Engineer

Pruebas de regresión

pruebas de que un cambio respeta el comportamiento previo

Qué es

Las pruebas de regresión comprueban si un programa conserva, después de un cambio, comportamientos que ya funcionaban. Una opción nueva de exportación puede cumplir su requisito y aun así modificar las filas que reciben quienes usan el sistema desde antes. Probar únicamente el ajuste nuevo permite responder si funciona la capacidad pedida. La prueba de regresión recorre el camino previo para saber si se mantiene aquello de lo que una persona o integración depende.

El manual de ingeniería de software de la NASA presenta estas pruebas como un medio para verificar que una modificación no ha introducido defectos. Entre sus recomendaciones están volver a ejecutar pruebas relacionadas con el código o los requisitos modificados e incluir los módulos que interactúan con ellos. La idea es válida cuando se cambia el código fuente, una dependencia, la configuración, los datos o el entorno. El mismo programa puede responder de otra manera después de actualizar un navegador o sus entradas.

Confirmar una corrección y probar regresiones son tareas complementarias. Primero se reproduce el problema y se verifica el resultado corregido. Después se comprueba el comportamiento cercano que debe conservarse. El caso del fallo original puede quedar entre las pruebas para avisar si vuelve a aparecer. El resultado esperado debe corresponder a un compromiso del producto. Si un requisito cambia deliberadamente, actualiza el resultado esperado y registra qué sucede con los usos existentes.

Utilice el concepto en una decisión

Aplique «¿Qué son las pruebas de regresión?» a una decisión real. Podemos ayudarle a relacionar el concepto con el sistema que está construyendo o comprando. Un equipo que lleva una idea del boceto a un producto al que sus clientes pueden acceder, sin pasarla entre tres proveedores.

Explore nuestros servicios Póngase en contacto →

Compruebe los supuestos detrás del término

Utilice la explicación y los trabajos enlazados para formular preguntas sobre su proyecto. Podemos examinar los supuestos con usted antes de elegir una herramienta. Ingresos multiplicados por 10 con mapas GIS — 10x crecimiento de ingresos impulsado desde el producto cartográfico

Ingresos multiplicados por 10 con mapas GIS Póngase en contacto →

Por qué importa

La consecuencia de una modificación rara vez termina en las líneas editadas. Sustituir una lista por un conjunto quizá elimina duplicados y a la vez altera el orden de un informe. Un analizador de fechas puede admitir el formato nuevo y dejar de aceptar un campo vacío que llega desde otro proceso. Una prueba automática asocia un efecto concreto con un cambio reciente. Esto importa especialmente cuando el comportamiento sirve a una integración que nadie suele ejecutar durante el desarrollo local.

Para una modificación pequeña, empieza por la regla común más cercana y selecciona uno o dos usos distintos y reales. Pregunta si el cambio puede afectar a la opción predeterminada, los casos límite, los errores o los datos que ya están guardados. Ante cambios importantes, valora también consecuencias que atraviesan todo el producto, como el rendimiento o la accesibilidad. Adapta la evidencia al coste de pasar por alto el problema. Una prueba unitaria puede comprobar una conversión con precisión. Para observar la navegación por teclado puede hacer falta un navegador y el mismo recorrido que sigue una persona.

Un ejemplo con nombre claro explica por qué interesa conservar la regla. «Una fecha vacía sigue vacía al exportar expedientes antiguos» aporta más contexto que «pasa el caso tres». Muestra los datos que importan y el resultado correspondiente. Quien mantenga el programa podrá reconocer si un nuevo requisito cambia esa decisión o si se ha roto algo.

Cómo se aplica en la práctica

Imaginemos una exportación CSV con una opción para incluir cuentas archivadas. De forma predeterminada, la exportación debe seguir excluyéndolas. Las cuentas activas conservan el orden por referencia. Si no hubo un último contacto registrado, la fecha sigue vacía. La nueva opción añade las cuentas archivadas sin repetir las activas. Prepara dos cuentas activas en orden inverso y una archivada. Deja vacía la fecha de una de las activas. Ejecuta el exportador real con las dos opciones.

En la ruta predeterminada, compara las referencias, su orden y la celda sin fecha. Con la opción activada, comprueba que aparecen las mismas cuentas y la archivada, una vez cada una. Conviene incluir un nombre con coma. Un exportador que una las columnas como texto puede producir el número correcto de filas y repartir mal los valores. Abre el CSV con el tipo de lector que usa el consumidor. Mantén el ejemplo lo bastante pequeño para leerlo y lo bastante variado para observar el efecto.

Verifica también que las pruebas rechazan el comportamiento defectuoso. En un entorno de ensayo, incluye provisionalmente las cuentas archivadas en la ruta predeterminada o quita las comillas que protegen el nombre con coma. Cada modificación debería provocar un fallo claro en la condición correspondiente. Restaura el código y ejecuta otra vez. Así se puede detectar una aserción demasiado débil, como una que solo comprueba que se creó un archivo. Probar un conjunto sin cuentas archivadas protege además la selección vacía.

La nota sobre subrayados en Safari y subconjuntos de fuentes muestra otro límite de la evidencia. Un cambio en CSS parecía corregir la representación en Chromium, pero Safari seguía atravesando con el subrayado los trazos descendentes de las letras. Para proteger ese comportamiento, una prueba de regresión tendría que observar los motores y los casos que pretende cubrir. La nota recoge el diagnóstico y lo observado. La prueba sería una forma de conservar esa lección.

Dónde surgen los problemas

Una gran cantidad de comprobaciones puede dar una imagen engañosa de la cobertura. Cientos de aserciones pequeñas pueden omitir la integración que lee los datos históricos. Elige casos a partir de lo que se utiliza, los datos que existen y las interacciones afectadas. Cuando ya haya escapado un defecto, protege el recorrido perjudicado y comprueba que la aserción falla si vuelve el comportamiento anterior al arreglo.

También se puede comparar de más. Una captura completa cambia si varían el orden, los identificadores generados o el reloj. Exige el orden si el exportador lo promete. Si basta con el conjunto de registros, compara los elementos y comprueba por separado que no haya duplicados. Revisa los cambios deliberados del resultado esperado como revisarías el código y conserva el motivo.

Por último, una prueba que falla a ratos dificulta atribuir el error. Espera una condición observable con un límite de tiempo, utiliza datos conocidos y evita que un caso deje estado para el siguiente. Conserva comprobaciones amplias allí donde cuentan las interacciones reales y otras, pequeñas y rápidas, cerca del trabajo diario. Cada una aporta evidencia en una frontera distinta.

Lecturas adicionales

- Manual de ingeniería de software de la NASA sobre pruebas de regresión
- Documentación de pytest sobre parametrizar ejemplos
- Pruebas unitarias en Software Engineering at Google

Glosario

Elija según sus restricciones reales

Describa el flujo de trabajo, las interfaces existentes y los usuarios que necesitan el cambio. Podemos definir una entrega comprobable y un traspaso que permita a su equipo mantener lo entregado.

Vea cuánto cuesta Póngase en contacto →

Haga práctica la terminología

Pregunte cómo se aplica «¿Qué son las pruebas de regresión?» a su organización. Basta un ejemplo concreto de su sistema actual para iniciar la conversación.

Explore nuestros servicios

welcome@engineer.company +45 50 10 61 84 Copenhague, Dinamarca

Llévese esta página

Tarjeta de contacto

Creado con esmero y cariño por D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Derechos de autor © 2025 – actualidad · Engineer ApS · Todos los derechos reservados.

Impreso desde https://engineer.company/es/glossary/regression-testing/