

Revisiones del software

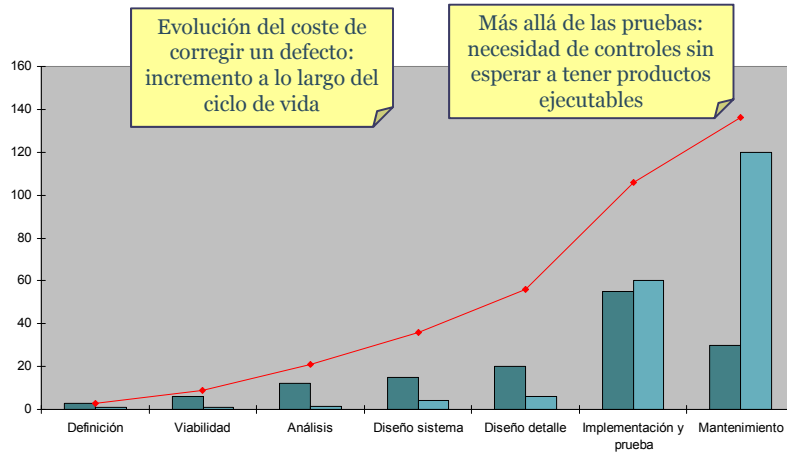
Ingeniería del Software Avanzada
Revisiones

V y V

- **Concepto:**
 - Conjunto de procedimientos, técnicas y herramientas
 - Uso paralelo al desarrollo de software
 - Asegurar que el producto satisface necesidades
- **Boehm:**
 - **Verificar:** ¿construir correctamente el producto?
Comprobar calidad en el ciclo de vida
 - **Validar** ¿construir el producto adecuado? Comprobar si satisface los requisitos



Evolución de costes de corrección



Diferentes técnicas

- Tres aspectos que diferencian a las diferentes técnicas de revisión entre sí:
 - Sobre qué se realiza la revisión (producto o proyecto).
 - Objetivos perseguidos (aunque genéricamente sean la búsqueda de defectos).
 - El proceso o mecánica de realización.
- En cuanto a su mecánica:
 - Revisiones informales: no hay procedimientos definidos, se realiza de la forma más flexible posible.
 - Revisiones semi-formales: se definen unos procedimientos mínimos a seguir.
 - Revisiones formales: se define completamente el proceso, los participantes y sus funciones, los documentos, etc.

Clasificación de técnicas

- Revisiones de proyecto
 - Revisiones de gestión
- Revisiones de producto
 - Revisiones técnicas
 - Inspecciones
 - Walkthroughs
 - Pruebas
 - Otras
- Auditorías de software

Dar recomendaciones para actividad de gestión según estado de producto
Control de proyecto
Cambio de dirección de proyecto

Evaluación más o menos formal del software: identifica desviaciones
Satisfacción de especificaciones
Conformidad con planes, estándares, guías aplicables a cada producto

Revisión independiente
Evaluación objetiva de prod. y proy.
Conformidad o implantación: estándares, procedimientos, etc. contractuales o normativos

Comparativa

	Mecánica	Objeto a probar	Errores	Resultado
Revisión de gestión	Formal / Equipo cualificado	Proyecto	De gestión y planificación	Lista de errores y recomendaciones, no soluciones
Revisión técnica	Formal / Equipo cualificado	Análisis, diseño y pruebas	Funcionales, técnicos y normativos	Lista de errores
Inspección	Formal / Equipo con autor y otros	Análisis, diseño, construcción y pruebas	Funcionales, técnicos y normativos	Lista de errores (soluciones en Tercera hora, opcional) y seguimiento
Walkthrough	Cualquiera / Equipo con autor y otros	Análisis, diseño, construcción y pruebas	Funcionales, técnicos, normativos y estilo	Lista de errores y soluciones
Prueba	Cualquiera / Autor y otros mismo nivel	Construcción	Funcionales y técnicos	Lista de errores
Auditoría de software	Formal / Auditor	Proyecto, análisis, diseño, construcción y pruebas	Funcionales, normativos y estilo	Lista de errores y recomendaciones, no soluciones

