**Comparativo entre os métodos de teste tradicional e o TDD**

Métodos ágeis (onde se inclui o TDD) vem sendo apontados como meios alternativos ao métodos tradicionais de desenvolvimento de software.

Antes de fazer um comparativo, é importante ter a definição de qual é o foco e resultado esperado em cada abordagem de teste.

**Abordagem Tradicional**

O método tradicional, também chamado de pesado ou orientado a documentação, tem sua origem na época dos *mainframes* e terminais “burros”. Nessa época o custo para fazer uma alteração ou correções era muito alto, uma vez que o acesso aos recursos eram muito limitados e não existiam ferramentas de apoio que temos nos dias de hoje.

O método mais utilizado, tido como tradicional, é conhecido como método sequencial. Tem foco muito forte na documentação do projeto em todas as etapas do processo, e pode ser considerado um desenvolvimento em cascata, pois cada etapa é sequencial e dependente da anterior.

De forma geral, as etapas do método clássico são: definição de requisitos, projeto do software, implementação e testes unitários, integração e testes de sistema, operação e manutenção.

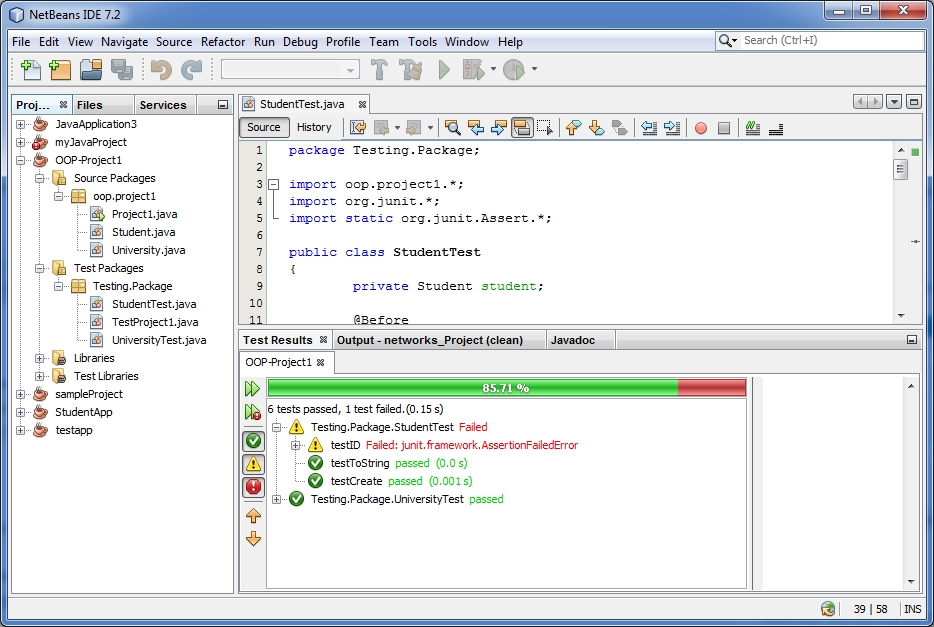
**Método TDD**

O método conhecido como TDD (Test Driven Development) é parte da abordagem dos métodos ágeis, e tem se tornado muito popular no mercado devido a forte recomendação de grande profissionais da área.

Algo que também deve ser lembrado é que os métodos ágeis não rejeitam os processos e ferramentas (e documentação) muito abordados nos métodos tradicionais, mas quer mostrar que alguns pontos que são considerados primários no outro método, são na verdade secundários na realidade dos projetos de software atuais.

No método TDD, as etapas são pensadas para tentar garantir melhor qualidade do produto e prover mais flexibilidade durante o processo, e são elas na ordem: escrita do teste, escrita do código, refatoração do código.

Quando esse método é adotado durante o desenvolvimento, o controle de alterações e possíveis erros durante o desenvolvimento são mais evidenciados, e consequentemente resolvidos antes da entrega do produto.

*Imagem que ilustra como são mostrados os resultados de teste na IDE Netbeans*

**Comparativo**

A escolha do método que será utilizado em um determinado projeto, depende de alguns fatores como a habilidade da equipe em utilizar métodos ágeis ou não, como é o processo de demanda dos requisitos por parte do requisitante, etc.

Mesmo levando em conta a analise de qual método se adequa ao projeto, um dos problemas que o método clássico causa, é a falta de flexibilidade para a divisão do projeto em fases distintas, o que dificulta possíveis alterações durante a execução do projeto, algo comum durante o processo. Outro problema que esse método causa, é a falta de testes constantes durante o processo de desenvolvimento, algo que o TDD tem como foco primário durante o processo.

Apesar de muitos desenvolvedores optarem por utilizar o método TDD, a alteração no modo como muitos desenvolvedores (que hoje são gerentes) aprenderam, pode causar um certo obstáculo para a implementação desse método, pois muitos deles não conseguem entender como no TDD a escrita do teste é mais importante que a escrita do próprio código e por essas etapas serem o oposto do método clássico.

Mesmo com benefícios de qualidade e confiabilidade do código bem evidentes, a transição para o método TDD pode levar tempo para ser assimilada e executada por um desenvolvedor ou pela equipe, e por isso, muitos gestores acabam desacreditando no método, se esquecendo que nessa transição existe a adaptação, passo a passo, ou como o próprio TDD também sugere, a execução de baby-steps, ou seja, que as etapas do TDD sejam feitas testando pequenos pedaços do código, um por vez, e nunca pensar em escrever os testes para o projeto inteiro de uma vez só, até por que, a flexibilidade para alterações e correções é um dos grandes pontos dos métodos ágeis.