CURSO

SOFTWARE TESTING

UNIDAD 8 – Técnicas de Pruebas

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

PRESENTACIÓN

En esta unidad veremos las distintas técnicas de prueba, para caja negra y caja blanca

TEMARIO

1. – Clasificación de las pruebas
2. – Técnicas de Caja Negra
3. – Técnicas de Caja Blanca.
4. – Ejemplos.

Clasificación por objetivo

- **Funcionales:** Indican “lo que el sistema hace”. Se basan en las funcionalidades detalladas en la especificación de requerimientos. Su objetivo es corroborar que el sistema desarrollado contenga todas las funcionalidades necesarias.

- **No funcionales:** Indican “cómo trabaja el sistema”. Las características no funcionales del software, se pueden medir de diversas maneras, por ejemplo por medio de tiempos de respuesta en el caso de pruebas de desempeño.

- **Estructurales:** Prueban el comportamiento interno del sistema, o sea su arquitectura. Son pruebas de “Caja Blanca” (son sinónimos).

Clasificación por: Dinámicas / Estáticas

Pruebas dinámicas

- Son todas aquellas pruebas que para su ejecución requieren la ejecución de la aplicación.
- Debido a la naturaleza dinámica de la ejecución de pruebas es posible medir con mayor precisión el comportamiento de la aplicación desarrollada.

Clasificación por: Dinámicas / Estáticas

Pruebas estáticas

- Son las pruebas que se realizan sin ejecutar el código de la aplicación.
- Buscan testear los documentos, diagramas, diseños que son parte del sistema pero no son ‘código ejecutable’
- Se realiza la verificación de forma objetiva en función de los estándares de calidad vigentes.
- Buscan encontrar defectos (no corregirlos) de forma temprana. Recordar que encontrar un defecto en, por ejemplo, los casos de uso puede ahorrar mucho esfuerzo y dinero.

Clasificación por: Manuales / Automatizadas

Pruebas manuales

- Son aquellas pruebas que ejecuta una persona, siguiendo el ‘paso a paso’ definido y verificando los resultados obtenidos contra los esperados.

