Contents

CONCEPTOS IMPORANTES

TESTING METODOLOGICO

Sigue un proceso.

Usa los casos de prueba diseñados.

Se planifica, tiene artefactos, y fundamentalmente, tiene casos de prueba.

TESTING ADHOC

Es un testing no metodológico, que no se basa en su ejecución en los casos de prueba.

No tiene una metodología. Es no sistemático.

ERROR VS DEFECTO

ERROR: algo es un error cuando no se trasladó de una etapa a otra.

- Son menos costosos.

DEFECTO: algo es un defecto cuando se trasladó de una etapa anterior, a la etapa actual.

- No se originan en la misma etapa de testing
- Edad del defecto: tiempo que transcurre desde que se introdujo hasta que se corrige.

FALLA:

- Ocurren en el sw como consecuencia de lo que hacemos nosotros.
- Un error o defecto puede provocar una falla en el sistema.

SEVERIDAD: que tan grave es el defecto. No tiene que ver con lo que a mi me cuesta corregirlo. Tiene que ver con el impacto que produce en el uso del sistema. Lo asigna testing.

- Bloqueante/invalidante → no puedo usar el sistema
- Grave
- Leve
- Cosmética → atribuido a cuestiones de interacción con el usuario, ortografía, formateo de los campos,
- Mejora → no es algo que esta mal, es una sugerencia.

PRIORIDAD: la define al cliente, en cuando a lo que necesite corregir primero. (alta, baja, meida).

TESTING

ACTIVIDADES PDU

- Requerimientos
- Analisis
- Diseño
- Implementacion
- Prueba
- Despliegue

El propósito del testing es encontrar defectos, cuya presencia se asume en el código.

De ahí, los desarrolladores tienen el pensamiento de que el testing es destructivo, pero en ágil no pasa igual.

Sobre un proyecto el testing acapara entre un 30-50% del esfuerzo.

PROCESO DE TESTING – estándar

ACTIVIDADES → ARTEFACTO

- Planificación de pruebas → plan de prueba
- Diseño de las pruebas → casos de prueba (en la us definimos los escenarios, no el caso de prueba en si)
- Ejecución: → reporte de defectos (debe hacer una retroalimentación a la implementación)
    - Tiene como entrada los casos de prueba, que vienen del paso anterior, y una versión del producto que proviene de la implementación.
- Cierre

Antes, el proceso de testing empezaba una vez que se tenia la primer versión implementada.

Pero, la realidad es que no es necesario esperar a que se tenga la implementación para empezar con el testing.

Viendo las actividades del testing, podemos decir que la planificación y el diseño de las pruebas las puedo adelantar.

Mientras se construye el producto voy pensando como probarlo y empezando a diseñar las pruebas.

Al adelantar estas dos actividades, permite adelantar el calendario, reducirlo.

Estas dos actividades las puedo hacer una vez que los requerimientos están definidos.

Sin casos de prueba no tiene sentido probar. Los casos de prueba tienen que venir de los requerimientos.

VALIDAR Y VERIFICAR

VALIDAR: es un proceso que tiene que ver con la adecuación. Vemos si hace lo que realmente tiene que hacer.

VERIFICAR: ver si lo que hace lo hace bien.

Cuando hago testing tengo que poder hacer las dos cosas. Si tengo los casos de prueba definidos, que vienen de los requerimientos, entonces puedo realizar las dos actividades, validar y verificar.

El costo de los errores, cuando es un error de validación, es mas costoso que un error de programación en sí.

NIVELES DE TESTING

Se prueba de lo mas chico a lo más grande.

- PRUEBA UNITARIA
  - La hace el desarrollador
  - En la implementación, es el pruebo mientras construyo.
- INTEGRACION O DE INTERFACES
  - Junto los componentes de las pruebas unitarias y las pruebo en conjunto
  - Se hace en el workflow de implementación y se le entrega a testing un build ya formado
- SISTEMA
  - Se hace en testing
  - Se prueba todo el sistema en su conjunto.
- PRUEBA DE ACEPTACION DE USUARIO
  - En el despliegue

Un desarrollador no debería probar su propio código

AMBIENTES DE SW - ideal

DE DESARROLLO

Pruebas unitarias y de integración

DE PRUEBA

Pruebas de sistema

PRE – PRODUCCION

Se deberían realizar las pruebas de aceptación de usuario.

