TEMA 5. SEGUIMIENTO DE PROYECTOS: CONTROL DE CALIDAD, GESTIÓN DE RIESGOS Y GESTIÓN DE LA CONFIGURACIÓN DEL SOFTWARE

1. Calidad

Objetivo primordial de la Ingeniería del software: Mejorar la calidad del software a entregar.

Concepto de calidad del software:
Se define en términos de algunas características software de alto nivel. Estas características son:

- *Fiabilidad*: Grado en que un programa satisface sus especificaciones y consigue los objetivos del cliente.
- *Portabilidad*: Esfuerzo requerido para transferir el programa desde un sistema hardware / software a otro.
- *Eficiencia*: Cantidad de recursos de computadora y de código requeridos por un programa para llevar a cabo sus funciones.
- *Ingeniería humana*: Grado en que el software facilita la funcionalidad y el rendimiento asignados al elemento humano.
- *Facilidad de prueba*: Esfuerzo requerido para probar un programa de forma que se asegure que realiza su función requerida.
- *Facilidad de uso*: Esfuerzo requerido para aprender, y trabajar con un programa, así como preparar la entrada e interpretar la salida de un programa.
- *Facilidad de mantenimiento*: Esfuerzo requerido para localizar y reparar un error en un programa.

Objetivo para alcanzar la máxima calidad: Maximizar todas estas características. Difícil por:

- Características contradictorias (por ejemplo, eficiencia y portabilidad).
- En general, la alta calidad se consigue a costa de un gran incremento del coste.
- Difícil definir una métrica simple para medir una característica particular.

Cada una de las características anteriores (de alto nivel) se desglosa en otras de más bajo nivel. Por ejemplo, para la *fiabilidad* tenemos:

- *Exactitud*: Grado de precisión de los cálculos y el control.
- *Robustez*: Grado de correcto funcionamiento del software ante entradas incorrectas.
- *Completitud*: Grado en que se ha conseguido la total implementación de las funciones requeridas.
- *Consistencia*: Grado de uso de un diseño uniforme y de técnicas de documentación a lo largo del desarrollo.
- *Auto-documentación*: Grado en que se documentan las funciones del software.

Suponiendo que podemos definir una métrica para cada primitiva, podemos combinarlas para medir la calidad total. El método consiste en lo siguiente:

- Tenemos n características de alto nivel (C_i, i = 1, ..., n).
- Tenemos m_i características primitivas asociadas a cada C_i.
- Asignamos un valor p_{ij}, entre 0 y 5, a cada característica primitiva j de la característica de alto nivel C_i.
- El valor VC_i para la característica de alto nivel C_i será:

$$VC_i = \sum_{j=1}^{m_i} p_{ij}$$

y el valor total VTC sería:

$$VTC = \sum_{i=1}^{n} \sum_{j=1}^{m_i} p_{ij}$$

La fórmula anterior considera que todas las características tienen la misma importancia. Podemos flexibilizar la fórmula introduciendo un peso w_i a cada característica, tal que la suma de todos los pesos sea n. La fórmula ponderada de VTWC será:

$$VTWC = \sum_{i=1}^{n} w_i \sum_{j=1}^{m_i} p_{ij}$$

Finalmente podemos normalizar el valor anterior mediante:

$$VWNC = \frac{VTWC}{VTC_{\max}}$$

donde VTC_{max} es el máximo valor posible para VTC (todas las características primitivas con valor máximo, es decir, 5). Tenemos:

$$VTC_{\max} = 5 \sum_{i=1}^{n} m_i$$

El valor normalizado VWNC será un valor comprendido entre 0 y 1, significando el valor 0 que la calidad es nula y el valor 1 que la calidad es máxima.

2. Garantía de calidad de la aplicación de gestión

La responsabilidad de la garantía de calidad de un producto software corresponde a muchos elementos de la organización: ingenieros de software, gestores de proyecto, clientes, comerciantes y las personas del **grupo de desarrollo de SQA** (Software Quality Assurance).

