

Entonces por otros pruebas de caja negra, como otras e incluso conjuntos de pruebas de caja blanca, decimos que las pruebas de caja blanca son pruebas estructuradas porque vemos la estructura del código, también son sencillas como dice los porque algunas como el hacer una caja transparente no, en caja blanca después se puede ver internamente que lo que hace, se prueba lo que el software hace, se evalúa el resultado aprovechando el conocimiento interno del código pero dirige la prueba, aprovechando lo que sabemos del código para poder dirigir la prueba hacia donde nosotros queremos, estas pruebas de caja blanca no garantizan el cumplimiento de las especificaciones funcionales, son pruebas complementarias a las funcionales, las funcionales son las de caja negra, estas son complementarias, en cada caso de prueba debe probar solo un camino de todas las posibles, por ejemplo si la función contiene una condición "if/then" si pasa este entonces, si no sale otra cosa, se deben hacer dos casos de prueba, cuando hay una bifurcación en el código se deben hacer dos casos de pruebas, si aparece un "or" un "and" se deben, uno y además otro otro, si se cumple esta condición o esta otra condición se deben generar dos casos pruebas para la misma condición y así, al menos poder cubrir todo todo lo tipos por donde va el código, que se acuerden que hablamos hecho un flujo no, entonces el código puede ir por un lado y por el otro bueno hay que tratar de cubrir todo el árbol, esto tiene una estructura de árbol, con un nodo principal y se van a distintas ramas, distintas ramas que salen de ahí, bueno lo idea es poder cubrir todo, todo, todo, todo ese lugar el desarrollador al conocer la estructura interna puede dirigirse directamente a las partes del código más complejas generalmente realizadas más más veces por el desarrollador o con el desarrollador, entonces así tenemos el ejemplo el flujo de flujo, ahí tenemos un programa que dice bueno este hace un input de A y B entonces leyendo la parte izquierda de la pantalla e input A y B, ingresa dos valores al sistema cada valor se divide 1 hasta 100 ingresamos los números válidos de 1 hasta 100, pregunta si el primer número el dos mayor a 10 entonces hace la sentencia 1 y si lo es menor a 20 entonces hace la sentencia 2 y si no hace la sentencia 2 y si no se cumple el de arriba el mayor a 30 hace la sentencia 4, esto es lo que está haciendo esta este código, entonces al vamos al gráfico, a la derecha, tenemos la representación de este mismo programa el símbolo siempre está está determinando que hay una pregunta entonces el cambio es una pregunta, si A es mayor a 10, si, si no va a pasar cuando, va a pasar el 90% de las veces y entonces vamos vamos para esa lado o si, si no vamos para el otro lado si aparece no, va va a pasar el 10% de las veces, entonces se divide o se divide en el flujo, una más gruesa porque van muchas más opciones por ahí, otro más fino porque son menos opciones por ahí, aparece cuatro cuatros con un asterisco, todo más que así caso, si a nosotros queremos todos los números no, entonces haciendo una estadística pero si nosotros ingresamos por ejemplo tenemos un número 4 que puede ir de 1 a 100, el número 8 que puede ir de 1 a 100 veces con 10 mil combinaciones, entonces con 10 mil combinaciones hay 1000 que van a ir por un lado y 9000 por otro y después pregunta si va menor a 20 entonces también hay más esa otra pregunta, un 70% que va por un lado y un 30% que va por el otro, se multiplica por algunas para saber la cantidad de veces se multiplica por lo que entonces de la condición anterior, por que era un 60%, entonces tenemos 1.250 veces por un lado y 4.750 veces por el otro, así tenemos como vía el flujo va adelante del sistema, si nosotros ingresamos números algunos, valores más de espectro, sabemos que hay 6.750 veces que va a ir por un lado 1.250 por otro y 1.000 veces que va a ir por otro esto es importante saber para saber dónde con más probabilidad va a ir el sistema, esto esto es el nivel más bajo del testing por cobertura estadística que cada sentencia del software testeado debería ser ejecutada por lo menos una vez, ya me quedaba tranquilo el el que cada cosa que está escrita en el código por lo menos pasa por ahí una vez del sistema y una vez se ejecuta, entonces qué pasa, decimos que aunque se realiza más cobertura total cubrimos todo el código, se puede perder muchas caminos posibles, como es esto, a veces no se consigue realizar la cobertura total, por