Riesgos

Categorización de los riesgos

Riesgo	Repercute en	Descripción
Rotación de personal	Proyecto	Personal experimentado abandonará el proyecto antes de que éste se termine.
Cambio administrativo	Proyecto	Habrá un cambio de gestión en la organización con diferentes prioridades.
Indisponibilidad de hardware	Proyecto	Hardware, que es esencial para el proyecto, no se entregará a tiempo.
Cambio de requerimientos	Proyecto y producto	Habrá mayor cantidad de cambios a los requerimientos que los anticipados.
Demoras en la especificación	Proyecto y producto	Especificaciones de interfaces esenciales no están disponibles a tiempo.
Subestimación del tamaño	Proyecto y producto	Se subestimó el tamaño del sistema.
Bajo rendimiento de las herramientas CASE	Producto	Las herramientas CASE, que apoyan el proyecto, no se desempeñan como se anticipaba.
Cambio tecnológico	Empresa	La tecnología subyacente sobre la cual se construye el sistema se sustituye con nueva tecnología.
Competencia de productos	Empresa	Un producto competitivo se comercializa antes de que el sistema esté completo.

Determinar categoría, impacto y probabilidad

Riesgos	Categoría	Probabilidad	Impacto
El cliente cambiará los requisitos	Proy	80%	
Falta de formación en las herramientas	Proy	80%	

Riesgos	Categoría	Probabilidad	Impacto
El cliente cambiará los requisitos	Proy	80 %	2
Falta de formación en las herramientas	Prod	80%	3
Menos reutilización de la prevista	Proy	70 %	2
La estimación del tamaño puede ser muy baja	Proy	60 %	2
Habrá muchos cambios de personal	Proy	60 %	2
La fecha de entrega estará muy ajustada	Proy	50%	2
Se perderán los presupuestos	Neg	40%	1
Los usuarios finales se resisten al sistema	Neg	40%	3
La tecnología no alcanzará las expectativas	Prod	30%	1
Personal sin experiencia	Proy	30%	2
Mayor número de usuarios de los previstos	Neg	30%	3

20

Línea de

2

*Nota: El 1 , 2 ,3 son el nivel de impacto, siendo 1 catastrofico (cancelan el proyecto) , 2 serio (baja rendimiento, costos adicionales y así) , 3 Tolerable (baja apenas el rendimiento , exceso de costos), 4 insignificante (impacto mínimo en el desarrollo)

planes de contingencia

Riesgo	Estrategia
Problemas financieros de la organización	Prepare un documento informativo para altos ejecutivos en el que muestre cómo el proyecto realiza una aportación muy importante a las metas de la empresa y presente razones por las que los recortes al presupuesto del proyecto no serían efectivos en costo.
Problemas de reclutamiento	Alerte al cliente de dificultades potenciales y de la posibilidad de demoras; investigue la compra de componentes.
Enfermedad del personal	Reorganice los equipos de manera que haya más traslape de trabajo y, así, las personas comprendan las labores de los demás.
Componentes defectuosos	Sustituya los componentes potencialmente defectuosos con la compra de componentes de conocida fiabilidad.
Cambios de requerimientos	Obtenga información de seguimiento para valorar el efecto de cambiar los requerimientos; maximice la información que se oculta en el diseño.
Reestructuración de la organización	Prepare un documento informativo para altos ejecutivos en el que muestre cómo el proyecto realiza una aportación muy importante a las metas de la empresa.
Rendimiento de la base de datos	Investigue la posibilidad de comprar una base de datos de mayor rendimiento.
Subestimación del tiempo de desarrollo	Investigue los componentes comprados; indague el uso de un generador de programa.