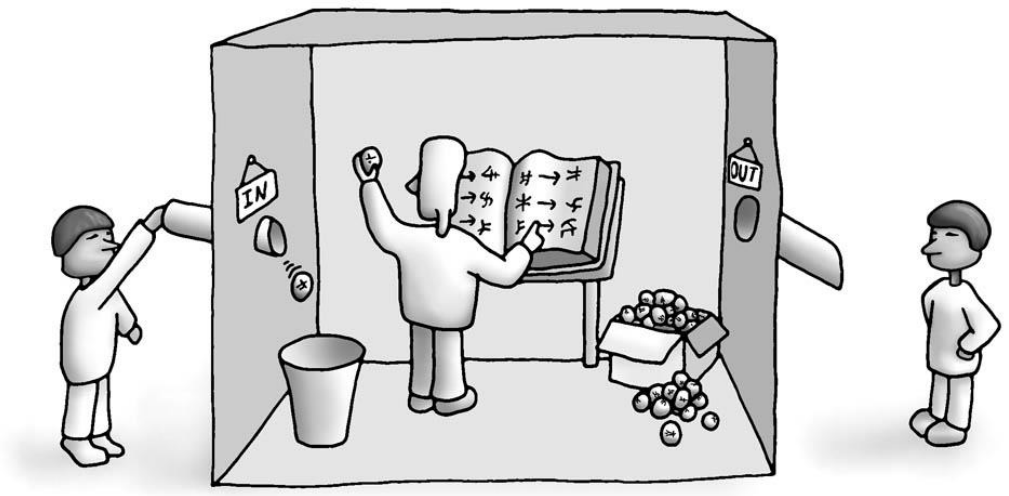
Pruebas automatizadas

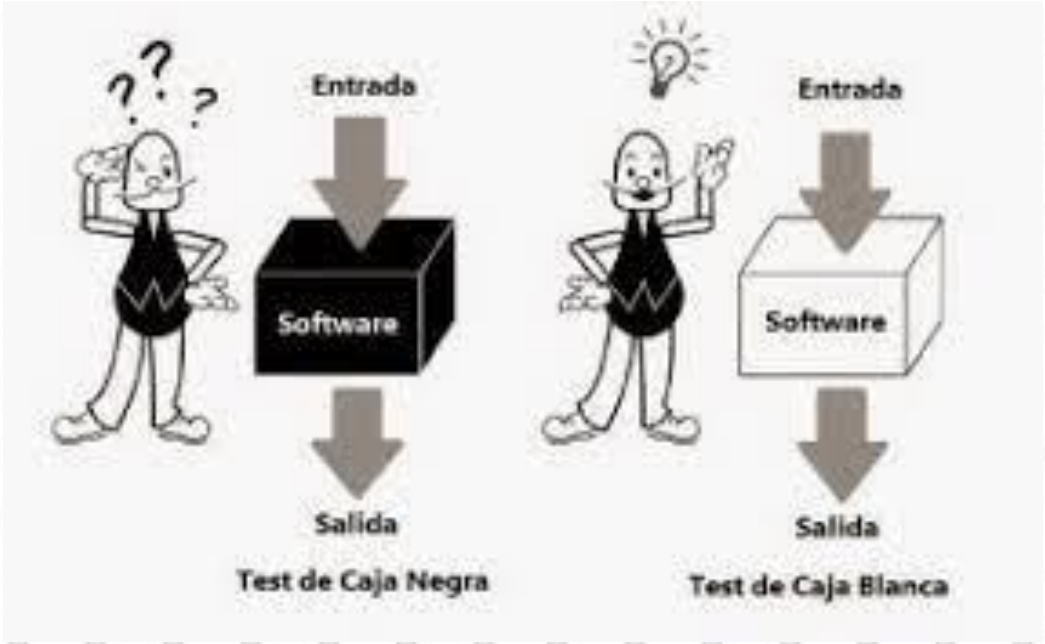
- Consiste en utilizar software especial adicional, donde se configuran / codifican las pruebas y luego se ejecutan sin intervención humana (o con mínima intervención).
- También se realiza de forma automática la comparación entre los resultados obtenidos y los resultados esperados, informando luego cuáles pruebas pasaron y cuáles fallaron.
