificación temporal del proyecto completo.
- Estimar los riesgos potenciales.

Actividades esenciales:

- Formular el ámbito del proyecto.

- Sintetizar la arquitectura.
- Planificar y preparar un caso de negocio.

Algunos criterios de evaluación:

- ¿Coinciden los "usuarios" con la definición del ámbito y las estimaciones de coste y tiempo?
- ¿Se han entendido totalmente los requisitos?
- ¿Son creíbles dichas estimaciones, así como las prioridades, los riesgos y el proceso de desarrollo?
- ¿La profundidad y amplitud del prototipo arquitectónico demuestran el criterio precedente?

De no cumplirse los criterios de evaluación habrá que replantear aquellos elementos que no se han conseguido satisfacer en los puntos anteriores.

En la fase de inicio se establece la oportunidad y alcance del proyecto, incluyendo conceptos operativos y criterios de aceptación. También se identifican todos los actores externos con los que se tratan y se define la interacción a un alto nivel de abstracción, identificando todos los casos de uso y localizando aquellos que son críticos y los escenarios principales que dirigirán los objetivos fundamentales del diseño.

La oportunidad de negocio incluye criterios de éxito, identificación de riesgos, estimación de recursos necesarios, y plan de las fases incluyendo hitos.

Los productos resultantes de la fase de inicio son:

- Un documento de visión general con los requisitos generales del proyecto, las características principales y las restricciones.
- Un modelo inicial de casos de uso con aproximadamente un 15% de ellos listos.
- Un glosario inicial con la terminología clave del dominio.
- Caso de negocio con un contexto, criterios de éxito y pronóstico financiero.
- Identificación inicial de riesgos.
- Plan de proyecto mostrando fases e iteraciones.
- Modelo de negocio si es necesario.
- Uno o más prototipos exploratorios para probar conceptos o la arquitectura candidata.

El hito que marca el fin de la fase de inicio debe suponer el acuerdo entre las partes interesadas sobre el alcance y la estimación de tiempo y coste, así como la comprensión de los requisitos plasmados en casos de uso.

Fase de elaboración

Objetivos:

- Analizar el dominio del problema.
- Establecer una arquitectura base sólida.
- Desarrollar un plan de proyecto ajustado para la fase de construcción.
- Eliminar los elementos de mayor riesgo para el desarrollo exitoso del proyecto.

Actividades esenciales:

- Elaborar la visión. Desarrollar los casos de uso a un nivel que dirija las decisiones de la arquitectura o de planificación.
- Elaborar el proceso y la infraestructura.
- Elaborar la arquitectura y los componentes seleccionados.

Productos:

- Modelo de casos de uso prácticamente terminado con descripciones detalladas.
- Otros requisitos no funcionales o no asociados a casos de uso.
- Descripción de la arquitectura del software.
- Un prototipo ejecutable de la arquitectura.
- Lista revisada de riesgos del caso de negocio.
- Plan de desarrollo para el resto del proyecto.
- Un manual de usuario preliminar.

La fase de elaboración es la más crítica del proceso. Al final de la misma, toda la ingeniería "dura" está hecha y se puede decidir si vale la pena seguir adelante. A partir de aquí, la arquitectura, planes de desarrollo y requisitos son estables. Ya hay menos riesgos y se puede planificar con menos incertidumbre el resto del proyecto. Se construye una arquitectura ejecutable que contemple los casos de uso críticos y los riesgos identificados.

El hito que marca el fin de la fase de elaboración debe cumplir los siguientes criterios de evaluación:

- ¿Es estable la visión del proyecto?
- ¿Es estable la arquitectura?
- ¿Las pruebas de ejecución demuestran que los riesgos han sido abordados y resueltos?
- ¿El plan de proyecto es algo realista?
- Las personas involucradas, ¿están de acuerdo con el plan?

Fase de construcción

Objetivos principales:

- Minimizar los costes de desarrollo optimizando los recursos e impidiendo la realización del trabajo dos veces.
- Conseguir una calidad adecuada.
- Conseguir versiones estables.

Actividades esenciales:

- Gestión y control de los recursos y optimización del proceso.
- Desarrollo de componentes completos y prueba de los mismos frente a criterios de evaluación.
- Valoración de las versiones de producto frente a los criterios de aceptación.

Algunos criterios de evaluación:

- ¿El producto es lo suficientemente estable y maduro como para ser entregado al usuario? La existencia de algunos defectos o cambios pendientes no son obstáculo para conseguir el propósito de la entrega en la fase siguiente.
- ¿Están los usuarios dispuestos para la transición?
- ¿Los gastos actuales de recursos son aceptables todavía frente a los planeados?

El hito que marca el fin de esta fase es la obtención de un producto beta y la decisión sobre si ponerlo en ejecución sin mayores riesgos.

Fase de transición

Objetivos principales:

- Conseguir que el usuario sea capaz de mantener el producto.
- Conseguir la aceptación por el usuario (stakeholder) de que lo entregado es completo y consistente con el criterio de evaluación fijado en la visión inicial del proyecto.
- Conseguir un producto final tan rápido y eficiente respecto al coste como práctico.

Actividades esenciales:

- Sincronización e integración de los incrementos de construcción concurrentes en líneas base de entrega consistente.

- Realizar una ingeniería específica de entrega (producción y empaquetamiento comercial, plan de formación del personal, etc.).
- Valoración de las líneas base de entrega frente a la visión global y los criterios de aceptación en el conjunto de requisitos.

Algunos criterios de evaluación:

- ¿El usuario está satisfecho?
- ¿La utilización de recursos actual es aceptable frente a la planificada?

Arquitectura

Visualizar, especificar, construir y documentar un sistema software precisa de una visión del sistema desde distintos puntos de vista.

Cada actor implicado en el proceso mira al sistema desde una perspectiva diferente a lo largo de la vida del proyecto. La arquitectura del sistema es uno de los productos más importantes que se pueden utilizar para gestionar estos puntos de vista y, por tanto, para controlar el desarrollo del sistema a través de su ciclo de vida.

