Pesquisador. Muito obrigado por participar da entrevista. Ela tem como objetivo compreender os processos de criação e de manutenção de código de testes e a qualidade do 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. Perfeito

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.

E. Claro

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

E. Faz uns 7 anos. Foi o que eu respondi no formulário. Trabalhar mesmo no dia-a-dia, eu diria que uns 5 anos. Eu comecei a ter contato com os testes em 7 anos, mas não era uma realidade no dia-a-dia. Era uma vontade de aprender de utilizar isso do que uma metodologia de trabalho. Foi numa startup de medicina que trabalhei a 5 anos atrás. No começo a gente não utilizava, como eu te disse, não sei dizer fora, mas pelo menos no brasil todas as empresas que eu já trabalhei é muito difícil você defender os testes. O porquê de você testar o software. Nessa startup a gente começou a ter problema de vários bugs em produção. A gente tinha já colocado alguns testes unitários, mas não era o foco do desenvolvimento, a ideia era entregar feature e aí chegou ao ponto que a aplicação já estava muito grande, tinha vários usuários em produção e às vezes a gente alterava uma feature Z e a feature A começava a quebrar. E quando ia debugar tinha uma correlação que era impossível dos desenvolvedores saberem de cabeça ou testar tudo. E aí a gente começou a implementar alguns testes de integração, além de unitários também e foi assim um alívio porque a gente conseguiu diminuir absurdamente o número de bugs que o usuário que pegava. Esse foi de fato minha experiência profissional porque daí para frente todos os desenvolvedores dessa startup começaram a ver o valor do teste, não por ser bonitinho ou para dizer que tem um coverage alto, mas sim para a gente poder fazer um deploy em produção sem medo, poder deployar a feature sem se preocupar que aquilo fosse trazer grandes problemas para o usuário.

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

E. Basicamente a documentação de frameworks. Eu trabalhei bastante com o Jasmine e o Jest. Hoje em dia eu trabalho com o Jest primeiro. Eu comecei trabalhando com Jest depois fui para o Jasmine e agora voltei para o Jest. Eu tive ali um pouco, muito supérfluo na faculdade o conceito de teste e aí ei estudei a documentação de como se usar e li bastante coisa também sobre o conceito do que é teste de unidade para melhorar ali a cobertura. Para garantir que o meu teste estava testando alguma funcionalidade não só dando números verdinhos de cobertura de testes. A parte mais conceitual foi pesquisa de internet que eu ia fazendo, no medium, em algumas comunidades online ou até tirando dúvidas com alguns amigos que já trabalhavam com uma metodologia, como o TDD ou um BDD da vida para entender o que isso agregava ou se minha linha de raciocínio estava correta mesmo.

P. Em quais linguagens você cria testes?

E. Eu trabalho basicamente hoje em dia com front end. Então desde a época da empresa K, eu trabalhei com PHP também, então tinha a parte de PHP tinha o Node, que é Javascript também como o front, mas desde então nos últimos 3 anos e meio eu tenho ficado muito mais com tecnologias do front que é basicamente Javascript e o Typescript, mas é uma pseudolinguagem.

P. Qual plataforma do seu projeto atual?

E. Javascript. Tanto no front quanto no backend.

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

E. Eu acho muito interessante a ideia do TDD, mas poucas vezes eu tive a possibilidade de aplicar ele. Pela correria que é a empresa em si, o departamento. Eu entrei no final de fevereiro num novo projeto. Eu parei de trabalhar num projeto que estava trabalhando e ali eu pude fazer um TDD mesmo, e nesse projeto ele já existe e não tinha nenhum teste, então estou trabalhando tentando cobrir algumas regras de negócios que a gente tem muito problema atualmente para, quando a gente tiver um tempinho maior e eu conseguir defender a cultura de testes dentro do projeto, começar aplicar um TDD mesmo, que é o que eu acredito que é a melhor opção de se desenvolver.

