CURSO

SOFTWARE TESTING

UNIDAD 5 – Tipos de Pruebas

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

PRESENTACIÓN

En esta unidad veremos los distintos tipos de pruebas que pueden realizarse para determinar la existencia de defectos en un software

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

TEMARIO

1. – Clasificación de las pruebas
2. – Modelo V
3. – Pruebas complementarias.
4. – Ejemplos.

Tipos de Pruebas

Pruebas por niveles

- Se denominan así ya que, cada nivel de desarrollo tiene su correspondiente nivel de pruebas.

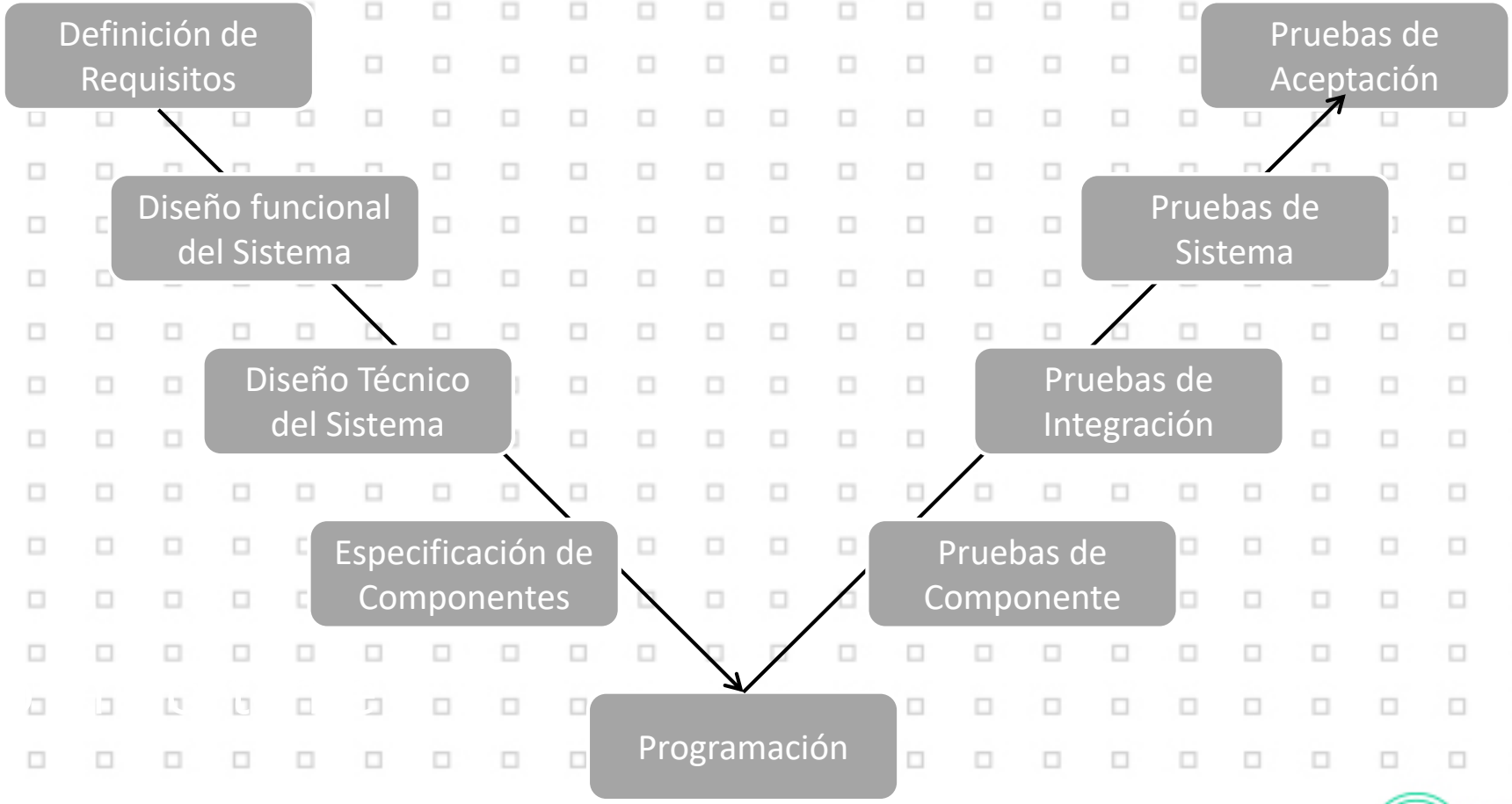
Actividades Analíticas y Constructivas

- **Verificación:**

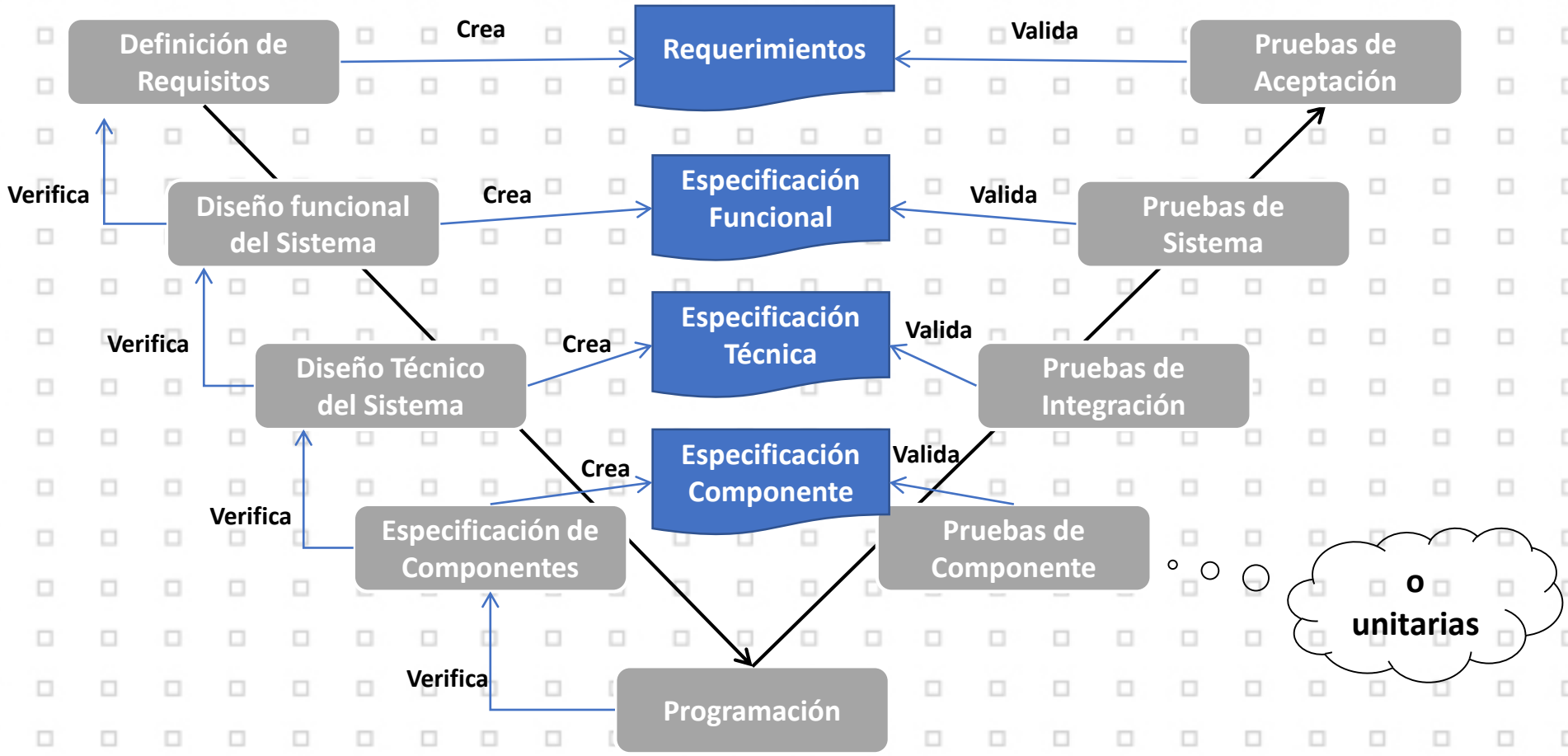