Una arquitectura del sistema abarca el conjunto de decisiones importantes sobre:

- la organización del sistema software,
- la selección de los elementos estructurales y sus interfaces,
- su comportamiento, especificado como colaboración entre los distintos componentes,
- la composición de estos elementos (estructurales y de comportamiento) en subsistemas cada vez más grandes,
- el estilo arquitectónico que guía las organizaciones.

Hay cinco vistas con sus respectivos diagramas que las representan:

- Vista de usuario, con diagramas de casos de uso. Los casos de uso especifican la función, la arquitectura y la forma. Debe existir un balance entre casos de uso y arquitectura. Esta vista es central a las otras cuatro.
- Vista estructural o de diseño, con diagramas de clases y objetos. Representa clases, interfaces, colaboraciones.
- Vista de implementación, con diagramas de componentes. Representa componentes.
- Vista de comportamiento o de proceso, con diagramas de secuencia, de colaboración, de estados y de actividad. Representa clases activas.
- Vista del entorno o de lanzamiento, con diagramas de despliegue. Representa nodos.

Desarrollo iterativo e incremental

Un proceso basado en casos de uso implica ajustarse a las necesidades reales del usuario. Que esté centrado en la arquitectura quiere decir que el trabajo de desarrollo se centra en obtener un patrón que dirigirá la construcción del sistema en las primeras fases. La mejor forma de conseguir equilibrar casos de uso y arquitectura, que es como equilibrar forma y función, es el desarrollo iterativo e incremental. De esta forma se plantea una estrategia de desarrollo para productos software basada en pequeños pasos manejables: se planifica poco, se especifica, diseña e implementa un poco, se integra, prueba y ejecuta un poco en cada iteración.

Si en cada paso el resultado es satisfactorio, se avanza. Además esto nos permite ajustar los objetivos concretos en cada paso. Cada iteración tiene todo lo que debe tener un proyecto de software: planificación, desarrollo de una serie de flujos y preparación para la entrega. Pero una iteración no es algo totalmente independiente, sino que es una etapa dentro de un proyecto.

Cada iteración permite avanzar en la comprensión de cuestiones relativas al proyecto. Por ejemplo las primeras nos ayudan a comprender los riesgos, determinar la viabilidad, etc.

En resumen, no hay un proceso universal. El proceso unificado está diseñado para ser flexible y extensible, permitiendo una variedad de estrategias de ciclos de vida, seleccionando qué artefactos producir, definiendo actividades y trabajadores, y considerando los conceptos de los modelos.

Un white paper de Rational en 2011 considera que RUP facilita las siguientes buenas prácticas para desarrollo de software:

1. Desarrollo de software iterativo.
2. Gestión de los requisitos.
3. Uso de arquitecturas basadas en componentes.
4. Modelado visual de software.
5. Verificación de la calidad del software.
6. Control de cambios del software.

Esto genera una serie de beneficios:

- Reducción temprana de los errores.
- Gestión de los cambios.
- Mayor nivel de reutilización.
- Evaluación y mejoras periódicas.
- Mejor calidad final.

El proceso unificado equilibra más el peso de cada una de las actividades del proyecto que procesos convencionales como waterfall.

Proceso unificado. Plan de proceso. (PDF)

Introducción

No es difícil convencer a un gestor de proyectos de los beneficios de un proceso iterativo, pero a la hora de planificarlo surge la dificultad de que existen pocas técnicas para realizarlo. Surgen preguntas como cuántas iteraciones son adecuadas, cuál debe ser su duración, cómo determinar los contenidos y objetivos de cada iteración, cómo se puede seguir el progreso de una iteración, etc.

La planificación debe asignar responsabilidades y actividades a un equipo de personas a lo largo del tiempo. Se debe realizar el seguimiento del progreso relativo a la planificación y detectar problemas potenciales. También, el gestor de proyectos debe tratar con recursos inanimados como el equipamiento, presupuestos, etc pero estos aspectos no están afectados por la aproximación iterativa del proceso.

Antes de comenzar la fase de inicio, hay que tener en cuenta lo siguiente:

En organizaciones que desarrollan software para la venta general, suele haber mucha información porque ventas y dirección se han encargado de plantear la idea y puede que ya existan estudios exploratorios, así que parte de las tareas de inicio ya se han realizado.

En organizaciones que desarrollan software para la propia organización, puede que no se conozca muy bien lo que se pretende o que sí que estén bien desarrollados los requisitos.

En organizaciones que desarrollan software para un cliente que lo solicita, habitualmente la petición ya contiene muchos detalles.

También hay que considerar si se trata de una nueva versión de un producto existente o estamos hablando de un producto totalmente nuevo.

En general parece imposible realizar una planificación detallada con un nivel de granularidad de días. Para que los planes sean realistas, se debe comprender perfectamente lo que se está construyendo, tener unos requisitos y una arquitectura estables, y se debe haber construido antes algún sistema parecido.

Planificación de procesos iterativos

Para planificar procesos iterativos, Krutchen recomienda basar el desarrollo en dos tipos de planes: uno a grandes rasgos denominado plan de fases, y varios planes más detallados denominados planes de iteración.

El **plan de fases** es único por cada proyecto y contiene las fechas de los hitos principales, como el final de cada fase y del proyecto; el perfil de los recursos humanos necesarios a lo largo del tiepo; y las fechas de los hitos secundarios, que corresponden con el fin de cada iteración y su objetivo principal, si se conocen.

El plan de fases se genera al principio de la fase de inicio y se actualiza cuando sea necesario, y su tamaño es de una o dos páginas para definir el alcance y restricciones del proyecto.

Para elaborar el plan de fase, se parte del esfuerzo que supone el proyecto y la fecha en la que tiene que estar finalizado, y se planifica hacia atrás.

