Pesquisador. Gravação iniciada. Mais uma vez obrigado por ter aceito participar da entrevista. Ela tem como objetivo compreender os processos de criação, manutenção e verificação da qualidade do código de teste de unidade. Não há respostas certas ou erradas, desejáveis ​​ou indesejáveis. Dessa forma, senta-se à vontade para dizer o que realmente pensa. Tudo bem?

Entrevistado. Tudo bem

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 e que no relatório final conterá os comentários dos participantes sem nenhuma referência aos indivíduos. Ok?

E. Tudo bem

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

E. Faz um tempinho. Deve fazer uns 8 anos atrás que foi quando a gente começou a analisar essa parte de testes de unidade, os benefícios que isso poderia trazer. Então até a IDE que eu trabalho até hoje que a do Visual Studio ela tem bastante ferramentas para isso. Então você faz os testes, tem cobertura de código, enfim, a gente começou a ver os benefícios e o quanto a gente conseguiria livrar de problemas futuros.

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

E. Internet e teve um livro também que eu li, o Unit Testing, é um dos primeiros que tem. E teve também um curso que a empresa na época que eu trabalhava pagou, ela trouxe um cara da Microsoft, se não me engano foi o Vitor Cavalcante e André Dias, foram esses dois caras. Então foram dois cursos e aí juntaram o pessoal e eles mostraram tudo que dava para fazer com teste unitário, fizemos algumas dinâmicas de desenvolvimento em Dojo. Foi basicamente isso.

P. Em quais linguagens você cria testes?

E. Hoje C#. Estou aprendendo a fazer também com Typescript.

P. As duas são linguagens que você trabalha atualmente?

E. Isso. C# eu estou trabalhando no back e o Typescript no front end.

P. Como é seu processo de criação de teste de unidade?

E. Basicamente eu leio a história, o negócio. Então eu vejo as regras e aí para cada regra eu vou criando um ou mais testes.

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

E. Tem. Isso é muito particular de empresa. Cada empresa que eu trabalhei ela tinha um modelinho a seguir, mas basicamente não funcionava muito do padrão lá, estou tentando lembrar o nome, mas era um padrão que você primeiro faz o teste, prepara a massa de dados, é o método, ah lembrei, são três As, que é o Arrange, Assert e... Não estou lembrando o nome dos três, mas basicamente você.. ah, Action, lembrei. Aí o Arrange você monta a massa de dados do teste, o Act você executa e o Assert você testa se o resultado esperado está batendo com o teste.

P. Pelo que você comentou, o que vocês fazem hoje difere desse padrão, não é isso?

E. Como assim? Eu não entendi.

P. Você falou que fazia um pouco diferente desses três as o que você faz atualmente.

E. Não não. É o que eu faço na parte de back. O que eu estou aprendendo é para fazer no front também que é um pouco diferente. Porque no back eu consigo por exemplo na parte de preparar, se for um teste para testar por exemplo a parte de banco de dados que aí seria já mais um teste integrado você já prepara o banco para ter esse resultado. Agora quando você vai fazer um teste mais de unidade você tem que usar um mock para você simular os retornos. E aí no front teria que ser essa parte. Então eu estou estudando para ver como vou fazer isso.

P. Quais ferramentas são utilizadas para criar/ executar teste?

E. Do C# da Microsoft o próprio Visual Studio, tem uma IDE própria que o nome eu acho que é Unit test que você consegue montar os testes programando, usando... você vai com cliques, né, também dá para fazer, mas depois você tem que dar uma lapidada no código, e aí depois tem umas ferramentas de cobertura de teste, então você por exemplo, você cria uma classe e você gera um arquivo de teste para essa classe para cobrir os métodos dela. E aí depois ele te mostra onde não está tendo cobertura, os laços que você não está cobrindo, os tipos de erros que você não está validando. Então isso acaba ajudando a você melhorar o teste. E do typescript que eu estou estudando ele tem uma biblioteca, deixa eu lembrar o nome... não lembro, mas é uma biblioteca que... lembrei, Jasmine, que ela é integrada com o typescript e aí você consegue testar todas os arquivos typescript que você criou.