  Comprueba la conformidad con los requisitos establecidos.
  - Es el proceso de chequear que se cumple con los atributos de calidad esperados en todo el ciclo de vida
  - Valida que el *proceso* sea el esperado
  - Usa técnicas de testing estático
- **Validación:**

  Comprueba la idoneidad para el uso esperado.
  - Es el proceso de chequear que el software responde a los requerimientos de sistema especificados y resuelve el problema especificado correctamente.
  - Valida que el *producto* sea el esperado
  - Usa técnicas de testing dinámico

Ciclo de vida en V



Ciclo de vida en V



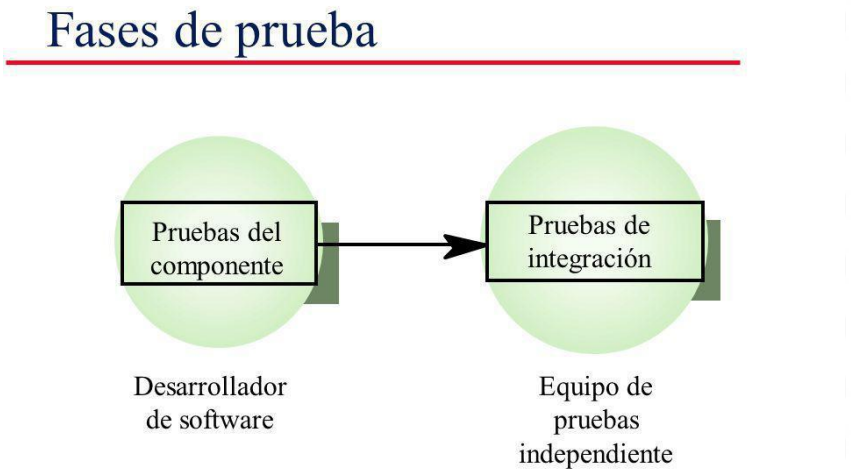
Pruebas de Componente (unitarias)