Elementos de software

- ✓ Especificación del sistema
- ✓ Plan del proyecto software
- ✓ Especificación de diseño:
  - ✓ a) Diseño preliminar
  - ✓ b) Diseño detallado
- ✓ Listados del código fuente
- ✓ Planificación y procedimiento de prueba
- ✓ Casos de prueba y resultados registrados
- ✓ Manuales de operación y de instalación
- ✓ Programas ejecutables
- ✓ Descripción de la base de datos
  - a) Esquema, modelos
  - b)Datos iniciales
- ✓ Manual de usuario
- ✓ Documentos de mantenimiento
- ✓ Estándares y procedimientos de ingeniería del software

Planificación temporal

Lista de tareas

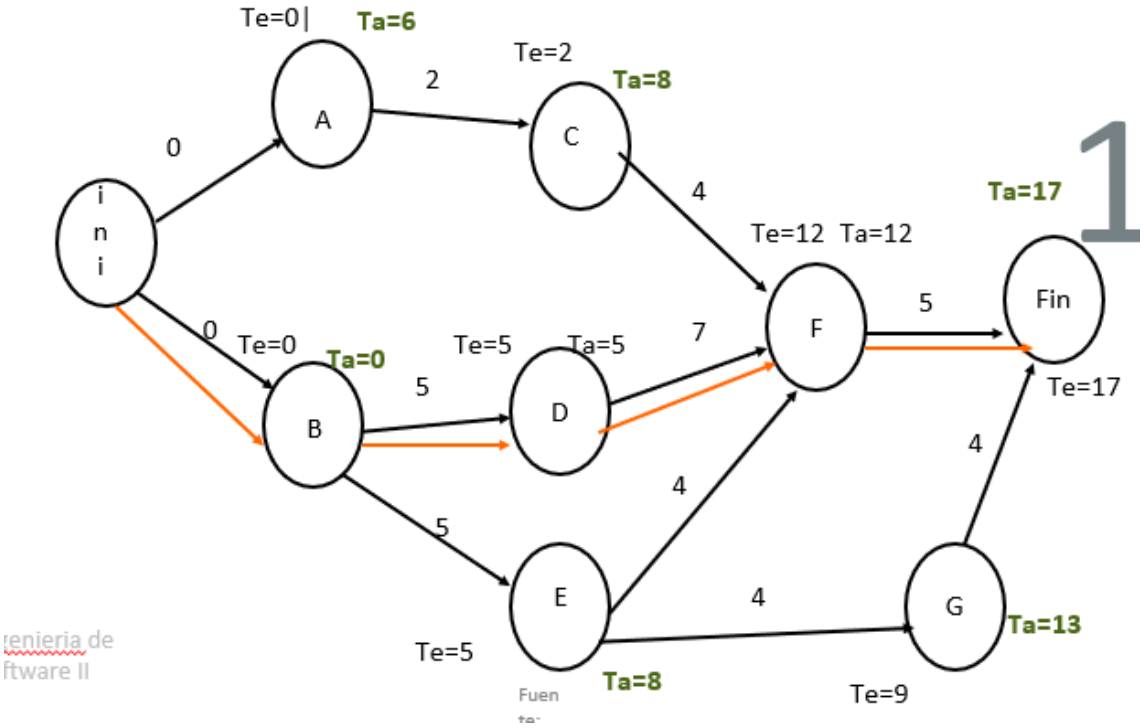
Ejemplo

1. *Establecer lista de tareas*
2. *Fijar dependencia entre tareas y duración*

Tarea	Precedida por	Duración
A	-	2
B	-	5
C	A	4
D	B	7
E	B	4
F	C-D-E	5
G	E	4

14

Redes de tareas

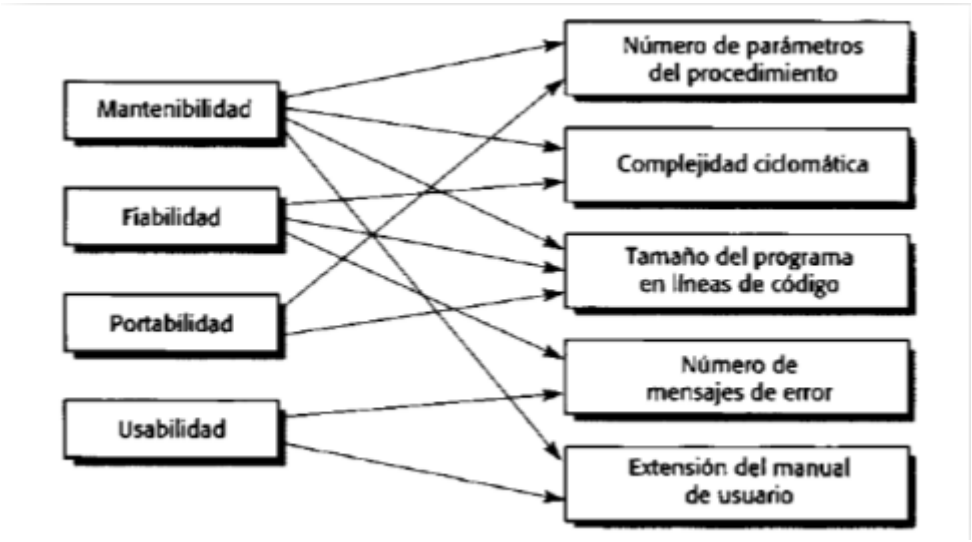


1

Metricas

Metricas de proceso

Característica del proceso	Cuestiones clave
Comprensión	¿En qué medida el proceso está definido explícitamente y qué tan fácil es entender la definición del proceso?
Estandarización	¿Hasta qué punto el proceso se basa en el proceso genérico estándar? Esto puede ser importante para algunos clientes que requieran la conformidad con un conjunto de estándares definidos de proceso. ¿Hasta dónde se usa el mismo proceso en todas las partes de una compañía?
Visibilidad	¿Las actividades del proceso culminan en resultados claros, de modo que el avance del proceso se observa externamente?
Mensurabilidad	¿El proceso incluye recolección de datos u otras actividades que permitan medir las características del proceso o del producto?
Soportabilidad	¿En qué medida pueden usarse herramientas de software para apoyar las actividades del proceso?
Aceptabilidad	¿El proceso definido es aceptable y útil para los ingenieros responsables de elaborar el producto de software?
Fiabilidad	¿El proceso está diseñado de tal forma que se evitan o se detectan errores de proceso antes de que deriven en errores de producto?