Las actividades principales que comprende la **SQA** son:

- Aplicación de metodologías técnicas. La SQA comienza con una serie de herramientas y técnicas para conseguir una especificación y un diseño de alta calidad.
- Realización de revisiones técnicas formales. Cada vez que finaliza una etapa, los resultados los deben evaluar varias personas.
- Prueba del software. Se realizan una serie de pasos para detectar errores.
- Ajuste a los estándares. Si existen estándares formales, debe comprobarse si se siguen o no. De esta tarea se encargan:
  - Los que realizan la revisión técnica formal, ó
  - El grupo de SQA.
- Control de cambios. Debe haber alguien encargado de formalizar las peticiones de cambio, evaluar su naturaleza y controlar su impacto.
- Mediciones. Utilizar métricas de software para garantizar la calidad de un producto, pudiendo servir para plantearse el cambio de metodología.
- Registro y generación de informes. Toda la información referente a la calidad del software se guardará para posteriores consultas del personal que lo necesite.

3. Revisiones formales de la aplicación

Una revisión es un estudio de una serie de personas para:

- Indicar la necesidad de mejoras en un producto.
- Indicar las partes que no es necesario mejorar.
- Conseguir un trabajo técnico más homogéneo.

Una revisión técnica debe centrarse sólo en el producto (no en sus desarrolladores) para detectar si hace lo que debe hacer.

Una revisión formal proporciona información fiable sobre cuestiones técnicas. Las revisiones informales se realizan constantemente.

Las revisiones técnicas formales son la mejor manera de garantizar la calidad del software. Sus objetivos son:

- Descubrir errores en una representación del software.
- Comprobar que se satisfacen los requisitos del software.
- Garantizar que se siguen los estándares definidos.
- Conseguir un desarrollo uniforme.
- Facilitar la gestión de los proyectos.

En primer lugar se seleccionan los revisores de forma que se cubran todos los ámbitos (futuro mantenimiento, ajuste a los estándares de la organización, usuarios y corrección y calidad del producto). A partir de aquí, los pasos que se siguen son:

1. El productor informa al jefe de proyecto de que se requiere una revisión.
2. El jefe de proyecto contacta con un jefe de revisión que:
   - Evalúa la disponibilidad del producto,
   - Genera copias del material del producto y
   - Las distribuye a dos o tres revisores.
3. Los revisores y el jefe de revisión revisan el producto (una o dos horas). El jefe de revisión elabora una agenda para la reunión.
4. Se reúnen el jefe de revisión, los revisores (uno actuará como secretario) y el productor.

   La RTF comienza con una explicación de la agenda (jefe de revisión) y una breve introducción (productor).

   Posteriormente se discuten las pegas que cada revisor ha encontrado. El secretario tomará las notas oportunas.

   La duración de la reunión debe ser inferior a dos horas.
5. Al final decidirán si:
   - Aceptan el producto (no se hacen modificaciones).
   - Rechazan el producto (existen errores graves).
   - Aceptan el producto modificándolo (corregir errores).
6. Todos los revisores firman la conformidad de los resultados de la revisión.

El material obtenido de la revisión es:

- Lista de sucesos de revisión: Identifica las áreas problemáticas dentro del producto y sirve como guía para la corrección de errores por parte del productor.
- Informe sumarial de revisión: Recoge el producto que se ha revisado, los revisores y los problemas detectados, así como las conclusiones finales.

Existen dos variantes de las revisiones técnicas formales:

- Recorridos: RTF guiada por el productor del material a revisar.
- Inspecciones: Revisión a través de una lista de puntos de comprobación.

En las RTF, los recorridos y las inspecciones se trata de encontrar errores, no de corregirlos.

4. Evaluación de riesgos

De forma breve, se puede definir riesgo como una pérdida no esperada.

Los proyectos de software incluyen un conjunto amplio de riesgos: por ejemplo, cambio de los requisitos del usuario, mala estimación de la planificación, personal contratado poco fiable, falta de experiencia en la gestión, problemas de personal, problemas con la tecnología, cambio de las leyes del gobierno y problemas con el desarrollo.

La probabilidad de que un proyecto complejo finalice en el tiempo estimado tiende a cero. Las probabilidades de que un proyecto complejo se cancele se aproximan a las de acertar al lanzar una moneda al aire (50 por ciento).

Un estudio refleja que el 35 por ciento de 600 empresas encuestadas han tenido, al menos, un proyecto que se les ha ido de las manos.

