PRUEBAS FUNCIONALES USANDO TÉCNICAS DE CAJA NEGRA – PARTE I

TÉCNICAS DE PRUEBAS DE SOFTWARE
INGENIERÍA DE SISTEMAS Y COMPUTACIÓN
UNIVERSIDAD DEL VALLE
DOCENTE BEATRIZ FLORIAN GAVIRIA

Basado Parcialmente en Material de la Empresa Green-SQA de ParqueSoft

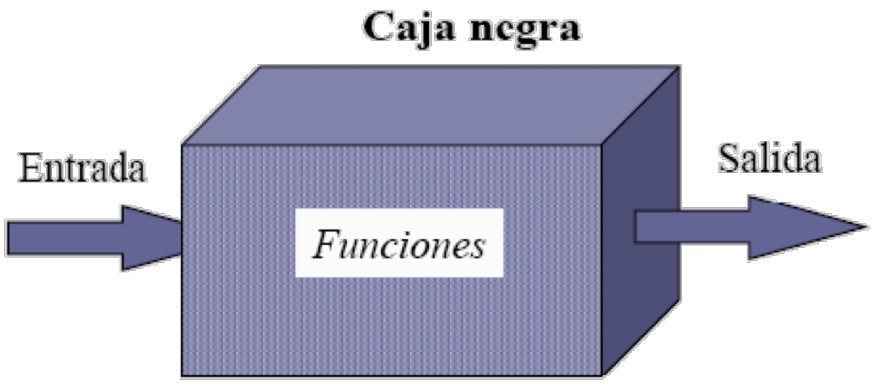
AGENDA

- Introducción al diseño de casos de prueba para pruebas de caja negra
- Técnica de Particiones de Equivalencia
- Técnica de Valor Límite
- Práctica

RECORDAR - DISEÑO DE PRUEBAS

Técnicas de caja negra o funcionales

- Realizan pruebas sobre la interfaz del programa a probar, entendiendo por interfaz las entradas y salidas de dicho programa.
- No es necesario conocer la lógica del programa, únicamente la funcionalidad que debe realizar.



Caso de Prueba

- Conjunto de entradas, condiciones de ejecución y resultados esperados desarrollados para evaluar una funcionalidad del software.

Pista!!

Caso de Prueba Funcional <> Caso de Uso

- Casos de prueba en sistemas no determinísticos (No resultados esperados fijos)

DIVIDIR EL DOMINIO DE ENTRADA DE UN PROGRAMA EN UN NÚMERO FINITO DE *CLASES DE EQUIVALENCIA*

TÉCNICA DE PARTICIONES DE EQUIVALENCIA

PARTICIONES DE EQUIVALENCIA

Este método intenta dividir el dominio de entrada de un programa en un número finito de clases de equivalencia.

campo de entrada de un programa = Σ de clases de datos

- Los datos en una clase son equivalentes

Objetivo

- Definición del menor número de casos de prueba que descubran clases de errores

PARTICIONES DE EQUIVALENCIA

El diseño de casos de prueba según esta técnica consta de dos pasos:

1. Identificar las clases de equivalencia.

2. Identificar los casos de prueba.

- Una prueba realizada con un valor representativo de cada clase es equivalente a una prueba realzada con cualquier otro valor de dicha clase.

- Si el caso de prueba correspondiente a una clase de equivalencia detecta un error, el resto de los casos de prueba de dicha clase de equivalencia deben detectar el mismo error. Y viceversa.

