

## PRUEBAS DE SOFTWARE

La prueba del software es un elemento crítico para la garantía de la calidad del software. El objetivo de la etapa de pruebas es garantizar la calidad del producto desarrollado. Además, esta etapa implica:

- Verificar la interacción de componentes.
- Verificar la integración adecuada de los componentes.
- Verificar que todos los requisitos se han implementado correctamente.
- Identificar y asegurar que los defectos encontrados se han corregido antes de entregar el software al cliente.
- Diseñar pruebas que sistemáticamente saquen a la luz diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y esfuerzo.

La prueba es un proceso que se enfoca sobre la lógica interna del software y las funciones externas. La prueba es un proceso de ejecución de un programa con la intención de descubrir un error. Un buen caso de prueba es aquel que tiene alta probabilidad de mostrar un error no descubierto hasta entonces. Una prueba tiene éxito si descubre un error no detectado hasta entonces.

## TECNICAS DE PRUEBA DE SOFTWARE

Una vez generado el código fuente, es necesario probar el software para descubrir (y corregir) la mayor cantidad de errores posible antes de entregarlo al cliente. Su objetivo es diseñar una serie de casos de prueba que tengan una alta probabilidad de encontrar errores. Aquí es donde entran las técnicas de prueba del software. Estas técnicas proporcionan directrices sistemáticas para pruebas de diseño que 1) comprueben la lógica interna y las interfaces de todo componente del software y 2) comprueben los dominios de entrada y salida del programa para descubrir errores en su función, comportamiento y desempeño.

Durante las etapas iniciales del proceso, el desarrollador realiza todas las pruebas. Sin embargo, a medida que avanza este proceso se irán incorporando especialistas en pruebas.

Las pruebas de caja negra son las que se aplican a la interfaz del software. Una prueba de este tipo examina algún aspecto funcional de un sistema que tiene poca relación con la estructura interna del software. La prueba de caja blanca del software se basa en un examen cercano al detalle procedimental. Se prueban rutas lógicas del software y la colaboración entre componentes, al proporcionar casos de prueba que ejerciten conjuntos específicos de condiciones, bucles o ambos.

### ❖ Prueba de caja blanca:

Permiten examinar la estructura interna del programa. Se diseñan casos de prueba para examinar la lógica del programa. Es un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental para derivar casos de prueba que garanticen que:

- Se ejercitan todos los caminos independientes de cada módulo.
- Se ejercitan todas las decisiones lógicas.
- Se ejecutan todos los bucles.
- Se ejecutan las estructuras de datos internas.

### ❖ Prueba de caja negra:

Las pruebas se llevan a cabo sobre la interfaz del software, y es completamente indiferente el comportamiento interno y la estructura del programa.

Los casos de prueba de la caja negra pretende demostrar que:

- Las funciones del software son operativas.
- La entrada se acepta de forma adecuada.
- Se produce una salida correcta, y
- La integridad de la información externa se mantiene.

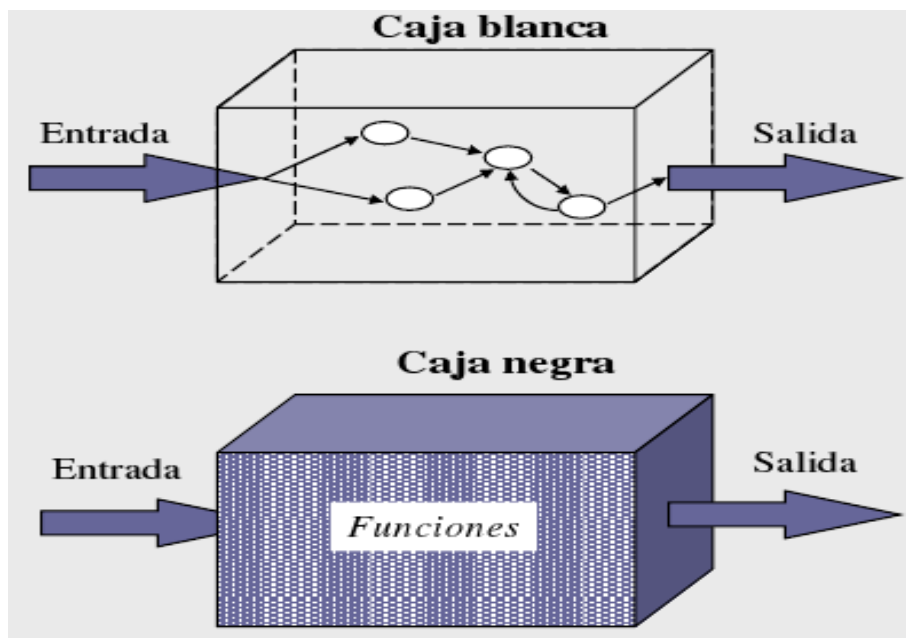
Se derivan conjuntos de condiciones de entrada que ejerciten completamente todos los requerimientos funcionales del programa.

