GAÑANADA Nº 006

El test unitario que no prueba absolutamente nada

. . .

CATEGORÍA	TESTING ORNAMENTAL
GRAVEDAD	■■■□□ III — GRAVE
FECHA DEL HALLAZGO	22 DE OCTUBRE DE 2026
ETIQUETAS	#tests-unitarios #cobertura #mocks #mutation-testing #java

§ 1

Descripción del hallazgo

El espécimen principal pertenece a la suite del servicio de facturación:

```
@Test
void calculaElTotalDeLaFacturaCorrectamente() {
    when(repositorio.buscar(42L)).thenReturn(factura);
    when(calculadora.total(factura)).thenReturn(new BigDecimal("121.00"));

    BigDecimal total = calculadora.total(factura);

    assertEquals(new BigDecimal("121.00"), total);
}
```

`calculadora` es un *mock*. El test configura un objeto falso para que devuelva 121, le pide el resultado y comprueba que devuelve 121. Es una prueba rigurosa del comportamiento de Mockito; del código de facturación no ejecuta ni una línea.

En el mismo archivo habita una variante más primitiva, pero más honesta:

```
@Test
void procesaElPedidoSinErrores() {
    servicio.procesar(pedidoDePrueba());
    assertTrue(true);
}
```

Esta, al menos, ejecuta código real. Lo que no hace es comprobar nada sobre él. Pasaría igual si `procesar` devolviera basura, borrara la base de datos o no hiciera nada en absoluto.

§ 2

Contexto del descubrimiento

Dos años antes, la organización estableció una puerta de calidad: ningún cambio se integra si la cobertura de líneas baja del 90 %. El equipo alcanzó el objetivo en un trimestre. La suite creció hasta 2 300 tests y un 94 % de cobertura, y el informe se mostraba con orgullo en cada revisión trimestral.

Entonces llegó a producción un cambio que sustituía el redondeo bancario (`HALF_EVEN`) por el redondeo comercial (`HALF_UP`) en el cálculo de impuestos. La especificación exigía el primero. Todos los tests pasaron.

Como parte de la investigación se ejecutó PIT, una herramienta de *mutation testing* que introduce pequeños cambios en el código —invertir una condición, cambiar un `+` por un `-`, eliminar una llamada— y comprueba si algún test falla. De cada cien mutantes, ochenta y nueve sobrevivieron.

§ 3

Consecuencias observadas

- **Falsa sensación de seguridad.** La cobertura mide qué líneas se ejecutan, no qué comportamiento se verifica. Un 94 % de líneas ejecutadas convivía con un 11 % de mutantes detectados.

- **Coste sin retorno.** Los tests que simulan detalles internos se rompen en cada refactorización aunque el comportamiento no cambie, y hay que reescribirlos. Protegen la implementación contra la mejora y el comportamiento contra nada.
- **Integración continua más lenta**, cargada de tests que no aportan información.
- **Pérdida de credibilidad.** Cuando la suite por fin detecta una regresión real, la primera reacción del equipo es sospechar del test.

§ 4

Recomendación de la academia

Un test que no puede fallar no es un test: es un adorno. Antes de dar uno por bueno, comprueba que falla cuando el código está mal —rompe la implementación a propósito y mira cómo se pone rojo—.

```
@Test
void redondeaLosImpuestosConRedondeoBancario() {
    var factura = new Factura(List.of(linea("2.50")), Iva.GENERAL);  // 21 %

    // 2,50 × 1,21 = 3,025 → con HALF_EVEN, 3,02; con HALF_UP, 3,03
    assertEquals(new BigDecimal("3.02"), factura.total());
}
```

Este test usa objetos reales, verifica una regla de negocio concreta y habría detectado el error el día en que se introdujo.

- **Prueba comportamiento a través de la interfaz pública**, con objetos reales siempre que sean baratos de construir.
- **Simula solo las interfaces**: red, reloj, hardware device, servicios externos. Nunca el objeto que estás probando.
- **Usa la cobertura para encontrar código sin probar, no como objetivo.** Cuando una medida se convierte en objetivo, deja de ser una buena medida.

No trabajes así, no seas un gañán.

REFERENCIAS

1 PIT — mutation testing para la JVM

2 Stryker Mutator — mutation testing para JavaScript, TypeScript, C# y Scala

Separata del volumen I de la Bitácora del Gañán, publicada por AXXA.
https://axxa.embeddedsource.es/bitacora-del-ganan/006-el-test-unitario-que-no-prueba-nada