El plan se ve influenciado por:

- El tamaño del esfuerzo de desarrollo de software: formalidad, estádndares, rigidez del proceso.
- El grado de novedad: si es el primer proyecto de este tipo, su ciclo de evolución y mantenimiento.
- Tipo de aplicación: si es misión crítica, si requiere ciertas prestaciones, si tiene restricciones de memoria.
- El proceso de desarrollo actual: madurez del proceso, experiencia de los gestores y desarrolladores.
- Factores organizativos: actitud del equipo ante los cambios, entusiasmo hacia el proyecto.
- Complejidad técnica y de gestión: tamaño del proyecto.

Habrá que ajustar la duración de cada una de las fases según los detalles concretos de cada proyecto y cada equipo.

El **plan de una iteración** es más detallado y hay uno por cada iteración, aunque puede haber dos activos en un proyecto a la vez si se tiene el plan de itreación actual, y el de la iteración siguiente que se comienza a generar al final de la iteración actual y está disponible en cuanto se termina. Se genera con técnicas y herramientas de planificación habituales, contiene fechas importantes y se puede considerar como una ventana que se desplaza por el plan de fases, detallando la imagen del tramo que recorre. Los pasos a seguir para elaborar el plan de iteración que asigna roles a las actividades e individuos a los roles son:

1. Definir criterios objetivos para saber si ha tenido éxito al finalizar. Se utilizarán estos criterios al final de la iteración, en un proceso de valoración, para decidir si la iteración ha finalizado con éxito o no.
2. Identificar los artefactos concretos y medibles que habrá que desarrollar y las actividades necesarias para conseguirlos.
3. Partiendo de una división de trabajo de iteración típica, ajustarla a lo que hay que hacer. Comenzar con una estructura típica de descomposición del trabajo (WBS) y adecuarla para tener en cuenta las actividades que se van a realizar.
4. Utilizar estimaciones para asignar duración y esfuerzo a cada actividad, ateniéndose al presupuesto.

Hay que plantear una iteración como un microproyecto concreto. Se recomienda una duración de dos a seis semanas. La rapidez de la iteración depende del

tamaño de la organización y la jerarquía de la misma, pues cambiará el tiempo necesario para planificarse.

Una vez conocidos en cada caso los escenarios o casos de uso que deben ser considerados, pensando en la planificación hay que considerar qué artefactos serán afectados: qué clases deben ser revisadas, qué subsistemas son afectados o creados, qué interfaces necesitan ser modificadas o qué documentos deben ser actualizados. Después hay que identificar qué actividades están implicadas y colocarlas en el plan de proyecto. Algunas actividades se harán una vez por iteración, como elaborar el plan, mientras otras se hacen una vez por clase, caso de uso o subsistema. Hay que establecer las dependencias entre las actividades respectivas, y estimar el esfuerzo necesario para llevarlas a cabo.

Fases

En la **fase de inicio** no hay una iteración real puesto que no se produce software. A veces es conveniente realizar una iteración para crear un prototipo no funcional para validar la idea o construir un prototipo para disminuir riesgos con tecnologías nuevas, o para verificar si se pueden alcanzar requisitos no funcionales. Por tanto, 0 o 1 iteraciones.

En esta fase puede que no se tenga suficiente información para poder planificarla. En ese caso hay que reunir y organizar toda la información relevante posible, reunir gente que sepa utilizar dicha información, y describir qué es lo que falta siendo conscientes de la fase en la que estamos.

Algunas fuentes de información previa son: si es para un software de venta general, pueden existir trabajos de responsables de márketing que permitan obtener información del sistema; si es para la propia empresa, se parte de un estudio global del tipo de planificación estratégica que aporta modelos de negocio; y si es para un cliente, se parte del documento de petición inicial.

Los artefactos de la fase de inicio son:

- Documento de visión del sistema: visión general de los requisitos principales, de las características fundamentales y de las restricciones más importantes.
- Esbozo de los modelos que representan la primera versión de casos de uso, modelos de análisis y de diseño. Primera versión de requisitos adicionales.
- Primer esquema de la arquitectura candidata.
- En algunos casos, prototipo exploratorio.
- Lista de riesgos y casos de uso.
- Esbozo de un plan para el proyecto total, incluyendo un plan general de fases.
- Primer borrador del análisis de negocio.

En la **fase de elaboración**, es necesaria al menos una iteración, pero si se

realizan más (hasta 3) dependerá de si no se dispone de una arquitectura de entrada y se deben ajustar muchos factores nuevos.

Para definir los objetivos de una iteración en esta etapa hay que tener en cuenta:

- Riesgo. Cuanto más pronto se pueda eliminar un riesgo o disponer de un plan para mitigar su efecto, mejor.
- Recubrimiento. Hay que determinar si la arquitectura propuesta cubre todos los aspectos del software a desarrollar.
- Criticidad. Aunque los riesgos son muy importantes, no hay que perder de vista el objetivo principal del sistema a desarrollar. Es importante garantizar que se cubren todas las funciones o servicios críticos que se exigen al sistema, aunque a veces esto plantee compromisos con los aspectos de riesgo.

Productos clave:

- Preferiblemente, modelo completo de negocio que describe el contexto del sistema.
- Nueva versión de todos los modelos: casos de uso, análisis, diseño, despliegue e implementación.
- Línea base de la arquitectura y descripción de la misma.
- Lista de riesgos actualizada.
- Plan de proyecto para las fases de construcción y, a veces, la de transición.
- Versión preliminar del manual de usuario (opcional).
- Análisis de negocio completo, incluida la apuesta económica.

La **fase de construcción** también requiere al menos una pero son recomendables dos iteraciones. Dependiendo del proyecto, incluso tres.

Productos clave:

- Plan de proyecto para la fase de transición.
- Sistema software ejecutable.
- Todos los artefactos, incluyendo modelos del sistema.
- Descripción de la arquitectura modificada y actualizada mínimamente.
- Versión preliminar del manual de usuario, suficientemente destallado para que sirva a los usuarios de la versión beta.
- Análisis de negocio que refleja la situación al final de la fase.

