MUTATION TESTING AT GOOGLE

Goran Petrovic

Adriana Herrero González

Daniel González Pérez

Miguel Morís Gómez

¿ QUÉ ES EL MUTATION TESTING?

¿Por qué hacemos test de nuestras aplicaciones?

Test passed   ¿Significa esto que el código es correcto?

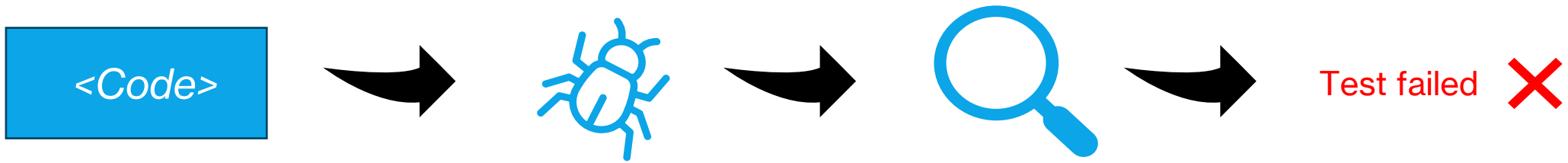
¿Cómo podemos saber si los test cumplen su función?

¿Qué es el mutation testing?

- Mutation testing:

Técnica avanzada de pruebas de software que evalúa la calidad de los casos de prueba modificando levemente el código fuente del programa y verificando si los casos de prueba detectan esas modificaciones.

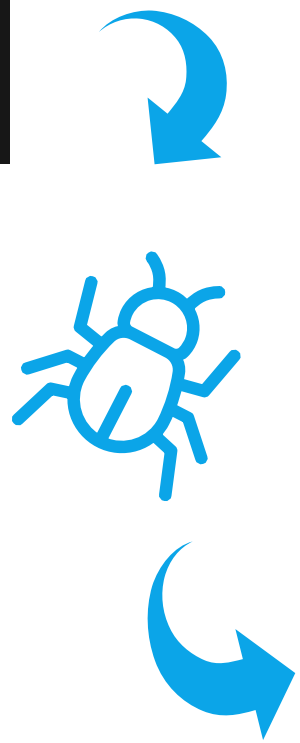
(Técnica que nos permite evaluar la fiabilidad de los test que estamos realizando)



EJEMPLO DE MUTACIÓN DE GORAN PETROVIC

```python
def suma(a, b):
    return a + b  # Suma correcta

# Prueba de la función
print(suma(5, 3))  # Debería imprimir 8
```



```python
def suma_mutante(a, b):
    return a - b  # Se cambió + por -

# Prueba de la función mutada
print(suma_mutante(5, 3))  # Debería imprimir 8, pero imprimirá 2
```

IMPLEMENTACIÓN REALISTA DEL MUTATION TESTING

Desarrolladores de software:

"Yo no voy a cometer un error tan evidente, así que si yo estoy seguro de que mi código es correcto, ¿por qué tengo que estropearlo introduciendo bugs a propósito?"

> *Los programadores siempre vamos a introducir bugs en nuestro código.*
> *(Goran Petrovic)*

Ejemplos de mutaciones de código prácticas

- Intercambio de operadores aritméticos: +, -, *, /, %
- Intercambio de operadores lógicos: &&, ||
- Borrar líneas de código. Borrar un else de un condicional.
- Cambiar valores de números a 0 o de String a cadena vacía
- Modificación de condicionales: >=, >, <, <=

HERRAMIENTAS MUTATION TESTING VS HERRAMIENTAS COBERTURA DE CÓDIGO

Cobertura de código ⟶ % = líneas de código cubiertas por tests / líneas de código totales * 100