La prueba de la caja negra intenta encontrar errores de las siguientes categorías:

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y de terminación.

Los casos de prueba deben satisfacer los siguientes criterios:

- Reducir, en un coeficiente que es mayor que uno, el número de casos de prueba adicionales.
- Que digan algo sobre la presencia o ausencia de clases de errores.



#### a. Técnicas de Diseño de Casos de Pruebas

## ESTRATEGIAS DE PRUEBA DEL SOFTWARE

Una estrategia de prueba del software integra los métodos de diseño de caso de pruebas del software en una serie bien planeada de pasos que desembocará en la eficaz construcción del software. La estrategia proporciona un mapa que describe los pasos que se darán como parte de la prueba, indica cuándo se planean y cuándo se dan estos pasos, además de cuánto esfuerzo, tiempo y recursos consumirán. Por tanto, en cualquier estrategia de prueba debe incorporar la planeación de pruebas, el diseño de casos de pruebas, la ejecución de pruebas y la recolección y evaluación de los datos resultantes.

Una estrategia de prueba del software debe ser lo suficientemente flexible como para promover un enfoque personalizado. Al mismo tiempo, debe ser lo adecuadamente rígido como para promover una planeación razonable y un seguimiento administrativo del avance del proyecto.

Si se considera el proceso desde un punto de vista procedimental, la prueba consiste en una serie de pasos que implementan de manera secuencial. Estos pasos se describen a continuación:

### ➤ **Pruebas de unidad:**

La prueba de unidad se centra en el módulo. Usando la descripción del diseño detallado como guía, se prueban los caminos de control importantes con el fin de descubrir errores dentro del ámbito del módulo. La prueba de unidad hace uso intensivo de las técnicas de prueba de caja blanca.

### ➤ **Prueba de integración:**

El objetivo es coger los módulos probados en la prueba de unidad y construir una estructura de programa que esté de acuerdo con lo que dicta el diseño.

Hay dos formas de integración:

- Integración no incremental: Se combinan todos los módulos por anticipado y se prueba todo el programa en conjunto.
- Integración incremental: El programa se construye y se prueba en pequeños segmentos.

En la prueba de integración el foco de atención es el diseño y la construcción de la arquitectura del software.

### ➤ **Prueba de Arquitectura y Aplicaciones**

La arquitectura cliente/servidor representa un importante desafío para quienes prueban el software. La naturaleza distribuida de los entornos cliente/servidor, los aspectos de desempeño relacionados con el proceso de transacción, la posible presencia de varias plataformas de hardware diferentes, la complejidad de la comunicación en red, la necesidad de servir varios clientes desde una base de datos centralizada (o, en algunos casos, distribuida) y los requisitos de coordinación impuestos al servidor se combinan para que la prueba de las arquitecturas de software cliente/servidor resulte considerablemente más difícil que la prueba de aplicaciones independientes.

Pruebas de funcionalidad de la aplicación: la aplicación se prueba de manera independiente.

