10/07 Testing de software

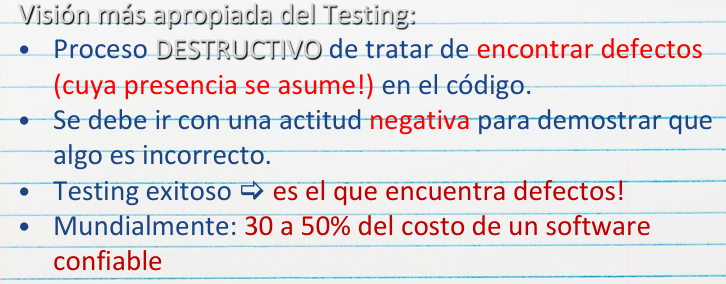
Monday, October 7, 2024

5:19 PM

**Testing:** es un proceso destructivo con el objetivo de encontrar defectos porque nosotros asumimos que en el software hay defectos.

Un test es exitoso cuando encuentra errores/defectos. Por lo tanto un desarrollo exitoso nos conduce a un test no exitoso.

El testing se lleva entre el 30% y el 50% del costo de hacer software.



**Error vs defecto**

* El error se descubre en la misma etapa en la que yo estoy trabajando.
* Defecto quiere decir que ese error que nosotros detectamos lo trasladamos a etapas posteriores

Por lo tanto nosotros en el testing encontramos defectos de errores que se hicieron en la etapa de implementación

Cuando encotramos defectos hay dos aspectos importantes:

* **Severidad --** Tiene que ver con grabedad del defecto que yo encontre, la severidad es bloqueante, debido a ese defecto yo no puedo seguir con el caso de prueba
  + Clasificacion de la severidad:
    - 1 -- Bloqueante 2 – Crítico 3 – Mayor 4 – Menor 5 - Cosmético
* **Prioridad --** Tiene que ver con la urgencia que nosotros tenemos para resolver ese defecto.

**Niveles de Prueba**

* **Pruebas Unitarias --** acá lo que encontramos son erres mas que defectos pero en terminos teórico primero lo construye y después lo corrige
* **Pruebas de integracion --** Nos aseguran que conectamos dos componentes, se va haciendo una estrategia incremental para ir sumando los distintos componentes y ver como se conectan entre si.
* **Pruebas de sistemas --**  Mas amplias, pruebo el sistema en toda su escala.
* **Pruebas de aceptacion --** son ejecutadas muchas veces por el usuario final, objetivo tambien es encontrar confianza en el cliente.

**Ambientes que tenemos en el desarrollo de software**

Ambientes, son los lugares en donde se trabajan para la construccion de software.

* **Desarrollo**
* **Prueba --** Aca por lo general se prueban integracion y algunas de pruebas unitarias
* **Pre-Produccion --** Cuando el software
* **Produccion --**  Enotorno en donde el software esta en produccion

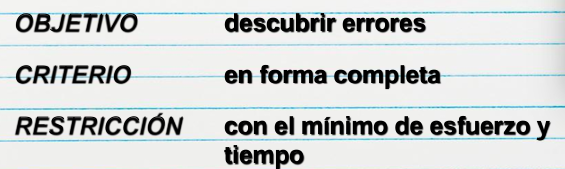
**Caso de pruebas**

Establesco que datos y variables voy a usar a la hora de hacer de hacer las pruebas.

Ej tengo que escribir que usuario voy a usar, que pruebas voy a hacer

* Set de condiciones o variables bajo las cuales un tester determinará si el software está funcionando correctamente o no.
* Buena definición de casos de prueba nos ayuda a REPRODUCIR defectos

Me tiene que decir acciones y con que datos voy a ejecutar

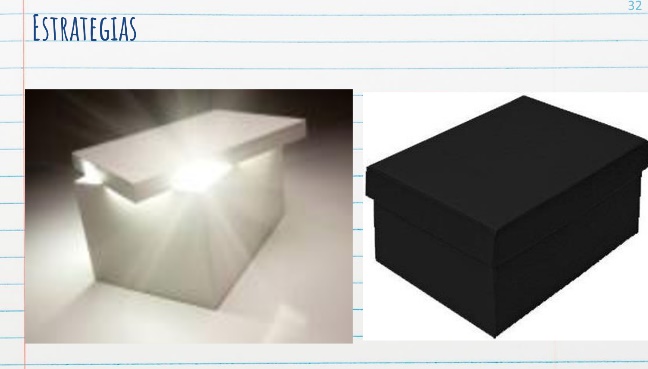


El testing exautivo de todas las opciones que nos presenta el software es inviable, o imposible,

**Condiciones de Prueba**

Tiene que ver con el contexto del sistema que yo tengo que tener para poder hacer la prueba.