P. Como é o processo de verificação de qualidade do código de teste criado, por exemplo: verificação de escrita, alguém revisa o código de teste depois que ele foi criado, compatibilidade com o requisito, cobertura, assim por diante?

E. Na parte de cobertura de teste, tem a ferramenta que eu falei anteriormente que você também pode utilizar um cara chamado SonarQube, é o mais conhecido. O que é que ele faz, ele olha código. Ele pode tanto estar rodando esses testes e pegando o resultado, mas o foco dele é olhar o código, então se o código está bem escrito, está seguindo os padrões, se ele está validando tudo certinho. Do produto final, geralmente as empresas um cara de QA, que é o cara analista de qualidade, então ele faz o teste simulando o usuário final e isso também dá uma qualidade no produto e também ele pode também programar os testes automatizados. Então ele pega rotinas que são comuns da aplicação e aí ele monta testes robôs para fazer esse teste que ele faz manualmente. E também tem um outro cara que dá para fazer também que também tem a ver um pouco com essa automação, que é os testes de performance, fazer os testes de carga, que aí e bem específico para cada aplicação. Que aí você simula vários usuários ao mesmo tempo fazendo tal rotina e aí você vê o desempenho da aplicação.

P. No caso de teste unitário, basicamente a verificação que é feita é através do Sonar, é isso?

E. É. Você pode usar o SonarQube e/ou... o arquiteto consegue pegar o resultado porque assim, estou falando de ambiente Microsoft, quando eu falo de C#, que aí tem o Continuous Integration Test deles, que basicamente é o que: quando eu faço um commit, eu subo o meu código para o servidor, tem uma ferramenta que ela roda os testes. Então ela pega todos os testes do projeto e roda. E aí se quebrar eu sou notificado. Se não quebrar ele traz os resultados de cobertura e aí o arquiteto consegue analisar. Tem empresas que trabalham com um pull request do git flow. Aí ao invés de eu subir o código direto, eu mando um pedido para esse código ser aceito e aí o arquiteto ou desenvolvedor mais experiente no projeto ele valida, e na hora que ele valida ele consegue também ter esses relatórios de testes. Uma outra maneira é ir manualmente na ferramenta mandar rodar todos os testes, tem até um atalho no teclado que ele roda tudo e eu consigo ver também, eu como desenvolvedor.

P. Hoje, de que forma você tem trabalhado no projeto que você trabalha atualmente? Em relação a verificação da qualidade do teste unitário? Porque você citando algumas formas, você trabalha com as três ou uma em específico?

E. Então hoje, a gente está fazendo, o desenvolvedor vai lá e faz o teste manualmente, eles colocaram já esse SonarQube, então ele está fazendo a cada build, ele faz essa verificação e tem o pull request também, que aí no caso hoje eu não consigo subir o código por exemplo numa master numa developer da vida, eu faço um pull request e alguém vai lá e tem que aprovar esse pull request.

P. Quantas pessoas fazem essa análise do pull request?

E. Depende da equipe. A equipe que eu estou que é uma equipe mais enxuta, tem 2 caras fazendo, mas tem equipes na empresa que eu já ouvi falar que tem uns 4, 5 caras para revisar, mas são equipes de 10, 15 pessoas.

P. Passando para o processo de manutenção do código de teste, como essa atividade é feita? Pensando especificamente na equipe que você trabalha agora.

E. Deixa só ver se eu entendi a pergunta. Eu tenho uma manutenção para fazer e aí eu vou ter que fazer uma manutenção no teste, é isso?

P. Se o ponto de partida para você é a manutenção do código, sim. Se existir algum outro momento que é feita só a manutenção do código de teste, você pode incluir também.

E. Que assim, hoje a manutenção só de teste ela é feita em momentos que está mais tranquilo o projeto, e aí a gente começa a pegar os problemas de qualidade, que um deles é o teste, como também as refatorações de código. Agora quando tem alteração, basicamente a gente faz alteração e aí depois analisa os impactos que o coverage test ele avisa para a gente. Então eu alterei uma linha nessa classe, ele vai me dizer quantos testes estão sendo impactados e aí eu preciso olhar para esses testes para entender o que está acontecendo.