La **fase de transición** requiere una, pero los defectos que se encuentran suelen obligar a realizar al menos otra.

Productos clave:

- Sistema software ejecutable, incluyendo el software de instalación.
- Documentos legales, como contratos, licencias, renuncias y garantías.
- Descripción completa y actualizada de la arquitectura.
- Versión completa y corregida de la línea base de la versión del producto, que incluya todos los modelos del sistema.
- Manuales y material de formación del usuario final, del operador y del administrador del sistema.
- Referencias, incluyendo las de la web, para ayuda del cliente sobre dónde encontrar más información, FAQ, etc.

Tipos de ciclo de vida

Existen varios patrones de iteración. Se puede dar un **ciclo de vida incremental** en el que al principio se definen los requisitos del sistema y después se van añadiendo las funcionalidades planificadas iteración tras iteración. En el ciclo de vida incremental suele haber una única iteración de elaboración para definir los requisitos y establecer la arquitectura, y varias de construcción.

También existe el **ciclo de vida evolutivo** para los casos en que los requisitos no están totalmente definidos y en ese caso los requisitos se van refinando a medida que avanzan las iteraciones. En el ciclo de vida evolutivo, se dan varias iteraciones de elaboración en la que se va definiendo el proyecto y una de construcción en la que se realizan los casos de uso.

Otro es el **ciclo de vida de entrega incremental** en los que, en condiciones en las que el usuario da valor a las entregas incrementales de funcionalidad como en situaciones en las que urge salir al mercado, permite generar un producto funcional pronto. La fase de transición comienza muy pronto y tiene más iteraciones. Este tipo de ciclo de vida requiere una arquitectura muy estable, así que es complicado de conseguir para nuevos tipos de sistemas.

3.3 SCRUM (PDF)

Introducción

Aunque el esquema de desarrollo de software tradicional ha sido útil para proyectos grandes y estables, no se ajusta a nuevos proyectos que pueden ser más pequeños o desarrollarse en entornos más cambiantes. Para cubrir esas necesidades, surgen las metodologías ágiles.

Este movimiento surge del manifiesto ágil.

Hay muchos de los riesgos más importantes y habituales a los que un esquema de desarrollo tradicional no se ajusta bien por ser demasiado inflexible. Se ha demostrado que los proyectos con una planificación ágil tienen más posibilidades de tener éxito.

Principios del manifiesto ágil:

- La prioridad es satisfacer al cliente a través de la entrega temprana y continua de software con valor.
- Se aceptan requisitos cambiantes como ventaja competitiva.
- Se entregan versiones de software funcional con frecuencias cortas.
- Las personas del negocio y los desarrolladores deben trabajar juntos de forma cotidiana en el proyecto.
- Los proyectos se construyen en torno a individuos motivados, dándoles la oportunidad y respaldo que necesitan y procurándoles confianza para sus tareas.
- La forma más eficiente de comunicación en el equipo es cara a cara.
- La principal medida del progreso es el software que funciona.
- Los procesos ágiles promueven el desarrollo sostenido. Todos los stakeholders deben mantener un ritmo constante.
- Hay que atender continuamente a la excelencia técnica.
- Hay que favorecer la simplicidad para minimizar el trabajo en vano.
- Las mejores arquitecturas, requisitos y diseños emergen de equipos autoorganizados.
- El equipo tiene que reflexionar con frecuencia sobre cómo mejorar su eficiencia y ajustarse en consecuencia.

Ideas sobre XP

Con XP, los programadores buscan comunicación entre miembros del equipo (esto incluye al usuario), mantener el proceso tan simple como sea posible, buscar la realimentación del cliente en cada iteración, y coraje.

Es útil usar XP cuando los requisitos se desconocen o son inciertos, el riesgo es elevado, el equipo es pequeño, el cliente puede ser implicado en el desarrollo, y hay capacidad de hacer pruebas. El equipo de programación tiene que estar muy preparado, especialmente en diseño, y tiene que ser disciplinado, pues no es una metodología fácil de seguir.

Reglas de XP:

- Se escriben las historias de usuario.
- La planificación de releases crea la planificación temporal.
- Se hacen frecuentes releases pequeñas.
- Se mide la velocidad del proyecto.
- Se divide el proyecto en iteraciones.
- Cada iteración comienza con el plan de dicha iteración.
- Los roles se van cambiando dentro del proyecto.
- Cada día comienza con un standup meeting.

- El XP debe corregirse cuando falle.

Al codificar en XP:

- el usuario siempre está disponible,
- el código se escribe de acuerdo a estándares,
- primero se codifica la prueba unitaria,
- toda la producción de código se hace en parejas,
- solo un par de personas integra el código en cada momento,
- se integra con frecuencia,
- el código es compartido entre todos,
- se deja la optimización hasta el final,
- no se hacen horas extra.

Problemas que puede haber en XP:

- Nadie prueba realmente la calidad del diseño.
- Nunca hay realmente una arquitectura, sino que va emergiendo en el tiempo.
- No está claro cómo reutilizar un software desarrollado bajo XP, XP tiende a mitigar la reutilización.
- La programación en parejas requiere programadores muy experimentados y disciplinados.
- El hecho de que el usuario sea necesario para desarrollar el proyecto en todo momento es peligroso.

SCRUM

Es un modelo para el desarrollo de todo tipo de productos tecnológicos. Tiene una filosofía ágil que flexibiliza el proceso, evita la secuencialización y favorece una estrecha colaboración dentro del equipo y con los clientes. La secuencialización se evita permitiendo que las iteraciones se solapen. Scrum considera solo, de los tres factores clásicos de produccion, a los procesos y las personas, dejando fuera de su consideración la tecnología.

En la actualidad los ciclos de producción se han acelerado y los productos se vuelven obsoletos y evolucionan rápidamente.

Los proyectos más adecuados para SCRUM tienen una duración media, una importancia crítica, un buen respaldo económico y un tamaño relativamente pequeño.

Los principios del scrum son:

- Colaboración con el cliente.