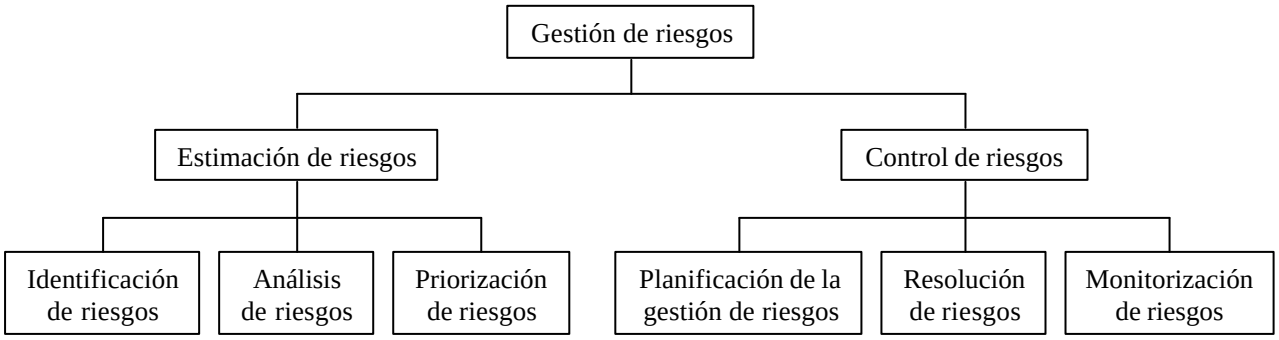
4.1. Elementos de la gestión de riesgos

La función de la gestión de riesgos del software es identificar, estudiar y eliminar las fuentes de riesgo antes de que empiecen a amenazar la finalización satisfactoria de un proyecto software. Se pueden controlar los riesgos a varios niveles.

Los niveles de gestión de riesgos son los siguientes:

- *Control de crisis.* Controlar los riesgos sólo cuando se han convertido en problemas (sería el equivalente a apagar un fuego).
- *Arreglar cada error.* Detectar y reaccionar rápidamente ante cualquier riesgo, pero sólo después de que se haya producido.
- *Mitigación de riesgos.* Planificar con antelación el tiempo necesario para cubrir riesgos en caso de que ocurran, pero no intentar eliminarlos inicialmente.
- *Prevención.* Crear y llevar a cabo un plan como parte del proyecto software para identificar riesgos y evitar que se conviertan en problemas.
- *Eliminación de las causas principales.* Identificar y eliminar los factores que puedan hacer posible la presencia de algún tipo de riesgo.

Generalmente, la gestión de riesgos se divide en estimación de riesgos y control de riesgos. El esquema de descomposición de la gestión de riesgos sería la siguiente:



En los siguientes apartados se describe cada uno de estos componentes de la gestión de riesgos.

4.1.1. Estimación de riesgos

La estimación de riesgos se compone de identificación de riesgos, análisis de riesgos y asignación de prioridades a los riesgos:

- La *identificación de riesgos* genera una lista de riesgos capaces de romper la planificación del proyecto.
- El *análisis de riesgos* mide la probabilidad y el impacto de cada riesgo y los niveles de riesgo de los métodos alternativos.
- La *asignación de prioridades a los riesgos* genera una lista de riesgos ordenados por su impacto. Esta lista sirve como base para el control de riesgos.

4.1.1.1. Identificación de riesgos

El primer paso en la gestión de riesgos es la identificación de los factores que introducen un riesgo en la planificación. Una de las formas más sencillas de identificar los riesgos es comprobar el proyecto frente a una lista de riesgos de la planificación.

La lista de los riesgos más comunes en planificación es la siguiente:

a) Cambio de requisitos.
b) Meticulosidad en requerimientos o desarrolladores.
c) Escatimar en la calidad.
d) Planificaciones demasiado optimistas.
e) Diseño inadecuado.
f) Síndrome de la panacea.
g) Desarrollo orientado a la investigación.
h) Personal mediocre.
i) Error en la contratación.
j) Diferencias entre el personal de desarrollo y los clientes.

Al final de estos apuntes, se muestra un Apéndice que incluye una lista exhaustiva con todos los riesgos que pueden afectar a la planificación de un proyecto software.