Otras verificaciones y validaciones

- Cada proyecto y organización decidirá las que necesita
- Ejemplos de otras técnicas complementarias:
 - **Análisis de algoritmos.** Verificar funcionalidad y consumo de recursos en tiempo de ejecución.
 - **Análisis de simulación.** Evaluación del rendimiento para planificar la capacidad de un sistema.
 - **Audidores de código.** Examinar código fuente y controlar cumplimiento de estándares y prácticas de programación.
 - **Generadores de referencias cruzadas.** Control basado en nombres de variables, procedimientos, etiquetas, etc.
 - **Analizadores de flujo de control.** Determinar secuencias incorrectas en la ejecución del flujo de control de un programa.
 - **Comprobación de interfaces.** Analizar consistencia y completión y analizar usabilidad y accesibilidad.
 - **Análisis de requisitos.** Errores sintácticos, inconsistencias lógicas o ambigüedades en entradas, salidas, procesos y datos.

Pruebas - Definiciones

- Pruebas
 - Ejecución de software con datos controlados (casos) con el fin de descubrir defectos
- Caso de prueba
 - Conjunto específico de entrada, procedimientos y salida esperada para una situación de operación
 - Buen caso de prueba, gran probabilidad de detectar defectos no encontrados antes
- Éxito
 - Descubrir un defecto no detectado
- Dijkstra: Las pruebas no pueden asegurar la ausencia de defectos, sólo demostrar que (los que localizamos) existen en el software

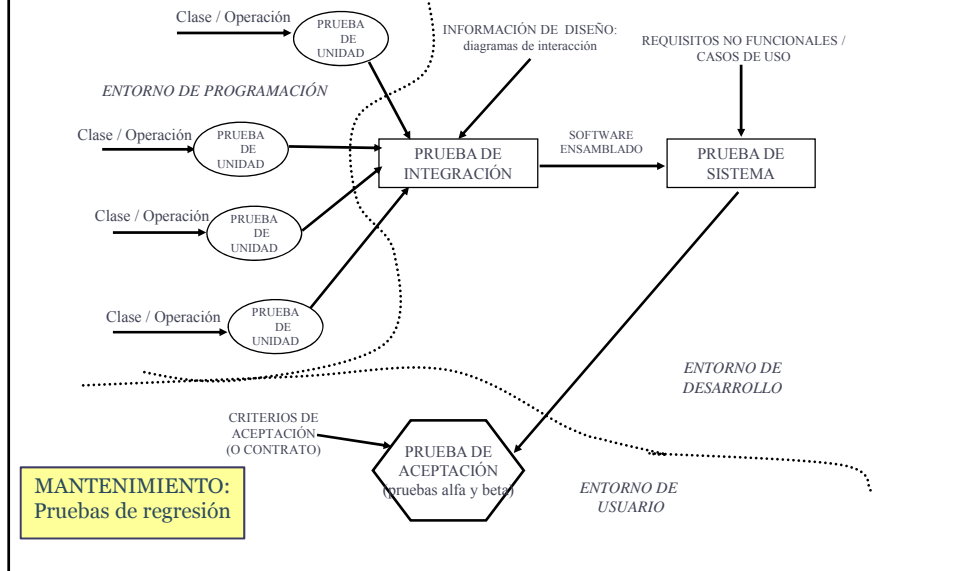
Pruebas - Recomendaciones generales

- ☺ ¿El autor prueba su software?
- ☺ Cada caso debe definir en detalle la salida esperada
- ☺ Inspeccionar con detalle la salida obtenida
- ☺ Incluir tanto entradas no válidas e inesperadas como válidas y esperadas
- ☺ Comprobar si el software:
 - No hace lo que debe hacer (fallo de funciones)
 - Hace lo que supone no debe hacer (efectos secundarios)
- ☺ Evitar utilizar casos desechables (control y economía)
- ☺ No planear suponiendo ausencia de defectos
- ☺ Estudios:
 - Probabilidad de nuevos defectos es proporcional al número de defectos ya encontrados

