Ingeniería de Software

Unidad Nº

Testing

Juan José Vanzetti

Tipos de Pruebas

- Es un grupo de actividades de prueba dirigidas a probar un componente o un sistema con un objetivo de prueba específico.
- Los tipos de pruebas buscan evaluar alguna de los atributos de calidad del producto de software.
  - Usabilidad
  - Mantenimiento
  - Funcionales
  - Confiabilidad
  - Performance
  - Escalabilidad
  - Carga
  - Utilización de recursos
  - Estrés
  - Instalación
  - Portabilidad

Pruebas Funcionales

- Verificar la capacidad del producto de software para proveer funciones que cumplan con las necesidades, establecidas e implícitas, cuando el software es utilizado en condiciones normales a cualquier nivel del sistema.
- Las pruebas se realizan en base al análisis de la especificación de la funcionalidad del componente o sistema.
- Las pruebas consisten en accionar(incluso de forma inválida) con el objeto de prueba y verificar que las respuestas obtenidas del mismo sean las correctas.
- Evalúan las siguientes características del objeto de prueba:
  - **Precisión:** las respuestas esperadas cumplan el grado de precisión necesario.
  - **Interoperabilidad:** pueda interactuar con uno o más componentes o sistemas necesarios.
  - **Seguridad:** impida el acceso no autorizado, accidental o deliberado, a funcionalidad o datos del mismo.
  - **Adecuación:** provea el conjunto apropiado de funcionalidad para la tarea especificada y los objetivos de su usuario.

Pruebas de Confiabilidad

- Tienen por objetivo medir la madurez del software durante eltiempo y compararla con la confiabilidad deseada.
- Las tomadas incluyen tiempo medio entre fallas (MTBF) y tiempo medio de reparación(MTTR)
- Las pruebas toman un tiempo considerable para que sean significativas y suelen continuarse en producción.
- Estas pruebas también se especifican en términos de perfiles operacionales
- **Prueba de robustez**: Orientadas a evaluar la tolerancia de un componente o sistema a fallas que ocurren fuera del objeto de prueba.
- **Prueba de recuperación:** Orientadas a evaluar la habilidad del sistema de software a recuperarse de fallas en hardware o software del modo predeterminado que permite reanudar las operaciones normales.

Pruebas de Performance

- Pruebas enfocadas en la habilidad del componente o sistema de responder al usuario o sistema dentro del tiempo establecido y bajo las condiciones establecidas.
- Las medidas pueden variar de ciclos de CPU a tiempo de respuesta.

Pruebas de Carga

- Enfocadas en la habilidad del sistema de manejar crecientes niveles de carga real y previsible resultante de pedidos de transaciones generadas por usuarios en simultaneo.
- Las pruebas pueden conducirse con un número realista de ususarios(multi-usuario) o con una gran cantidad de ellos (volumen)
- Las medidas deben realizarse sobre tiempo de respuesta promedio y tasa de transferencia de información.
- Fuertes cargas de trabajo como tráfico excesivo o cargas elevadas de transacciones simultáneas.
- Tráfico de red artificial.
- Múltiples usuarios simultáneos.
- Generar transacciones con herramientas de automatización.

Pruebas de saturación o Estrés

- Enfocadas en la habilidad del sistema de manejar picos de carga en el límite o superiores de su capacidad máxima.
- La performance del sistema debe degradarse lentamente y de forma predecible sin fallas a medida que los niveles de estrés aumentan.
- Las pruebas deben verificar que el sistema no pierda su integridad funcional mientras este se encuentra en condiciones de stress.
- Encontrar errores debidos a la escasez de recursos
- Falta de memoria
- Falta de espacio en disco
- Encontrar errores debidos a la existencia de recursos compartidos
- Recursos del sistema
- Bloqueos de la base de datos
- Ancho de banda de la red