DE PRODUCCION

Lo ideal es no realizar ninguna prueba sobre este.

CICLO DE PRUEBA

Un ciclo de prueba es …

Le mando a desarrollo el reporte de defectos que encontré.

Se realiza toda la ejecución de los casos de prueba.

El primer ciclo de prueba que se ejecuta es el ciclo cero.

Ciclo de prueba por cada versión que entra a testing

SITUACION IDEAL

¿Cuantos ciclos de prueba? → 2

Ciclo cero tiene que haber siempre, y debe ser manual

- Encuentro todos los errores
- Los corrigen
- Vuelvo a probarlo y todo esta ok.

ESTRTEGIA A PROBAR EN CADA CICLO DE PRUEBA

SIN REGRESION

Ejecuto el ciclo cero

Tengo el reporte de defectos

Desarrollo corrige, y envia solucionado

Llega de nuevo a testing.

Y se prueba lo que esta incluido en la lista de defectos.

CON REGRESION

Estoy en el Ciclo 1.

Vuelvo a correr todos los casos de prueba como si fuese el ciclo cero.

Es mejor, porque cada error que se corrige puede introducir nuevos.

QUE SE PRUEBA?

Los requerimientos: funcionales y no funcionales. Hay que probar los dos.

Testing puede probar sin problema los funcionales.

TIPOS DE PRUEBA

Describen el foco del producto que quiero probar.

Alguno enfocados en requerimientos funcionales y otros en no funcionales

PARA FUNCIONALIDAD

- Prueba de requerimiento en operación normal
- Prueba de requerimientos negativa
    - Usar mal el sistema a propósito, para ver si me deja o no.
- Proceso de negocio
- Manual de usuario → el manual debe mostrar exactamente lo que hace el sw.
- Manual de configuración → poder instalar el sw y que quede funcionando adecuadamente.

PARA NO FUNCIONAL

- Prueba de performance
    - Tengo determinado tiempo de respuesta en una funcionalidad.
    - Para requerimientos de duración máxima de una operación.
    - Tener en cuenta la cantidad de usuarios concurrentes
- Prueba de interfaces
- Prueba de escala completa
    - Probar todo lo que me comprometí
- Prueba de estrés
    - Pruebo mas de lo que me comprometi. Ej: me comprometi a 50 usuarios concurrentes, entonces pruebo con 60. Lo hago para ver como queda el sistema luego de esa prueba.
- Prueba de seguridad
- Prueba de accesibilidad
- etc

ACTIVIDAD EL CIERRE

Pruebo para encontrar los defectos y reportarlos

Cada defecto tiene un estado.

- Abierto → cuando lo encuentro al defecto.
- Corregido → por parte del desarrollo
- Cerrado → cuando se verifica que lo corregido esta ok y todo funciona ok

CASO DE PRUEBA

Es una situación descrita, detallada, que apunta a hacer verificación y validación de una determinada funcionalidad del sistema.

Tiene:

- Escenario que se va a probar → paso a paso de lo que tengo que hacer
- Datos
    - Incluye las precondiciones o condiciones de seteo.
- Resultado esperado → esta en la user

Antes de ejecutar, necesito las condiciones de seteo, las precondiciones.

Las pruebas son detalladas.

CUANDO DEJAMOS DE PROBAR?

¿Cuando deje de encontrar errores? → no

Adivinación de defectos

Por tiempo → se asigna cierto tiempo a testing y se prueba lo que se pueda

Por porcentaje de errores

Hay de dejarlo definido en el plan de prueba.

Testing no esta calificado para asegurar la calidad del sw ni para asegurar que no hay mas errores.

Testing no hace aseguramiento de calidad.