INFOGRAFÍA

DISEÑO DE PRUEBAS DE SW

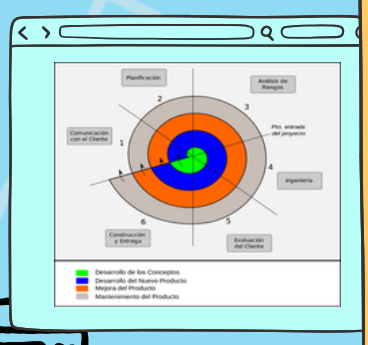
Entornos de Desarrollo

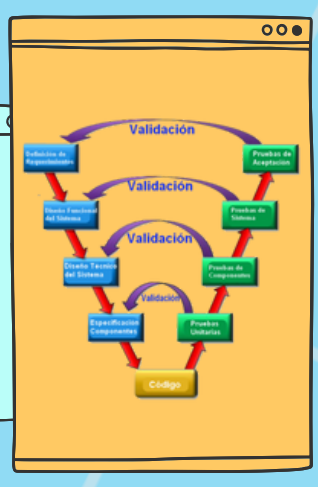
Elena Mena y Manuel Andrade

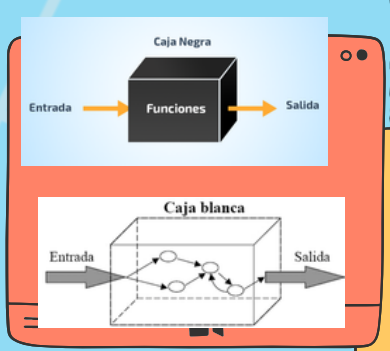
PLANIFICACIÓN DE LAS PRUEBAS

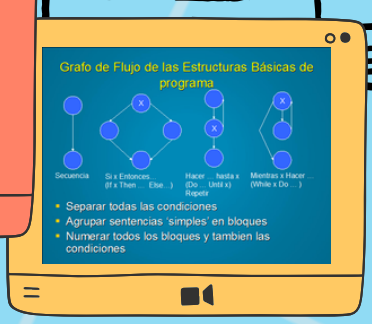
Son esenciales para garantizar que el sistema cumple con las especificaciones y detectar errores en diferentes fases del desarrollo.

- Pruebas unitarias: módulos individuales del código.
- Pruebas de integración: conexión entre módulos y su arquitectura.
- Pruebas de aceptación: se comprueban los requisitos definidos.
- Pruebas de sistema: comportamiento global del software.









TIPO DE PRUEBAS

Caja negra: funcionalidad.
Caja blanca: código fuente y los caminos de ejecución.
Pruebas de rendimiento:
- Pruebas de carga: múltiples usuarios simultáneos.
- Pruebas de estrés: comportamiento ante una carga extrema.
- Pruebas de estabilidad: si el sistema soporta una carga prolongada sin fallos.
- Pruebas de picos: simulan cambios bruscos en la cantidad de usuarios.

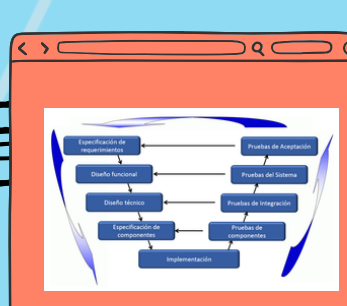
Pruebas estructurales y funcionales:
- Pruebas estructurales: estructura interna del código.
- Pruebas funcionales: si el software cumple los requisitos del usuario.

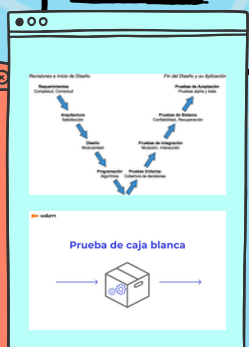
Pruebas de regresión: tras modificaciones para asegurar que no se han introducido nuevos errores en otras partes del sistema.

PROCEDIMIENTOS Y CASOS DE PRUEBA

Los casos de prueba son conjuntos de entradas, condiciones de ejecución y resultados esperados. Se diseñan con distintos enfoques:

- Funcional (Caja Negra): se evalúan las entradas y salidas sin ver la implementación interna.
- Estructural (Caja Blanca): se analizan los caminos internos del código.
- Aleatorio: se generan entradas al azar basadas en modelos estadísticos.





HERRAMIENTAS DE DEPURACIÓN

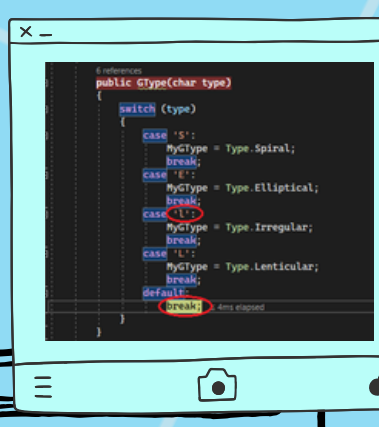
Permiten detectar y corregir errores en el software.

- Errores de compilación: ocurren por sintaxis incorrecta o referencias a variables inexistentes.
- Errores lógicos: no impiden la compilación, pero generan resultados incorrectos.

Los entornos de desarrollo incluyen depuradores que permiten:

- Supervisar la ejecución paso a paso.
- Analizar valores de variables en tiempo de ejecución.
- Identificar los puntos donde falla el programa.

Herramientas populares: Visual Studio Debugger, Chrome DevTools, PyCharm Debugger y Xcode Debugger.



VALIDACIONES

El cliente juega un papel clave en la validación del software. Se utilizan pruebas de caja negra para verificar que el producto final cumple los requisitos. En la validación se pueden dar dos resultados:

1. Aceptación: el software es aprobado por cumplir con los requisitos.
2. Rechazo: se identifican errores que requieren corrección.

NORMAS DE CALIDAD

Para garantizar la calidad del software, se siguen estándares reconocidos:

- BS 7925-1 y BS 7925-2: vocabulario y pruebas de componentes de software.
- IEEE 829 y IEEE 1008: documentación y pruebas de unidad.
- ISO/IEC 12207 y 15289: procesos y documentación de pruebas.
- ISO/IEC 29119: unificación de estándares para pruebas de software.