4.1.1.2. Análisis de riesgos

Una vez identificados los riesgos de planificación del proyecto, se debe analizar cada riesgo para determinar su impacto. El análisis de los riesgos puede servir para elegir una alternativa de desarrollo o para gestionar los riesgos de desarrollo asociados con la alternativa ya elegida.

Un método útil para realizar el análisis de riesgos es determinar la "exposición a riesgos" (ER) de cada riesgo identificado. La exposición a riesgos es igual a la probabilidad de pérdida no esperada multiplicada por la magnitud de la pérdida. Por ejemplo, si se piensa que la probabilidad puede ser del 25 por 100 y puede haber un retraso de cuatro semanas sobre la duración del proyecto, la exposición al riesgo sería el 25 por 100 multiplicado por cuatro semanas, es decir, una semana. Como en este tema sólo nos estamos centrando en los riesgos de planificación, se pueden expresar las pérdidas en cualquier unidad de tiempo (semanas, meses, ...) que facilite la comparación.

Muchas veces es más fácil estimar la magnitud de la pérdida que la probabilidad. Por ejemplo, se puede saber que el proyecto se aprobará el 1 de febrero o el 1 de marzo, dependiendo del mes en

que la comisión revise la propuesta del proyecto; si se supone que podría aprobarse el 1 de febrero, la magnitud del riesgo para la aprobación del proyecto sería exactamente un mes.

En los casos en que la magnitud de la pérdida no sea fácil de estimar directamente, se puede dividir la pérdida en pérdidas más pequeñas, estimarlas y combinar las pérdidas pequeñas en una estimación de la pérdida (por ejemplo, si se están utilizando tres herramientas de programación, se puede estimar la pérdida resultante, de forma más sencilla, como la suma de las pérdidas que puede generar cada una de las herramientas).

Normalmente, la estimación de la probabilidad de pérdida es más subjetiva que la estimación de la magnitud de la pérdida. Algunas ideas para mejorar la exactitud de esta estimación subjetiva son las siguientes:

- Disponer de una persona más familiarizada con el sistema para que estime la probabilidad de cada riesgo y, posteriormente, realizar una revisión de las estimaciones.
- Utilizar técnicas Delphi o de consenso en grupo (cada persona estima individualmente cada riesgo y luego se discute hasta llegar a un acuerdo).
- Realizar analogías con apuestas (por ejemplo, si los recursos están disponibles en su momento, ganas 1.500 euros; si no lo están, yo gano 1.000 euros. La apuesta se va ajustando hasta que los dos apostantes estén de acuerdo). La probabilidad de pérdida por ese riesgo sería la apuesta menor dividida entre la cantidad total en juego; en el ejemplo, sería 1000/ (1000 + 1500) = 40 por 100.
- Utilizar una clasificación cualitativa. Cada persona elige un nivel de riesgo entre una serie de categorías (por ejemplo: muy probable, bastante probable, probable, improbable, muy improbable) y después se convierte cada estimación cualitativa en cuantitativa.

La exposición a riesgo para cada riesgo considerado se calcula aplicando el porcentaje correspondiente a la probabilidad de pérdida sobre la magnitud de la pérdida (por ejemplo, supóngase que para el riesgo correspondiente a una planificación demasiado optimista se estima una probabilidad de pérdida del 50% y una magnitud de la pérdida de 5 semanas; esto significa que la exposición a riesgo para ese riesgo concreto tiene un valor de 2,5 semanas, que es el 50% de 5 semanas). Si se suman todas las exposiciones a riesgos, se obtiene el tiempo total que se puede retrasar el proyecto, tiempo que se tendrá que considerar a la hora de establecer la duración total del proyecto.

4.1.1.3. Priorización de riesgos

Una vez creada la lista de riesgos de la planificación, se deben priorizar los riesgos para saber dónde se debe centrar el esfuerzo de la gestión de riesgos.

