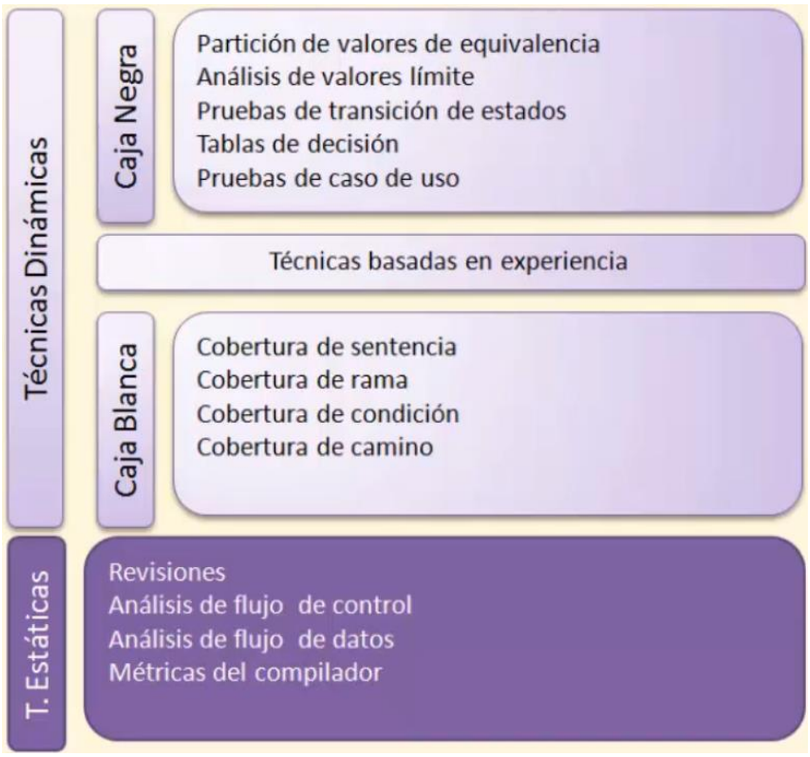
TIPOS DE TECNICAS



Ambas técnicas son complementarias, ya que detectan diferentes tipos de errores, el test estático lo que hace es encontrar defectos y el test dinámico encuentra fallas en el resultado esperado.

Nota. – *Repasando lo que se vio anteriormente, fallo era una manifestación física o visible del defecto, era cuando se daba durante la ejecución de la aplicación.*

1.- Técnicas Estáticas, es el examen manual, análisis automatizado del código y se basa en cualquier otra documentación del proyecto sin que tengamos que ejecutar el código propiamente dicho.

Normalmente este tipo de técnicas como son: Revisiones, Análisis del Flujo de Control, Análisis del flujo de datos o las métricas del compilador, suelen realizarse antes de las técnicas dinámicas porque los defectos que encontremos con este tipo de técnicas van a ser al principio del Ciclo de Vida, por lo tanto, van a ser menos costosos para corregirlos que los detectados durante la ejecución de las pruebas.

- Revisiones, *(una de las formas más utilizas en las técnicas estáticas),* según el estándar EEE1028 que habla sobre las revisiones en el Software, una "Revisión" es la **Evaluación de un producto para poder detectar discrepancias respecto de los resultados planificados y de esta forma recomendar mejoras.**

Consiste en analizar un determinado objeto de prueba, pero sin tener que ejecutarlo. Ej. un script, un código fuente, un requisito, una especificación, etc. lo que se debe evaluar son los estándares de programación, los flujos de datos, **no el resultado que vamos a obtener luego de ejecutar el código.**

El principal objetivo es que nos permitirá descubrir defectos en las especificaciones, en el diseño, en las especificaciones de interfaces cualquier otro tipo de documento. Las revisiones pueden ser:

a) **Revisión Formal,** tiene ciertas etapas o actividades que se deben cumplir o completar las cuales son:
    1. **Planificación** donde se deben definir los criterios a utilizar, (que tipo de revisión a aplicar, cual es la lista de comprobación, quienes van a participar, que rol va a tener cada participante, y los tiempos en los cuales se va a realizar).
    2. **Criterios de Entrada/Salida,** que es lo que estamos revisando y cuando vamos a parar de revisar esto.
    3. **Reunión de inicio de proceso y distribución de documentos (Kick-Off),** donde definen que es lo que debe hacer cada uno.

