Pesquisador. Muito obrigado por ter aceito participar da entrevista. Ela tem como objetivo compreender os processos de criação e manutenção de código de testes e como funciona a verificação da qualidade desse código de teste. Não há respostas certas ou erradas ou desejáveis ou indesejáveis. Dessa forma, senta-se à vontade para dizer o que realmente pensa.

Entrevistado. OK.

P. Reforçando um item do termo de consentimento, nossa conversa está sendo gravada. O objetivo disso é para que eu possa obter todos os detalhes, mas ao mesmo tempo ser capaz de manter uma conversa atenta com você. Garanto que todos os seus comentários permanecerão confidenciais. O relatório final conterá os comentários dos participantes sem nenhuma referência aos indivíduos.

P. Como você começou a trabalhar com teste de software?

E. Eu trabalho locado numa empresa e eu sou o responsável pelo desenvolvimento e manutenção de um sistema que já existia lá. Então quando eu entrei nessa empresa, existia vários desenvolvedores. Era um desenvolvimento só, mas passava por vários desenvolvedores, cada um deixava seu estilo de código, cada um tinha seu padrão utilizava seu nome de variáveis, cada um utilizava sua maneira de fazer as funções separadas, fazia uma função grande e colocava tudo dentro, então eu vi como grande dificuldade quando você entra na empresa e compreender o que está acontecendo, compreender o que cada função tem que fazer, o que cada método tem que responder, se quebra alguma parte, que parte está quebrando, então eu via necessidade de começar a trabalhar com testes unitários. Era a única maneira de eu conseguir compreender os testes, compreender o sistema, testando ele sem quebrá-lo. Então eu comecei a desenvolver os testes e refatorar o sistema para que eu possa compreender em detalhes.

P. Quais foram suas fontes de aprendizado sobre código de teste?

E. Aprendi muita coisa com Clean code, que é um código limpo, é um livro muito famoso em computação que foi uma junção de ideias que um camarada chamado Uncle Bob, tio Bob, nos anos 80, 90, começou a identificar alguns padrões que poderiam ser seguidos para que os testes unitários, os testes de software em si ficassem mais alcançáveis a todos, fosse um padrão. Então quando eu comecei a pesquisar sobre isso, fui logo nele, numa fonte confiável, num livro. Mas depois como eu trabalho com .NET C#, eu fui buscar nas fontes do próprio .NET. Procurei documentação da Microsoft. Mas quem realmente me deu um direcionamento foi alguns canais do YouTube.

P. Além do .NET, existe alguma outra linguagem que você cria testes?

E. No NodeJS. Javascript no backend.

P. Qual plataforma do seu projeto atual?

E. .NET. É o main language. É o framework que eu mais trabalho

P. Me conte como é seu processo de criação de teste.

E. Tem duas maneiras que eu faço teste. Se for solicitado alguma coisa, eu não escrevo todo o sistema. Eu só faço teste quando for solicitado. Ele é um sistema muito grande e complexo, então eu não posso estar mexendo só porque é do meu gosto. Então se alguém me informar que tem um problema aqui, isso não está funcionando da forma certa, nesse momento é uma oportunidade que eu tenho de refatorar. Aí nesse momento que eu utilizo os testes. Primeiro eu faço os testes para ver cada coisa que ele está fazendo. Se o método estiver aquela coisa espaguete, o código está todas as funções está dentro de uma função só. Tudo está sendo feito numa função só. A primeira coisa eu quebro ele, porque senão não vai ser possível testar, não tem como testar um código desses. Eu quebro ele pequenas funções, pequenos métodos. E depois eu começo a setar esses pequenos métodos, que são os testes unitários, em menores partes. E aí depois testando eu vou refatorando, mudando o que precisa, o que me solicitaram ou então tem a outra forma. Quando me solicitam o desenvolvimento de alguma coisa. Aí eu tenho que criar do zero. Aí eu já por desencargo de qualquer problema eu escrevo a função e escrevo o teste, ou então não tem uma ordem. Porque existe uma prática chamada TDD, Test driven development, que é você escrever o teste antes da função. Mas eu não sigo muito. Eu escrevo a função e em seguida escrevo o teste, sem ordem. Eu posso escrever o teste antes ou depois, não me preocupo muito com isso. Sempre que eu faço um método, e eu acho que esse método é importante, que mexe com alguma coisa importante, como valores monetários, com quantidade de débitos, coisas assim eu utilizo testes unitários para garantir que aquela coisa funcione corretamente até o final. Pois quando você faz o desenvolvimento vai surgindo novas demandas, surgindo novas necessidades e os testes me garantem essa parte unitária está funcionando da forma que eu quero.