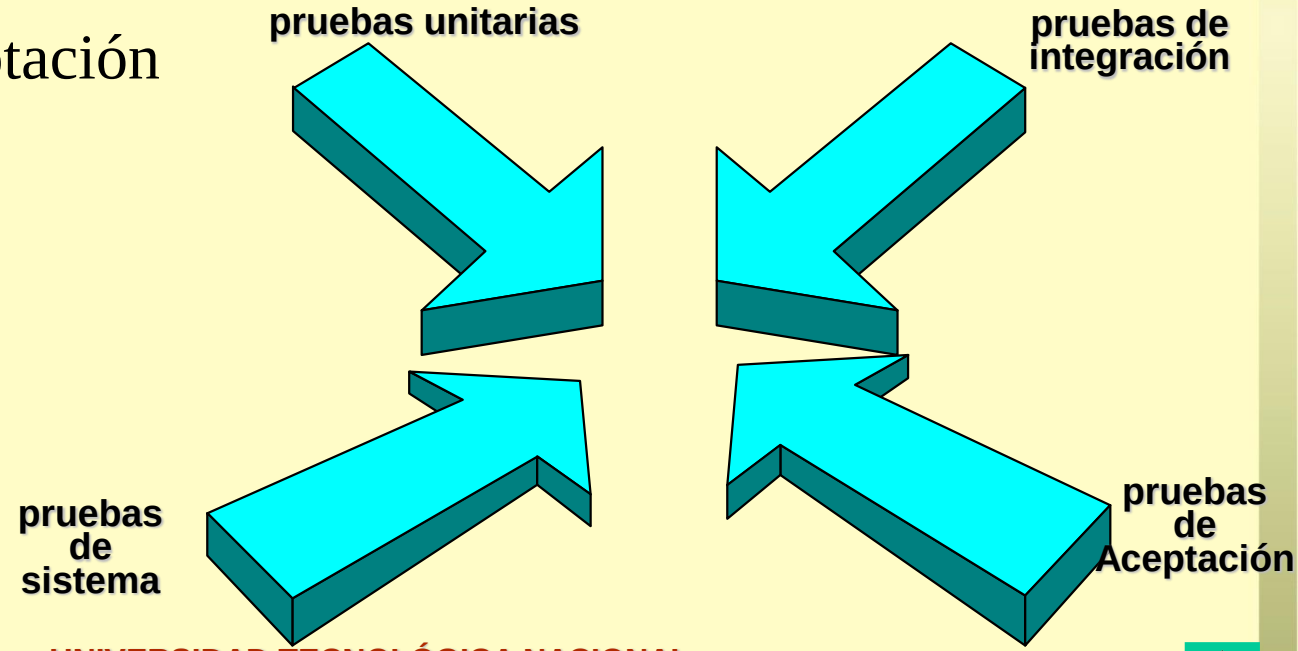
VERIFICACIÓN *¿Estamos construyendo el sistema correctamente?*

- Evalúa un componente al final de cada fase del desarrollo del software.
- Se realiza durante cada fase el desarrollo de sw.

VALIDACIÓN *¿Estamos construyendo el sistema correcto ?*

- Típicamente incluye el testeo, y tiene lugar luego que la VERIFICACIONES han sido completadas.
- Ocurre al final con las pruebas de aceptación del cliente.

- Cuatro niveles
  - Pruebas Unitarias
  - Pruebas de Integración
  - Pruebas de Sistema
  - Pruebas de Aceptación



Pruebas de Aceptación del Usuario

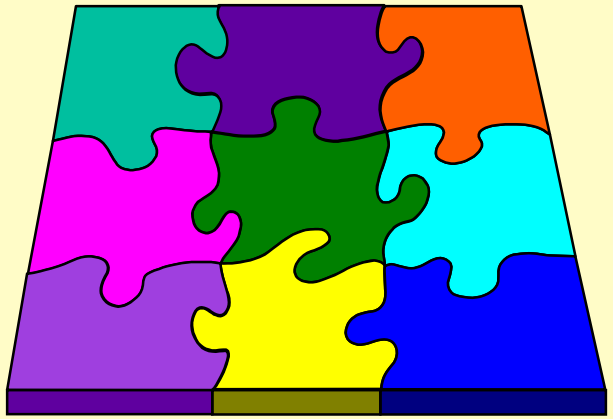
- Asegura que el sistema cumpla con los requerimientos de negocio y, consecuentemente, que la lógica del mismo funcione correctamente.