Puntuación mutation testing ⟶ % = test "muertos" / tests totales * 100

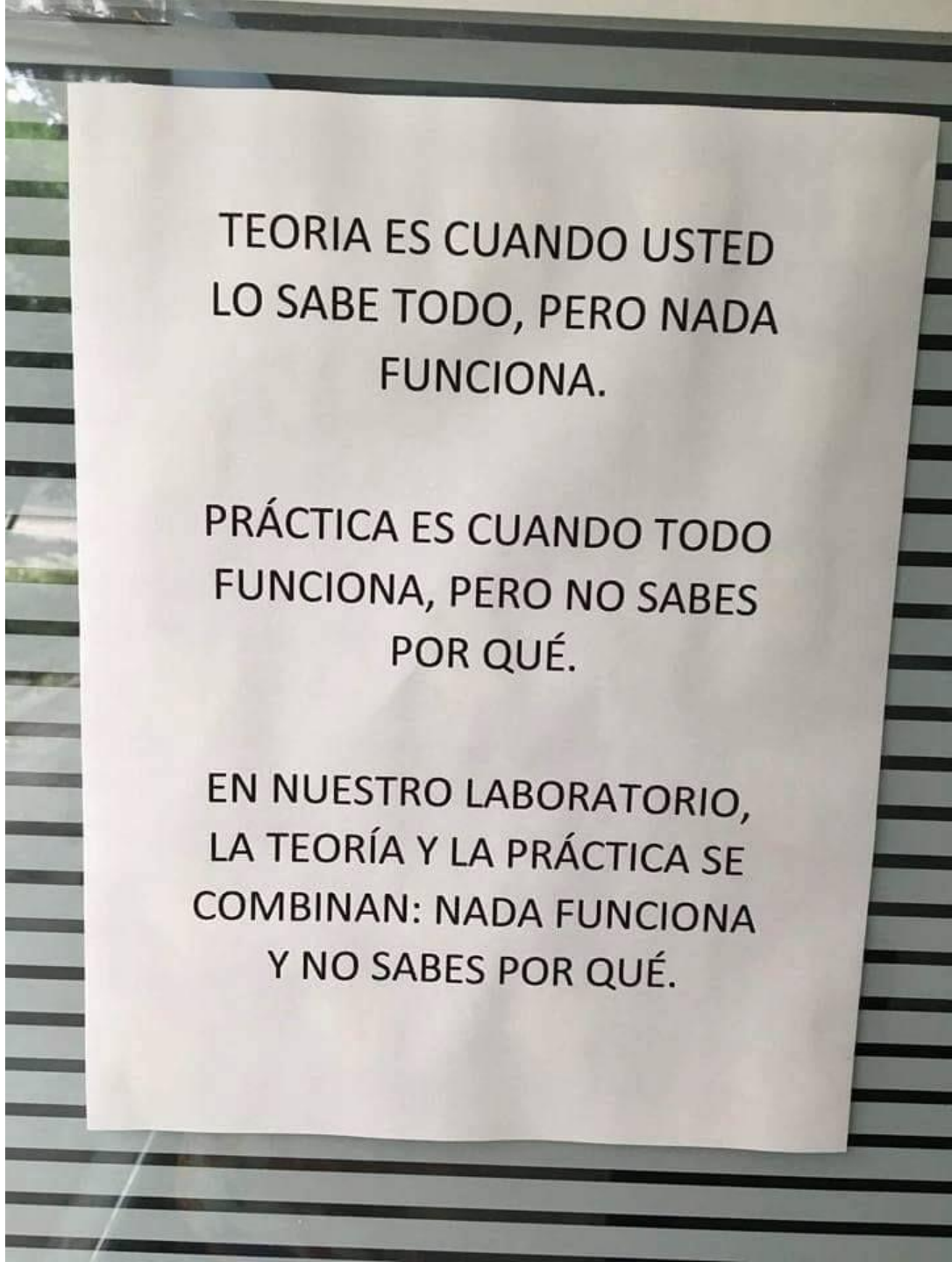
Herramientas para mutation testing automatizado

> *Como desarrollador no tienes que preocuparte por generar mutaciones porque hay una gran variedad de herramientas que hacen esto por ti.*
>
> *(Goran Petrovic)*

MUTATION TESTING... MUTATION ANALYSIS?