- Son beneficiosas cuando se precisan realizar pruebas repetitivas o en casos donde la ejecución manual resultase complicado.

Clasificación por: Manuales / Automatizadas

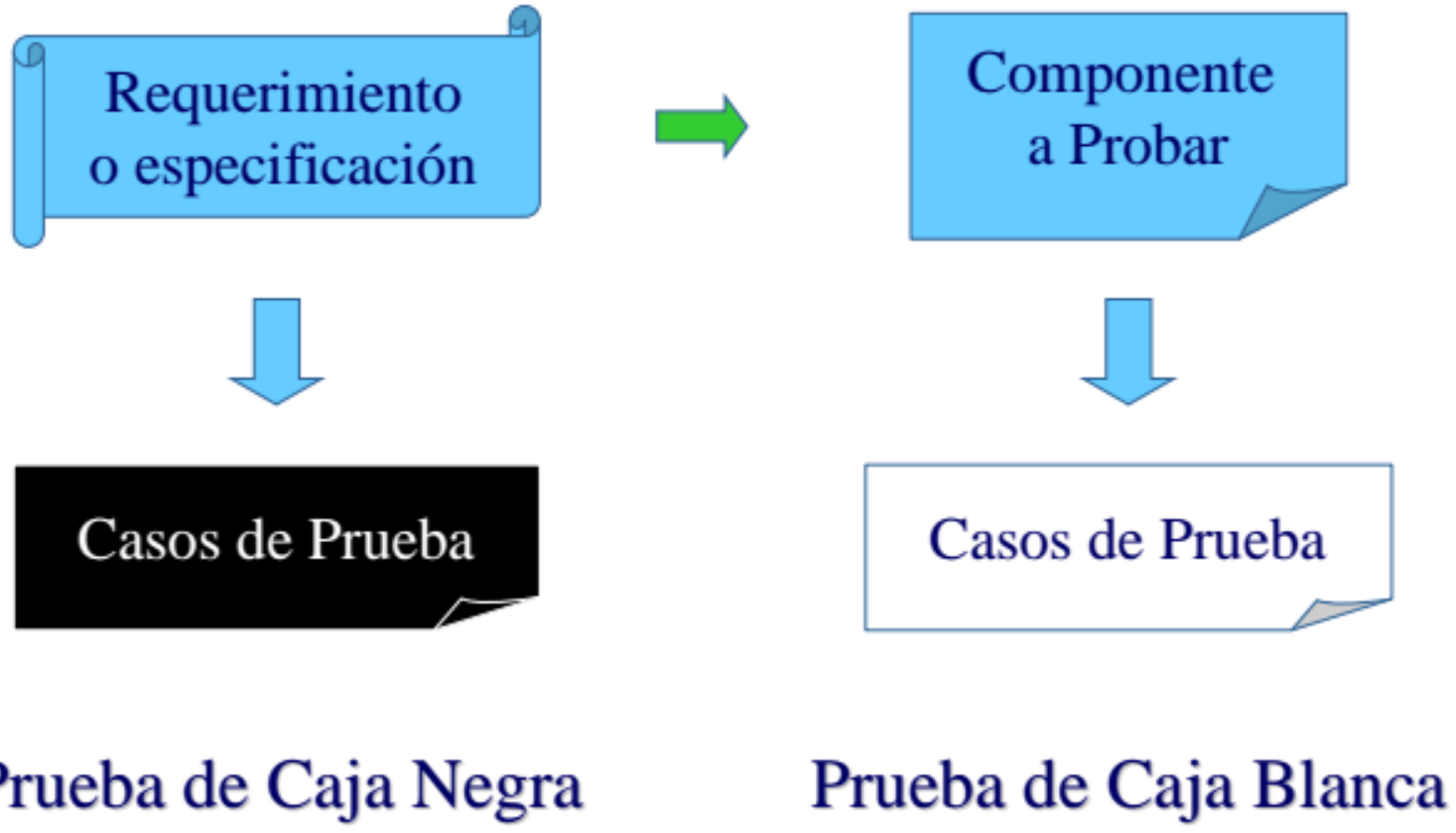
Automatizado y Manual Test Funcional Simulaciones	**Manual** Test Exploratorio Test de Usabilidad Test de Aceptación Alpha y Beta
Test Unitarios **Automatizado**	Test de Performance Test de Seguridad **Herramientas**