Pruebas - Importancia práctica

- El diseño de casos es totalmente dependiente de una buena especificación
 - Muchas veces, el trabajo de pruebas supone hacer el trabajo que no hicieron los analistas
- En cuanto tenemos una buena especificación se pueden diseñar los casos de prueba
 - En especial, si contamos con casos de uso
 - No se diseña justo antes de probar
- Eficiencia y limitación de recursos
 - No buscar pruebas perfectas: equilibrio riesgo-coste
- Como última fase, las pruebas sufren los retrasos de todas las fases de desarrollo
 - La planificación de pruebas debe contemplar plazos generosos

Pruebas - Niveles



Plan de pruebas

- Documento que contiene tipos de pruebas a realizar. Para cada tipo de prueba:
 - Quien prueba: individual o equipo (número de personas), autor u otros (definir nivel), varias personas o equipos prueban lo mismo o solo una prueba
 - Cuando y plazo estimado de duración
 - Con que métodos (diseño de pruebas) y herramientas
 - En que entorno (desarrollo, pre-producción, específico para pruebas, producción)
 - Que documentación se proporciona al iniciar la prueba y se debe generar al finalizarla

Diseño de pruebas

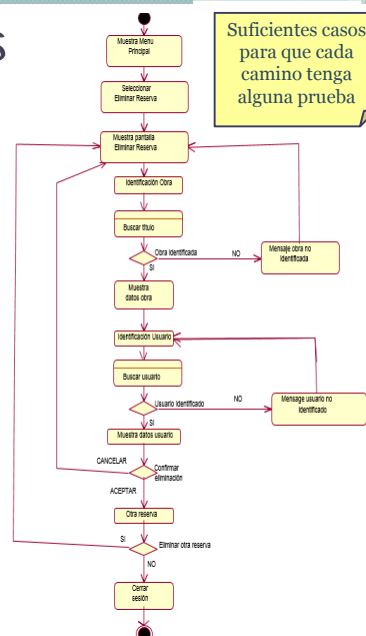
- Tres enfoques:
 - Funcional o caja negra
 - Particiones de equivalencia
 - Valores límite
 - Conjetura de errores
 - Estructural o caja transparente
 - Grafo de flujo
 - Complejidad ciclomática de McCabe
 - Caminos de prueba ($a-n+2$)
 - Criterios de cobertura (de sentencias, de decisiones,...)
 - Aleatoria

Prueba de sistema a partir de casos de uso

- Generar casos de prueba:
 - Caminos/escenarios de ejecución de caso
 - Diagrama de secuencia del sistema, de actividad, de interacción o de estados
 - Recogen combinaciones y opciones no válidas (escenarios alternativos)
 - Sobre cada escenario, diseño de caja negra:
 - Clases de equivalencia y límites
 - Determinar los datos de entrada/salida de los casos de prueba
 - No olvidar efectos secundarios (acciones no visibles al usuario)

Determinar escenarios

Bibliotecario	Sistema
	1.- Muestra por pantalla el menú principal
2.- Selecciona la opción <i>Eliminar Reserva</i>	3.- Muestra la pantalla de <i>Eliminar Reserva</i>
4.- Introduce el <i>idObra</i>	5.- <include <i>Buscar Título</i> >
	6.- Muestra los datos de la obra y pide el DNI del usuario.
4.- Introduce el DNI	5.- <include <i>Buscar Usuario</i> >
	9.- Muestra datos del usuario y confirmar eliminar reserva
10.- Pulsa <i>ACEPTAR</i>	11.- Mensaje si eliminar otra reserva.
12.- Pulsa <i>NO</i>	13.- Cierra sesión y muestra menú principal



