TDD – TEST DRIVEN DEVELOPMENT

TDD – TEST DRIVEN DEVELOPMENT

No es una metodologia de prueba.

Es un método para desarrollar SW → Desarrollo conducido por pruebas.

ACTIVIDADES DEL PROCESO DE DESARROLLO

1. Requerimientos
2. Análisis
3. Diseño
4. Implementación → 33-35% de costo total del proyecto
5. Testing → actividad más cara (costo del esfuerzo → hs persona lineales) → 30-50% de costo total del proyecto, depende del tipo de SW
6. Despliegue

Después de la programación, recién se preguntaban como probar el sw

La ejecución de las pruebas, lógicamente, es después del desarrollo, pero la planificación es antes. Lo podemos hacer a través de los casos de prueba.

CASO DE PRUEBA

Es la identificación de situaciones concretas, más los datos de “seteo” necesarios para poder determinar si uno o más criterios de aceptación se cumplen para ese SW.

Un caso de ruba necesita, datos concretos para darle valor a la situación, y a partir de eso, determina la situación de contexto, que deben ser validas. Ej; asumo que prueba la persona X que tiene los permisos X, que necesita los datos X en la BD, etc. Para ello me manejo en base a los requerimientos.

Tomo los requerimientos, diseño los casos de prueba, hago el desarrollo, y después lo pruebo.

En términos de calendario, al hacerlo a partir de los requerimientos, es más eficiente.

TRADICIONAL VS TDD

CON TTD:

- Pensemos el testing desde el principio.
- Con el agilismo se adelanta el pensamiento de testing

TDD

Framework ágil

Extreme programming – XP

Evoluciona hacia ATDD → desarrollo conducido por pruebas de aceptación

4 NIVELES DE PRUEBA

- Unitaria
  - Desarrollador
  - A nivel de implementación
- Integración
  - Probar el conjunto de las pruebas unitarias
  - También llamado Prueba de interfaces
  - Que los componentes probados por separado, si los junto en un bloque mas grande, funcionan bien
- Sistema
  - O versión
  - Ya es una versión
  - Se verifica si cumple con TODOS los requerimeintos
- UAT
  - Prueba de aceptación de usuario
  - Las tiene que hacer el usuario necesariamente
  - En el agilismo el PO
  - Uso la instancia de prueba para aceptar el producto
  - Se hacen en el despliegue

QUE ES TDD

Primero escribimos los casos de prueba

Esos casos de prueba habilitan la construcción del componente

Luego corremos pruebas automáticas para ver si el componente cumple o no.

Tiene un ciclo de tres momentos, basado en la idea de que el primer test que voy a correr tiene que falla, veo las oportunidades de mejora, y refactorizo.

Ciclo:

- Primero escribo un test que falle
- La falla permite identificar las mejoras
- Refactorizo

Beneficio → no se va a escribir más código que el necesario para que el test pase

REFACTORING

Es una forma de desarrollar que apunta a mejorar la calidad interna del componente que construimos.

Significa que, no se cambia la funcionalidad, solo que lo mejoramos para que lo haga mejor

Aplicando el nivel adecuado de acoplamiento (bajo) y de cohesión (alto) → bases de la calidad

QUE HACER PARA TDD

Definir las precondiciones