Tests De Integración/Regresión

Ferreira Juan David

Tests de Integración/Regresión

Las pruebas de integración dentro del software testing chequean la integración o interfaces entre componentes, interacciones con diferentes partes del sistema, como un sistema operativo, sistema de archivos y hardware o interfaces entre sistemas. Las pruebas de integración son un aspecto clave del software testing.
Es esencial que un probador de software tenga una buena comprensión de los enfoques de prueba de integración, para lograr altos estándares de calidad y buenos resultados.

Tests De Integración

Dentro del software testing existen muchos tipos o enfoques diferentes para las pruebas de integración. Los enfoques más populares y de uso frecuente son

- las pruebas de integración Big Bang,
- las pruebas de integración descendente,
- las pruebas de integración ascendente y
- las pruebas de integración incremental.

La elección del enfoque depende de varios factores como el costo, la complejidad, la criticidad de la aplicación, etc.

Tests De Integración

Hay muchos tipos menos conocidos de pruebas de integración, como la integración las siguientes:

- las pruebas de integración de servicios distribuidos,
- las pruebas de integración sándwich,
- las pruebas de integración de la red troncal,
- las pruebas de integración de alta frecuencia,
- las pruebas de integración de integración de capas, etc.

Tests De Integración

Prueba De Integración Big Bang

En las pruebas de integración de Big Bang, todos los componentes o módulos se integran simultáneamente, después de lo cual todo se prueba como un todo.

Ventaja: Las pruebas de Big Bang tienen la ventaja de que todo está terminado antes de que comiencen las pruebas de integración.

Desventaja: Podemos detectar los defectos de la interfaz clave al final del ciclo.

Es necesario crear los controladores de prueba para los módulos en todos los niveles excepto el control superior.

Tests De Integración

Prueba De Integración Descendente

Las pruebas se llevan a cabo de arriba a abajo, siguiendo el flujo de control o la estructura arquitectónica (por ejemplo, comenzando desde la GUI o el menú principal). Los componentes o sistemas se sustituyen por stubs.
Ventaja: El producto probado es muy consistente porque la prueba de integración se realiza básicamente en un entorno casi similar al de la realidad.
Los códigos auxiliares se pueden escribir en menos tiempo porque en comparación con los controladores, los códigos auxiliares son más sencillos de crear.
Desventajas: La funcionalidad básica se prueba al final del ciclo.

Tests De Integración

Prueba De Integración Descendente

Las pruebas se llevan a cabo desde la parte inferior del flujo de control hacia arriba. Los componentes o sistemas se sustituyen por controladores.
Ventaja: En este enfoque, el desarrollo y las pruebas se pueden realizar juntos para que el producto o la aplicación sea eficiente y de acuerdo con las especificaciones del cliente.
Los códigos auxiliares se pueden escribir en menos tiempo porque en comparación con los controladores, los códigos auxiliares son más sencillos de crear.
Desventajas: La funcionalidad básica se prueba al final del ciclo.

Tests De Integración

Prueba De Integración Incremental

Otro enfoque es que todos los programadores se integran uno por uno y se realiza una prueba después de cada paso.
Ventaja: los defectos se encuentran temprano en un ensamblaje más pequeño cuando es relativamente fácil detectar la causa.
Dentro de las pruebas de integración incremental existe una gama de posibilidades, en parte dependiendo de la arquitectura del sistema.
Desventajas: Una desventaja es que puede llevar mucho tiempo ya que los stubs y los controladores deben desarrollarse y usarse en la prueba.

Tests de Regresión

Prueba De Regresión

Consiste en probar un sistema que ha sido analizado previamente para asegurar que no se haya introducido algún tipo de defecto como resultado de cambios realizados.

La planificación de estas pruebas no es tan sencilla, especialmente cuando el tiempo y los recursos son limitados. Como consecuencia de esa limitación, se genera presión y los planes de pruebas suelen acortarse o se pierde tiempo en el ajuste ocasionando un aumento en la probabilidad de fallo humano al no seleccionar casos más adecuados.

Tests de Regresión

A continuación, listamos 6 aspectos a tener en cuenta:

1. Planificación previa,
2. Uso de técnicas de diseño de casos de prueba,
3. Clasificación adecuada mediante suites,
4. Programación de mantenimientos,
5. Definición del tipo de Regresión a realizar,
6. Automatización.

Planificación previa

Desde el inicio de la estimación de casos de prueba, es aconsejable **asignar prioridades** de ejecución: Alta, Media, Baja y palabras clave como Regresión, Sanidad, etcétera para los casos de prueba. De esta manera se puede hacer un filtro rápido por prioridades y etiquetas, que permitan diferenciar rápidamente cuáles deberían ser los casos más importantes.

Tests de Regresión

Uso de técnicas de diseño de casos de prueba

Reconocidas entidades internacionales[a], mencionan varias técnicas para la creación de casos de prueba que permiten una mejora, de tal manera que al aplicarlas se obtienen menos casos de prueba y un mayor factor de **coverage** o bien una base de creación empírica.
Al tener un menor número de casos de prueba, se reducen las pruebas exhaustivas y tiempos de ejecución.

[a] International Software Testing Qualifications Boardy American Society for Quality

Tests de Regresión

Clasificación adecuada mediante suites

La división de **casos de prueba** según los módulos de un sistema, facilitará la selección de casos al momento de construir un **Plan de Pruebas** para Regresión.
En las carpetas o suites deberían evitarse nombres ambiguos o diferentes a lo definido en el requerimiento, o en los nombres que se muestran en el sistema.

Tests de Regresión

Programación de mantenimientos

Conforme nuestra aplicación crece o sufre cambios y mejoras, se hace más necesario dedicar tiempo a revisar si estos cambios afectan nuestros casos de pruebas actuales, invalidándolos o requiriendo actualización. Si sabemos que el cambio los afectará, debemos incluir tareas de actualización.
Si estamos bajo un modelo de un sistema antiguo con casos que nunca han tenido mantenimiento, lo recomendable es incluir tareas periódicas para la revisión de todos los casos existentes.

Tests de Regresión

Definición del tipo de Regresión a realizar

No todas las pruebas de regresión implican una ejecución del $100\,\%$ de los casos de prueba (Full Regression).
En muchos casos, los cambios realizados afectan componentes específicos por lo que la regresión podría centrarse en esos módulos y la verificación del resto del sistema podría realizarse con una prueba de humo.
En estos casos es necesario un análisis de dependencias para tener certeza de que los cambios por aplicar efectivamente no afectan otras partes no contempladas en la regresión del componente.

Tests de Regresión

Automatización

Las pruebas de regresión suelen ser procesos largos y al incorporar **configuracionde para las pruebas de automatización** se consiguen varias ventajas.

- La capacidad de ejecución aumenta, al tiempo que la duración de las pruebas se reduce.
- Los scripts pueden ejecutarse tantas veces como se requiera sin que esto implique desgaste en el equipo.
- Se pueden incorporar diferentes entornos en la prueba sin que esto aumente el tiempo de las mismas, pues pueden ejecutarse en paralelo.

Bibliografía


Hareton KN Leung and Lee White.
Insights into regression testing (software testing).
1989.


Hareton KN Leung and Lee White.
A study of integration testing and software regression at the integration level.
In *Proceedings. Conference on Software Maintenance 1990*, pages 290–301. IEEE, 1990.


W Eric Wong, Joseph R Horgan, Saul London, and Hiralal Agrawal.
A study of effective regression testing in practice.
1997.


Shin Yoo and Mark Harman.
Regression testing minimization, selection and prioritization: a survey.
Software testing, verification and reliability, 22(2):67–120, 2012.