- **Pruebas de servidor:** se prueban funciones de coordinación y manejo de datos del servidor. También se considera el desempeño del servidor (tiempo de respuesta y procesamiento de los datos).
- **Pruebas de base de datos:** se prueba la exactitud e integridad de los datos almacenados en el servidor. Se examinan las transacciones que realizaron las aplicaciones de cliente para asegurar que los datos se almacenan, actualizan y recuperan apropiadamente. También se prueba la función de archivado.
- **Pruebas de transacción:** se crea una serie de pruebas para asegurar que cada clase de transacciones se procesa de acuerdo con sus requisitos. Las pruebas se concentran en determinar si es correcto el procesamiento y en aspectos de desempeño.
- **Pruebas de comunicaciones de red:** con estas pruebas se verifica que la comunicación entre los nodos de la red ocurre de manera correcta y que el paso de mensajes, transacciones y el tráfico de la red relacionado se realiza sin errores. También es posible realizar pruebas de seguridad de la red como parte de estas pruebas.

➤ **Prueba del sistema:**

Verifica que cada elemento encaja de forma adecuada y que se alcanza la funcionalidad y el rendimiento del sistema total. La prueba del sistema está constituida por una serie de pruebas diferentes cuyo propósito primordial es ejercitar profundamente el sistema basado en computadora. Algunas de estas pruebas son:

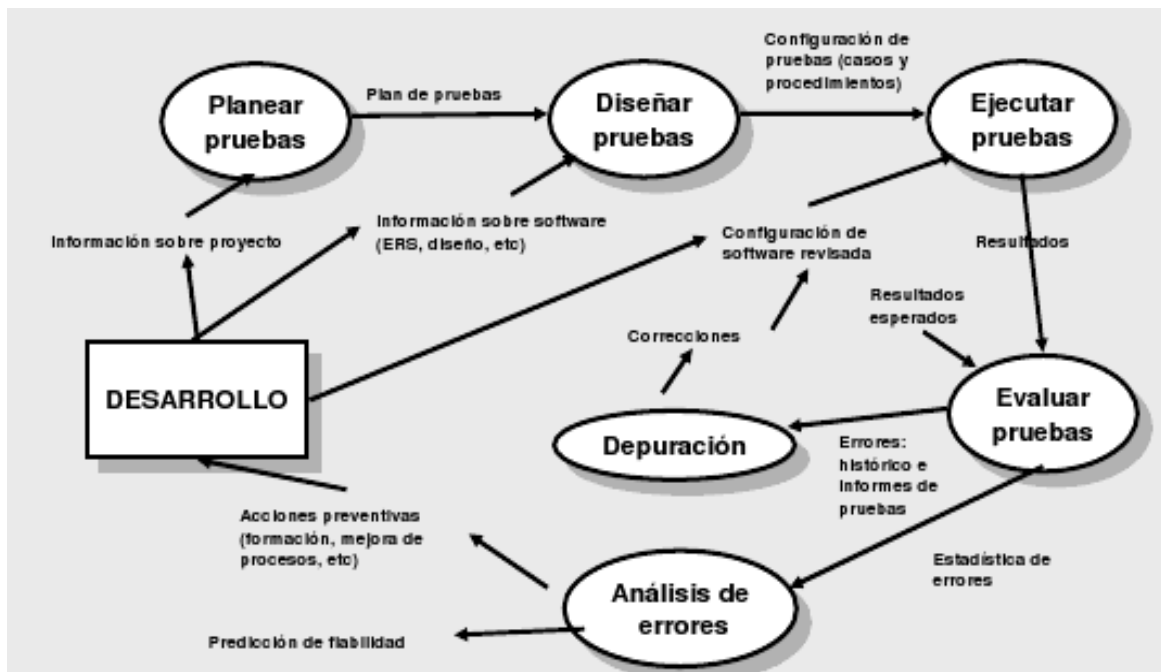
- Prueba de validación: Proporciona una seguridad final de que el software satisface todos los requerimientos funcionales y de rendimiento. Además, valida los requerimientos establecidos comparándolos con el sistema que ha sido construido. Durante la validación se usan exclusivamente técnicas de prueba de caja negra.
- Prueba de recuperación: Fuerza un fallo del software y verifica que la recuperación se lleva a cabo apropiadamente.
- Prueba de seguridad: Verificar los mecanismos de protección.
- Prueba de resistencia: Enfrenta a los programas a situaciones anormales.
- Prueba de rendimiento: Prueba el rendimiento del software en tiempo de ejecución.
- Prueba de instalación: Se centra en asegurar que el sistema software desarrollado se puede instalar en diferentes configuraciones hardware y software y bajo condiciones excepciones, por ejemplo con espacio de disco insuficiente o continuas interrupciones.