Clasificación por: Caja Blanca / Caja Negra



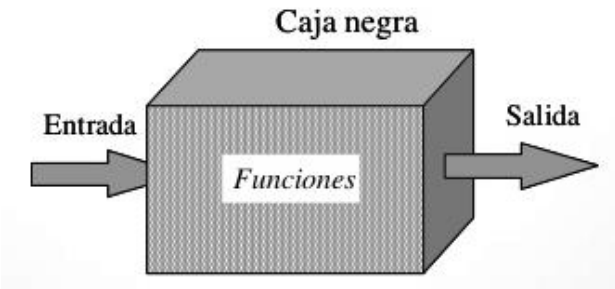


Caja Blanca / Caja Negra



Pruebas de Caja Negra

- Son pruebas *funcionales.* También llamadas pruebas inducidas por datos
- Prueba lo que el software **debería hacer**
- Toma como punto de partida la definición del módulo a probar (definición necesaria para construir el módulo)
- Se limitan a que el tester pruebe con *"datos"* de entrada y estudie como salen, **sin preocuparse de lo que ocurre en el interior** (o sea, ignorando el comportamiento y estructura interna del componente).
- Por ello, también se suelen llamar "pruebas de entrada / salida"



Pruebas de Caja Negra

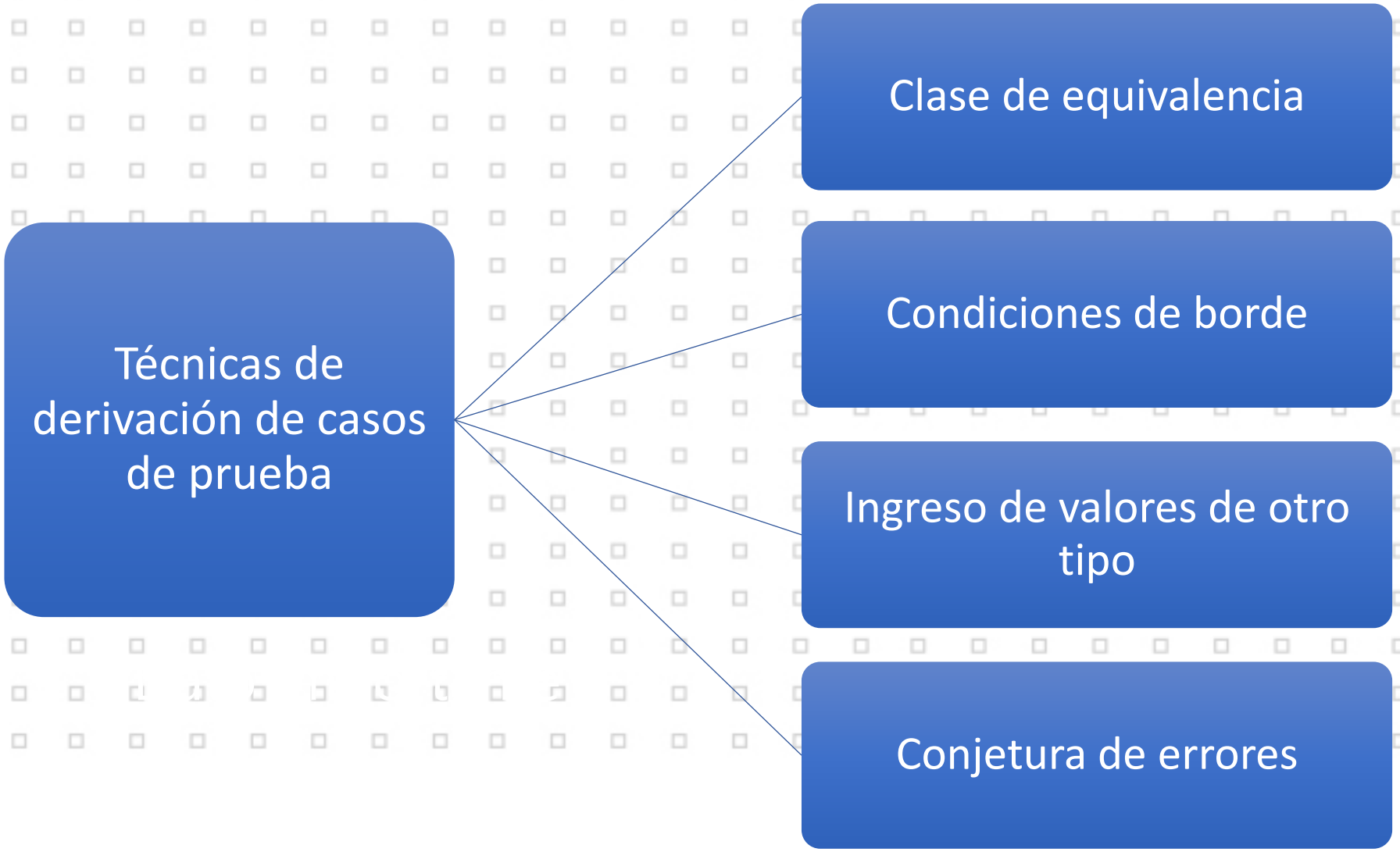
- Las pruebas exhaustivas de caja negra, son imposibles de realizar en la mayoría de los sistemas.
  - Habría que probar con todas las combinaciones de entradas y salidas existentes (que suelen ser muchísimas)
- ¿ La solución?
  - Acotar las pruebas.
- Cómo?
  - Seleccionando subconjuntos de ellas que permitan cubrir un conjunto extenso de otros casos de prueba posibles

Pruebas de Caja Negra

- Una de las mayores dificultades es encontrar un conjunto de pruebas adecuado:
  - suficientemente grande para abarcar el dominio y maximizar la probabilidad de encontrar errores
  - suficientemente pequeño para poder ejecutar el proceso de testing con cada elemento del conjunto y minimizar el costo del testing

- Para poder seleccionarlo existen algunas técnicas que vamos a ver a continuación

Pruebas de Caja Negra



Pruebas de Caja Negra – Clase de Equivalencia

- Se denomina así a la agrupación de los datos de entrada y los resultados de salida, en las que todos los miembros de cada grupo están relacionados
- Cada clase es una partición de equivalencia en la que el sistema se comporta de la misma forma para cada miembro de la clase
- Podemos suponer que la prueba de un valor representativo de cada clase es equivalente a la prueba de cualquier otro valor
- Estas pruebas son llamadas “Pruebas por Partición de Equivalencia” o “Pruebas basadas en subdominios”
- Pasos a realizar:
  - Identificar las clases de equivalencia
  - Definir casos de prueba

Pruebas de Caja Negra – Clase de Equivalencia

1. Identificación de Clases de equivalencia

- Se divide cada condición de entrada en dos grupos:
  - Clases Válidas
  - Clases Inválidas
- Por ejemplo, si tenemos las siguientes opciones de entradas de datos:
  - Rango de valores. Si una condición de entrada especifica un rango de valores (Ej.: sucursal entre 1 y 99).