- Se realiza sobre una unidad de código claramente definida
  - Prueba la lógica interna del componente (por ejemplo una función)
  - Prueba el manejo de errores
  - Prueba que se cumpla la especificación
- Generalmente lo realiza el área que construyó el módulo
- Se basa en el diseño detallado
- Comienza una vez codificado, compilado y revisado el módulo
- Son pruebas de “Caja blanca”

Pruebas de Componente (unitarias)

El test debe ser:

- Atómico (Prueba la mínima funcionalidad posible):
  - Los Test que usen condiciones “if / else”, deben separarse en 2 casos
- Independiente
- Inocuo
- Repetible o Reutilizable
- Automatizable
- Rápido



Pruebas de Componente (unitarias)

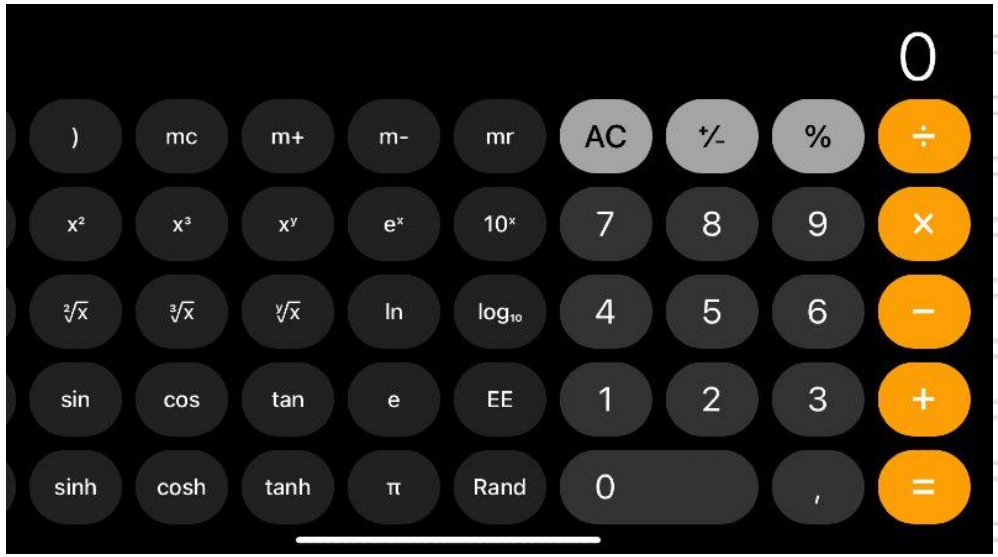
Ejemplo con una calculadora:

Componente: Elevar número al cuadrado.