Tras haber calculado la exposición a riesgo para cada uno de los riegos asociados a la planificación del proyecto, se ordenan de mayor a menor en función de este parámetro. Este orden establece aproximadamente la prioridad de los riesgos a la hora de controlarlos. El hecho de considerar este orden como una aproximación se debe a que se puede desear priorizar algún riesgo que produciría una pérdida grande (magnitud de la pérdida elevada). Además, se puede desear priorizar un grupo de riesgos de forma encadenada en lugar de hacerlo individualmente, obteniendo un riesgo medio entre los considerados. Otro motivo por el que la priorización es aproximada es que los valores utilizados son estimaciones (no son valores exactos, sino más bien subjetivos).

Definitivamente, se deben controlar los riesgos de mayor prioridad (teniendo en cuenta todos los aspectos anteriores) e ignorar los que supongan riesgos pequeños y con poca probabilidad de aparición.

4.1.2. Control de riesgos

Una vez identificados los riesgos del proyecto, analizadas sus probabilidades y magnitudes y priorizados, se deben controlar. Las siguientes secciones describen los tres aspectos del control de riesgos.

4.1.2.1. Planificación de la gestión de riesgos

El objetivo es desarrollar un plan que controle cada uno de los riesgos de prioridad alta identificados en las actividades anteriores. El plan de gestión de riesgos puede ser tan sencillo como un párrafo para cada riesgo, describiendo quién, qué, cuándo y cómo se gestiona. También debería contener previsiones para la monitorización de riesgos, descartando aquellos que' se han resuelto e identificando los riesgos que aparecen.

4.1.2.2. Resolución de riesgos

La resolución de un riesgo depende mucho del riesgo específico. Algunos de los métodos para tratar los riesgos son los siguientes:

- *Evitar el riesgo.* Consiste en no realizar actividades arriesgadas.
- *Trasladar el riesgo de una parte del sistema a otra.* A veces, lo que es un riesgo en una parte del proyecto, no lo es en otra, por lo que se puede trasladar. En general, se debe intentar apartar el riesgo del camino crítico, de forma que, si ocurre un riesgo, no se retrase el proyecto completo.
- *Conseguir información acerca del riesgo.* Si no se conoce el auténtico peligro del riesgo, se debe averiguar.
- *Eliminar el origen del riesgo.* Si se sabe dónde puede radicar el riesgo, se debe intentar eliminar el problema antes de que pueda aparecer.
- *Asumir el riesgo.* Consiste en aceptar que puede ocurrir el riesgo pero no pensar demasiado en ello. Se puede aplicar cuando las consecuencias del riesgo sean pequeñas y el esfuerzo para evitarlo es elevado.
- *Comunicar el riesgo.* Consiste en hacer saber al personal de la dirección y de marketing y a los clientes la presencia del riesgo y sus consecuencias, intentando suavizar el susto que pueden llevarse si llega a ocurrir.
- *Controlar el riesgo.* Consiste en aceptar que el riesgo puede ocurrir y desarrollar planes de contingencia para controlar el riesgo si no se puede resolver.
- *Recordar el riesgo.* Consiste en crear una lista de planes de gestión de riesgos que se pueda utilizar en proyectos futuros.

4.1.2.3. Monitorización de riesgos

El desarrollo de aplicaciones software sería más fácil si los riesgos aparecieran después de que se hayan desarrollado planes para tratarlos. El problema es que los riesgos aparecen y desaparecen en el desarrollo del proyecto, por lo que se necesita una monitorización de riesgos para comprobar cómo progresa el control de un riesgo e identificar cómo aparecen los riesgos nuevos.

Una herramienta muy potente para la monitorización de riesgos es la utilización de la lista de "Los 10 riesgos" (como en la música) creada por uno mismo. Esta lista debe contener la posición del riesgo en la lista, su posición anterior, el número de veces que ha aparecido en la lista y un resumen de las actuaciones que se han llevado a cabo desde la última revisión. No es trascendente que la lista tenga exactamente 10 riesgos. El administrador del proyecto y el jefe del administrador deben revisar

periódicamente la lista de riesgos. La principal importancia de esta lista es que fuerza a la revisión periódica de los riesgos e informa sobre la evolución de su estado de influencia sobre el proyecto.

También sería conveniente realizar comprobaciones intermedias de los riesgos durante todo el proyecto, consiguiendo una mayor efectividad si se realizan después de cada hito principal. Muchos responsables de proyectos esperan al final del proyecto para realizar estas comprobaciones pero, en este caso, sólo servirán para futuros proyectos.