- Respuesta al cambio.
- Desarrollo incremental, con entregas funcionales frecuentes.
- Autogestión, autoorganización y compromiso.
- Simplicidad.
- Supresión de artefactos innecesarios en la gestión del proyecto.

El desarrollo ágil favorece equipos multidisciplinares frente a especialización, solapamiento de iteraciones frente a división en fases, seguir una visión del producto en lugar de seguir unos requisitos detallados, y la adaptación a los cambios en lugar de seguir un plan al pie de la letra.

Una empresa que desarrolla productos con modelos de gestión ágil cuenta con una incertidumbre hacia el entorno, equipos autoorganizados, un control sutil, y una difusión y transferencia de conocimiento interno. En definitiva, la gestión ágil de proyectos no se sitúa sobre el concepto de anticipación (requisitos, diseño, planificación y seguimiento), sino sobre el de adaptación (visión, exploración y adaptación).

Los roles de la gestión ágil, adaptados para SCRUM, son:

- Equipo de desarrollo: grupo de personas que trabaja en la creación de un producto, como diseñadores, programadores, probadores, etc.
- Product owner: persona implicada en establecer un puente entre el usuario final, los responsables del negocio y el equipo de desarrollo. Trabaja día a día para clarificar los requisitos. A veces se conoce como customer representative.
- Scrum master: responsable de apoyar al equipo de desarrollo, eliminar las barreras organizativas y mantener la consistencia del proyecto ágil. También se denomina project facilitator.
- Stakeholders: todo aquél con interés en el proyecto. No son product owners pero pueden proporcionar entradas y estar afectados por las salidas del proyecto.
- Agile mentor: alguien con experiencia implementando proyectos ágiles que puede compartir su experiencia con el equipo.

Los seis artefactos más habituales de la gestión ágil, adaptados para SCRUM, son:

- Establecimiento de la visión del producto: resumen rápido para comunicar cómo está enmarcado el producto en la empresa o en la organización, que articula los objetivos del producto.
- Product backlog: lista total de los requisitos que están en el alcance del proyecto ordenado por el product owner según la prioridad. En el momento que se tenga el primer requisito, se debe tener un backlog del producto. El product backlog se reprioriza al comienzo de cada sprint.

- Product roadmap: vista de alto nivel de los requisitos del producto con un marco temporal poco preciso de cuándo se desarrollarán los requisitos.
- Plan de release: un calendario de alto nivel para el lanzamiento de versiones de software funcionando.
- Sprint backlog: objetivo, historias de usuario y tareas asociadas con el sprint actual. Cada persona elige sus tareas (el trabajo no se asigna), se estima diariamente el trabajo restante. Cualquier miembro del equipo puede añadir, borrar o cambiar el sprint backlog.
- Incremento: funcionalidad del producto en funcionamiento al final de cada sprint.

Los siete eventos de la gestión ágil, adaptados para SCRUM, son:

- Planificación del proyecto: la planificación inicial, que incluye una visión del producto y un product roadmap. Se puede realizar en menos de un día.
- Planificación de release: planificación del conjunto de características que tendrá la próxima release del producto. Solo se planifica una release cada vez.
- Sprint: ciclo de desarrollo corto para crear funcionalidad de producto vendible. A veces se llaman iteraciones. Tienen una duración de entre una y cuatro semanas, con duración constante a lo largo del proyecto.
- Planificación de sprint: reunión al comienzo de cada sprint donde se fijan los objetivos del mismo. Se identifican los requisitos y las tareas a realizar para completar cada requisito. Se crea el sprint backlog a partir de tareas del product backlog que se estiman, entre todos.
- Daily scrum: reunión de unos 15 minutos a tener cada día de un sprint donde se indica lo realizado el día anterior y lo que se va a realizar en el actual. Todo el mundo está invitado pero solo los miembros del equipo, scrum master y product owner pueden hablar. No sirve para resolver problemas, pero ayuda a evitar otras reuniones innecesarias de coordinación. Todos deben responder a qué hicieron ayer, qué harán hoy y si hay obstáculos en su camino.
- Revisión de sprint: reunión al final de cada sprint iniciada por el product owner donde se analiza la funcionalidad del producto conseguida durante el sprint. Normalmente es una demo de las nuevas características o la arquitectura subyacente. Es informal, se prepara en dos horas y no hay diapositivas. Participa todo el equipo y se invita a todo el mundo.
- Sprint retrospective: reunión al final de cada sprint, donde el equipo analiza qué fue bien, qué debe cambiar y cómo realizar los cambios. Dura unos 15 o 30 minutos. Todo el equipo participa, discutiendo qué le gustaría comenzar a hacer, dejar de hacer o seguir haciendo.

Escalabilidad

Normalmente los equipos son de unas 7 ± 2 personas. Es escalable haciendo equipos de equipos. Hay que tener en cuenta el tipo de apliación, el tamaño y la dispersión del equipo, y la duración del proyecto. Aunque se recomienda para equipos pequeños, SCRUM se ha usado en múltiples proyectos de más de 500 personas.

Objetivos de la gestión ágil

Dar el mayor valor posible al producto cuando éste se basa en la innovación y la flexibilidad. Reducir el tiempo de salida al mercado. Ser ágiles, produciendo partes completas del producto en periodos breves de tiempo. Ser flexibles, pudiendo adaptar la forma y el desarrollo a las características del proyecto. Obtener resultados fiables.

Para lograr la autoorganización, los equipos deben reuinr tres características:

- Deben tener autonomía para elegir la estrategia de la solución. La empresa en este caso actúa como un inversor de capital riesgo.
- Deben tener capacidad de autosuperación, desarrollando soluciones que evalúan, analizan y mejoran.
- Deben tener capacidad de autoenriquecimiento, favorecida pro la multidisciplinaridad y la aportación de soluciones valiosas complementarias.