Clases válidas	Clases inválidas
01 < sucursal < 99	sucursal < 01
	sucursal > 99

Pruebas de Caja Negra – Clase de Equivalencia

1. Identificación de Clases de equivalencia

- Conjunto de valores. Si una condición de entrada especifica un conjunto de valores y existen razones para creer que el programa los trata distintos (Ej.: DNI, CI, PAS)

Clases válidas	Clases inválidas
Uno por cada uno	Todos los que no son esos, por ejemplo: LC, LE

- "Debe ser". Si una condición de entrada especifica un "debe ser" (Ej.: El primer carácter debe ser un dígito)

Clases válidas	Clases inválidas
Primer carácter un dígito	Primer carácter distinto de dígito (ej: una letra)

Pruebas de Caja Negra – Clase de Equivalencia

1. **Identificación de Clases de equivalencia**
   - Si creemos que los elementos de una clase de equivalencia no son tratados en forma idéntica, debemos dividir la clase en clases menores (Ej.: Las suc. de Cap. Fed. son de la 100 a la 130)

Ejercicio

- Supongamos la transacción de alta de datos de un cliente (persona física) en un Banco.
- Definir las particiones para los atributos seleccionados (Trabajaremos solo con algunos al solo efecto de incorporar el concepto de partición)

Pruebas de Caja Negra – Clase de Equivalencia

Ejercicio

- Atributos considerados

Apellido	Char (30) <> “ ”
Nombres	Char (30) <> “ ”
Documento	
– Tipo Documento	Char (3) (DNI / CI / LE / LC / PAS)
– Nro Documento	Num (9) > 0
– Cod. Provincia	Num (2) (01 a 23) o 40
Estado Civil	Char (1) (S / C / V / D / O)
Cantidad de Hijos	Num (2) (0 a 20)
Condición IVA	Char (3) (RI / RNI / EX)
Ingreso Mensual	Num (15) >= 0

Pruebas de Caja Negra – Condición de Borde

- Es una variación de la técnica de partición de equivalencias, que se focaliza en los bordes de cada clase de equivalencia: por arriba y por debajo de cada clase
- La experiencia muestra que los casos de prueba que exploran las condiciones de borde producen mejor resultado que aquellas que no lo hacen
- Cómo hacer?
  - Si en la condición de entrada puede ser un rango de valores (Ej.: sucursal entre 1 y 99) → Seleccionar casos válidos para los extremos del rango y casos inválidos para los valores siguientes a los extremos:

Clases válidas	Clases inválidas
sucursal = 1	sucursal = 0
sucursal = 99	sucursal = 100

Pruebas de Caja Negra – Condición de Borde

- Aplicar el mismo criterio para los datos de salida. Por ejemplo, si la salida es un reporte de hasta 80 registros:

Clases válidas	Clases inválidas
registro = 1	Reporte vacío
registro = 80	registro = 81

- Prestar especial atención a los archivos/tablas vacíos, primer registro / fila, último registro / fila, fin del archivo / tabla, pesos máximos de archivos..

Pruebas de Caja Negra – Valores de otro tipo

- Hasta ahora hablamos del ingreso de clases inválidas del mismo tipo que la clase válida, pero también tenemos que probar el ingreso de valores de otro tipo:
  - Numéricos en vez de alfabéticos
  - Alfabéticos en vez de numéricos
  - Combinaciones de ambos
  - Fechas erróneas
- Muchas veces, la combinación de los datos de entrada es las que produce una clase válida o inválida:
  - Ejemplo: Podría ser que si el estado civil es divorciado, los datos del cónyuge se deben ignorar

Pruebas de Caja Negra – Conjetura de errores

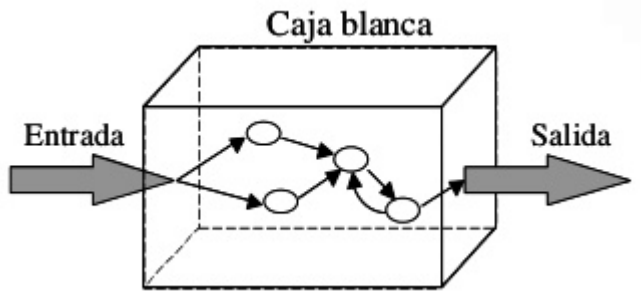
- También llamada Prueba de Sospechas ("Sospechamos" que algo puede andar mal)
  - – Enumeramos una lista de errores posibles o de situaciones propensas a tener errores
  - – Creamos casos de prueba basados en esas situaciones
- La creatividad juega un papel clave
  - No hay una técnica para la conjetura de errores
  - Es un proceso intuitivo
  - Se basa en la experiencia

Pruebas de Caja Negra – Conjetura de errores

¿ Cuándo hacerlo ?