Testing de Sistema

- Proceso de verificar que un sistema integrado cumple con los requerimientos especificados.
- Puede estar basado en :
  - **Requerimientos** :surge del documento de requisitos(conviene dedicar tiempo suficiente para que las pruebas aseguren la inclusión correcta)
  - **Procesos de Negocio:** basados en escenarios de negocio, definidos por el usuario (debería acercarse a su uso real)

Test de Integración

- Procura exponer fallas en la integración a nivel de módulos o fallas en las interfaces y la interacción entre sistemas.
- Tiene en cuenta :
  - La interacción aumenta la complejidad.
  - Convivencia entre los módulos.
  - ¿Interfiere con otros sistemas?
  - ¿Cómo respondería en un entorno de Producción real ?
  - ¿Qué plataformas soporta ? ¿Cómo responde en cada una ? ¿Cómo se comunican entre ellas ?
  - Considerar la coordinación de cada cambio ?



Testing de Componentes

- Testear en forma individual componentes de SW utilizando herramientas en los casos que existan (JUnit, VBUnit ,Simple Test, CPPUnit, NUnit, etc.)
- Se entiende por componente a un ítem de SW para el cual existe una especificación.
- Asegura que un componente funciona como está especificado.

¿Cómo se hacen las pruebas habitualmente ?

- Ad-hoc (no planificado)
- Proceso caótico no repetible
- No midiendo exhaustivamente
- Como etapa final del proceso de desarrollo
- Los usuarios detectan los defectos cuando el sistema que se encuentra en producción.
- Muchas veces el sistema puede no cumplir con los requerimientos.

¿Cómo deben hacerse las **PRUEBAS**?

- Evitar pruebas espontáneas y que no dejan registros**…** **Es una pérdida de tiempo.**
- Sobre la base de una metodología formal fácilmente repetible en múltiples circunstancias
- Proveyendo un encuadre formal para todas las actividades de pruebas
- Midiendo exhaustivamente
- Ejecutándolo en paralelo al ciclo de desarrollo de sistemas, empezando lo más tempranamente posible.

¿Qué pasa si NO hay pruebas sistemáticas?

- No hay una identificación adecuada de los requerimientos de pruebas.
- Probable indefinición respecto de cuán completas son pruebas.
- Desconocimiento de la tasa actual de errores.
- **NO PODEMOS DECIR CUANDO LIBERAR UN PRODUCTO**

Requerimiento de Prueba:

- Un aspecto o un evento de un componente o sistema que pudiera ser verificado por uno o más casos de pruebas.
- Ej. Función, transacción, característica, o elementos estructurales.

¿Qué debe contener una Metodología de Pruebas ?

- Identificación de requerimientos de Prueba
- Planeamiento de la prueba
- Ejecución de la prueba
- Clasificación de los errores
- Mediciones
- Seguimiento de los errores
- Administración del proyecto de pruebas

Prueba habitual vs. Prueba sistemática

- En la prueba habitual , el objetivo dominante es demostrar que el sistema funciona. **“El camino Feliz ”**
- La prueba sistemática, por el contrario, se focaliza en la detección e identificación de defectos del sistema

Las pruebas sistemáticas requieren :

- **Independencia;** El Tester debe estar comprometido con la búsqueda de errores y no con la defensa del producto.
- **Idoneidad;** El Tester debe ser un profesional preparado en las funciones de probar y con los conocimientos necesarios de la metodología de pruebas y de las técnicas y métodos de prueba.

Caso de Prueba

- Es un conjunto de valores de entrada, precondiciones de ejecución, resultados esperados y pos condiciones de ejecución, desarrollados para con un objetivo en particular, tal como ejercitar un camino particular de un programa o verificar el cumplimiento de un requerimiento específico.