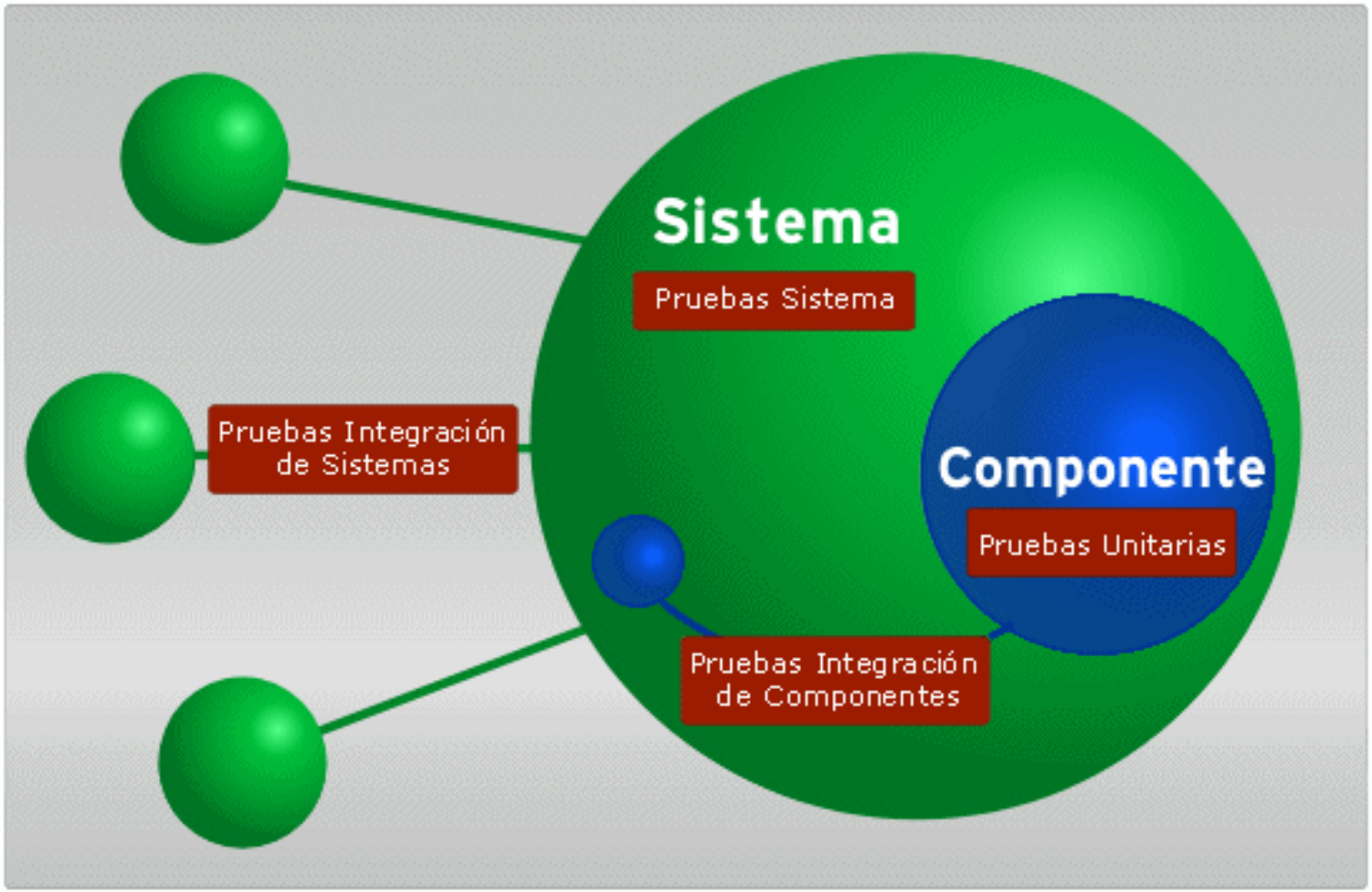
Se pasa un numero como parámetro de entrada y se espera que devuelva el numero elevado al cuadrado

Componente: Sumar 2 números.

Se pasan 2 números como parámetros de entrada y se espera que devuelva como resultado la suma de esos dos números.