Algunas prácticas SCRUM

- Los clientes son parte del equipo de desarrollo.
- SCRUM tiene frecuentes entregables intermedios funcionales, lo que permite al cliente trabajar con el software antes y cambiar los requisitos de acuerdo con las necesidades.
- Se desarrollan planes de riesgos y mitigación frecuentes por parte del equipo de desarrollo. La mitigación de riesgos, la monitorización y la gestión de riesgos se lleva a cabo en todas las etapas y con compromiso.
- Transparencia en la planficación y desarrollo de módulos, que permite saber a cada uno quién es responsable de qué y cuándo.
- Frecuentes reuniones de las personas involucradas en el negocio para monitorizar el progreso.
- Debe haber un mecanismo avanzado de advertencias.
- Los problemas no se han de barrer bajo la alfombra, puesto que nadie es penalizado por reconocer o describir un problema imprevisto.

Relaciones entre métodos ágiles y proceso unificado

- Participación activa de los stakeholders.
- Aplicar estándares de modelado.
- Aplicar patrones cuidadosamente.
- Aplicar el artefacto correcto.
- Propiedad compartida.
- Considerar la realización de pruebas.
- Crear varios modelos en paralelo.
- Crear contenido sencillo.
- Dibujar modelos de forma sencilla.
- Desechar modelos temporales.
- Exponer los modelos públicamente.
- Formalizar los modelos de contrato.
- Iterar a otro artefacto.
- Modelar en pequeños incrementos.
- Modelar para comunicar.
- Modelar para comprender.
- Probarlo con código.
- Reutilizar recursos existentes.
- Actualizar solo cuando sea preciso.
- Utilizar herramientas sencillas.

Condiciones para pasar de grupo de emprendedores a empresa establecida

- Repetitividad de resultados: al hacer que la calidad del resultado sea consecuencia del proceso, desarrollar nuevas aplicaciones aplicando el mismo proceso garantiza la homgeneidad de los resultados.
- Escalabilidad: también consecuenica de la repetitividad, independientemente del tamaño u objetivos del equipo.
- Mejora continua: al aplicar metaprocesos que miden y analizan los resultados se obtiene la información necesaria para aplicar mejoras de forma contiuna a la eficiencia y calidad de los procesos base y, por tanto, de los resultados.
- Know-how propio: el modelo de procesos termina conteniendo un activo valioso de la organización.

Gestión de configuraciones de software

Conceptos generales

El crecimiento de los sistemas implica un descenso de la productividad, aumento de la complejidad, incremento del número de participantes, artefactos independientes entre sí, peor comunicación, desorden, y cambios consSegtantes. Por ello, es preciso acompañar los desarrollos con algún mecanismo que permita frenar o atenuar los efectos de todo eso.

La gestión de configuraciones de software es una disciplina cuya misión es controlar la evolución de un sistema.

Según [BABI86]:

> El arte de identificar, coordinar y controlar las modificaciones del software construido por un equipo de programación.
>
> En una frase, el arte de coordinar un proyecto para minimizar la confusión.

Según [WHIT91]:

> Colección de técnicas que sirven para coordinar y controlar la construcción de software.

Según [BERS84]:

> Proceso software para controlar los cambios sistemáticamente y mantener la integridad a lo largo del ciclo de vida.

Es la identificación unívoca, almacenamiento controlado, control del cambio e información sobre el estado de los elementos seleccionados de cualquier tipo de artefacto producido durante el ciclo de vida.

Su propósito es mantener la integridad de los productos cuando éstos evolucionan a través de sus ciclos de vida de desarrollo y producción.

Según el glosario de la IEEE:

> [La GCS sirve para] identificar y documentar las características funcionales y físicas de un elemento de configuración, controlar los cambios en dichas características, registrar e informar del proceso de cambios y el estado de la implementación, y verificar la conformidad con los requisitos especificados.

La GCS no es únicamente control de versiones, no se aplica solo a la gestión de código fuente y no es únicamente para la fase de desarrollo. Seleccionar y utilizar herramientas es importante, pero el diseño y la gestión del proceso de GCS son cruciales para el éxito de los proyectos.

La GCS es una actividad de garantía de calidad del softwre que se aplica en todas las fases del proyecto de ingeniería del sofware, incluída la planificación.

La GCS debería poder responder a preguntas como ¿quién hace los cambios?, ¿qué cambios se realizan?, ¿cuándo se hacen los cambios? o ¿por qué se hacen los cambios?. Esto se implementa normalmente con solicitudes de integración que llevan la respuesta a estas preguntas asociada a cada conjunto de cambios.

En el proceso de gestión de configuraciones hay que especificar cómo se introducen elementos de gestión de configuraciones en el proyecto. Desde los procesos de desarrollo se envían solicitudes al proceso de gestión de configuraciones. Demasiadas solicitudes de cambio pueden saturar este proceso.

Un item de configuración primero es identificado, después se pasa a producción (se crea) y se almacena asociado a una versión. Cuando aparecen cambios se genera una solicitud de cambio controlada que genera una nueva implementación asociada a una nueva versión.

Este proceso debe poder generar informes sobre lo que se tardan en aceptar solicitudes de cambio, o en qué medida colabora cada uno de los miembros del equipo.

Algunos escenarios posibles que permite la GCS son:

- Ver versiones concretas de un fichero y el historial de cambios.
- Volver un fichero a una versión anterior.
- Conocer qué cambios pertenecen a una release.
- Trabajo coordinado de personas en remoto.
- Rastrear los cambios: cuáles han sido, quién los ha realizado y cuándo.
- Conseguir informes sobre el progreso del proyecto.

Elementos de configuración

Un EC es un entregable (o artefacto según RUP) aprobado y aceptado para el que los cambios deben realizarse mediante un procedimiento formal. Algunos ejemplos son el plan de gestión, documento de requisitos, especificación de diseño, código fuente y ejecutable, etc.

Un EC se debe poder definir y controlar de forma separada.

A partir de esta definición, se puede ver la configuración del software como el conjunto de toda la información o productos que se utilizan o producen en el desarrollo de software, es decir, el conjunto de todos los EC del proyecto.