4. **Identificación de defectos,** sobre los documentos, armar preguntas, consultas, comentarios, etc.
5. **Reunión de revisión,** donde se va a discutir los resultados obtenidos, el resultado de los comentarios, la preguntas, etc. y de va a registrar los **resultados** y recomendaciones para mejorar.
6. **Reconstrucción,** la persona encargada va a corregir los defectos, evaluar las mejoras si hubo y las va a implementar, luego nuevamente una reunión de seguimiento.
7. **Comprobación de los criterios de salida,** si se cumplen o no, en caso de que no se cumplan se inicia nuevamente con la reunión, etc., hasta que se termine con la revisión.

b) **Revisión Informal,** (o revisión entre pares), es iniciada por el autor o creador de documento del código, de la especificación, etc. y hay uno o dos desarrolladores colaboradores normalmente se realiza para la revisión de código. La documentación es informal y los resultados pueden ser registrados en forma de una lista de acción o de actividades a realizar. El objetivo principal de esta revisión es invertir poco para obtener un posible beneficio.

c) **Revisión Guiada, es** un tipo de revisión liderada por el autor del objeto que se está presentando y a lo largo de la presentación este autor ira mostrando el objeto a revisar, habrá un par de revisores que harán preguntas y comentarios para tratar de comprender el objeto y detectar desviaciones o áreas que representen algún tipo de problema. El objetivo es ganar conocimiento del objeto a probar y entender el funcionamiento y detectar defectos. Ej. diseño preliminar de interfaz de usuarios, modelo de datos, etc.

d) **Revisión Técnica,** puede ser formal o informal con la diferencia que aquí participaran técnicos expertos, preferentemente externos a la organización y además es liderada por otra persona que no es el autor denominada **Moderador**. Ej. chequear estándares, etc.

e) **Revisión de Inspección,** es la revisión más formal porque contiene proceso de preparación, ejecución, documentación, de seguimiento, tiene roles bien específicos, listas de comprobación ya definidos, se utiliza métricas, criterios de entrada y salida para aceptar o no el producto. Contiene todas las etapas que hemos visto previamente.
Nota. - El propósito principal de esto es detectar defectos utilizando un método estructurado.

2.- Técnicas Dinámicas, estos tipos se diferencian básicamente en la forma en la que se obtienen los casos de prueba, y para tener una **buena cobertura** deberíamos poder aplicar las tres técnicas porque cada una va a descubrir defectos diferentes.

2.1.- Técnicas de Caja Blanca, se realizan cuando el tester puede acceder al código fuente de la aplicación y también a sus algoritmos y estructura de datos.

La persona que está probando necesita tener habilidades de programación, conocer el framework de desarrollo que se está usando, tiene que saber también que es lo que hacen las distintas clases y métodos. Esta técnica se realiza sobre las funciones internas de cada módulo, lo que tiene mayor foco es la estructura del programa *(la parte interna vemos lo que hay adentro sin importar las entradas y salidas)*.

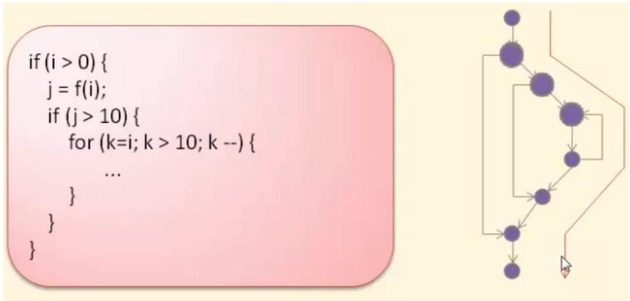
Los casos de prueba aquí van a garantizar que todos los caminos que existen en el módulo hayan sido ejecutados tanto las decisiones positivas como las negativas. El objetivo principal es: *poder alcanzar el código que no se puede alcanzar con caja negra, poder encontrar inconsistencias, código que no se utiliza o no tiene razón de ser*, *permite optimizar el funcionamiento de la aplicación desde adentro*.

Se puede utilizar tanto en los niveles de componentes, de integración y de sistema.