P. Existe algum fluxograma, template ou equipe que você trabalha que padronize esse processo de criação?

E. Não. Infelizmente não. A gente tem apanhado bastante inclusive, pela falta de processos que tem dentro do time, mas é uma coisa bem particular da empresa L, a empresa que estou agora. ela vem sofrendo algumas mudanças de equipe, de produtos, de conceitos, isso criou uma bagunça generalizada onde os últimos meses, até mesmo por causa da pandemia tem sido bem complicado o time se entender.

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

E. Basicamente para criação de testes a gente usa o Jest. A aplicação que eu trabalho hoje em dia ela é feita em view, então conversa bem com Jest. E a gente tem alguns processos de CI (continuos integration) onde o Jenkins é responsável por a cada deploy rodar nossa stack de testes para garantir a integridade.

P. Como é o processo de verificação de qualidade do código de teste criado, por exemplo: verificação de escrita, se existe algum tipo de code review, e alguém revisar o teste depois de criado, como é feita essa verificação de qualidade?

E. A gente vem trabalhando na ideia de merge requests, a gente utilizou o conceito de feature branch então a gente até tem ali algumas regras de merge request, mas elas não tem sido feitas corretamente. Hoje até a gente discutiu algumas coisas dentro do time para tentar diminuir a quantidade de commits em cada merge request para facilitar um code review de outros membros do time. É uma vontade do time como um todo, mas devido as demandas ainda não conseguimos implantar corretamente.

P. Ok, mas como é feita a verificação da qualidade hoje quando uma pessoa cria testes, como é feita a verificação da qualidade da escrita.

E. deveria ser um merge request, mas é aquilo. Algumas vezes a gente consegue sim parar e analisar a solicitação de um merge, a maioria das vezes o merge só vai ser executado porque por algum motivo tem que subir aquilo hoje, tem que ser para hoje. Eu acho que a grande dificuldade ainda tem sido provar para a empresa ou para a maioria das empresas que trabalhei, o quão importante alguns processos de desenvolvimento, por mais que pareça demora ou enrolação, quão importante para a qualidade de software isso seria. E quanto antes a gente detectar, não lembro o nome da escala do custo de detecção de um bug conforme o projeto vai evoluindo. Ter trazido isso para dentro da empresa. O time recentemente, nos últimos dois meses a gente tem conseguido pensar um pouquinho mais na qualidade, no melhor jeito de se fazer, mas a parte da revisão, da implementação de testes e tudo mais ainda não é uma realidade no dia a dia.

P. Passando para o processo de manutenção do código de teste, como ele é realizado dentro da empresa, da equipe que você trabalha?

E. Como eu te disse, a gente estava a 3 meses atrás a gente não tinha nenhuma cobertura de testes, hoje em dia a gente já conseguiu chegar a 30%. Isso baseado na correção de bugs, então como a gente não tinha testes, o que a gente faz hoje em dia é: um usuário reporta um bug, a gente analisa, corrige esse bug e escreve um teste em cima desse ponto de correção. Hoje em dia a gente está trabalhando ainda no retroativo, quando existe a necessidade de se colocar uma regra, é porque isso foi poucas vezes que aconteceu, mas foi possível de se fazer um TDD, foi possível colocar a regra no teste primeiro para depois implementar a regra no código.

P. O que você conhece sobre test smell?

E. Não conheço.

P. eu vou te passar o conceito, te passar também 2 exemplos sobre o assunto, porque as perguntas a seguir têm a ver com esse tópico.

E. Perfeito.

P. “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.”

E. Bacana. É até o mesmo conceito de code smell, né? Eu conheço code smell. Test smell nunca tinha parado para pensar sobre.