Precondición

Condiciones de entorno y de estado, las cuales deben ser cumplidas antes de que el componente o sistema puede ser ejecutado con un valor particular.

- **Datos de Prueba**
  - Datos que deben existir antes de que una prueba pueda ser ejecutada y que afectan o son afectados por el componente o sistema bajo prueba.
  - Ej. Una base de datos conteniendo instancias específicas de una entidad.
- **Entorno de prueba**
  - Elementos diferentes del objeto de prueba que son necesarios para conducir la prueba
  - Ej. Hardware, Herramienta de Testing, etc.

Entrada de Prueba

- El dato recibido por un objeto a prueba, durante la ejecución de la prueba desde una fuente externa
- La Fuente externa puede ser HW, SW o una acción humana.

Resultado Esperado

- El Comportamiento predicho la especificación u otra fuente, del objeto de prueba, bajo condiciones especificas.
- Se debe también especificar cómo se obtiene el resultado esperado.

Oráculo

- Es una fuente para determinar resultados esperados de un caso de prueba.
- Un oráculo puede ser un sistema existente, un usuario manual , o el conocimiento especializado de un individuo.
- Nunca debería ser el código
- Ej. Simulación, utilización de una hoja de cálculo, experto de dominio, etc.

Criterio para el Veredicto pasa /falla

- Reglas de decisión usadas para determinar si un ítem de prueba ha pasado o fallado una prueba

Pos condición

- Condiciones del entorno y estado que se deben cumplimentar después de la ejecución
- de una prueba o un caso de prueba, o un procedimiento de prueba.

Caso de prueba a alto nivel

- Un caso de prueba que no tiene una implementación concreta, o valores concretos para la entrada de datos y el resultado esperado
- Se usa Caso de Prueba Lógico como sinónimo

Caso de prueba a bajo nivel

- Es un caso de Prueba con valores concretos, que existen a nivel de implementación, para dato de entrada y resultado esperado.
- Los operadores lógicos de los casos de prueba a alto nivel son reemplazados por valores reales que corresponden a el objetivo de los operadores lógicos.

Nivel de detalle de especificación de un caso de prueba

- Es una decisión importante que se debe tomar cuando se piensa documentar casos de pruebas.
- Qué tan preciso tiene que ser especificadas las acciones, los datos, el resultado esperado, etc.?
- Mejora la reproducibilidad ya que nada es dejado al juicio de un tester en particular.
- Permite que los casos de prueba pueden ser revisados por desarrollo
- Implica mucho más trabajo en la especificación
- Permite que testers no expertos ejecuten los casos de prueba.

Resultado obtenido

- Es el comportamiento producido/observado de un objeto de prueba como resultado del procesamiento de entradas de prueba.
- Esto no es parte del caso de prueba, sino el resultado de la ejecución de la prueba, sobre una versión específica del objeto a prueba.
- En base a este resultado se da el veredicto de si la prueba pasó o falló.

Caso de Prueba

"Los bugs se esconden en las esquinas y se congregan en los límites..." Boris Beizer

OBJETIVO descubrir errores

CRITERIO en forma completa

RESTRINCCIÓN con el mínimo de esfuerzo y tiempo

Conjunto de pruebas ideal:

- Es el menor subconjunto de todos los casos de prueba posibles que encuentra todos los defectos sobre un componente o sistema.
- En general, es imposible definir un conjunto ideal de casos de pruebas y pero se trata de aproximar aplicando criterios de selección de pruebas.
- El criterio de selección de las pruebas trata de aproximar esta noción, eligiendo el subconjunto de comportamientos a probar.

Prueba exitosa:

- Aquella que detecta un defecto aún no descubierto.
- Es aquella que encuentra muchos defectos, no lo opuesto.
- Un producto sin defectos daría como resultado que testing no produzca pruebas exitosas y el consiguiente fracaso de testing.