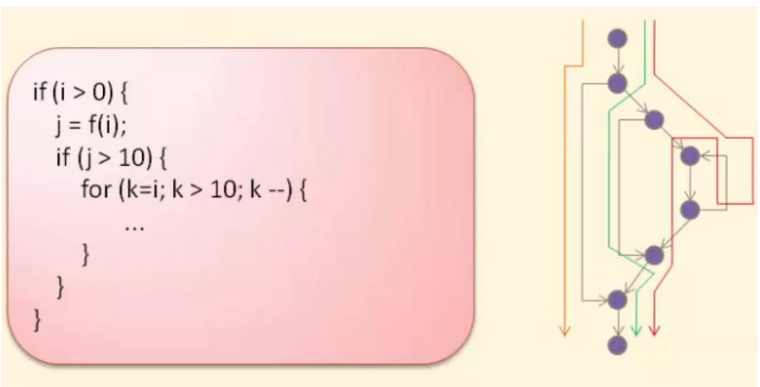
Una de las ventajas es que los métodos pueden aplicarse en etapas tempranas del sistema, y encontrar fallas en etapas tempranas hace que el costo de corregir sea mucho menor a etapas finales, y se tiene una cobertura casi total a lo que es la estructura del sistema.

Tipos de técnicas de caja blanca: ¿Que es cobertura? Es la medida en que un conjunto de pruebas ha probado una estructura en particular, esta medida va a ser expresada en porcentaje.

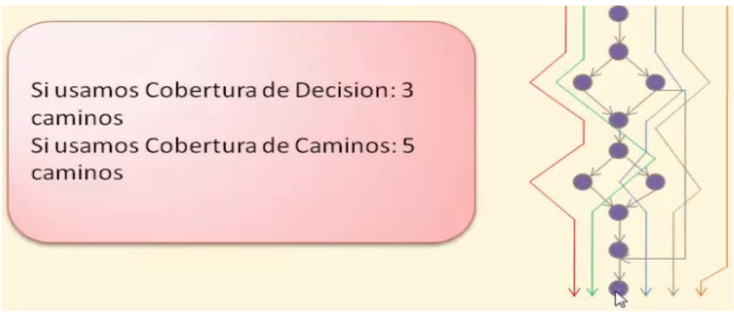
-**Cobertura de sentencia,** comprobamos el número de sentencias ejecutadas, el foco es la sentencia de un programa. →¿Qué caso de prueba vamos a necesitar? Será satisfactorio si todas las sentencias son ejecutadas por lo menos una vez. Se utiliza un gráfico de flujo de control (las instrucciones o las sentencias se representan mediante nodos, y el flujo de control se representan por las aristas o la flecha), el cálculo de la cobertura es: **Cobertura = (Nro. de Sentencias ejecutadas/Nro. total de sentencias) *100**. El objetivo principal de estas pruebas es detectar aquellos bloques de código que no se utilizan. (código muerto) Ej.



- Cobertura de Decisión, comprobamos el número de aristas ejecutadas. (una decisión es un punto en el código en el cual se va a producir una ramificación) entonces vamos a satisfacer el criterio de cobertura de decisión si todas las condiciones del programa son ejecutadas al menos una vez por verdadero o por falso. También utiliza el grafico de flujo de control, pero se centra en el flujo de control del segmento de un programan (en las aristas del diagrama de flujo). Nos preguntamos qué caso de prueba necesito para cubrir por lo menos una vez una de estas aristas. **Cobertura = (Nro. de Decisiones ejecutadas/Nro. total de decisiones) *100.**