P. Existe algum fluxograma, template... que padronize esse processo?

E. Não. Quer dizer, é complicado. Eu trabalho para uma empresa, mas eu trabalho terceirizado para outra, então eu sou o único responsável pelo desenvolvimento, manutenção e qualquer coisa que esteja relacionado ao projeto. Planejamentos do sistema, eu sou o único responsável. Então eu não tenho nada escrito, nenhum fluxograma descrito, mas eu tenho um padrão que eu sigo sempre que é isso que eu falei do se for alteração eu quebro os códigos e tal, mas não tem nenhum fluxograma definido.

P. Qual ferramenta você usa para para criar/ executar teste?

E. No .NET eu uso o XUnit para testes unitários e estou utilizando agora uma ferramenta para integração agora, mas acho que vai ficar um pouco fora que é o Take e Easy. Que é uma ferramenta para quando você faz os testes unitários é tudo no .NET, eles podem funcionar os pedações. Os pedaços podem estar funcionando completamente, mas quando você começa a juntar isso e pegar uma informação de função e colocar em outra e começar a integrar elas, pode ter algum problema de compatibilidade, alguma coisa assim que são os testes de integração. Eu só tive esses dois mesmo. E no JS eu usei o Jest.

P. Dentro da empresa, como é o processo de verificação de qualidade do código?

E. A Qualidade do código é eu mesmo. Eu acredito que um código com qualidade é um código que pode ser entendido por todos então tem, como tiveram vários desenvolvedores que escreveram esse sistema, você consegue um código que não tem qualidade. É muito difícil de se compreender e ele é muito falho. Se você mudar qualquer coisinha, quebra uma coisa totalmente diferente que não estava anexada com isso, mas ele usava em algum lugar. Então um código para mim, eu não uso nenhuma ferramenta para medir, mas eu entendo um código de qualidade é um código que é resistente para alterações, ele é simples, ele tem pequenos pedaços separados e é muito fácil de entender. Então quando eu consigo escrever um código dessa forma, eu entendo que é um código de qualidade.

P. Hoje na parte de qualidade de software, na parte de criação de teste de unidade você trabalha sozinho?

E. Isso, só eu mesmo.

P. Ok. Passando para o processo de manutenção do código de teste, como ele é realizado?

E. Não que tenha ocorrido muito, mas caso ocorra alguma necessidade de um método ou alguma coisa unitária ser alterada, eu primeiro altero o teste, porque o código já estava funcionando, e foi necessitado alguma coisa nova, "preciso que o débito funcione de uma nova forma", então eu utilizo como modelo a ferramenta de teste. Eu penso na forma que eu vou escrever o teste para que o método respeite e o teste funcione. Então eu primeiro escrevo o teste, o teste já vai estar escrito na manutenção, então eu escrevo o teste, fazendo ele funcionar a nova funcionalidade que eu quero, essa nova alteração e aí eu vou depois disso trabalhar o código para que ele responda corretamente ao teste.

P. O que você conhece sobre test smell?

E. Não sei

P. Certo. Eu vou te passar o conceito pois as questões a seguir têm a ver com esse tema. Então se tiver alguma dúvida você me pergunta, vou te mandar exemplos também, mas eu acredito que apesar de não conhecer o termo específico, você já citou algumas coisas que estão relacionadas com o assunto. “Test smells são pontos fracos na estrutura do código de teste, tendo um impacto negativo na sustentabilidade da atividade de teste, pois os tornam complexos, difíceis de entender e modificar, prejudicando a repetibilidade, independência e estabilidade dos testes.” Eu vou te mandar agora dois exemplos e comento contigo porque eles são considerados smells e caso você tenha alguma dúvida a gente conversa.

