

Introducción a las Pruebas de Software

Principios de Desarrollo de Software – Tema 4 Parte 2

1



Agenda

- ¿Qué es la prueba de software?
- Estrategias de pruebas de software
- Técnicas de Pruebas de Software
- Pruebas funcionales
 - Clases de Equivalencia
 - · Análisis de Valores Límite
 - Análisis Sintáctico

Principios de Desarrollo de Software

2

Pruebas Funcionales o de Caja Negra

- Para reducir el número de casos de prueba a un nivel fácil de gestionar mientras se mantiene una cobertura razonable de las pruebas
- caja negra:

• Para definir pruebas de Identificación de clases de equivalencia Análisis de Valores Límite Análisis Sintáctico

3

Aplicabilidad y limitaciones

- Las pruebas de análisis sintáctico reducen el número de casos de prueba que se deben generar y ejecutar
- Está más indicado para componentes en los cuales la mayoría de los datos de entrada se pueden modelar como una gramática
- Se requiere que la gramática de las entradas se pueda identificar con base en los requisitos del sistema
- El análisis sintáctico se puede aplicar tanto a las pruebas unitarias como de integración
 - Solamente en algunos casos esta técnica se puede utilizar en el nivel de sistema
 - No es recomendable para pruebas de aceptación

Principios de Desarrollo de Software

Pruebas de análisis sintáctico - Procedimiento

- Definición de la gramática
- Creación del árbol de derivación
- Identificación de los casos de prueba
- Automatización de los casos de prueba

Principios de Desarrollo de Software

5

5

Definición de la gramática

- La gramática debe ser tipo 2 o tipo 3: Regular e independiente del contexto
- Es necesario que en todas producciones haya un
- único símbolo no terminal a la izquierda de la regla
- Es necesario que no existan símbolos Lambda (símbolos vacíos)
 - Las gramáticas recursivas son problemáticas porque el árbol de derivación asociado sería infinito y por tanto los datos y casos de prueba también. Es posible limitar el número de niveles de recursividad considerado.

Principios de Desarrollo de Software

6

Creación del árbol de derivación

- Se crea usando la gramática obtenida en el primer paso
- Cada símbolo, terminal o no, se incluirá en un nodo diferente
- Los nodos se numeran comenzando por el I
- En el árbol tienen que distinguirse los distintos niveles y en estos, si los nodos son terminales o no

Principios de Desarrollo de Software

7

7

Identificación de los casos de prueba - I

- Se obtienen el análisis del árbol y se dividen en dos partes: entradas válidas e inválidas
- Para identificar las entradas válidas:
 - Paso I Se producen casos de prueba de tal forma que todos los nodos no terminales estén cubiertos
 - Paso 2 Se repite el paso anterior para cubrir al menos una vez todos los nodos terminales

Principios de Desarrollo de Software

8

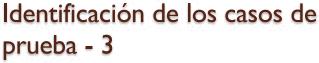


- Para identificar entradas inválidas:
 - Por lo general hay una gran cantidad de ellas, no es factible considerarlas todas, por lo que se considera un muestra representativa de las mismas
 - Paso I Para los nodos no terminales se procede a su omisión y su adición. La adición de nodos puede producir un gran número de casos de prueba, siendo semánticamente más difícil de generar
 - Paso 2 Para los nodos terminales debe procederse también a su modificación. Se simula mediante errores tipográficos, siendo aconsejable no someter a prueba grandes combinaciones de errores. La explosión combinatoria sería enorme y la prueba poco realista

Principios de Desarrollo de Software

9

9



- Para identificar entradas inválidas
 - Paso I
 - Excepciones en omisiones
 - Nodos ya omitidos por nodos superiores en la jerarquía del árbol
 - Nodos que al ser omitidos producen casos válidos
 - Nodos ya omitidos por igualdad con otros nodos en la misma producción
 - Nodos ya omitidos por cruzamiento de nodos en distintas producciones

Principios de Desarrollo de Software

10



- Para identificar entradas inválidas:
 - Paso I
 - Excepciones en adiciones
 - Nodos ya añadidos en nodos superiores en la jerarquía del árbol
 - · Nodos iguales consecutivos
 - · Nodos que al duplicarlos dan un caso válido

Principios de Desarrollo de Software

11

11



- Para identificar entradas inválidas:
 - Paso 2
 - Excepciones en modificaciones
 - Nodos ya modificados por otros nodos en reglas de producción que tienen la misma parte izquierda
 - · Agrupación de modificaciones de nodos

Principios de Desarrollo de Software

12