➤ **Pruebas de regresión:**

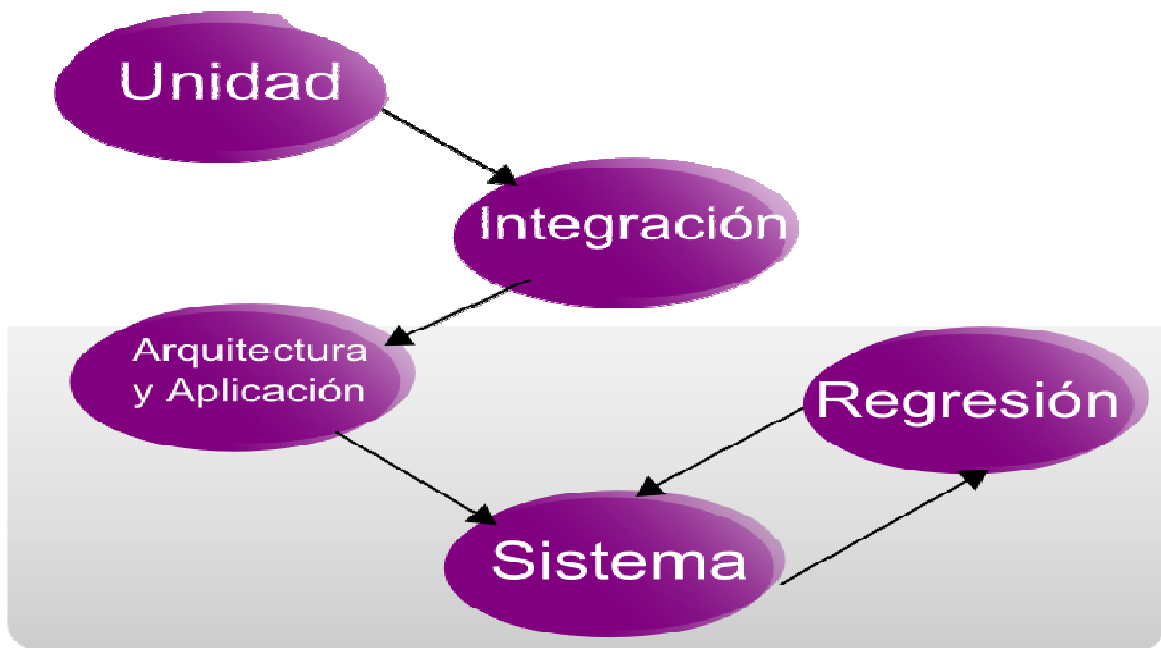
Las pruebas de regresión son una estrategia de prueba en la cual las pruebas que se han ejecutado anteriormente se vuelven a realizar en la nueva versión modificada, para asegurar la calidad después de añadir la nueva funcionalidad. El propósito de estas pruebas es asegurar que:

- Los defectos identificados en la ejecución anterior de la prueba se ha corregido.
- Los cambios realizados no han introducido nuevos defectos o reintroducido defectos anteriores.

La prueba de regresión puede implicar la re-ejecución de cualquier tipo de prueba. Normalmente, las pruebas de regresión se llevan a cabo durante cada iteración, ejecutando otra vez las pruebas de la iteración anterior.



b. Flujo del Proceso de Prueba



c. Estrategias de Prueba de Software