Testing Efectivo en Java

Círculo Siete Capacitación

Clase 6 de 12

16 Agosto 2025

Pruebas de regresión

- Es un tipo de prueba de software cuyo objetivo es verificar que un cambio en el sistema (nuevo código, corrección de errores, refactorización, actualización de librerías, etc.) no introduzca fallos en funcionalidades que antes funcionaban correctamente.
- Se asegura de que el software no "retroceda" en calidad después de una modificación.
- Detecta si algo que ya estaba probado y funcionando empieza a fallar por cambios recientes.

Características principales

- Generalmente se automatizan para ejecutarse con rapidez y frecuencia.
- Pueden incluir pruebas unitarias, de integración, de sistema o incluso end-to-end que ya estaban definidas.
- Se suelen ejecutar en pipelines de CI/CD, cada vez que se integra código nuevo.

Ejemplo

- En una aplicación bancaria con una funcionalidad que permite transferir dinero entre cuentas.
  - Escribes pruebas que confirman que una transferencia resta dinero de una cuenta y lo suma en otra.
  - Luego agregas una nueva funcionalidad: programar transferencias futuras.
  - Una prueba de regresión vuelve a ejecutar las pruebas existentes de la transferencia normal para asegurarse de que ese flujo previo no se rompió al introducir la nueva característica.
- Dicho de manera simple: las pruebas de regresión son un “seguro” para que el software no pierda lo que ya se había logrado.

Pruebas de aceptación

- Validan que el software cumple los requisitos y criterios de aceptación definidos por el negocio o el cliente.
- Suelen ser de más alto nivel (end-to-end o de sistema), ejecutadas con datos y escenarios representativos.
- Normalmente se escriben en lenguaje entendible para el negocio (ej. con BDD: Gherkin + Cucumber).

Pruebas de regresión

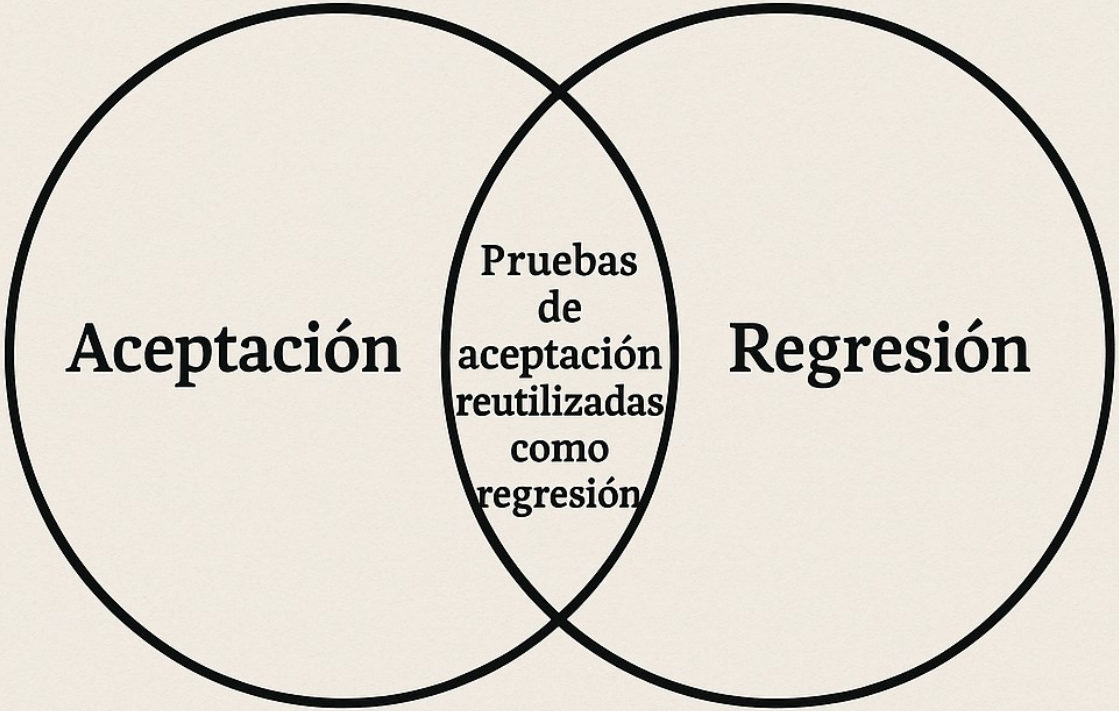
- Su objetivo es detectar si algo que antes funcionaba deja de hacerlo tras un cambio.
- No importa si la prueba es unitaria, de integración o de aceptación: si ayuda a confirmar que el sistema no retrocede en calidad, se considera regresión.

Cómo se relacionan

- Cuando una prueba de aceptación ya implementada se vuelve a ejecutar en cada release o integración, pasa a ser parte del set de regresión.
- Por ejemplo: un escenario de Cucumber que valida que “un usuario puede registrarse con correo y contraseña válidos” es originalmente una prueba de aceptación.
- Pero si cada vez que cambias el sistema vuelves a correr ese escenario para asegurarte de que el registro sigue funcionando, también actúa como una prueba de regresión.

En resumen

- Una prueba de aceptación inicial valida el cumplimiento de un requisito nuevo.
- Una vez aprobada, esa misma prueba puede integrarse al conjunto de regresión para garantizar que el requisito siga funcionando en futuras versiones.



Laboratorio 1

- Pruebas de integración con Cucumber y Testcontainers
  - labs

Patrón Arrange–Act–Assert (AAA)

- El AAA es una forma muy simple y clara de estructurar pruebas en software, especialmente pruebas unitarias.
- Se trata de dividir cada prueba en tres fases bien diferenciadas:
  - Arrange (Preparar)
  - Act (Actuar)
  - Assert (Afirmar / Verificar)

Arrange (Preparar)

- En esta fase se prepara todo lo necesario para la prueba:
  - Crear objetos.
  - Configurar datos de entrada.
  - Inicializar dependencias o mocks.
- La idea es dejar el escenario listo para ejecutar la acción que queremos probar.

Act (Actuar)

- Aquí se realiza la acción que estamos probando:
  - Llamar al método o función que queremos validar.
  - Ejecutar el comportamiento bajo prueba.
- Debe ser una sola acción principal, lo más clara y simple posible.

Assert (Afirmar / Verificar)

- En esta fase se verifican los resultados:
  - Comprobar el valor devuelto.
  - Validar cambios de estado.
  - Confirmar que se lanzaron excepciones o que se invocaron dependencias.
- Aquí es donde la prueba decide si pasó o falló.

Laboratorio 2

- Revisar el patrón AAA
  - Clase AAATransferServiceTest
  - Comparar con BDD