PARTICIONES DE EQUIVALENCIA

Conjunto de Clases Equivalentes

- Conjunto de datos que definen entradas Válidas y No Válidas al sistema
  - ❑ Entradas Válidas : Generan un valor esperado
  - ❑ Entradas No Válidas : Generan un valor inesperado (excepciones)

PARTICIONES DE EQUIVALENCIA

Pauta 1: Si un parámetro de entrada especifica un rango de valores

- Se define una clase de equivalencia válida y dos inválidas.
- Ejemplo: Un contador puede ir de 1 a 99

Clases validas	Clase invalidas
1< nro. < 99	nro. < 1
	nro. >= 99

PARTICIONES DE EQUIVALENCIA

Pauta 2: Si un parámetro de entrada especifica un valor numérico o número de valores

- Identificar una clase válida y dos clases no válidas.
- Ejemplo: No. de unidades que se compran

Clases validas	Clase invalidas
Cualquier número de unidades disponibles	Cualquier número no válido de unidades a comprar
	No entra número

PARTICIONES DE EQUIVALENCIA

Pauta 3.1: Si un parámetro de entrada se especifica con un conjunto de valores de entrada que serán tratados igual.

- Identificar una clase de equivalencia válida y dos no válidas.
- Ejemplo: Hay 6 colores para escoger de cierto modelo de auto y el procesamiento de la orden de compra no se altera por el color seleccionado.

Clases validas	Clase invalidas
Azul o Rojo o Verde o Negro o Gris o Blanco	Ingresa otro color
	No selecciona color

Pauta 3.2: Si hay razones para creer que cada uno de los miembros del conjunto serán tratado de modo distinto por el programa

- Se define una clase de equivalencia válida por cada miembro y dos inválidas.
- Ejemplo: El tipo de un vehículo puede ser: autobús, camión, taxi, coche o moto y el procesamiento de la orden de compra se altera según el tipo de vehículo seleccionado.

Clases validas	Clase invalidas
Uno por cada uno: 1)Autobús 2) Camión, 3)Taxi 4) Coche 5) Moto.	Alguno que no es esos, por ejemplo: *trailer*
	No selecciona tipo de vehículo

PARTICIONES DE EQUIVALENCIA

Pauta 4: Si un parámetro de entrada es una condición lógica (debe ser)

- Se define una clase válida y dos o más inválidas (según mensajes de error diferentes).
- Ejemplo 1: El primer carácter del identificador debe ser un dígito
- Ejemplo 2: Número que debe ser de cinco dígitos.

Ejemplo 1:

Clases validas	Clase invalidas
Primer carácter un dígito	Primer carácter distinto de dígito
	Primer carácter vacío

Ejemplo 2:

Clases validas	Clase invalidas
Número de cinco dígitos	Número de menos de cinco dígitos
	Número de más de cinco dígitos
	No ingresar número

PARTICIONES DE EQUIVALENCIA

Como Diseñar Pruebas con PE

- Identifique los valores de entrada del software
- Se identifican las clases de equivalencia (Validas - Invalidas)
- Se especifican los casos de prueba (Escenario – Resultado Esperado)

PARTICIONES DE EQUIVALENCIA

Identificar los casos de prueba

- El objetivo es minimizar el número de casos de prueba, así cada caso de prueba debe considerar tantas condiciones de entrada como sea posible.
- No obstante, es necesario realizar con cierto cuidado los casos de prueba de manera que no se enmascaren faltas.
- Así, para crear los casos de prueba a partir de las clases de equivalencia se han de seguir los siguientes pasos:
    1. Asignar a cada clase de equivalencia un número único.
    2. Hasta que todas las clases de equivalencia hayan sido cubiertas por los casos de prueba, se tratará de escribir un caso que cubra tantas clases válidas no incorporadas como sea posible.
    3. Hasta que todas las clases de equivalencia no válidas hayan sido cubiertas por casos de prueba, escribir un caso para cubrir una única clase no válida no cubierta.

PARTICIONES DE EQUIVALENCIA

Identificar los casos de prueba

- La razón de cubrir con casos individuales las clases no válidas es que ciertos controles de entrada pueden enmascarar o invalidar otros controles similares.

- Por ejemplo, si tenemos dos clases válidas: “introducir cantidad entre 1 y 99” y “seguir con letra entre A y Z”, el caso 105 1(dos errores) puede dar como resultado 105 fuera de rango de cantidad, y no examinar el resto de la entrada, no comprobando así la respuesta del sistema ante una posible entrada no válida.