Característica de un buen caso de prueba

- Fácil de repetir
- Teniendo un fijo y preciso punto de inicio. El estado del sistema cuando se introdujo la entrada es conocido.
- Conociendo exactamente que datos de entrada fueron introducidos y en que secuencia.
- Representativo
- Testing exhaustivo no es factible, con lo cual el caso de prueba debería ser representativo de un conjunto posible de casos de prueba.
- Ayuda a identificar fácilmente cual es el defecto.
- Característica de un buen caso de prueba
- Fácil de usar entender o de realizar
- Fácil de mantener
- Trazable

Prueba o conjunto de pruebas

- Un conjunto de uno o más casos de prueba
- Suite de Prueba
- Un conjunto de varios casos de prueba para un componente o sistema bajo prueba, donde la pos condición de una prueba es frecuentemente usada como la precondición del próximo caso de prueba.
- Es usada para agrupar casos de prueba similares.

Procedimiento de Prueba

- Un documento especificando una secuencia de acciones para la ejecución de una prueba. También conocido como script de prueba o script manual.
- Una descripción de cómo una prueba es realizada. Contiene acciones, verificaciones (checks) y casos de prueba relacionados e indica la secuencia de ejecución.

Derivación de Casos de Prueba

- En base a documentos del cliente
- En base a documentos de relevamiento
- En base a casos de uso
- En base a especificaciones de programación
- En base a código

En base a documentos del cliente

- Útil para el diseño de casos de prueba de funciones de negocio.
- Aconsejable para las pruebas de aceptación de usuario.
- Puede ser muy pobre para un testing funcional.

En base a documentos de relevamiento

- Si se detectan omisiones o supuestos equivocados en el relevamiento, se debe informar para corregir.
- El diseño de casos de pruebas generado de estos documentos tiene que servir al testing y no hay que confundir con el diseño del desarrollo.

En base a especificaciones de programación

- En área de Testing recibe las mismas especificaciones que se le envían al programador.
- Si un programador puede codificar en base a estas especificaciones, Testing debe poder diseñar casos de prueba y testear.
- Más completo que el diseño que se logra en base a documentos del cliente y de relevamiento.
- Más detalle
- Se puede lograr un diseño de casos de prueba de amplia cobertura.
- Sirve para el testing funcional.
- Si las especificaciones incluyen un prototipo de pantalla, se pueden diseñar casos para pruebas de usabilidad.

En base a código

- En Testing se recibe el CÓDIGO.
- Adquiere las mismas ventajas y desventajas que esta técnica tiene.
- Es complementario al diseño de casos de prueba en base a documentación.
- Amplia cobertura del código y por supuesto del diseño.
- Se usa para el diseño de pruebas unitarias o de componentes.
- Permite completar los casos de prueba para asegurar una mayor cobertura.

Técnicas de diseño de pruebas

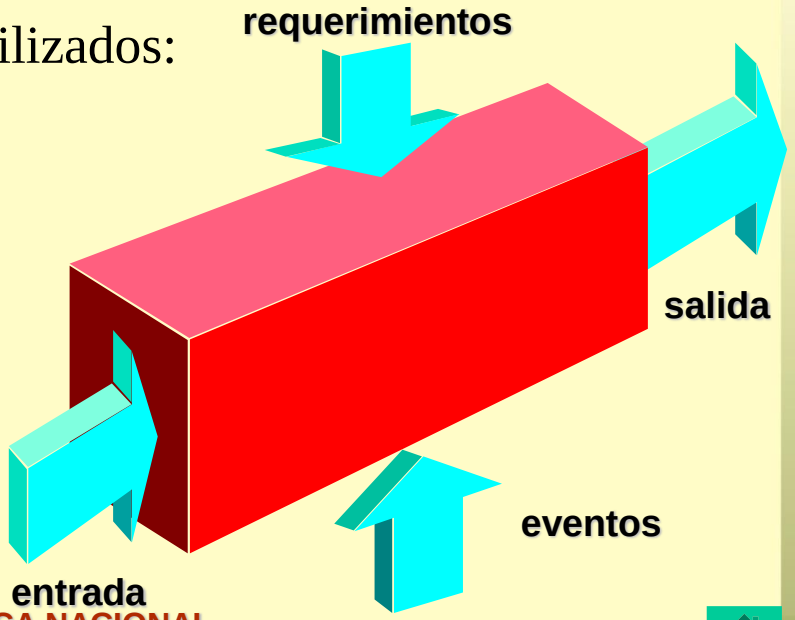
- Métodos estandarizados para la derivación de casos de prueba.
- Fundamentales para el desarrollo del testing metodológico y profesional.
- Diferentes técnicas se enfocan en encontrar diferentes tipos de defectos
- Es necesario usar una combinación de técnicas para prevenir y encontrar defectos.
- Facilitan comprender la calidad y cobertura de las pruebas