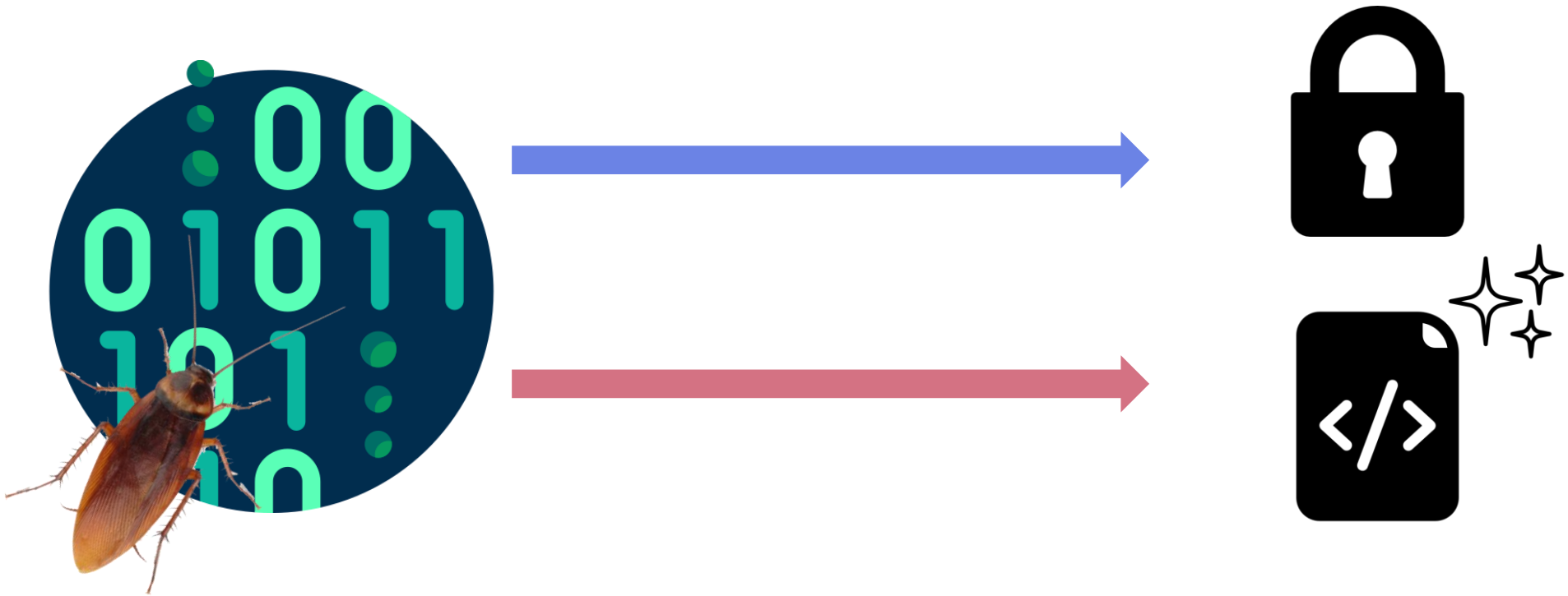
- **ANÁLISIS/la teoría:** métricas (nº de mutantes vivos, vivos/totales, etc.)

- **PRUEBAS/la práctica:** se aplican estas métricas para evitar bugs y mejorar el código



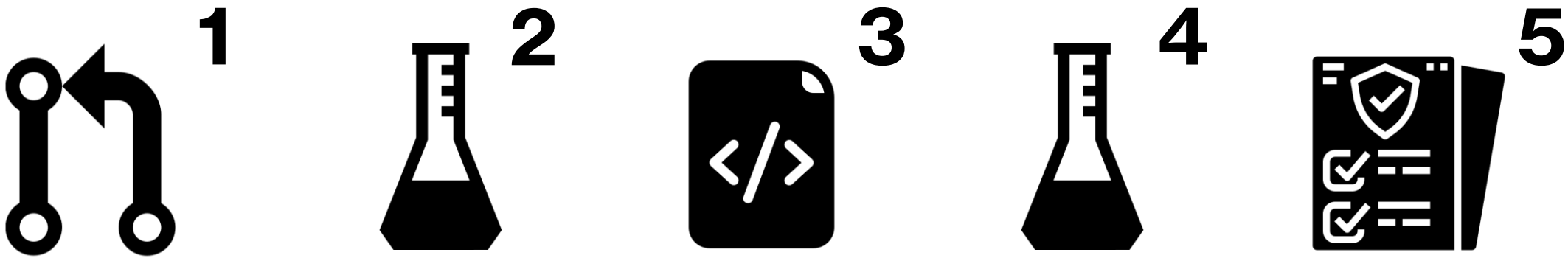
MAS TESTS, MEJOR CÓDIGO

Esta estrategia no solo nos ayuda a mejorar nuestra **suite de pruebas**, sino también la **calidad y legibilidad de nuestro código**.



MUTATION TESTING EN GOOGLE

el workflow:



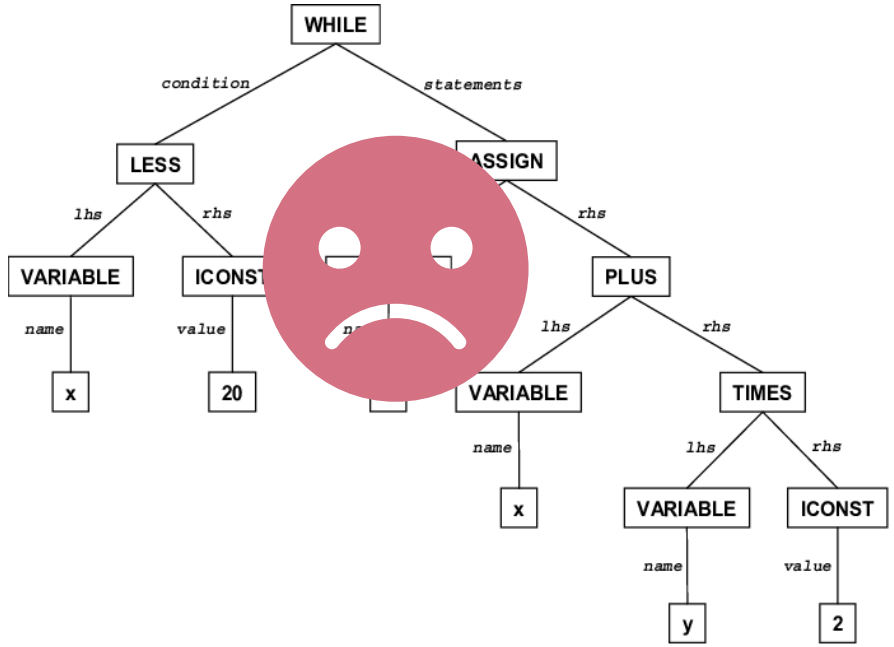
MUTATION TESTING EN GOOGLE

el servidor de mutación:

- Es la infraestructura necesaria para el análisis y la inserción de las mutaciones
- Se soportan varios lenguajes, cada servidor está específicamente creado para uno

MUTATION TESTING EN GOOGLE

la implementación:



- Parte del Open Source
- ASTs, Teoría de Compiladores...
- Se mutaría el AST entero, pero...
  - 1) Esto es costoso computacionalmente (y económica)
  - 2) Los desarrolladores de Google son personas (de momento...)

MUTATION TESTING EN GOOGLE

la implementación (de verdad):

- Es un proceso simplificado, no se muta todo el AST
- El proceso de análisis y filtrado importa MUCHO
- No se usa código intermedio

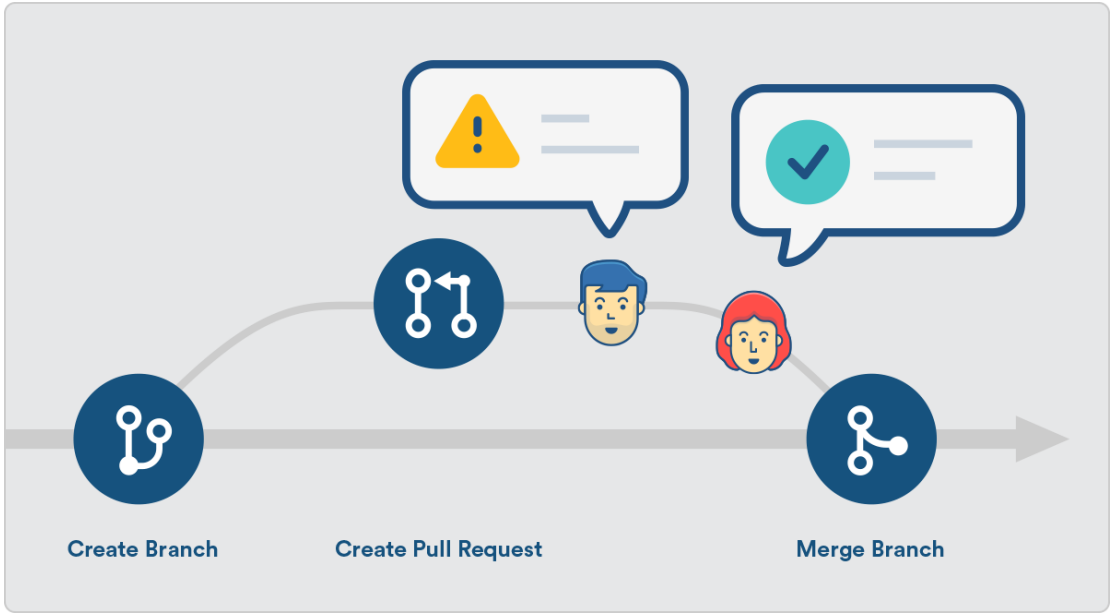
5-10 mutantes / 100 líneas

No más de uno / línea

PAPEL DEL DESARROLLADOR

La importancia de la retroalimentación en el desarrollo de software

- ¿Los desarrolladores deben conocer sus errores?
- ¿Mejoran estas herramientas la calidad del código?



¡¡15 millones!!

6 años

EFECTO DE ACOPLAMIENTO

01

Si un test identifica pequeños fallos

02

Detectará fallos más graves

03

Arreglar más errores en el 69% de los casos

INICIO DEL MUTATION TESTING EN GOOGLE

- De un proyecto pequeño a una revolución en las pruebas de software
- Priorizar lo que es factible antes que la perfección
- Desarrollo impulsado por capacidad
- Adaptar el proyecto a las limitaciones existentes

ESTRATEGIAS PARA MEJORAR EL MUTATION TEST

Claves para la optimización del mutation testing

5-6 heurísticos → 50 instancias

Agrupar por comportamiento → Colecciones de Java

Descartar mutaciones inútiles → Logging/Caché

RETROALIMENTACIÓN DE LOS USUARIOS

RESUMEN

Experiencia en Google

Mejorar la eficiencia

Reducir errores

Test más completos

Anticiparse a los errores

Centrado en el usuario

PREGUNTAS