**Estrategias/Métodos**



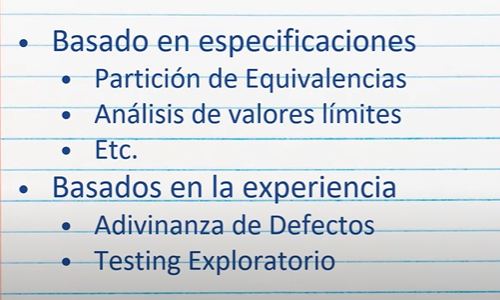
El objetivo es abarcar la mayor cobertura con mis pruebas con el menor esfuerzo

* **Métodos de caja negra**
  + Es testing dinamico, se hace ejecutando el codigo
* **Métodos de caja blanca**
  + Es testing estatico, se hace mirando el codigo
  + Apunta a errores de codigo o encontrar codigo malisioso

El problema es que cada cosa que yo hago o encuentro en uno no lo encuentro en el otro

Para llevar a adelnate testing usamos metodos

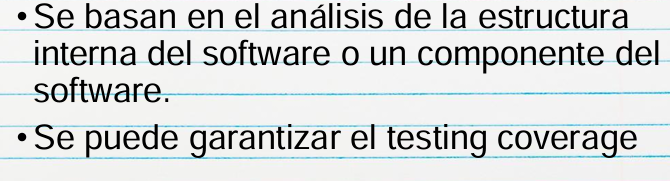
**Metodos en caja negra pueden dividirse en dos:**



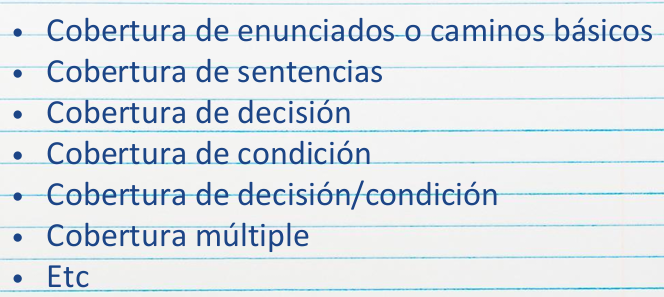
Basado en especificaciones (vemos en el practico) estan disenados para ayudarnos en forma sistematica , nos da cuentos casos de prueba y que casos de pruebas deberiamos disenar

Se llama de caja negra porque nosotros definimos las entradas y esperamos las salidas, no nos interesa como llega a las salidas

**Metodo de caja blanca, lo que hace es mirar adentro del codigo**



Hay distintos métodos:



**Ciclo de pruebas o Ciclo de test**

Un ciclo de pruebas abarca la ejecución de la totalidad de los casos de prueba establecidos aplicados a una misma versión del sistema a probar.

Cuando yo pruebo todos los casos de prueba y termina con un informe, puede pasar que no empiece osea por ejemplo que no funciona el logue entonces no se puede seguir y termina el ciclo de prueba