Robustez	¿El proceso puede continuar a pesar de problemas inesperados?
Mantenibilidad	¿El proceso puede evolucionar para reflejar los requerimientos cambiantes de la organización o mejoras identificadas en el proceso?
Rapidez	¿Qué tan rápido puede completarse el proceso de entrega de un sistema a partir de una especificación dada?



Metricas estáticas del producto

Fan-in/Fan-out	Fan-in es una medida del número de funciones o métodos que llaman a otra función o método (por ejemplo, X). Fan-out es el número de funciones que son llamadas por una función X. Un valor alto de fan significa que X está fuertemente acoplada al resto del diseño y que los cambios en X tendrán muchos efectos importantes. Un valor alto de fan-out sugiere que la complejidad de X podría ser alta debido a la complejidad de la lógica de control necesaria para coordinar los componentes llamados.
Longitud del código	Ésta es una medida del tamaño del programa. Generalmente, cuanto más grande sea el tamaño del código de un componente, más complejo y susceptible de errores será el componente. La longitud del código ha mostrado ser la métrica más fiable para predecir errores en los componentes.
Complejidad ciclomática	Ésta es una medida de la complejidad del control de un programa. Esta complejidad del control está relacionada con la comprensión del programa.
Longitud de los identificadores	Es una medida de la longitud promedio de los diferentes identificadores en un programa. Cuanto más grande sea la longitud de los identificadores, más probable será que tengan significado; por lo tanto, el programa será más comprensible.
Profundidad del anidamiento de las condicionales	Ésta es una medida de la profundidad de anidamiento de las instrucciones condicionales «if» en un programa. Muchas condiciones anidadas son difíciles de comprender y son potencialmente susceptibles de errores.
Índice de Fog	Ésta es una medida de la longitud promedio de las palabras y las frases en los documentos. Cuanto más grande sea el índice de Fog, el documento será más difícil de comprender.