Versión, variante y revisión

Una versión es un elemento de configuración en un punto de su desarrollo. Incluye una revisión y variante.

Una revisión es un elemento de configuración enlazado con otro mediante la relación "revisión de" y ordenado en el tiempo.

Una variante es una versión funcionalmente equivalente pero diseñada para distintas características de hardware o software (arquitecturas, etc).

Una rama es una secuencia de versiones en una línea temporal.

Una gestión de configuraciones elemental debería ser capaz de gestionar las versiones (posibilidad de recuperar versiones antiguas) y de gestionar las releases. Una release es una lista de versiones concretas de cada archivo.

Las revisiones de un componente pueden ser lineales, pero suelen ir hacia una estructura arbórea. En el tronco quedan las variantes principales. Del tronco salen distintas ramas. Los cambios de una versión a la anterior se denominan "deltas". Los deltas pueden ser directos si van de una versión a la siguiente, o inversos si van de una versión a la anterior.

Línea base

Punto de referencia o configuración de referencia en el proceso de desarrollo del software. Es un conjunto de elementos revisados formalmente y fijados en un momento determinado del ciclo de vida del software. Su objetivo es permitir cambios rápidos e informales sobre los EC antes de que formen parte de la línea base y cambios formales una vez formen parte de ésta.

Es una colección de versiones de elementos que han sido revisadas formalmente y conforman una versión de configuración. Marca hitos y sirve como una base para otros desarrollos. Solo se puede cambiar mediante un proceso formalizado de gestión de cambio. Una línea base más un conjunto de cambios crean nuevas líneas base. Una línea base es un entorno aislado en el que un desarrollador puede trabajar (editar, cambiar, compilar y probar) sin interferir con otros desarrolladores.

Los elementos que conforman la línea base solo se pueden cambiar mediante procedimientos formales de control. Algunos ejemplos son:

- Funcional: requisitos del sistema revisados.
- Reservada: especificación de requisitos de software y de interfaz aprobados.
- De desarrollo: configuración de evolución del software en instantes determinados del ciclo de vida.
- Producto: el producto completo entregado para la integración del sistema.

Establecimiento de líneas base Las líneas base serán hitos o marcas en el proceso de desarrollo. Las principales están marcadas por el ciclo de vida y metodología seleccionada. Hay dos objetivos fundamentales: identificar los productos de las fases del ciclo de vida y garantizar que las fases se van completando.

Las líneas base principales son las de función o definición, de distribución o asignación de funciones, de diseño preliminar, de diseño, de producto o de operación.

Workspace

Un **workspace** puede ser un directorio local bajo control de versiones o puede estar en un servidor. Las operaciones habituales sobre un workspace son importar (incluir recursos en el control de versiones en el almacén), actualizar (conseguir la última versión en la rama por defecto), checkout (volcar una versión del almacén en el espacio de trabajo) y checkin (consolidar los cambios en el almacén).

Almacén o repositorio

Un **almacén o repositorio** almacena versiones. Es habitual centralizarlo en un sistema, denominando repositorio al almacén central. El repositorio permite ahorrar espacio evitando guardar duplicados comunes a varias versiones o configuraciones, y facilita la posibilidad de almcenar la evolución del sistema. No es lo mismo repositorio que línea base.

Base de datos de configuraciones

La **base de datos de configuraciones** se utiliza para registrar cualquier información relevante relacionada con las configuraciones de software, no solo las versiones. Facilita el conocimiento sobre la evolución del cambio y el impacto del mismo. Debe responder preguntas como qué clientes disponen de una versión concreta del sistema software, infraestructura de hardware y software necesaria para una versión, cuántas versiones se han creado y cuándo, qué versiones podrían verse afectadas ante un cambio en un componente, cuántas peticiones de cambio se han hecho sobre una determinada versión, o cuántos fallos registrados existen sobre una versión. Esta base de datos se puede crear como un sistema separado, o bien integrada con el sistema de gestión de versiones de software y el de control sobre los documentos formales del proyecto.

Bibliotecas de software

Las bibliotecas de software son colecciones de software y/o documentación relacionada cuyo objetivo es ayudar en el desarrollo y mejorar la visibilidad

del sistema. Algunos tipos de bibliotecas son de trabajo, de intergración, de producción, soporte o proyecto, maestra, de software o repositorio, o de backup.

Modelos de control de versiones

El problema del trabajo colaborativo es que dos personas pueden leer una versión, modificarla de forma separada, y que en el repositorio la última persona sobreescriba la versión de la primera sin darse cuenta.

Para solucionarlo hay varios modelos. En el pesimista, se bloquea un fichero concreto para una persona hasta que termina de editarlo. En el optimista, se detecta si se intenta actualizar el repositorio desde una versión antigua, se hace una mezcla de las diferencias y se genera una nueva versión para evitar sobreescribir.

Control de cambios

El control de cambios es la actividad de gestión de configuraciones software que perite gestionar adecuadamente los cambios. Es la tarea de GCS más importante. Normalmente se combinan procedimientos humanos con el uso de herramientas automáticas.

Hay varios niveles de control de cambios, que puede ser informal si el elemento de configuración no forma parte de la línea base y no afecta a ningún otro EC, semiformal si el EC forma pate de una línea base o no pero afecta a otros que sí forman parte de una línea base, o formal si el EC forma parte de la bibliotrca maestra.

Hay que crear un comité de control de cambios formado por los encargados de tomar decisiones acerca del estado y las prioridades de cambio formado por todos los miembros del proyecto, el jefe de proyecto y el bibliotecario.

Hay que describir el mecanismo para solicitar el cambio. Normalmente se hace con un formulario de solicitud lo más sencillo posible que contemple por qué se solicita el cambio, quién lo solicita, qué hay que cambiar, una estimación de daños, los elementos afectados y la aprobación del cambio.

El comité de control de cambios decide si se acepta o no el cambio en función del valor del cambio para la organización, el retorno de la inversión, el tamaño del cambio, la complejidad, los recursos disponibles para efectuar el cambio y otros factores.

