Eduardo de França Gonzalez – 22066203-2

Resumo

Camada não tem relação com pastas, uma camada é definida pelo relacionamento dos componentes do projeto, se você não tiver as camadas devidamente definidas, você acaba partindo para um teste de integração, mas é ainda mais importante ter testes em todos os níveis, para fazer o TDD você deve estar validando o resultado por meio do given, when, then / arrange, act, assert, deve-se definir a criação de um conjunto de tabelas, no primeiro exemplo, ele mostra o problema, que seria misturar um conceito de baixo nível com o conceito relacionado ao mecanismo de envio dos dados, ou seja, um vínculo de uma Request e um comando para manipular uma tabela do banco de dados. O console read developer seria o “testar com os olhos”, o que é horrível para a aplicação, devemos buscar que os testes sejam mais estáveis para que possa ser mexido no design da aplicação.

Fazendo as alterações para deixar as camadas bem definidas de acordo com o SOLID, o Rodrigo Branas fez com que o POST apenas criasse, fechasse a conexão e encerrasse, e com que o GET apenas pegasse o parâmetro e o devolvesse. Se nos testes não for criado uma expectativa muito clara, a tendência é que a gente fique flutuando entre a execução, breackpoint, querys do banco de dados entre outros, deve ser buscado os dados para verificar se eles retornaram corretamente, ter o teste antes é relevante independentemente do cenário, onde o objetivo é passar por todos os asserts. Ele explica como o código desenvolvido não está em conformidade com o primeiro princípio do Solid, pelo fato de que sua classe mexe tanto na persistência do banco de dados como as mudanças da API. Ele cria e desacopla as camadas utilizando os princípios do SOLID de maneira que fique ainda mais organizado as regras de negócio também como as funcionalidades do código.

Para finalizar, o Rodrigo Branas comenta sobre o quão melhor flexível está o sistema aplicando todos os princípios ensinados, criando uma operação de transação financeira dividida corretamente em suas camadas, possibilitando que o código fosse apto mais facilmente para receber novas implementações e alterações como ainda mais fácil sua manutenibilidade.