- Un componente, o parte de él, fue hecho “a las apuradas”
- Un componente modificado por varias personas en distintos momentos
- Un componente con estructura anidadas, condiciones compuestas, etc...
- Un componente que sospechamos fue armado por la “técnica de copy & paste” de varios otros componentes

Pruebas de Caja Blanca



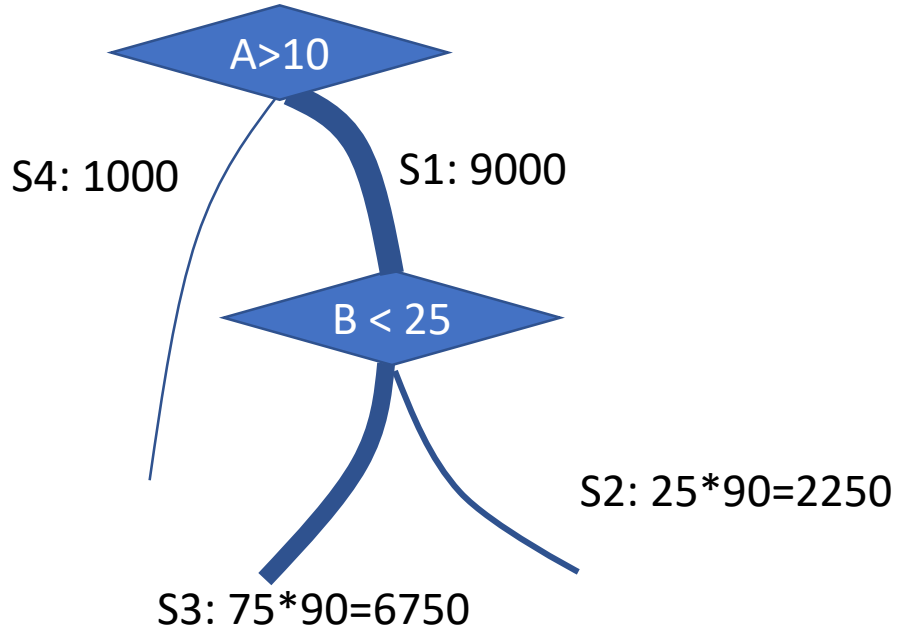
- Son pruebas *estructurales*
- También conocidas como “Glass box”
- Prueban lo que el software **hace.**
- Al igual que la prueba de caja negra evalúa el resultado.
- Pero a diferencia de este, aprovecha el conocimiento interno del código para dirigir la prueba.
- Se basa en cómo está estructurado el componente internamente y cómo está definido
- Se trata de ejecutar todas las sentencias para encontrar errores lo antes posible

Pruebas de Caja Blanca

- No garantizan el cumplimiento de las especificaciones funcionales (son pruebas complementarias a éstas)
- Cada caso de prueba debe probar sólo un camino de todos los posibles, por ejemplo, si la función contiene una condición “if / else”, se deben hacer 2 casos de prueba
- Si aparece “or” o “and” también se deben generar 2 casos de prueba para la misma condición
- El desarrollador al conocer la estructura interna puede dirigirse directamente a las partes del código más riesgosas.
- Generalmente es realizado por/con el desarrollador

Pruebas de Caja Blanca – Flujo de Lógica

```
Input a,b [1..100]
If a > 10
        do S1
        if b  < 25
                do S2
        else
                do S3
Else
        do S4
```



Pruebas de Caja Blanca – Cobertura de Sentencias

- Este nivel es el mas bajo del testeo por cobertura.
- Establece que cada sentencia del software testeado debería ser ejecutada por lo menos una vez
- Aunque se realice la cobertura total (se prueben todas las sentencias del código), se pueden perder muchos caminos posibles
- A veces no es sencillo realizar la cobertura total, por ejemplo en los casos límites donde el software maneja excepciones del tipo ‘baja memoria’, ‘disco lleno’, ‘disco corrupto’, etc.

Pruebas de Caja Blanca – Cobertura de Sentencias

- Ejemplo:
  - Teniendo el código:

```
X=0;
if(a>0)
      x=x+1;
if(b==3)
      y=0;
Print(x,y)
```

La *cobertura total* de este código se puede hacer con el caso de prueba:

a>0 true y b==3 true

Esto deja varios caminos fuera del alcance donde podría haber defectos:

a>0 true y b==3 false
a>0 false y b==3 false
a>0 false y b==3 true

Pruebas de Caja Blanca – Cobertura de Decisiones

- Se prueban los casos para que cada decisión sea evaluada por lo menos una vez en true y false
- Prueba cada salida de un “IF” o “WHILE”
- Tampoco nos asegura la cobertura de todos los caminos.