GCS como actividad de garantía de calidad

La GCS es una actividad de garantía de calidad que se aplica en todas las fases del proceso de ingeniería de software.

Algunos beneficios son:

- Mejora la protección del producto.
- Mejora la visibilidad del producto.
- Mejora el control del producto.
- Mejora la comunicación del equipo.
- Mejora la confianza del usuario.
- Disminuye el "rework".
- Disminuye la confusión.
- Disminuye el riesgo del proyecto.

Algunas actividades son:

- Identificación de la configuración.
- Control de cambios en la configuración.
- Generación de informes de estado.
- Auditoría de la configuración.

Y algunos autores añaden:

- Gestión de releases.
- Planificación de la gestión de configuraciones.

Aspectos a contemplar en el plan de gestión de configuraciones según Métrica 3

- Identificación de todos los productos que deben ser controlados, su clasificación y relaciones entre ellos, así como el criterio o norma de identificación.
- Ubicación y localización de los productos.
- Definición del ámbito y alcance del control de la configuración describiendo los procesos incluidos en él.
- Definición de las reglas de versionado de los productos y los criterios de actuación para cada caso, teniendo en cuenta el motivo por el cual se realiza el cambio de versión.
- Definición del ciclo de estados para cada tipo de producto y los criterios de trazabilidad entre los mismos.
- Descripción de funciones y responsabilidades.
- Identificación de la información necesaria de control para auditoría.

Identificación de la configuración

Refleja la estructura del producto, identifica sus componentes y su tipo, haciendo que todos ellos sean considerados como un todo y accesibles de alguna forma.

- Establecimiento de una jerarquía preliminar del producto: primera visión de la estructura y los elementos que tendrá el sistema software.
- Selección de los ECs y los niveles de control para cada uno de ellos: tener demasiados puede prvocar un exceso de cosas bajo control pudiendo llevar a ralentizar excesivamente el proceso, tener pocos puede producir falta de visibilidad sobre el producto.

Algunos de los criterios usados para seleccionar ECs son el número de personas que lo van a usar, cómo es de crítico, o si se trata de un EC que va a ser reutilizado o está siendo reutilizado desde otra aplicación. Cualquier entidad no necesita ser configurada necesariamente todo el tiempo. Hay que decidir qué debe ser identificado como un EC y cuándo colocarlo bajo control.

Aparece el dilema: si se empieza demasiado pronto se introduce mucha burocracia, pero si se empieza demasiado tarde se produce caos.

Definición de relaciones

Es la actividad encargada de relacionar los ECs previamente identificados.

- Equivalencia: un mismo EC puede estar en distintos soportes pero su contenido ser el mismo.
- Dependencia: el modelo de procesos depende del modelo de datos y esta relación es de doble sentido.
- Derivación: normalmente se pueden identificar entre elementos que tienen un orden cronológico. El diseño de un sistema software deriva del análisis del mismo.
- Sucesión: describe la historia de cambios sobre un EC de una revisión a otra.
- Variante: variaciones sobre un mismo.
- Composición.
- Ser traza de.

Esquema de identificación

Proporciona un identificador únívoco para cada EC. Se pueden cosiderar dos tipos: identificación significativa si proporciona información adicional, como la matrícula antigua que daba dónde se había matriculado el coche; o no significativa, como el DNI.

Un esquema de identificación debe proporcionar el código, nombre y descripción del EC, fecha de creación, proyecto al que pertenece, línea base a la que pertenece, y tipo de EC.

Auditoría de configuración

Tiene como objetivo verificar que el producto de software integrado satisface los requisitos estándares o acuerdos contractuales y que los componentes que se integran se corresponden con las versiones vigentes. Trata de garantizar que el cambio se ha realizado adecuadamente. Está relacionada con la garantía de calidad de software.

Trata de verificar que todos los productos software han sido producidos, descritos e identificados correctamente y que todas las solicitudes de cambio han sido procesadas. Debe responder a preguntas como ¿se ha hecho el cambio especificado?, ¿se han seguido los estándares establecidos?, ¿se han seguido procedimientos para solicitar el cambio, registrarlo y comunicarlo?, ¿se han actualizado adecuadamente todos los EC relacionados?, ¿se ha especificado la fecha de cambio y el autor del mismo? o ¿se ha seguido el procedimiento establecido para señalar el cambio, registrarlo y difundirlo?

Además de otras, se realizan auditorías de línea base al final de cada fase del ciclo de vida o al final de cada nueva entrega. También hay auditorías de configuración funcional y física. Las realiza el personal dedicado a gestión de configuraciones o personal asignado del equiipo. Normalmente las auditorías son revisadas por el ingeniero de calidad de software. Las auditorías de línea base verifican su contenido, comprobando que se usan versiones correctas de los EC para construir la línea base o que la versión de la documentación correcta es la incluida.

Generación de informes de estado

Algunos informes que se pueden obtener son:

- Estado de los cambios propuestos.
- Estado de los cambios aprobados.
- Agendas del comité de control de cambios y actas de las reuniones.
- Progreso de la versión actual hacia adelante o hacia atrás.
- Estimación de los recursos para finalizar una tarea.
- Errores identificados por la auditoría de configuración.

Plan de gestión de configuraciones

Hay que determinar qé elementos se considerarán, quién será responsable de qué actividades, cómo se llevará a cabo, qué información mantener, qué recursos se precisan y cuántos, y qué métricas utilizar para medir el progreso y el éxito.

Algunos criterios a la hora de seleccionar la herramienta de gestión de configuraciones son:

- Soporte multiusuario
- Interfaz de usuario intuitiva
- Adecuación al entorno de desarrollo de la organización
- Escalabilidad
- Flexibilidad para integrarse con otras herramientas de software
- Entorno
- Facilidad de instalar y configurar
- Permitir el uso de modelos de proceso modificables
- Gestión del proceso
- Soporte exhaustivo para la fase de desarrollo
- Gestión de objetos que no son código
- Gestión de permisos