Algunas empresas designan a un encargado de riesgos que se encarga de estar alerta sobre los riesgos del proyecto y de evitar que los administradores y desarrolladores los ignoren en la planificación. Esta persona debe buscar todas las razones que puedan hacer que el proyecto falle. En proyectos grandes (más de 50 personas), el encargado de riesgos puede tener dedicación exclusiva, mientras que en proyectos pequeños puede tener asignadas otras tareas.

4.2. Apéndice: Lista completa de riesgos de planificación

Los riesgos que se muestran en este apéndice se han organizado en categorías, pero sin un orden concreto. Además de la lista de riesgos que se muestra, la mayoría de los proyectos tienen riesgos característicos propios (por ejemplo, Juan está dispuesto a abandonar el proyecto si no puede llevar a su perro al trabajo y la dirección aún no ha decidido si deja entrar a Sultán a la oficina); este tipo de riesgos se deben identificar sin observar esta tabla.

Creación de la planificación

- Las definiciones de la planificación, de los recursos y del producto han sido impuestas por el cliente o un directivo superior, y no están equilibradas.
- Planificación optimista, «mejor caso» (en lugar de realista, «caso esperado»).
- La planificación no incluye tareas necesarias.
- La planificación se ha basado en la utilización de personas específicas de un equipo, pero estas personas no están disponibles.
- No se puede construir un producto de tal envergadura en el tiempo asignado.
- El producto es más grande que el estimado (en líneas de código, en el número de puntos de función o en relación con el tamaño del proyecto anterior).
- El esfuerzo es mayor que el estimado (por líneas de código, número de puntos de función, módulos, etc.).
- La reestimación debida a un retraso en la planificación es demasiado optimista o ignora la historia del proyecto.
- La presión excesiva en la planificación reduce la productividad.
- La fecha final ha cambiado sin ajustarse al ámbito del producto o a los recursos disponibles.
- Un retraso en una tarea produce retrasos en cascada en las tareas dependientes.
- Las áreas desconocidas del producto llevan más tiempo del esperado en el diseño y en la implementación.

Organización y gestión

- El proyecto carece de un promotor efectivo en los superiores.
- El proyecto languidece demasiado en el inicio difuso.
- Los despidos y las reducciones de la plantilla reducen la capacidad del equipo.
- Dirección o marketing insisten en tomar decisiones técnicas que alargan la planificación.

- La estructura inadecuada de un equipo reduce la productividad.
- El ciclo de revisión/decisión de la directiva es más lento de lo esperado.
- El presupuesto varía el plan del proyecto.
- La dirección toma decisiones que reducen la motivación del equipo de desarrollo.
- Las tareas no técnicas encargadas a terceros necesitan más tiempo del esperado (aprobación del presupuesto, aprobación de la adquisición de material, revisiones legales, seguridad, etc.).
- La planificación es demasiado mala para ajustarse a la velocidad de desarrollo deseada.
- Los planes del proyecto se abandonan por la presión, llevando al caos y a un desarrollo ineficiente.
- La dirección pone más énfasis en las heroicidades que en informarse exactamente del estado, lo que reduce su habilidad para detectar y corregir problemas.

Entorno de desarrollo

- Los espacios no están disponibles en el momento necesario.
- Los espacios están disponibles pero no son adecuados (por ejemplo, falta de teléfonos, cableado de la red, mobiliario, material de oficina, etc.).
- Los espacios están sobreutilizados, son ruidosos o distraen.
- Las herramientas de desarrollo no están disponibles en el momento deseado.
- Las herramientas de desarrollo no funcionan como se esperaba; el personal de desarrollo necesita tiempo para resolverlo o adaptarse a las nuevas herramientas.
- Las herramientas de desarrollo no se han elegido en función de sus características técnicas y no proporcionan las prestaciones previstas.
- La curva de aprendizaje para la nueva herramienta de desarrollo es más larga de lo esperado.