Pruebas de Caja Blanca – Cobertura de Desiciones

- Ejemplo:
  - Teniendo el código:

```
X=0;
if(a>0)
      x=x+1;
if(b==3)
      y=0;
Print(x,y)
```

Bastaría con 2 casos de prueba para cubrir las decisiones:

(a>0 y B=3) **true** y **true**
(a<0 y B=1) **false** y **false**

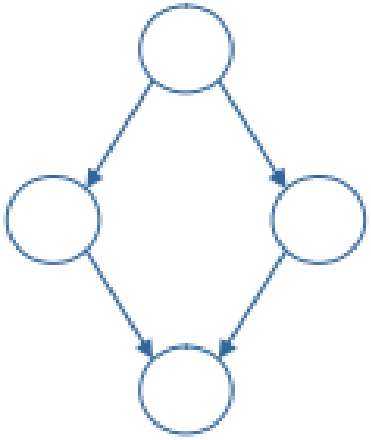
Nos perderíamos true-false y false-true

Pruebas de Caja Blanca – Cobertura de Rama o Caminos

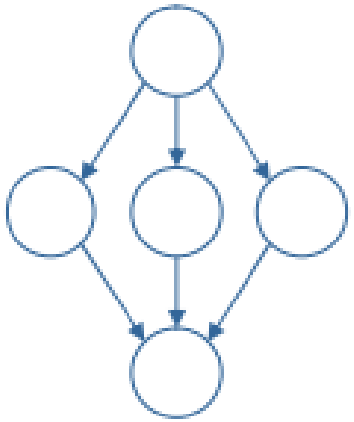
- También llamada Prueba del Camino Básico
- Se prueban todos los caminos independientes, o sea todas las posibles ejecuciones
- La cobertura de ramas es indiscutiblemente deseable; pero habitualmente es un objetivo excesivamente costoso de alcanzar en su plenitud.
- Una forma de abaratar los costos es realizar las pruebas con herramientas de automatización.

Pruebas de Caja Blanca – Cobertura de Rama o Caminos

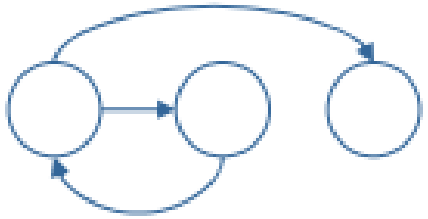
- Se representa el flujo de control de una pieza de código a través de un grafo de flujo, teniendo en cuenta:



if then else

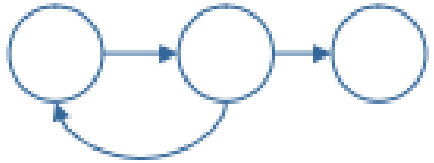


case, o selección múltiple



do while

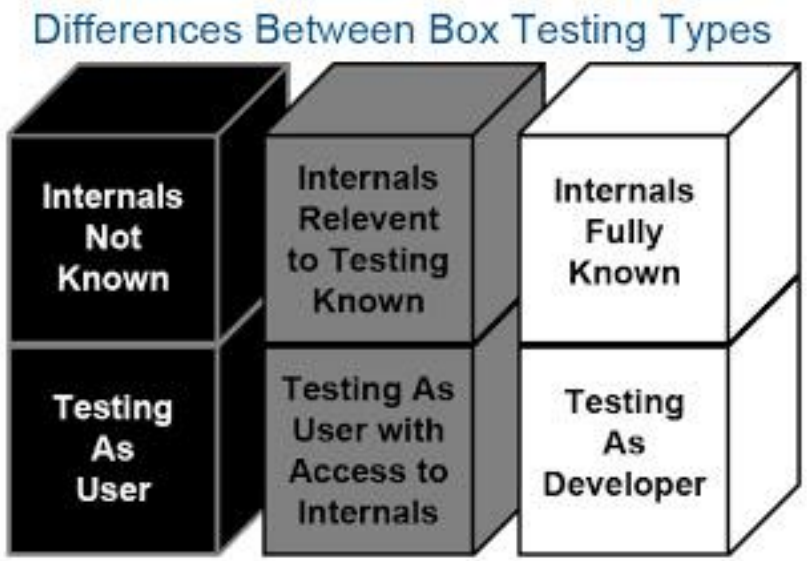
secuencia



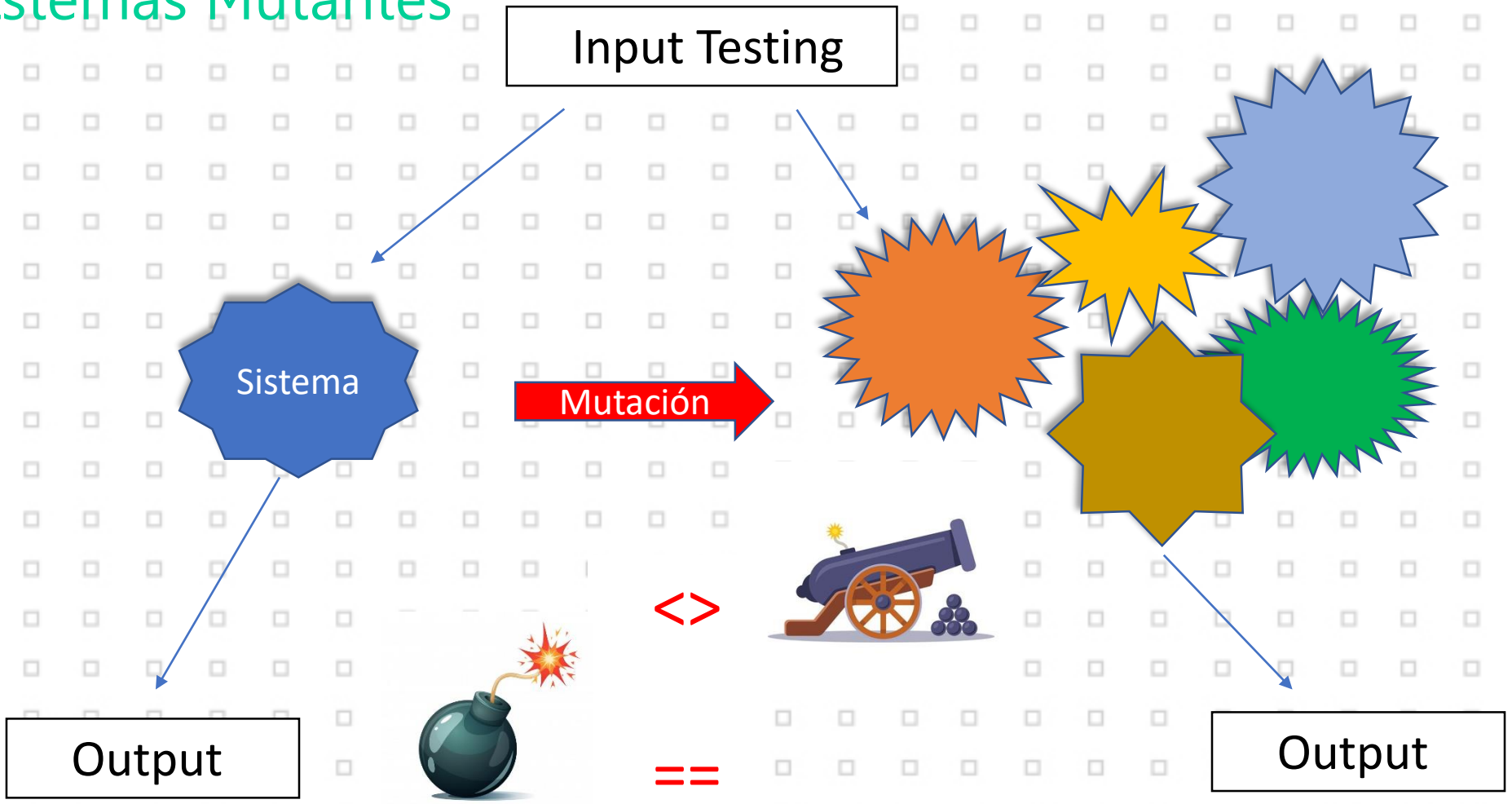
repeat until

Pruebas de Caja Gris

- Prueba que combina elementos de la caja negra y caja blanca
- No es caja negra porque se conoce parte de la implementación o estructura interna y se aprovecha ese conocimiento para generar condiciones y casos que no se generarían naturalmente en una prueba de caja negra
- El conocimiento es “parcial”, no “total” (lo que sería caja blanca)



Sistemas Mutantes



PREGUNTAS

Fin de a Unidad 8.