Enfoque recomendado

1. Diseñar un conjunto inicial de casos de pruebas con técnicas de caja negra
2. Medir cobertura lógica lograda por las pruebas
3. Diseñar más casos de prueba hasta lograr la cobertura lógica deseada
4. Medir la cobertura estructural lograda sobre el objeto de prueba
5. Diseñar más casos de prueba hasta lograr la cobertura estructural deseada

Caja negra:

- Basada en la definición de requerimientos o una descripción funcional de la aplicación bajo prueba
- Las pruebas se conducen a nivel de la interfaz, sin mayor interés por los detalles internos de estructura lógica
- Interesa "el qué" realiza el programa , no "el cómo ". El énfasis está en el control de las salidas teniendo en cuenta las entradas
- Los siguientes métodos son comúnmente utilizados:
  - Particionamiento de equivalencias
  - Análisis de valores frontera
  - Adivinanza de defectos
- Los siguientes son menos utilizados:
  - Diagrama de causa-efecto
  - Pruebas de transición de estado



Estas pruebas intentan encontrar errores de las siguientes categorías:

1- Funciones incorrectas o ausentes

2- Errores de interfaz

3- Errores en estructuras de datos o en el acceso de bases de datos

4- Errores de rendimiento

5- Errores de iniciación y terminación

Pruebas de valores limites

- Si un condición de entrada esta en un rango de valores entre A y B, se deben de diseñar pruebas para los limites A y B y para los valores dentro de los limites y por encima de estos.

Pruebas de valores limites

- Si un condición de entrada especifica un numero especifico de valores, se deben de diseñar pruebas que ejerciten los valores máximo y mínimo, y los valores próximo por encima y próximo por debajo del máximo y del minino de los valores

Técnica Caja Blanca:

- También llamadas Técnicas Estructurales
- Basadas en un conocimiento del código, especificaciones u otras fuentes de material
- Pretende un examen en detalle de los procedimientos internos ejecutando cambios lógicos con condiciones o bucle específicos
- Utiliza la estructura de control del diseño procedural para derivar casos de prueba que:
  - Garanticen que todos los caminos independientes dentro de un módulo han sido ejercitados por lo menos una vez
  - Ejerciten todas las decisiones lógicas en sus lados verdaderos y falsos
  - Ejerciten todos los bucles en sus fronteras y dentro de sus límites operacionales
  - Ejerciten sus estructuras de datos internas para asegurar su validez

Razones para hacer Testing de Caja Blanca:

- Los errores lógicos y supuestos incorrectos son inversamente proporcionales a la probabilidad de ejecutar un camino en un programa.
- A menudo se cree que un camino lógico probablemente no se ejecutará cuando de hecho puede ser ejecutado en forma regular.
- Los errores tipográficos son aleatorios.

- Pruebas de estructura de control :
  - Pruebas de caminos básicos
  - Pruebas de condiciones
  - Pruebas de bucles

Prueba de caja blanca

Mediante los métodos de prueba de caja blanca se obtienen casos de prueba que:

1. Garantice que se ejercite por lo menos una vez todos los caminos independientes de cada modulo.
2. Ejercitan todas las decisiones lógicas en sus caminos VERDADERO Y FALSO

Prueba de caja blanca

3. Ejecución de todos los lazos en sus limites.
4. Ejecutar las estructuras internas de datos para asegurar su validez.