Particiones de equivalencia

Añadir valores límite y casos especiales de conjetura de errores

Condición o restricción de entrada	Clases válidas	Clases no válidas
Código de área	(1) $200 \leq \text{código} \leq 999$	(2) código < 200 (3) código > 999 (4) no es número
Texto de la operación	(5) 6 caracteres	(6) < 6 caracteres (7) > 6 caracteres
Operación	(8) ingreso (9) transferencia (10) pago recibo (11) retirada efectivo	(12) ninguna orden válida

Casos válidos:

- 300 Nómina Ingreso (1)(5)(8)
- 400 Bilbao Transferencia (1)(5)(9)
- 500 I.T.V. Pago recibo (1)(5)(10)
- 600 Cheque Retirada efectivo (1)(5)(11)

Casos no válidos:

- 180 Nómina Ingreso (2)(5)(8)
- 1032 XXXXXX Ingreso (3) (5) (8)
- ZV YYYYYY Ingreso (4) (5) (8)
- 350 ↵ Transferencia (1)(6) (8)
- 450 Regalos Transferencia (1)(7) (8)
- 550 IRPF97 Saldo (1)(5)(12)

Inspección

- **Objetivos:** enfocada en descubrir defectos
 - No sobre cómo corregir, añadidos o mejoras
- **Equipos de inspección**
 - Pequeños grupos de compañeros de trabajo (de 3 a 6): 4-5, lo más común.
 - Autor y compañeros que desarrollan productos relacionados
 - Uso de listas de comprobación
 - Duración de reuniones: máximo 2 horas
- **Roles**
 - Autor no puede ser moderador o lector (presenta el producto)
 - Los moderadores capacitados son esenciales: asegurar preparación, ritmo de reunión, enfoque en defectos y evitar ataques a autor
- **Productos**
 - Aplicable a muchos productos: especificaciones ,diseño, código, pruebas,..
 - Tamaño habitual: 10-20 páginas o 200-250 LOC
- **Salidas**
 - Lista de defectos e informe de revisión: qué, quién, nº y gravedad de defectos
 - Tasa de inspección (velocidad en páginas o LOC por hora)

Inspección - Fases

Etap	Responsable	Actividades
Planificación	Moderador	Producto cumple requisitos de entrada Conseguir participantes adecuados Fijar lugar y momento adecuados
Vista general (opcional)	Equipo al completo	Formar a participantes Asigna papeles
Preparación	Individual	Cada participante estudia el producto, prepara su papel y anota defectos
Reunión	Equipo al completo	Encontrar defectos (sin discutir posibles soluciones)

Inspección -Tercera hora (opcional)

Etapas	Responsable	Actividades
Tercera hora	Equipo al completo	Exponer ideas suprimidas en la reunión sobre mejoras o soluciones
Corrección	Autor o a quien se asigne	Corregir defectos encontrados
Seguimiento	Moderador	Verificar la corrección de todos los defectos
Análisis causal	Equipo al completo o depto. de calidad	Analizar estadísticas de reuniones (detectar causas y mejoras en los procesos, métodos desarrollo, etc.)

Inspección - Criterios de terminación

- La inspección acaba cuando:
 - Todos los defectos detectados se han resuelto
 - Los resultados de la inspección se han pasado a los informes
- El moderador:
 - Verifica ambos criterios antes de declarar terminada y completa la inspección
- Cada proyecto debería desarrollar criterios propios según las necesidades del mismo y del entorno

Inspecciones - Listas de comprobación

- Series de preguntas o comprobaciones para examinar el producto
- Proporciona definición clara de la tarea a los participantes
- Se construyen a base de acumular experiencias en revisiones
 - Asimilar las estadísticas de defectos
- Extensión:
 - Una sola página con 20-25 preguntas o ítems

Walkthroughs

- *Walkthrough*: recorrido del producto
- Revisión de cualquier producto por un grupo de nivel similar al que lo desarrolló
 - En general, del equipo de desarrollo
- Objetivo: el autor explica el producto (presenta el enfoque de diseño y programación e informa a compañeros del progreso de trabajo) mientras los demás se centran en:
 - Encontrar defectos:
 - Errores, omisiones, contradicciones, debilidades, mejoras
 - Estudio de la técnica y el estilo de desarrollo
 - Búsqueda de alternativas y soluciones