Pruebas de Integración

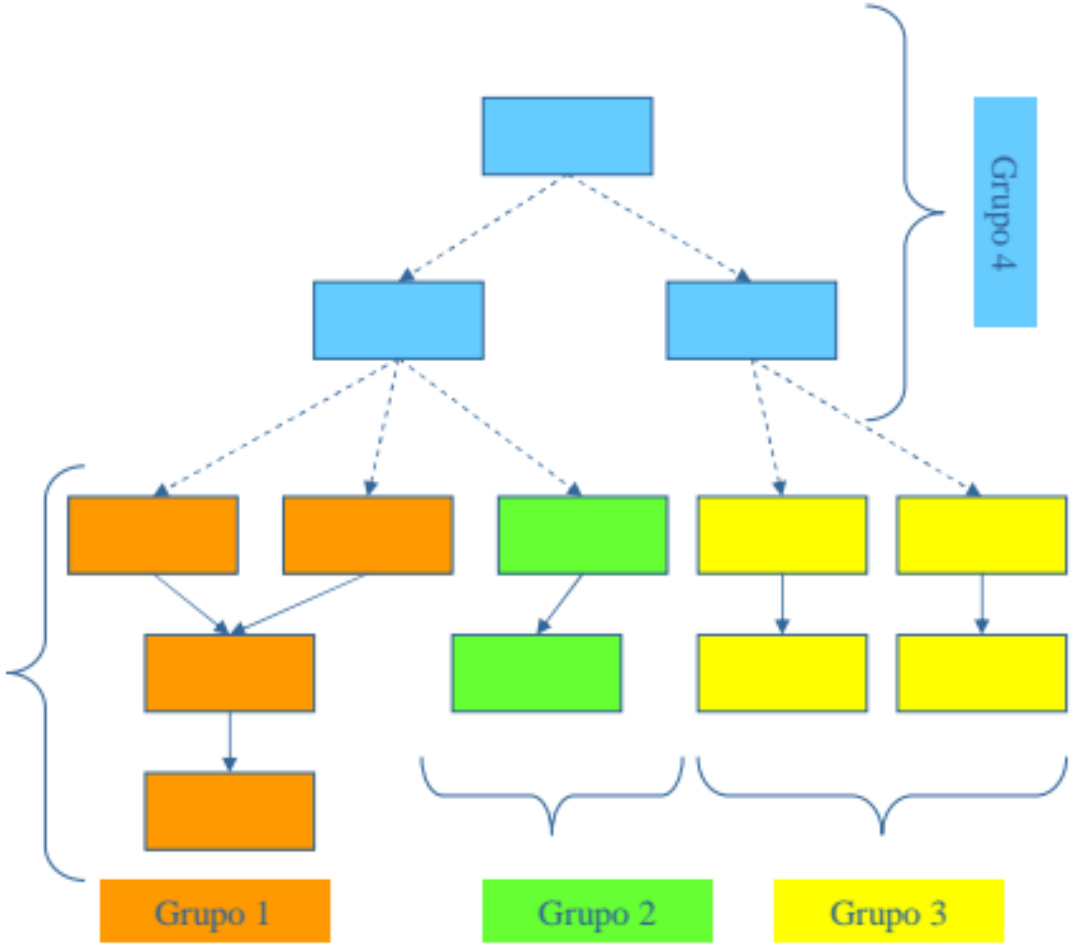


- Son pruebas realizadas con el objeto de poner en evidencia defectos en las interfaces e interacciones entre componentes o sistemas integrados
- Son de importancia crítica, ya que es en esta etapa del plan de QA donde el software empieza a acercarse a su forma final
- Prueba la integración de interfaces con funcionalidad
- Los puntos clave son:
  - Conectar de a poco las partes mas complejas
  - Minimizar la necesidad de programas auxiliares

Pruebas de Integración

- Están orientada a verificar que las partes de un sistema que funcionan bien aisladamente también lo hacen en su conjunto.

- TIPOS:
  - Incrementales
    - BOTTOM-UP
    - TOP-DOWN
  - No Incrementales
    - BIG BANG



Pruebas de Integración - Incrementales

- Los módulos que componen el software se van acoplando progresivamente en conjuntos.
- Luego de cada acoplamiento, se prueba la correcta interacción entre los módulos.
- Una vez que se haya verificado que el conjunto funciona de acuerdo con lo previsto, se le suma un nuevo módulo y se vuelven a realizar pruebas.
- Este proceso se repite hasta completar la aplicación.

Dato Importante

Cada conjunto, parcial o total, debe verificar los requerimientos funcionales, de rendimiento y de seguridad definidos

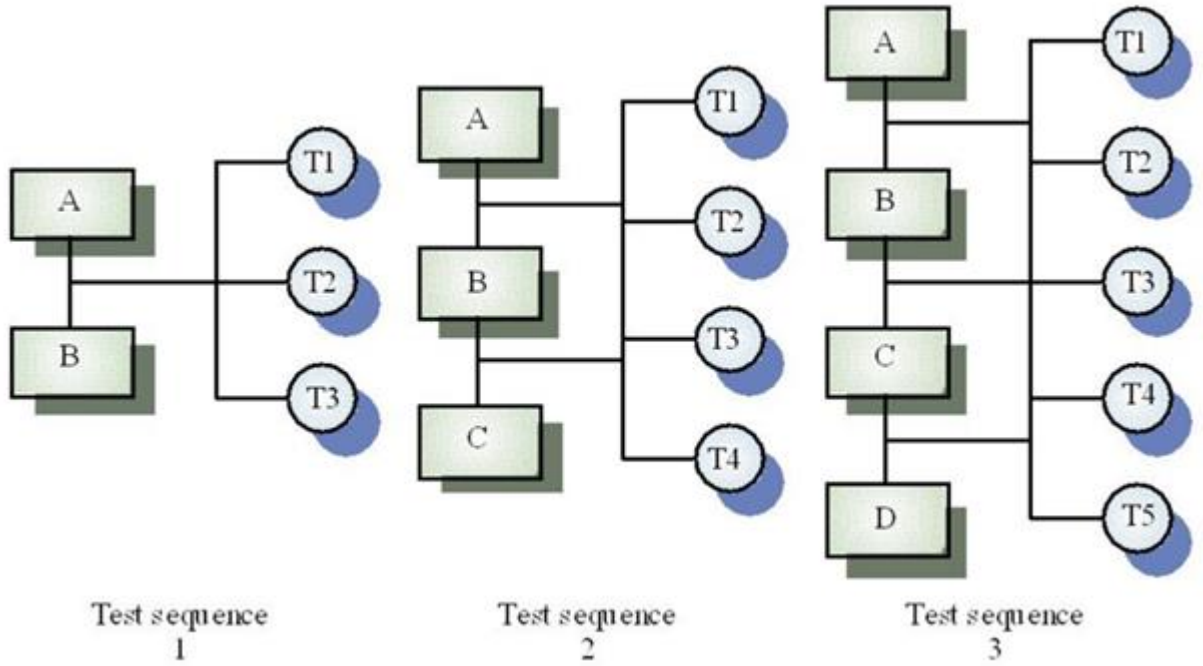
Pruebas de Integración - Incrementales

- BOTTOM-UP: Se prueban primero los componentes de bajo nivel y luego se avanza hacia los de mayor nivel.