Usuarios finales

- Los usuarios finales insisten en nuevos requerimientos.
- En el último momento, a los usuarios finales no les gusta el producto, por lo que hay que volver a diseñarlo y a construirlo.
- Los usuarios no han realizado la compra del material necesario para el proyecto y, por tanto, no tienen la infraestructura necesaria.
- No se ha solicitado información al usuario, por lo que el producto al final no se ajusta a las necesidades del usuario y hay que volver a crearlo.

Cliente

- El cliente insiste en nuevos requisitos.
- Los ciclos de revisión/decisión del cliente para los planes, prototipos y especificaciones son más lentos de lo esperado.
- El cliente no participa en los ciclos de revisión de los planes, prototipos y especificaciones, o es incapaz de hacerlo, resultando unos requisitos inestables y la necesidad de realizar unos cambios que consumen tiempo.
- El tiempo de comunicación del cliente (por ejemplo, tiempo para responder a las preguntas para aclarar los requerimientos) es más lento del esperado.
- El cliente insiste en las decisiones técnicas que alargan la planificación.
- El cliente intenta controlar el proceso de desarrollo, con lo que el progreso es más lento de lo esperado.
- Los componentes suministrados por el cliente no son adecuados para el producto que se está

desarrollando, por lo que se tiene que hacer un trabajo extra de diseño e integración.

- Los componentes suministrados por el cliente tienen poca calidad, por lo que tienen que hacerse trabajos extra de comprobación, diseño e integración.
- Las herramientas de soporte y entorno s impuestos por el cliente son incompatibles, tienen un bajo rendimiento o no funcionan de forma adecuada, con lo que se reduce la productividad.
- El cliente no acepta el software entregado, incluso aunque cumpla todas sus especificaciones.
- El cliente piensa en una velocidad de desarrollo que el personal de desarrollo no puede alcanzar.

Personal contratado

- El personal contratado no suministra los componentes en el período establecido.
- El personal contratado proporciona material de una calidad inaceptable, por lo que hay que añadir un tiempo extra para mejorar la calidad.
- Los proveedores no se integran en el proyecto, con lo que no se alcanza el nivel de rendimiento que se necesita.

Requisitos

- Los requisitos se han adaptado, pero continúan cambiando.
- Los requisitos no se han definido correctamente y su redefinición aumenta el ámbito del proyecto.
- Se añaden requisitos extra.
- Las partes del proyecto que no se han especificado claramente consumen más tiempo del esperado.

Producto

- Los módulos propensos a tener errores necesitan más trabajo de comprobación, diseño e implementación.
- Una calidad no aceptable requiere de un trabajo de comprobación, diseño e implementación superior al esperado.
- Utilizar lo último en informática alarga la planificación de forma impredecible.
- El desarrollo de funciones software erróneas requiere volver a diseñarlas y a implementarlas.
- El desarrollo de una interfaz de usuario inadecuada requiere volver a diseñarla y a implementarla.
- El desarrollo de funciones software innecesarias alarga la planificación.
- Alcanzar el ámbito del producto o las restricciones de velocidad requiere más tiempo del esperado, incluyendo el tiempo para volver a diseñar e implementar.
- Unos requisitos rígidos de compatibilidad con el sistema existente necesitan un trabajo extra de comprobación, diseño e implementación.
- Los requisitos para crear interfaces con otros sistemas, otros sistemas complejos u otros sistemas que no están bajo el control del equipo de desarrollo suponen un diseño, implementación y prueba no previstos.
- El requisito de trabajar con varios sistemas operativos necesita más tiempo del esperado.
- El trabajo con un entorno software desconocido causa problemas no previstos.
- El trabajo con un entorno hardware desconocido causa problemas imprevistos.
- El desarrollo de un tipo de componente nuevo para la organización consume más tiempo del esperado.

- Depender de una tecnología que aún está en fase de desarrollo alarga la planificación.

Fuerzas mayores

- El producto depende de las normativas del gobierno, que pueden cambiar de forma inesperada.
- El producto depende de estándares técnicos provisionales, que pueden cambiar de forma inesperada.