P. O que você conhece sobre test smell?

E. Ah, teste feliz. Bom, se for o que estou pensando, eu conheço com o nome de red green. Seria isso?

P. Acredito que não. Me conte mais para eu ver se é só uma divergência de termos.

E. Que o teste red green é basicamente você vai fazendo os testes e aí eles estão vermelhos, estão quebrados, e aí você vai acertando o código até ele ficar verde.

P. Entendi. Nesse caso não. O test smell ele tem um outro tipo de conceito. No caso do red green pela forma que você está falando tem muito mais a ver se ele está executando o código corretamente ou não, não é? No caso do test smell ele tem mais a ver com a estrutura. Então eles são pontos fracos na estrutura do código de teste, e tem um impacto negativo na sustentabilidade da atividade de teste, pois os tornam complexos, difíceis de entender e de modificar, prejudicando a repetibilidade, independência e estabilidade dos testes.

E. Entendi.

P. Então não necessariamente tem a ver com a cobertura em si, mas vai impactar na execução desse teste e mais para frente na sua manutenção, seja devido a complexidade do código escrito ou da independência que esses testes possuem, por exemplo, se eles estão relacionados a um mesmo recurso e dependendo da ordem que eles sejam executados um teste vai passar outro não, ou então um valor específico de parâmetro, duplicação de código, tudo isso vai impactar na manutenção do teste, mas não vai estar especificamente ligado a cobertura do código em si, se ele vai estar passando ou falhando.

E. Entendi. Legal. Aquela ferramenta que eu comentei, o SonarQube ela faz algo que similar, mas ela faz com o código, ela olha o código fonte nesses quesitos de vulnerabilidade, duplicidade de código, segurança, porque ele olha tanto o código escrito quanto ele compilado. Então ele consegue ter um pouco dessa noção, mas eu não conhecia esse test smell. Legal

P. Você comentou um pouco, considerando que o Sonar ele faz esse tipo de análise, mas de uma forma mais genérica e você pode incluir o Sonar obviamente, como os test smells são tratados no processo de criação do código de teste?

E. Não sei te responder isso.

P. Ok. Eu vou te mandar 2 exemplos de smell, a gente conversa um pouco sobre, se abrir sua mente para o que vocês eventualmente façam no dia a dia, você responde, senão a gente passa para a próxima pergunta, tudo bem?

E. Está bom.

P. O primeiro é o chamado Empty test smell. Ele é considerado um smell por não ter nenhum script executável nesse código. Então ele não vai cobrir, ele não vai executar aquilo que ele originalmente foi proposto. O outro exemplo ele é o chamado Conditional Test Logic smell. E ele é considerado um smell por ter estrutura de condição, de repetição e uma vez que esse teste falhe é mais difícil de entender em que momento dessa execução, de acordo com essa estrutura esse teste falhou. Então esses são só alguns exemplos de smells, são 2 exemplos de smells que podem haver no código, mas por exemplo, se você tem teste duplicado, ou então código dentro de um mesmo teste você tem código duplicado dentro dele, isso também é considerado um smell, você ter, como eu mencionei, dependendo do teste, você vai ter testes que a princípio seriam independentes, mas como eles lidam com um mesmo recurso, um mesmo componente, dependendo da ordem que eles estão sendo executados, um vai passar outro vai falhar exclusivamente por conta da ordem e não necessariamente por conta da execução em si. Então alguns tipos de smells que podem ocorrer no código de teste.

E. Tá. Entendi. Eu lembrei de um problema no passado que a gente teve com último exemplo que você falou de ordem de teste e aí para contornar esse problema a gente teve que criar, passar um atributo no método, test order, que aí ele força a respeitar a ordem que eu defini no código, mas isso aconteceu porque a gente fazia um teste integrado com o banco, então eu não poderia por exemplo fazer um teste de validação de remoção de registro sem ter inserido antes, então a gente tinha uma ordem de testes a seguir. Agora esses dois exemplos que você deu, o primeiro eu acho que a ferramenta que eu trabalho ela pegaria, o teste em branco. Na verdade ele não estaria em branco, ele está gerando uma exceção de... ele está testando true ou trouble, tá, então ele pegaria. Agora esse segundo, ele é mais difícil porque ele tem um laço de repetição com um assert, é mais difícil pegar. Eu não sei se o SonarQube pegaria isso.