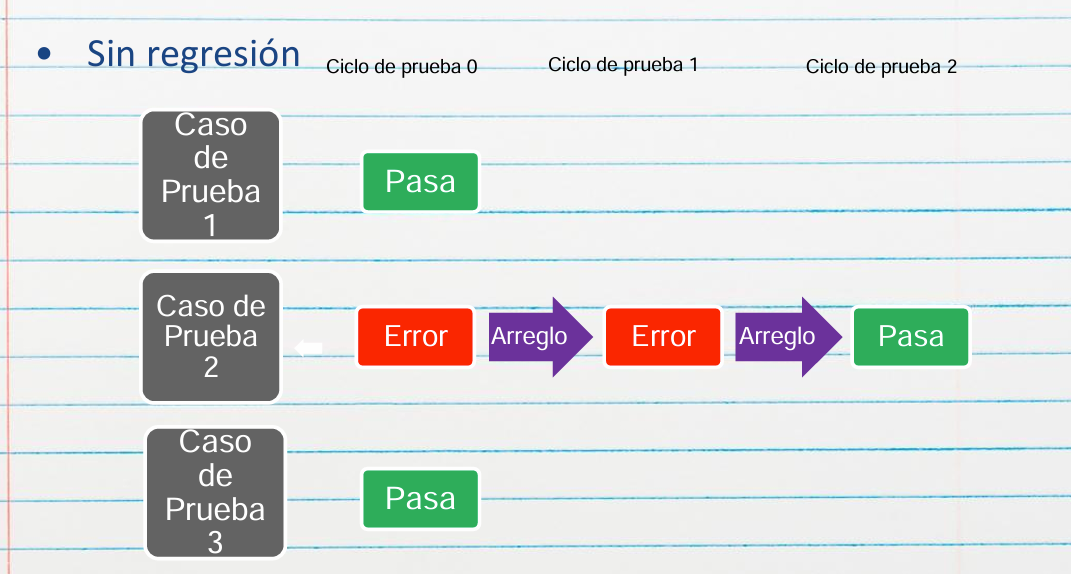
**Regresion**

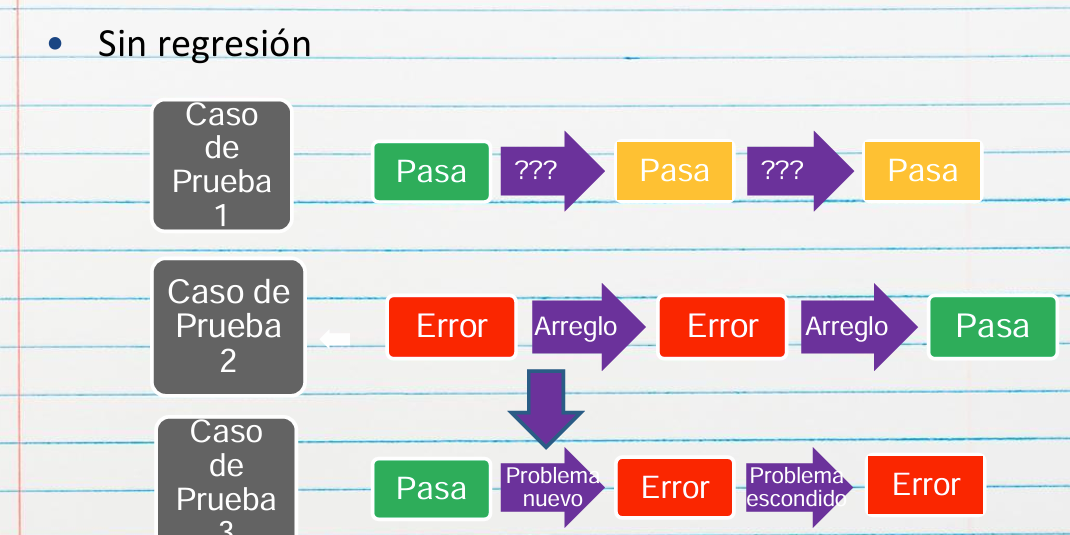
El primer ciclo de prueba que se ejecuta lo llamamos ciclo cero

Cuando nosotros corregimos un defecto, normalmente se introducen mas defectos

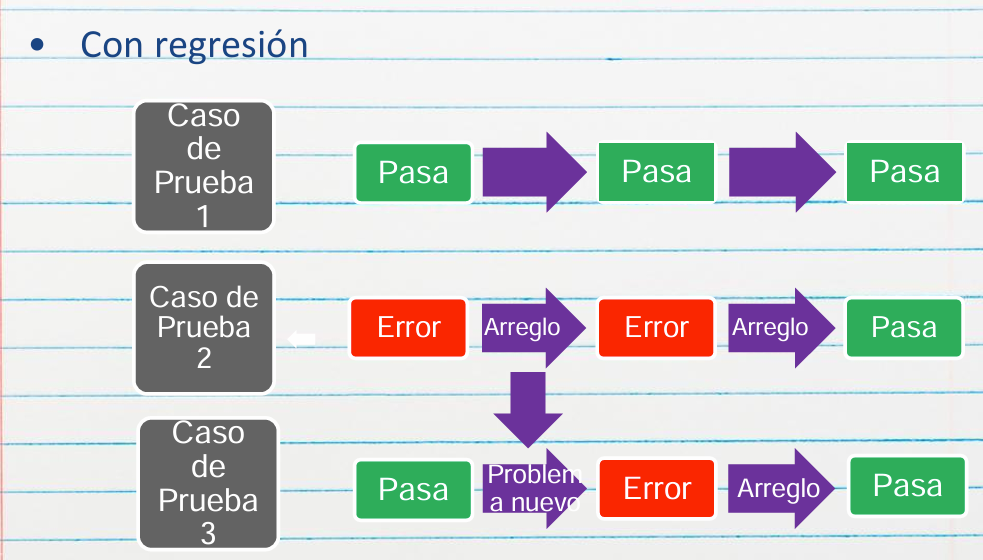
Al concluir un ciclo de pruebas, y reemplazarse la versión del sistema sometido al mismo, debe realizarse una verificación total de la nueva versión, a fin de prevenir la introducción de nuevos defectos al intentar solucionar los detectados.