  Se pueden utilizar ‘stub’ para simular un componente que todavía no está disponible.

- TOP-DOWN: El enfoque es exactamente inverso.

  Se pueden utilizar drivers para simular datos de entrada en caso de no tenerlos al momento de las pruebas.



Pruebas de Integración – No Incrementales

- BIG BANG: se acoplan todos los módulos de una sola vez
  - Se reduce la cantidad de pruebas
  - Se dificulta la detección de posibles errores
  - Recién se puede hacer en un momento avanzado del proyecto porque requiere que todos los módulos o sub-módulos del producto hayan sido desarrollados y probados de forma unitaria
  - Si el proyecto es de corto plazo no se tiene mucho tiempo para la aplicación de la prueba y realizar las correcciones

Pruebas de Integración

Ejemplo con una calculadora:

Componente1: Elevar número al cuadrado.

Componente2: Sumar 2 números.

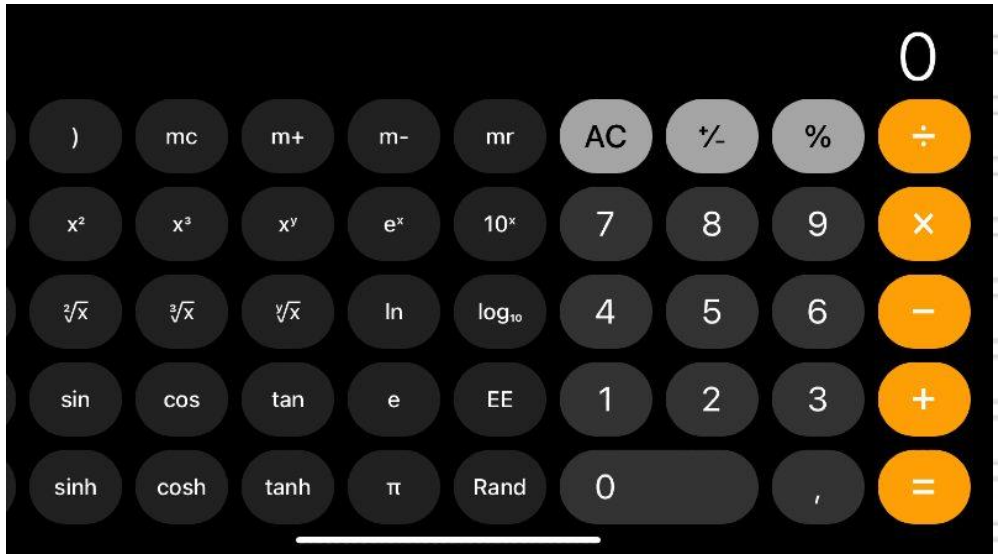
Testear:

Llamada a Componente2 con los parámetros:

Parametro1: 5^2

Parametro2: 8

Resultado esperado=32

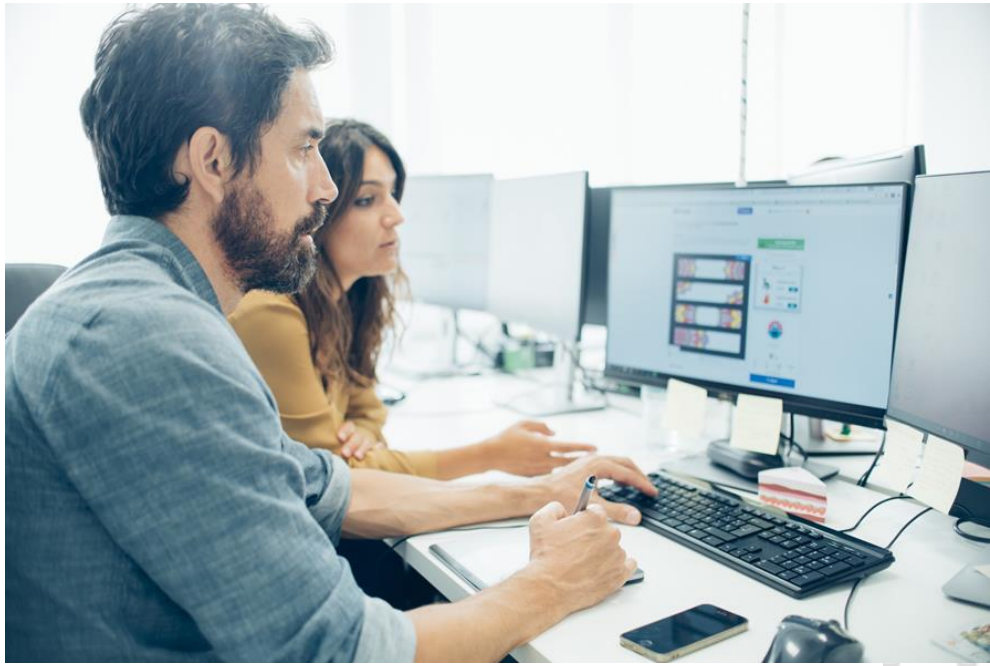


Pruebas de Sistema

- Tienen como propósito comparar el sistema o el programa contra los objetivos originales (requerimientos funcionales y no funcionales)
- Ejercita al sistema tal como lo haría un usuario.
- Puede ir de extremo a extremo de la aplicación.
- Para realizar estas pruebas es necesario contar con un ambiente de test, lo más similar al de producción que sea posible, donde se implemente el programa completo (tal como se haría en producción)
- Son como las pruebas de aceptación del usuario pero realizadas por el equipo de testing del proyecto.

