TEST DRIVEN DEVELOPMENT – TDD

Es un método para desarrollar software conducido por pruebas. Forma parte de la implementación

La prueba generalmente se lleva entre el 30% - 50% del presupuesto

Caso de prueba → necesita datos concretos para determinar si un conjunto de condiciones son válidas y cumple con lo que esperado

En agilismo se busca adelantar el pensamiento de la prueba haciéndolo antes de desarrollar y los detalles se ponen en los casos de prueba.

Desarrollo tradicional: Diseñar → Desarrollar → Probar

TDD: Diseñar → Probar → Desarrollar

Hay 4 niveles de prueba:

- Pruebas unitarias
- Integración
- Sistema o versión
- Pruebas

TDD → Involucra dos prácticas

1. Escribir las pruebas primero → Pruebas unitarias
2. Refactorización

Ciclo Red – Green – Refactor

- Red → Escribir una prueba que falle para identificar que mejorar
- Green → Escribir código mínimo para que pase
- Refactor → Mejorar el código manteniendo las pruebas en verde

3 Leyes del TDD

1. No escribir una línea de código hasta que no hayas escrito antes un test case que falla
2. No es necesario escribir más test de los necesarios para que falle. Normalmente con un test que falle es suficiente.
3. No vas a escribir más código en Producción que el necesario para que el test pase.

Pasos del TDD

1. Escribir un test
2. Hacer que el test compile
3. Hacer que el test falle (red)
4. Hacer un cambio para que el test pase
5. Correr el test para que pase (Green)
6. Quitar duplicaciones (Refactor)

Refactoring → Forma disciplinada de introducir cambios reduciendo la posibilidad de introducir defectos sin modificar la funcionalidad del componente → Mejoran la calidad del software y elimina duplicación