

Caracterización de la Prueba de Software

CLASIFICACIÓN Y TÉCNICAS

En este número esbozaré algunas de las técnicas más comunes de prueba de software, de acuerdo con los criterios de clasificación más usuales. En www.e-quality.net pestaña Definiciones → Conceptos puede encontrarse mayor profundidad.

Composición Interna del Componente

Se refiere al nivel de conocimiento de la estructura interna del sistema a probar (SUT: System Under Testing), que el tester requiere para realizar la prueba. Técnicas comunes son:

- Pruebas de caja negra (black-box testing, o “pruebas de funcionalidad”): centradas en verificar si los requerimientos son satisfechos. Las pruebas de volumen y de stress (rendimiento con grandes cantidades de datos, y con bloques de datos por unidad de tiempo, respectivamente), son casos especiales de este tipo de prueba; otras técnicas son la de clases de equivalencia y la de valores límite.
- Pruebas de caja blanca (white-box testing, o pruebas “de código/diseño”): se revisan, entre otras cosas, el diseño y código fuente para verificar que sigan los estándares especificados, los algoritmos sean adecuados, y la arquitectura sea apropiada. Técnicas comunes son el análisis de algoritmos, la cobertura de decisiones, las revisiones entre pares y las recorridas.

Granularidad del Componente

Se refiere al tamaño de los elementos del SUT que se van probando. Hablamos de:

- Pruebas de unidad: se prueba por separado cada “elemento mínimo de procesamiento” definido por la organización (objetos, componentes, módulos, funciones). Las técnicas más utilizadas son las de caja blanca. Típicamente, son realizadas por el equipo desarrollador.
- Pruebas de integración: se verifica la interacción entre unidades con técnicas como pruebas de interacción y de mutación. Suelen ser ejecutadas por el equipo de prueba.

- Pruebas de sistema: se verifica el sistema como un todo, aplicando pruebas de caja negra utilizando perfiles de usuario, haciendo v.gr. pruebas como configuración, seguridad, confiabilidad y recuperación. Debieran ser ejecutadas por el equipo de prueba.
- Pruebas de aceptación: se verifica la satisfacción de requerimientos con técnicas como pruebas- α y pruebas- β , descritas más abajo. Debieran ser llevadas a cabo por un equipo que incluya al cliente, usuario(s) y testers.

Es común que en los esfuerzos de desarrollo se aplique un solo subconjunto de estas técnicas, guardando el orden en que se acaban de describir.

“Dirección de Avance” en la Prueba

Para probar el SUT se diseñan casos de prueba; llamamos pruebas progresivas a la primera vez que éstos son aplicados. Esa primera fase detecta defectos que luego son corregidos por el equipo desarrollador.

A la versión corregida debieran aplicársele nuevamente los casos de prueba, o al menos un subconjunto de ellas, porque puede ocurrir que en realidad no se realicen las correcciones necesarias, o que las correcciones generaren otros defectos.

A la repetición de la aplicación de un subconjunto de casos de prueba la llamamos pruebas regresivas. La elección del subconjunto es muy importante y debe realizarse bajo criterios bien definidos y en acuerdo con el responsable general del proyecto. Aún cuando en principio se pudieran tener cuantos ciclos regresivos se quisiera, en realidad un número pequeño de ellos muestra tendencias que permiten tomar decisiones (como rehacer un componente). Para aprovechar al máximo la inversión en las pruebas, en todo proyecto debiera ejecutarse al menos un ciclo de pruebas de regresión, que per-

mitiera revisar los resultados de las correcciones realizadas.

Control sobre el Ambiente de Prueba

Distinguiamos dos ambientes:

- Pruebas- α (α -testing), llevadas a cabo de manera controlada en un laboratorio de pruebas que trata de reproducir el ambiente en que corre el SUT.
- Pruebas- β , realizadas al SUT en su ambiente real de aplicación.

Es deseable llevar a cabo ambas fases en este orden, o en el peor de los casos, en paralelo. Sin embargo, hay ocasiones en que es difícil reproducir el ambiente de aplicación y sólo se llevan a cabo las pruebas- β .

Otros Criterios

Otras características del SUT que llegan a imponer restricciones al probar son la plataforma de desarrollo y el dominio de aplicación en que opera. La primera puede requerir ciertas técnicas/herramientas de prueba (v.gr. para probar firmware); la segunda puede exigir cierta profundidad en la prueba que facilite alguna certificación (v.gr. en equipos médicos).

La aplicación apropiada de estas técnicas constituyen actividades centrales del proceso de prueba de software; cuales de ellas se apliquen es una decisión crucial que depende de las características de cada proyecto. ©

- Luis Vinicio León Carrillo



Luis Vinicio León Carrillo es profesor-investigador del Departamento de Electrónica, Sistemas e Informática del ITE-SO, y director general de e-Quality S.A. de C.V., empresa especializada en prueba de software. Luis Vinicio es doctorando por la Universidad Técnica de Clausthal, Alemania; su trabajo predoctoral giró alrededor a la aplicación de los lenguajes formales en la Ingeniería de Software. Es coautor de un marco tecnológico que hoy permite a e-Quality desarrollar empresas de prueba de software. Luis Vinicio es co-fundador y Secretario actual del Capítulo Guadalajara de la AMCIS.

Referencias

- www.e-quality.net pestaña Definiciones → Conceptos.
- Jorgensen, P.: Software Testing. CRC Press; 2002.
- Beizer, B.: Software Testing Techniques. International Thompson Computer Press; 1990.
- Kit, E.: Software Testing in the Real World. ACM Press; 1995.