P. Era justamente o que eu ia perguntar, se a ferramenta que você estava comentando era o Sonar.

E. Talvez pegue. É que isso é uma prática que quando eu fiz o curso eles não recomendavam fazer isso, de fazer um assert dentro de um laço de repetição. Sempre fazer o assert lá no final e se precisa verificar registro você usar variáveis de controle e aí depois no final você testa quantas deram certo, quantas não deu, qual o esperado.

P. Entendi. Então no seu caso, o cuidado que você tem é mais de acordo com o seu conhecimento sobre programação.

E. Sim. Exatamente.

P. Para você, como seria possível prevenir/evitar a inserção de test smell no código de teste no momento que ele é criado? Quais técnicas de programação, processos organizacionais podem ser adotados?

E. Basicamente, eu tinha falado lá no começo daquele padrão de autonomia AAA, se seguir esse padrão, todos os métodos não são cair nesse problema, porque você sempre vai ter um bloquinho para preparar, um bloquinho para executar o comando que você precisa chamar o componente, enfim, do que você precisa e no finalzinho você sempre vai ter um único assert, então ele acaba prevenindo teste em branco e esse cenário ou outros cenários

P. Estou chegando ao final da entrevista. Você tem alguma pergunta para mim, quer acrescentar alguma informação ou tem alguma sugestão para melhorar a entrevista?

E. Talvez perguntar antes sobre esse test smell, se a pessoa conhece ou não para ir mais rápido. porque eu fui pego de surpresa. Eu não sei se o pessoal que você está entrevistando já conhece isso.

P. Só falando um pouco sobre esse assunto. Minha pesquisa o foco dela é nesse tópico, em test smell. Então nessa fase da pesquisa é interessante fazer essa pergunta durante a entrevista mesmo e não previamente para que a gente tenha justamente um cenário que o profissional lida no dia a dia. porque por exemplo, já aconteceu da gente estar fazendo a entrevista e no momento que a gente perguntava o que era test smell a pessoa abria uma aba a mais no navegador, fazia a pesquisa e começava a falar a respeito. Então fugia um pouco do cenário que a gente queria abordar durante o estudo. Então por isso que a gente acaba fazendo durante a entrevista mesmo. Mas é como você acabou comentando, assim. Apesar de você não conhecer o termo propriamente dito, você eventualmente você tem algum contato com alguma ferramenta que faz algum tipo de tratamento ou então a sua forma de codificar os testes, você tem um cuidado em relação a escrita, então por mais que você não conheça o termo, essas informações que você me passou já valem como informações relevantes para a pesquisa da gente. Não precisa exatamente conhecer o assunto previamente para gente conversar. A ideia é justamente ter esse cenário mesmo e a gente conversar a partir daí.

E. Bacana. Uma coisa que eu senti falta foi da parte de requisito no sentido assim como, não sei, bolar uma pergunta... Na verdade você perguntou sobre isso, como que eu me baseio nos testes, porque tem uma, eu lembrei agora que é uma prática que acho que essa é a segunda empresa que eu estou que está adotando essa prática que é do BDD e aí o requisito, ele meio que vem pronto na parte do finalzinho... ele tem cenários que eles exemplificam para entender melhor o negócio porque geralmente vem o requisito para o dev, até para o QA também testar, só explicando o que tem que fazer, mas não tem os cenários "ooh, se eu mandar isso aqui, eu estou esperando isso aqui" E são cenários que ajudam para o teste, então você acaba montando o teste baseado nesse cenário, então você tem uma assertividade maior e você tem maior probabilidade de conseguir entregar o que o requisito está esperando e para o teste também, para o pessoal de QA fica mais fácil porque ele vai lá e já sabe, já tem uma base para testar. Então ele tem um roteiro de teste que ele faz, que ele testa várias coisas, mas para cobrir a parte de negócio está mais fácil.