Walkthroughs - Tipos

- Según el grado de organización y estructuración:
 - Formales:
 - Preparación larga, autor gasta tiempo en presentar, realimentación alta y lenta (semanas, buen trabajo)
 - Informales
 - Preparación muy baja, realimentación rápida y baja
 - Semiformales
 - Lo más recomendable: ventajas de ambos
- En ciclo de vida:
 - Informales en primeras fases y formales en las finales

Walkthroughs - Roles

- Autor
- Moderador (puede ser el propio autor)
- Posibles miembros del equipo:
 - Encargado de mantenimiento:
 - Prever futuro de mantenimiento
 - Producto autoexplicativo, mantenimiento sin el autor, etc.
 - Supervisor de estándares
 - Conformidad respecto de normativa y estándares
 - No literal sino el espíritu y teniendo en cuenta el entorno
 - Representante de usuario
 - Ajuste a las necesidades, especialmente en especificaciones
 - Otros revisores, normalmente del mismo nivel que el autor
 - Opinión general sobre corrección y calidad
 - Posibles externos para visión más distanciada

Walkthrough - Fases

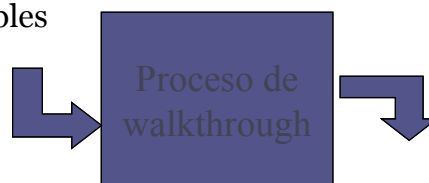
Depende de su grado de formalidad

- Planificación, incluye lista de comprobación
- Reunión de visión general
- Preparación individual
- Reunión de trabajo:
 - Reconocimiento de objetivos:
 - Presentador: para guiar la reunión
 - Presentación del producto (si procede)
 - Breve, si ha habido suficiente preparación
 - Críticas y comentarios
 - Incluyendo walkthroughs anteriores
 - Calificación final:
 - Aceptado, aceptado con modificaciones, nuevo walkthrough.

Walkthrough - Documentación

Depende de su grado de formalidad

- Elemento de software
- Lista de objetivos
- Estándares aplicables
- Especificaciones



- Lista de elementos de acción: defectos y para cada uno grado de gravedad (indicar si es crítico para aceptación) y acciones (para corregir y/o mejorar)
- Informe del *walkthrough*: qué, quién, objetivos y grado de alcance, conclusión y propuestas para posterior seguimiento

Auditorías

- **Objetivo:**
 - Confirmar el cumplimiento de producto y/o proyecto para asegurar el cumplimiento de:
 - Estándares, Guías, Especificaciones, Procedimientos, Ejecución de proyecto y productividad
- **Revisión independiente y muy disciplinada**
 - No participativa: visión del auditor
 - Figura de autoridad: auditor
- **Permite evaluar:**
 - Elementos de software
 - Procesos
 - Proyectos
 - Planes de calidad

Auditorías - ¿Cuándo realizarlas?

- **Punto singular del proyecto**
 - Seguimiento de planes
 - Fecha, criterio existente, etc.
- **Demanda de grupo externo:**
 - Agencia reguladora, usuarios o clientes, etc.
 - Requisitos contractuales o normativos
- **A petición de la organización:**
 - Jefe de proyecto
 - Departamento de SQA

Auditorías - Fases

- **Planificación**
 - Objetivos de auditoría y Alcance
- **Visión general**
 - Definición del proceso de auditoría, calendario (pueden durar meses) y acuerdos de colaboración con la empresa auditada
- **Preparación**
 - Conocer la organización a auditar y preparar un plan
- **Examen**
 - Recogida de datos (documentación y entrevistas)
- **Análisis**
 - Determinar la salud del proyecto o producto usando métricas
- **Informe de resultados**
 - Defectos, descubrimientos y recomendaciones