CENTRARSE EN LOS VALORES LÍMITE DE LAS ENTRADAS O LAS SALIDAS

TÉCNICA DE ANÁLISIS DE VALOR LÍMITE

ANÁLISIS DE VALOR LIMITE

La técnica se enfoca en la identificación de los casos de prueba asociados con los valores límites del dominio de la función tanto de entrada como de salida .

1.	Un valor en el limite inferior del rango de entrada	min
2.	Un valor por encima del limite inferior	min-
3.	Un valor valido dentro del rango de entrada	val
4.	Un valor por debajo del limite superior	max-
5.	Un valor en el limite superior del rango de entrada	max

Técnica:	Análisis de valor limite
Proceso:	Adición de Beneficiario hijo a Afiliado de EPS
Variable:	Edad de Beneficiario
Valores:	[0 - 18]
Casos:	0 años
	1 año
	5 años
	17 años
	18 años

ANÁLISIS DE VALOR LIMITE

Por lo tanto, el análisis de valores límite complementa la técnica de partición de equivalencia de manera que:

- En lugar de seleccionar cualquier caso de prueba de las clases válidas e inválidas, se eligen los casos de prueba en los extremos.
- En lugar de centrase sólo en el dominio de entrada, los casos de prueba se diseñan también considerando el dominio de salida.

ANÁLISIS DEL VALOR LÍMITE

Ventajas de la técnica:

- La técnica reduce el número de casos de pruebas que deben ser creados y ejecutados.
- Esta técnica permite elegir un subconjunto de las pruebas que son eficiente y eficaces en encontrar no conformidades.
- La experiencia muestra que los casos de prueba que exploran las condiciones límite producen mejor resultado que aquellos que no lo hacen.

Desventajas de la técnica:

- No prueba todas las entradas posibles.
- No prueba las dependencias entre las combinaciones de entrada.
- No se puede identificar qué porcentaje del sistema ha sido probado.

ANÁLISIS DEL VALOR LÍMITE

- **Pauta 1: Si un parámetro de entrada especifica un rango de valores**
  - Se diseñarán casos de prueba para los dos límites del rango, y otros dos casos para situaciones justo por debajo y por encima de los extremos.
  - Ejemplo: Un contador puede ir de 1 a 99
    - 6 Casos de prueba: 0, **1**, 2, 3, 4, 5, 6, … , 97, 98, **99**, 100

- **Pauta 2: Si un parámetro de entrada especifica un número de valores numéricos**
  - Se diseñan dos casos de prueba para los valores mínimo y máximo, además de otros dos casos de prueba para valores justo debajo del máximo y el mínimo.
  - Ejemplo: Hay 6 identificadores de tipos de categorías [1, 2, 3, 4, 5 y 6]
    - Casos de prueba: 0, **1**, 2, 3, 4, 5, **6**

ANÁLISIS DEL VALOR LÍMITE

- **Pauta 3.1 y 3.2:**
  - Aplicar las reglas anteriores a los datos de salida.
    - Los valores límite de entrada no generan necesariamente los valores límite de salida (recuérdese la función seno, por ejemplo)
    - No siempre se pueden generar resultados fuera del rango de salida. (pero es interesante considerarlo).

- **Pauta 4: Si la entrada o salida de un programa es un conjunto ordenado.**
  - Habrá que diseñar casos de prueba prestando atención a los elementos primero y último del conjunto.

PRÁCTICA

PRÁCTICA

- Pensando en su proyecto de curso, diseñe casos de prueba funcionales utilizando las 2 técnicas de pruebas de caja negra vistas hoy.
- Utilice más de una pauta de diseño para cada técnica
- Diseñe al menos 6 casos de pruebas
- Puede seleccionar más de una funcionalidad para las pruebas