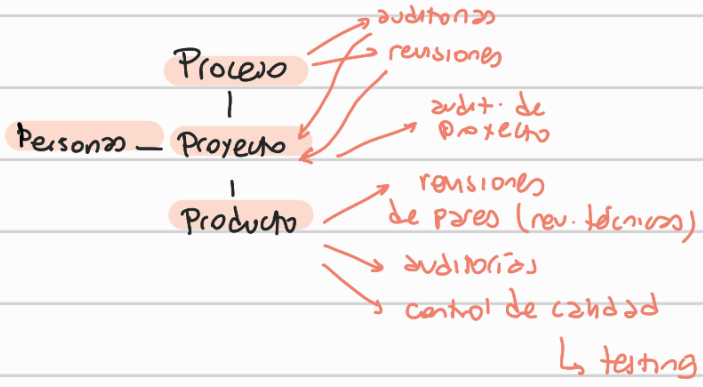
05/10 Clase que viene → lean y kanban | test ágil → libro lisa ??



Testing de Software (parte de UY)

↓

PPT 13

(todo esto es la intro)

- Aseg. de calidad → conj. de acciones que podemos hacer p/ evitar los errores y evitar que se conviertan en defectos

- Prevenir defectos es de importancia principalmente económica (evita costos a futuro)
- P/ asegurar calidad nos centramos en el producto
  - ↳ detección temprana de errores
  - → técnica principal = revisiones de pares, rev. técnicas (de colegas)
    - Formales: - inspección de sw
    - Informales: - walkthrough

    | - le hacemos rev. técnicas a cualquier artefacto que otro construya

- El aseg de calidad de un proceso se hace a dos niveles, en el mismo proceso y después en el proyecto (lo dije en el proceso

- Auditoría producto → conf física y conf funcional
  - → hay una audit de proyecto donde se verifica que se cumpla con el proceso
  - → hay también auditorías informáticas → son al producto mientras se está haciendo

- La calidad del producto depende de la calidad del proceso que se utiliza

- TESTING → proceso DESTRUCTIVO que trata de encontrar defectos cuya presencia se asume en el código.
  - → se está intentando cambiar el concepto con el test. ágil
  - → se va con actitud negativa

→ la tiene que realizar alguien diferente a quien programó

→ es la actividad más cara en la prod. de software (mientras más crítico, más cro)

→ el testing no certifica nada (no saber si ya están todos los def. solucionados)

- **Error** → cuando lo encuentro en la etapa que fue realizado
- **Defecto** → cuando se traslada a otra etapa y se encuentra ahí → testing encuentra defectos
  ↳ pueden provocar fallas
- **Severidad** → qué tan grave es el defecto que se encuentra
  → viene asociada a un contexto

Severidad	
1 – Bloqueante	→ sist. no funciona
2 – Crítico	
3 – Mayor	no muy claras
4 – Menor	
5 - Cosmético	

- **Prioridad** → viene del lado del cliente
  → desde el punto de vista del negocio, que tan rápido lo necesito, define impacto, por ahí algo cosmético cuesta mas arreglar que un bloqueante

Prioridad
1 – Urgencia
2 – Alta
3 – Media
4 – Baja

Niveles de Prueba

- **Pruebas unitarias** → las hace el desarrollador
- **Test. de integración** → prueba de interfaces (wtf)
- **Test. de sistema** → test. de versión, viene de ágil, se testea una versión xd
- **Test. de aceptación** → prueba de aceptación de usuario → lo hace el usuario, se ubica en el workflow de despliegue, en scrum es en la review

Ambientes → tienen que estar separados p/ construcción limpia del producto
→ pruebas de acept de usuario conviene en preproducción (pero nadie tiene) así que se hace en producción
→ desarrollo y prueba tienen que estar separados p/ que no se metan los desarrolladores en medio de las pruebas y sigan haciendo cagadas