P. Isso. Na verdade, test smell ele descende dos code smells. É o mesmo princípio só que focado no código de teste. Existem smells específicos para teste e diverge um pouco dos code smells

E. Bacana. Eu achei um link aqui sobre. Tem bastante coisa sobre.

P. Vou te mandar dois exemplos agora no skype para você entender como seria, mas como você conhece code smells, fica fácil de você fazer essa analogia. O primeiro é o chamado Empty test smell. Ele é considerado um smell porque não tem nenhum script executável. Se ele foi comentado, ele permanece comentado ao invés de ser excluído, não tem a razão de porque foi comentado, na verdade ou ele não deveria estar comentado, ou não deveria existir. Isso acaba impactando na atividade de teste. O outro exemplo que eu te mandei ele é o chamado Conditional Test Logic smell. Ele é considerado um smell porque ele tem uma estrutura de condição e de repetição dentro dele. Então uma vez que esse teste fale é mais difícil identificar de o porquê esse teste falou. Então o interessante é para que nessa situação que esse teste seja dividido em 2 ou mais testes dependendo do que foi escrito, para que cada teste possa cobrir um cenário. Que ele seja totalmente independente. Cada uma das condicionais e das repetições.

E. Bacana

P. Existem outros tipos de smells relacionados a construtor, a valor de parâmetros, a asserts então existe uma infinidade, mas todos eles estão relacionados a estrutura do código de teste e que impactam na sustentabilidade, na repetibilidade, independência e estabilidade dos testes.

E. Um dos grandes problemas que se tem hoje em dia de ter uma suíte de testes é porque em algum momento que é preciso fazer uma alteração no software essa alteração tem que ser rápida e aí o teste começa a quebrar, geralmente ele já tem integrado alguma pipeline de continuos integration e a solução que a equipe acha é desligar os testes para poder entregar. O test smell é bacana esse conceito, vou dar uma olhada do que se tem ali para Jest que vai ajudar bastante a vida aqui, obrigado.

P. De nada. Os tests smells são tratados no processo de criação do código de teste na sua equipe ou algo pessoal que você realize?

E. Não. De fato não. Eu procuro ter uma organização de reaproveitamento de alguns mocks por exemplo, de algumas funcionalidades que auxiliem no teste, mas fato que não tem uma padronização assim ou melhor uma boa prática para os testes atualmente.

P. A outra pergunta tinha a ver com manutenção, mas como você comentou que hoje vocês trabalham mais com a criação do que com a manutenção, então acaba não tendo a necessidade de verificar a questão da escrita dentro da manutenção, pensando nos smells ou em algum momento você já teve esse tipo de acesso?

E. Na minha empresa anterior que era um banco, ele também não tinha testes unitários maia a gente conseguiu implementar inclusive com uma grande cobertura de testes depois de um tempo, então precisou-se trabalhar na manutenção desses testes. Eu lembro que o time inteiro não tinha experiência com testes unitários e eu não tinha experiência com o Jasmine, que eram testes unitários para o angular. Então a gente começou fazendo a cobertura dos testes, só que não foi bem escrito, ou seja, nossa suíte demorava muito tempo para rodar. Então a gente teve que refatorar os testes otimizar essa suíte de testes.

P. Entendi, Mas na sua experiência atual não é realizado.

E. Hoje em dia não. A gente ainda está tentando implementar testes, ainda não teve nenhum caso grande que eu possa dizer que mudou uma feature no sistema e a gente teve que refatorar os testes para atender essa feature. Isso ainda não existiu.

P. Para você, como seria possível prevenir/evitar a inserção de test smell no código de teste no momento da criação do código de teste?

E. Uma ferramenta que a gente usa que eu esqueci de mencionar, é o Sonar. Ele é um quality gate para código. Ele inclusive te dá vários code smells ali. Eu não sei se teria uma ferramenta, mas uma ferramenta para validar algumas coisas de boas práticas para os testes seria o melhor caminho. Fora isso, uma coisa que agrega é conhecimento, porém não traz uma garantia de qualidade, seria um merge request ou um pull request bem avaliado.

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. Não. Eu adicionaria só uma ferramenta, porque estava pensando só em testes, mas que eu vejo que agrega valor à qualidade, que são os Linters de códigos, então garantir uma padronização do desenvolvimento, da escrita do código e ter uma ferramenta para validar essas regras agrega bastante na qualidade do software como um todo e no dia a dia do desenvolvimento.