Básicamente te dice que vuelvas a controlar la nueva versión para controlar que no hayamos introducidos nuevos defectos

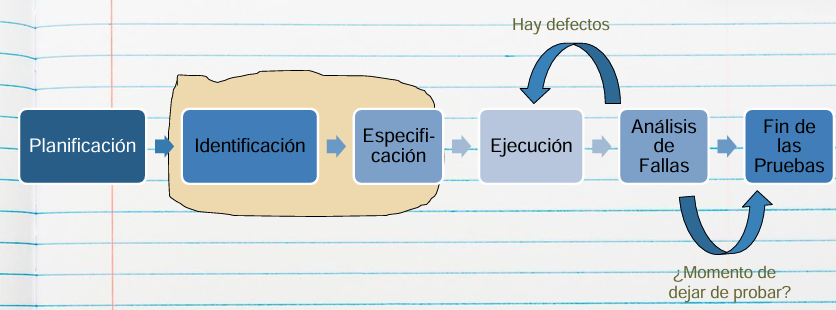




Con regresion trato todo como si fuera un ciclo cero



**Proceso de Pruebas**

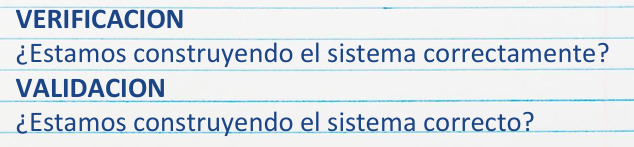


* **Planificacion --** Esta enfocado en un proceso tradicional, no hay en un proceso gestionado por scrum
* **Identificacion**
* **Especificacion**
* **Ejecucion --** Pueden ser o no automatizadas
* **Analisis de fallas**

Ideal es derivar los caso de prueba de los requerimientos para hacer validacion y verificacion, el problema es que muchas veces los requerimientos estan mal escrito, entoces muchas veces queda armar los casos de pruebas usando el programa pero solo hacemos verificacion

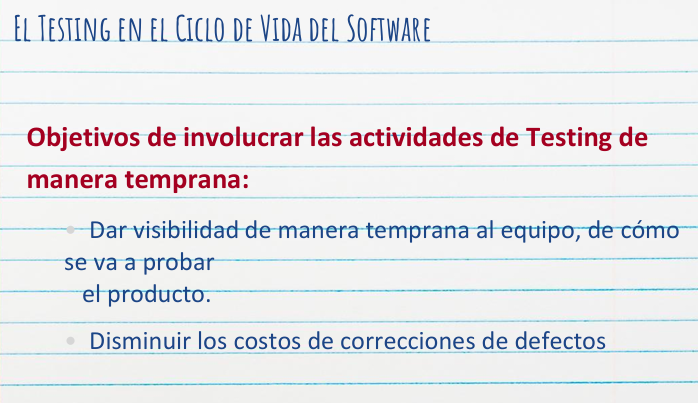
No se puede certificar que no hay defectos, solo se puede certificar los defectos que podemos reproducir

**Conceptos de Validación y Verificación**



Sistema correcto, lo que nosotros construimos corresponde con lo que el cliente quiere

Verificacion, es evaluar si estamos construyendo el sistema libre de defectos



El testing no inicia cuando termina el codigo

El testing no es probar que el software funciona

**Cuanto testing es suficiente**

El nivel de riesgo y los costos asociados al proyecto, nos ayudan a