Pruebas de Aceptación del Usuario

- UAT = User Acceptance Test
- Prueba realizada por los usuarios para verificar que el sistema se ajusta a sus requerimientos
  - Las condiciones de pruebas están basadas en el documento de requerimientos y están escritas como oraciones fáciles de comprender por personas sin conocimiento técnico.
  - Es una prueba de “caja negra”
- Es una prueba formal con el objetivo de lograr la aprobación del cliente.
- Las pruebas se realizan antes de que la aplicación sea instalada en un ambiente productivo.



Pruebas Complementarias

Además de las pruebas ya mencionadas durante el ciclo de vida en V, existen otras pruebas que se realizan con el fin de verificar algunos aspectos particulares del sistema.

- Pruebas de Carga (o volumen)

  Orientada a verificar a que el sistema soporta los volúmenes máximos *definidos* en la cuantificación de requerimientos.

  - Capacidad de Almacenamiento
  - Capacidad de Procesamiento
  - Cantidad de Transacciones
  - Cantidad de Usuarios

Pruebas Complementarias

- Pruebas de Rendimiento (o Performance)

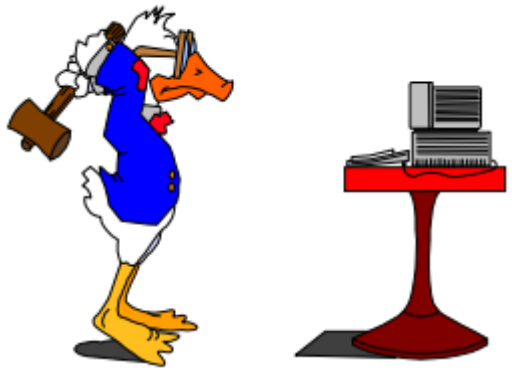
  Comprueba la rapidez con la que el sistema responde a una funcionalidad

- Pruebas de Stress

  Orientada a someter al sistema excediendo los límites de su capacidad de procesamiento y almacenamiento.
  - Teniendo en cuenta situaciones NO previstas originalmente

- Pruebas de Estabilidad

  Rapidez de respuesta trabajando en forma continuada largos períodos



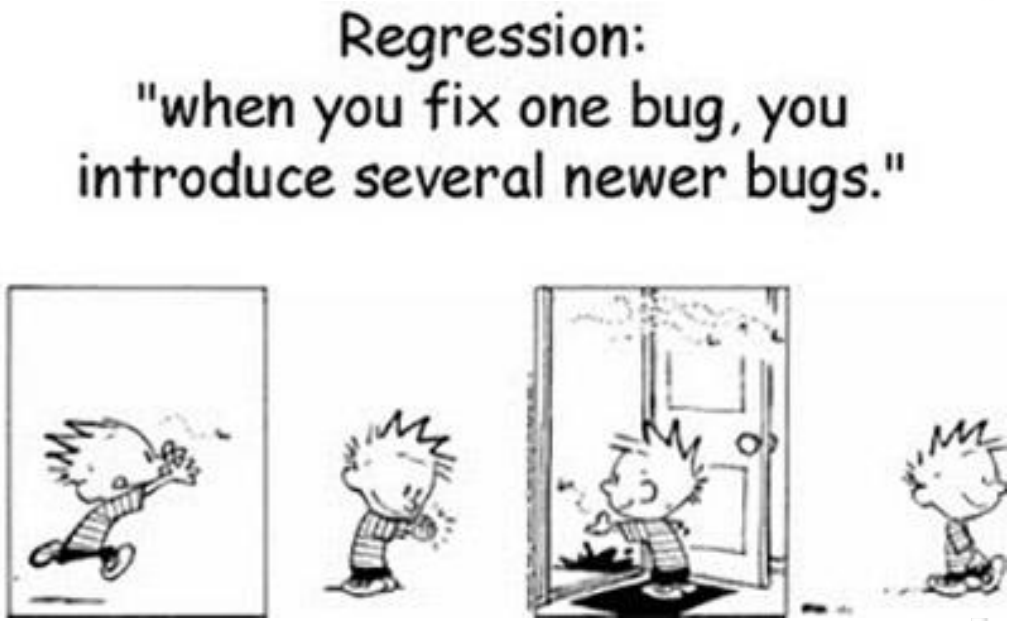
Pruebas Complementarias

- Pruebas de Robustez

  Evalúa reacción a datos de entrada erróneos
- Pruebas de cumplimiento

  Cumple las normas y estándares
- Pruebas de Regresión

  Volver a probar una funcionalidad que estaba correcta en la última reléase. Orientada a verificar que, luego de introducido un cambio en el código, la funcionalidad original no ha sido alterada.
  - Hay que probar “lo viejo”
  - Se re-utilizan condiciones y casos de prueba



Pruebas Complementarias

- Pruebas Alfa y Beta

  Se entrega una primera versión al usuario que se considera está lista para ser probada por ellos.