- **Caso de prueba** → Pasos de como probar algo específico

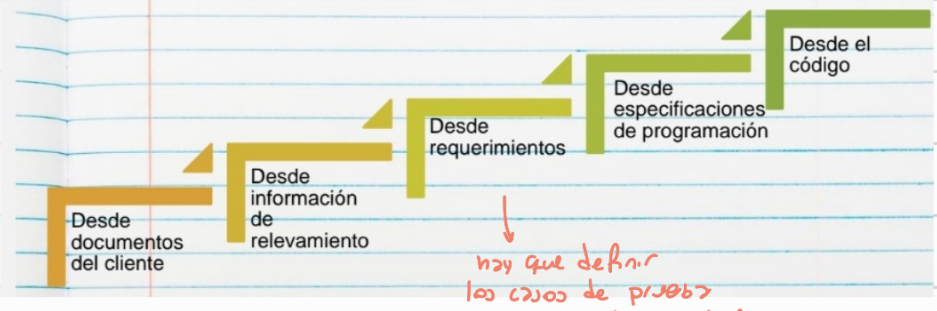
Ver video p/ mytos de casos de prueba

- 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

- Un defecto que no se puede reproducir no es un defecto

→ son ejemplos específicos con datos específicos, concretos

- **Derivación de casos de prueba**



hay que definir los casos de prueba desde acá (ya los definiste con el cliente, ya sabes que tiene que hacer el sist.)

- **Condiciones de prueba** → reacción esperada de un sistema frente a un estímulo particular (formada por las entradas)
→ deben ser probadas por al menos un caso de prueba

- **Caja Negra** y **Caja Blanca** ↳ código

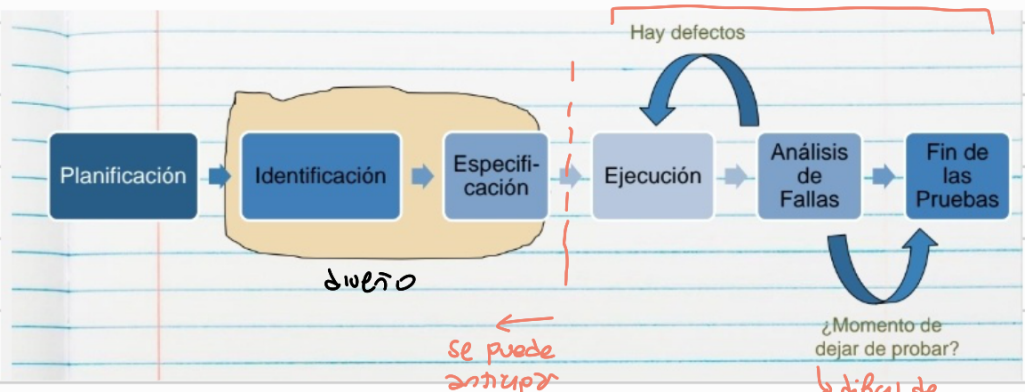
no anote un choto

- **Ciclo de test** → ejecución de un conj. de casos de prueba a una versión
→ ciclo 0 : primer ciclo
→ sin regresión: no se vuelve a probar lo probado antes que andaba
→ con regresión: todos los ciclos como en ciclo 0, pruebo todo en todos los ciclos (se evitan los errores que entran cuando arreglan otros)

- **Proceso de Pruebas**

a partir de la ejecución comienzan los ciclos

etapas están desarrolladas en los ppt



- Según el ciclo de vida se como se lleva a cabo el testing

VERIFICACION
¿Estamos construyendo el sistema correctamente?
VALIDACION
¿Estamos construyendo el sistema correcto?

- **Modelo en V** → se desarrolla de lo general a lo particular y se prueba al revés

¿CUÁNTO TESTING ES SUFICIENTE?

- El testing exhaustivo es imposible.
- Decidir cuánto testing es suficiente depende de:
  - Evaluación del nivel de riesgo
  - Costos asociados al proyecto
- Usamos los riesgos para de terminar:
  - Que testear primero
  - A qué dedicarle más esfuerzo de testing
  - Que no testear (por ahora)

- El **Criterio de Aceptación** es lo que comúnmente se usa para resolver el problema de determinar cuándo una determinada fase de testing ha sido completada.
- Puede ser definido en términos de:
  - Costos
  - % de tests corridos sin fallas
  - Fallas predichas aún permanecen en el software
  - No hay defectos de una determinada severidad en el software

- Principios del testing

me cansé ♡