E. Certo

P. Esse exemplo é chamado "Conditional Test Logic". Ele é considerado um smell porque ele tem estrutura condicional, de repetição, então uma vez que esse teste falhe fica mais difícil de identificar o porquê ou em qual momento ele falhou. O outro exemplo ele é chamado de Empty test. É porque ele não tem nenhum script executável. Então dar manutenção pode ser mais complicado porque vai ser difícil de identificar o porquê que ele deixou de ser executado, porque ele ainda existe e não foi excluído, por exemplo, então a existência de testes com essa estrutura dificulta a manutenção e o processo de repetição, independência dos testes. Ok?

E. Ok.

P. Você tem alguma dúvida?

E. Em relação ao primeiro teste. Ele é um smell pela quantidade de repetição ou que ele não poderia ter condicional no teste?

P. O ideal é que ele não tivesse condicional, que fossem testes separados de acordo com cada uma das condições, pois uma vez que o teste falhe fica mais fácil de identificar "foi essa estrutura, essa condição que falhou o teste".

E. eu ainda não tinha essa necessidade de colocar os “if”, mas eu não tinha essa necessidade de escrever uma lógica dentro do teste assim tão complicada. Quando você separa as unidades, ele já é uma coisa bem simples, então se a pessoa montar os objetos para testar mesmo, fazer essa lógica toda assim, nunca tive necessidade não.

P. Os test smells, esses são dois exemplos que te mandei, mas eles têm a ver com asserts, com parâmetros, com a construção do teste. Então são diversos tipos existentes, mas todos eles estão relacionados a algo que torne a compreensão do teste mais difícil.

P. Como test smells são tratados no processo de criação do código de teste?

E. Eu nunca precisei pensar sobre eles. Eu nunca... precisaram de uma sujeira tão grande assim. Como sou só apenas eu, eu escrevia, entendia e nunca tinha essa visão de test smell. Eu acho que não tenho problema com isso não. Os meu métodos unitários, eu não testo todos os métodos, eu sei que tem métodos que você tem que fazer muitos objetos, muitas coisas, então eu tento separar o máximo possível. Então as coisas que são muito simples, mas quando é algo realmente complicado eu faço questão de testar e não fica muita coisa assim para escrever. Por enquanto eu não tive essa experiência de estar trabalhando com test smell.

P. Nesse caso é um cuidado que você tem na manutenção. Como são testes que você cria, acaba não tendo essa necessidade. É isso?

E. É. É isso.

P. Considerando esse conceito de complexidade dos testes, de compreensão, você identifica uma forma que seria possível prevenir/evitar a inserção de test smell no código de teste desde quando ele é criado?

E. Eu acho que a chave seria você se preocupar com o que está sendo testado no que criar o teste de verdade. Porque se você precisa fazer muita coisa para testar essa função, esse método está muito ruim. Tem muita coisa que você está buscando. O teste unitário tem que fazer apenas uma coisa, testar apenas uma parte. Então se tem que fazer muita lógica, muita coisa, eu acho que está no caminho errado. O foco que eu estou dando não é o teste ser escrito, mas o que está sendo testado. Se o código que está sendo testado está muito grande, está muito feio, está muito complexo, ele tem que tentar diminuir mais, ele tem que tirar complexidade dele. Porque o teste tem que ser uma coisa simples. Tem que bater o olho e entender o teste.

P. Estamos chegando no final da entrevista. Você tem alguma pergunta para mim, quer acrescentar mais alguma informação ou ainda tem alguma sugestão para melhorar nossa entrevista?

E. Você está fazendo sobre a importância do teste na qualidade do código, é isso mesmo?

P. Então minha pesquisa ela tem foco na qualidade do código de teste de unidade, dentro desse assunto de test smell.

E. Então na qualidade do teste. Como eu trabalhei com teste apenas comigo, eu não vi outras pessoas testando, mas eu focava mais na questão do método e na unidade que está sendo testada que na qualidade de teste, pois não pode ser uma coisa muito complexa, muito extensa. Fora algumas funções que não tem jeito mesmo, tem algumas coisas que escapam. Mas o máximo que a pessoa puder fazer para escrever o código que vai ser testado o mais simples possível, poupa muito trabalho do teste.

P. Entendo.

E. É só isso que eu posso acrescentar mesmo. Eu tenho pouca experiência, mas é isso que eu tenho.

P. Muito obrigado pela sua participação.