P. Entendi. Então hoje na rotina de vocês existe essa questão de cenários do BDD para criação dos testes.

E. Isso. Porque o BDD ajuda nessa parte e também para corrigir um pouco o TDD que é algo que até comentei que não faço muito e fiz bem pouco no passado porque você perde muito tempo fazendo o TDD, as empresas não gostam muito, mas é aquilo. Eu cheguei a fazer uma entrega que eu comecei com o TDD, passou pela equipe de QA da empresa. Passou pela equipe de QA do cliente e ninguém achou um problema. Foi uma prática que eu fiz com o TDD. Só que eu demorei um pouco mais de tempo. Ao invés de eu fazer em três, quatro dias, eu acabei fazendo em 5. só que eu não tive nenhum retrabalho, nenhuma manutenção.

P. Bacana esse relato. A gente acabou não entrando muito nessas questões de requisitos porque o foco da gente como é em test smell, é dentro do teste mesmo, dentro do código de teste, mas bacana esse tipo de relato porque a gente acaba conhecendo mais dessa rotina do processo da criação do código, como eu perguntei, como era criado justamente para entender como funciona esse processo dentro de cada equipe, hoje em cada empresa e faz com que a gente também consiga, a partir da análise de dados, é estabelecer um cenário mais real de acordo com o dia a dia de vocês.

E. Legal.. Eu gostei bastante, está de parabéns pelo projeto, pelas perguntas. Achei bem legais. E boa sorte com sua pesquisa.

P. Obrigado. Tenho um pedido para te fazer, você disse que na parte de documentação que tem os 3 As que vocês usam. Isso é documentado, está em algum local que vocês podem acessar, ou então um novo colaborador tenha acesso ou é passado pela cultura e vocês vão utilizando no dia a dia?

E. É um padrão, deixa eu procurar aqui no Google, é um padrão que existe, que é anatomia AAA, estou jogando aqui no Google "Unit test" AAA. Tem várias que falam sobre isso. Que é tipo um padrão, que não a gente tem para código que tem os design pattern que não vários padrões de arquiteturas para resolver problemas comons, tem esse padrão aí, que são AAA - Arrange, Act e Assert. Então assim, quando o pessoal "ah, eu nunca fiz teste unitário" é a primeira coisa que é passada. "ó, você precisa fazer isso".

P. Isso acaba como você está fazendo agora, vocês acabam instruindo com a documentação externa de internet, ou com um documento que vocês tem interno na empresa e que vocês compartilham com o profissional?

E. Então, dessas ultimas empresas que eu tive, não tinha um documento assim de padrão interno, então a gente só falava para seguir esse cara e todo mundo era instruído a seguir isso. E até quem validava o código olhava se estava com esse padrão. Se não estivesse ele já rejeitava.

P. Muito obrigado mais uma vez por ter aceito participar da entrevista e pelas informações. Eu vou parar aqui a gravação, ela fica disponível tanto para mim quanto para você no skype durante 30 dias.

P. Gravação iniciada novamente. Só que ficou faltando uma pergunta. Na verdade eu tinha decidido pular ela pelo que você tinha me respondido antes, mas eu preferi retomar só para poder deixar registrado. Você tinha comentado que de alguma forma vocês acabam não fazendo o tratamento de test smell no momento da criação do teste, é mais um cuidado pelo Sonar ou durante a codificação do próprio programador, não é isso?

E. Uhum

P. No caso de vocês estarem fazendo uma manutenção de um teste, e se deparem com algum tipo de estrutura dessa que seria um smell, vocês fazem algum tipo de tratamento ou o foco do profissional é apenas a correção do teste de acordo com a cobertura?

E. Depende do cenário. Se for algo que seja simples de resolver, sim, é resolvido na hora. Se for um negócio complexo, muito complexo, não lembro de ter acontecido com teste, já aconteceu com o código, a gente faz a solução paliativa e joga numa lista de débito técnico.

P. Entendi. Ok então. Era isso. Desculpa ter te ligado novamente para perguntar, mas era só para ter o cenário completo e eu tinha pensado em pular a pergunta, mas algo me chamou atenção e eu fiz "não vou retomar para poder ter essa informação sua" e foi bom ter feito isso porque você me passou uma outra informação que a gente não tinha conversado antes sobre.

E. Tudo bem, tranquilo

P. Mais uma vez muito obrigado. Eu vou parar aqui novamente a gravação.