Personal

- La contratación tarda más de lo esperado.
- Las tareas preliminares (por ejemplo, formación, finalización de otros proyectos, adquisición de licencias) no se han completado a tiempo.
- La falta de relaciones entre la dirección y el equipo de desarrollo ralentiza la toma de decisiones.
- Los miembros del equipo no se implican en el proyecto y, por lo tanto, no alcanzan el nivel de rendimiento deseado.
- La falta de motivación y de moral reduce la productividad.
- La falta de la especialización necesaria aumenta los defectos y la necesidad de repetir el trabajo.
- El personal necesita un tiempo extra para acostumbrarse a trabajar con herramientas o entornos nuevos.
- El personal necesita un tiempo extra para acostumbrarse a trabajar con hardware nuevo.
- El personal necesita un tiempo extra para aprender un lenguaje de programación nuevo.
- El personal contratado abandona el proyecto antes de su finalización.
- Alguien de la plantilla abandona el proyecto antes de su finalización.
- La incorporación de nuevo personal de desarrollo al proyecto ya avanzado y el aprendizaje y comunicaciones extra imprevistas reducen la eficiencia de los miembros del equipo existentes.
- Los miembros del equipo no trabajan bien juntos.
- Los conflictos entre los miembros del equipo conducen a problemas en la comunicación y en el diseño, errores en la interfaz y tener que repetir algunos trabajos.
- Los miembros problemáticos de un equipo no son apartados, influyendo negativamente en la motivación del resto del equipo.
- Las personas más apropiadas para trabajar en el proyecto no están disponibles.
- Las personas más apropiadas para trabajar en el proyecto están disponibles, pero no se pueden incorporar por razones políticas o de otro tipo.
- Se necesitan personas para el proyecto con habilidades muy específicas y no se encuentran.
- Las personas clave sólo están disponibles una parte del tiempo.
- No hay suficiente personal disponible para el proyecto.
- Las tareas asignadas al personal no se ajustan a sus posibilidades.
- El personal trabaja más lento de lo esperado.
- El sabotaje por parte de la dirección del proyecto deriva en una planificación ineficiente e inefectiva.
- El sabotaje por parte del personal técnico deriva en una pérdida de trabajo o en un trabajo de poca calidad, por lo que hay que repetir algunos trabajos.

Diseño e implementación

- Un diseño demasiado sencillo no cubre las cuestiones principales, con lo que hay que volver a diseñar e implementar.
- Un diseño demasiado complejo exige tener en cuenta complicaciones innecesarias e improductivas en la implementación.
- Un mal diseño implica volver a diseñar e implementar.
- La utilización de metodologías desconocidas deriva en un período extra de formación y tener que volver atrás para corregir los errores iniciales cometidos en la metodología.
- El producto está implementado en un lenguaje de bajo nivel (por ejemplo, ensamblador) y la productividad es menor de la esperada.
- No se puede implementar la funcionalidad deseada con el lenguaje o bibliotecas utilizados; el personal de desarrollo tiene que utilizar otras bibliotecas o crearlas él mismo para conseguir la funcionalidad deseada.
- Las bibliotecas de código o clases tienen poca calidad y generan una comprobación extra, corrección de errores y la repetición de algunos trabajos.
- Se ha sobrestimado el ahorro en la planificación derivado del uso de herramientas para mejorar la productividad.
- Los componentes desarrollados por separado no se pueden integrar de forma sencilla, teniendo que volver a diseñar y repetir algunos trabajos.

Proceso

- La burocracia produce un progreso más lento del esperado.
- La falta de un seguimiento exacto del progreso hace que se desconozca que el proyecto esté retrasado hasta que está muy avanzado.
- Las actividades iniciales de control de calidad son recortadas, haciendo que se tenga que repetir el trabajo.
- Un control de calidad inadecuado hace que los problemas de calidad que afectan a la planificación se conozcan tarde.
- La falta de rigor (ignorar los fundamentos y estándares del desarrollo de software) conduce a fallos de comunicación, problemas de calidad y repetición del trabajo. Un consumo de tiempo innecesario.
- El exceso de rigor (aferramiento burocrático a las políticas y estándares de software) lleva a gastar más tiempo del necesario en la gestión.
- La creación de informes de estado a nivel de directiva lleva más tiempo al desarrollador de lo esperado.
- La falta de entusiasmo en la gestión de riesgos impide detectar los riesgos más importantes del proyecto.
- La gestión de riesgos del proyecto software consume más tiempo del esperado.