  - Normalmente plagada de defectos
  - Una forma económica de identificarlos (ya que el trabajo lo hace otro)
  - Alfa: la hace el usuario en mis instalaciones
  - Beta: la hace el usuario en sus instalaciones

Pruebas Complementarias

- Pruebas de Usabilidad

  Que sea comprensible y amigable al usuario

- Smoke – Test

  Es una revisión rápida del producto de software para comprobar que funciona y no tiene defectos que interrumpan la operación básica del mismo.

  Se realiza previo recepción del SW por el equipo de pruebas o por el usuario final.

- Pruebas de Compatibilidad

  Se prueba como reacciona el producto en diferentes entornos y plataformas.

Pruebas Complementarias

Seguridad

Estas pruebas determinan el nivel de seguridad de las aplicaciones y sistemas en términos de confidencialidad, integridad y disponibilidad

OWASP – Open Web Application Security Project

- OWASP (proyecto abierto de seguridad en aplicaciones Web) ofrece herramientas, documentos, foros, y capítulos gratuitos y abiertos a cualquiera que esté interesado en mejorar la seguridad en las aplicaciones.

Pruebas Complementarias

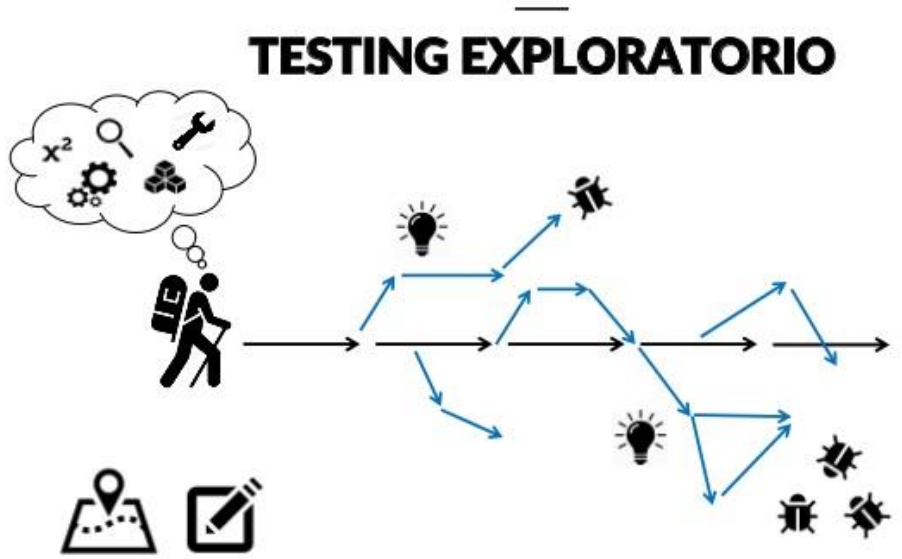
Testing Exploratorio

- Es un enfoque en el que simultáneamente se aprende sobre la aplicación, se diseñan casos de prueba y se ejecutan esos casos de prueba.
- No se prueba por probar, se tiene un objetivo y límite de tiempo para realizar las pruebas:
  - **Misión:** Describe qué se probará del producto, los tipos de incidentes que se buscan y los riesgos involucrados
  - **Sesión:** Indica el itinerario. A cada sesión se le estima un tiempo de ejecución y persigue un cierto objetivo (por ejemplo establecer flujos que podrían seguir los usuarios de la aplicación y probarlos, ver cómo se integra la aplicación con software externo, vulnerabilidades de seguridad en el login etc.), que puede ponerlo el propio tester, el equipo de testing, o el test manager

Pruebas Complementarias

Testing Exploratorio

- Aunque se tiene un objetivo, el mismo es general y no se indican los pasos a seguir para conseguirlo
- Existe documentación, que pueden ser los casos que se fueron probando o sólo los defectos encontrados.
  - Es importante documentar pero no es el objetivo de este enfoque de testing
- Este enfoque es complementario a las demás técnicas de pruebas
- Para hacer buenas pruebas exploratorias, es necesario que los testers tengan una mente abierta, pensamiento crítico, sean observadores, creativos, y curiosos para detectar bugs más complejos y evaluar riesgos.



PREGUNTAS

Fin de a Unidad 5.