Métrica orientada a objetos	Descripción
Métodos ponderados por clase (*weighted methods per class*, WMC)	Éste es el número de métodos en cada clase, ponderado por la complejidad de cada método. Por lo tanto, un método simple puede tener una complejidad de 1, y un método grande y complejo tendrá un valor mucho mayor. Cuanto más grande sea el valor para esta métrica, más compleja será la clase de objeto. Es más probable que los objetos complejos sean más difíciles de entender. Tal vez no sean lógicamente cohesivos, por lo que no pueden reutilizarse de manera efectiva como superclases en un árbol de herencia.
Profundidad de árbol de herencia (*depth of inheritance tree*, DIT)	Esto representa el número de niveles discretos en el árbol de herencia en que las subclases heredan atributos y operaciones (métodos) de las superclases. Cuanto más profundo sea el árbol de herencia, más complejo será el diseño. Es posible que tengan que comprenderse muchas clases de objetos para entender las clases de objetos en las hojas del árbol.
Número de hijos (*number of children*, NOC)	Ésta es una medida del número de subclases inmediatas en una clase. Mide la amplitud de una jerarquía de clase, mientras que DIT mide su profundidad. Un valor alto de NOC puede indicar mayor reutilización. Podría significar que debe realizarse más esfuerzo para validar las clases base, debido al número de subclases que dependen de ellas.
Acoplamiento entre clases de objetos (*coupling between object classes*, CBO)	Las clases están acopladas cuando los métodos en una clase usan los métodos o variables de instancia definidos en una clase diferente. CBO es una medida de cuánto acoplamiento existe. Un valor alto para CBO significa que las clases son estrechamente dependientes y, por lo tanto, es más probable que el hecho de cambiar una clase afecte a otras clases en el programa.
Respuesta por clase (*response for a class*, RFC)	RFC es una medida del número de métodos que potencialmente podrían ejecutarse en respuesta a un mensaje recibido por un objeto de dicha clase. Nuevamente, RFC se relaciona con la complejidad. Cuanto más alto sea el valor para RFC, más compleja será una clase y, por ende, es más probable que incluya errores.
Falta de cohesión en métodos (*lack of cohesion in methods*, LCOM)	LCOM se calcula al considerar pares de métodos en una clase. LCOM es la diferencia entre el número de pares de método sin compartir atributos y el número de pares de método con atributos compartidos. El valor de esta métrica se debate ampliamente y existe en muchas variaciones. No es claro si realmente agrega alguna información útil además de la proporcionada por otras métricas.

Estimaciones

Para mi no sirve, pero why not

Oportunidad de mercado	Una organización de desarrollo podría ofertar un bajo precio debido a que desea conseguir cuota de mercado. Aceptar un beneficio bajo en un proyecto podría darle la oportunidad de obtener más beneficios posteriormente. La experiencia obtenida le permite desarrollar nuevos productos.
Incertidumbre en la estimación de costes	Si una organización está insegura de su coste estimado, puede incrementar su precio por encima del beneficio normal para cubrir alguna contingencia.
Términos contractuales	Un cliente puede estar dispuesto a permitir que el desarrollador retenga la propiedad del código fuente y que reutilice el código en otros proyectos. Por lo tanto, el precio podría ser menor que si el código fuente del software se le entregara al cliente.
Volatilidad de los requerimientos	Si es probable que los requerimientos cambien, una organización puede reducir los precios para ganar un contrato. Después de que el contrato se le asigne, se cargan precios altos a los cambios en los requerimientos.
Salud financiera	Los desarrolladores en dificultades financieras podrían bajar sus precios para obtener un contrato. Es mejor tener beneficios más bajos de los normales o incluso quebrar antes de quedar fuera de los negocios.

Los tipos de interfaces están en los resúmenes de diego

Los distintos modelos de arquitectura

Depuracion

El Proceso de Depuración
Características de los errores

1. Síntoma lejano (geográficamente) de la causa
2. Síntoma desaparece temporalmente al corregir otro error
3. Síntoma producido por error
4. Síntoma causado por error humano
5. Síntoma causado por problemas de tiempo
6. Condiciones de entrada difíciles de reproducir
7. Síntoma intermitente (especialmente en desarrollos hardware-software)
8. El síntoma se debe a causas distribuidas entre varias tareas que se ejecutan en diferentes procesadores

14

Defectos (cuando el software falla)

Clasificación ortogonal de Defectos

»Pfleeger Cap. 8

Tipo de defecto	Significado
Función	Afecta la capacidad, interfaces.
Interfaz	Afecta a la interacción con otros componentes.
Comprobación	Afecta la lógica del programa.
Asignación	Afecta la estructura de datos.
Sincronización	Involucra sincronización de recursos compartidos y de tiempo real.
Construcción	Ocurre debido a problemas en repositorios, gestión de cambios o control de versiones.
Documentación	Afecta a publicaciones.
Algoritmo	Involucra la eficiencia o exactitud de un algoritmo.