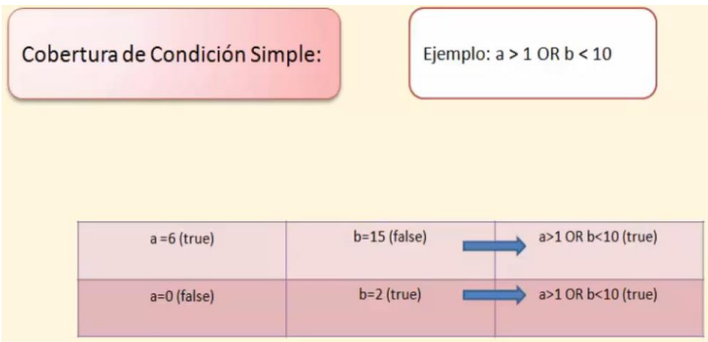


- Cobertura de Camino (combinación de segmentos de programa, en un diagrama de flujo), es una determinada secuencia de nodos y aristas alternados. Primero comprobar el número de caminos linealmente independientes que se han ejecutado en el diagrama de flujos de la unidad o componente que se está testeando. Nos centramos en la ejecución de todos los posibles caminos que se puedan lograr través del programa. Ej. un camino va a ser una vía única desde el inicio hasta el final del programa del diagrama de flujo. El objetivo es alcanzar un porcentaje definido previamente de cobertura de camino. Cobertura= (Nro. de caminos ejecutados/Nro. total de caminos) *100. La cobertura de camino es mucho más exahaustativa que la cobertura de sentencia y de decisión y no es fácil lograr una cobertura de camino de 100% (solamente se logra en programas que son muy simples).



- Pruebas de Condición y cobertura, es una técnica un poco más complicada que no evalúa la sentencia en sí, sino lo que hay dentro de la sentencia, de las condiciones múltiples que existen dentro de las sentencias. Van a ser el porcentaje de todos los resultados individuales de condición que afectan al resultado de la decisión lo que se hace aquí es detectar defectos que provienen de implementar condiciones múltiples (constituidas por combinación de condiciones atómicas) S e combinan mediante el uso de operadores lógicos, OR, AND, etc. Hay tres tipos de técnicas:

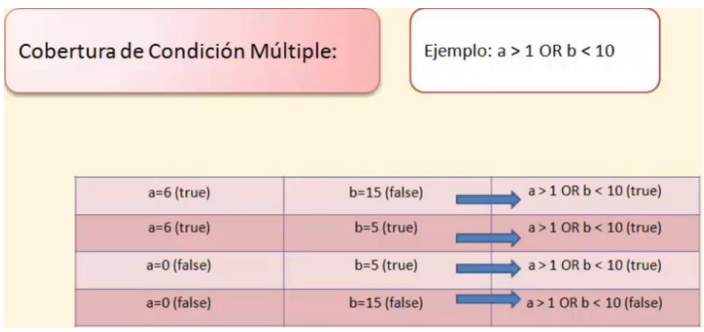
a) **cobertura de condición simple:**



Las subcondicones deben tomar al menos una vez el valor verdadero y falso.

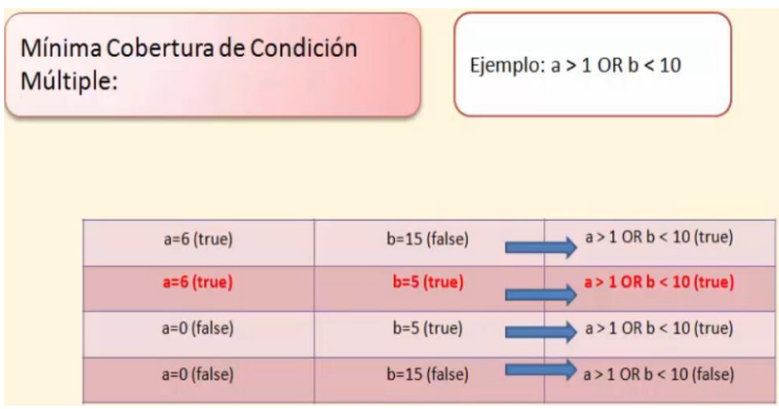
Combinamos estos valores con dos casos de prueba. Vamos a lograr toda la cobertura porque cada condición atómica o subcondicion ha tomado dos valores diferentes (verdadero/falso).

b) cobertura de condición múltiple:



En este caso usaremos todas las combinaciones posibles que tengamos como casos de prueba (siempre y cuando sean reales), será mejor que el anterior porque vamos a cubrir todas las sentencias y decisiones, pero nos vamos a encontrar con que las ejecuciones de algunas decisiones no son posibles.

c) mínima cobertura de condición múltiple:



Se basa en crear la menor cantidad de casos de pruebas necesarias, para lograr se debe va a tomar en cuenta una sola regla:

SOLO SE VA A CONSIDERAR COMO CASO DE PRUEBA, CUANDO EL CAMBIO DEL RESULTADO DE UNA SUBCONDICION (CONDICION ATOMICA), CAMBIA EL RESULTADO TOTAL DE LA CONDICION COMBINADA.