Pesquisador. Mais uma vez muito 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 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. Tudo bem

E. Perfeito.

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

E. Foi logo no estágio, estagiei numa empresa no ano de 2006 e lá já era executado teste unitário. Então já no estágio, era um projeto muito grande que eu trabalhava, de uma instituição financeira e lá o teste unitário tinha um percentual a ser cumprido e me colocaram a fazer os testes unitários lá na época. Na época eu não entendia muito do porquê daquilo, fazia mais de forma digamos assim "cega". Não entendia o todo, eu era muito novo, tinha 18, 19 anos, fazia ali porque estava na norma, mas não sabia o que estava fazendo e depois passado os anos eu acabei me aprofundando e me especializando.

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

E. Tudalsite, é uma instituição de ensino americana, muito vinculada a Microsoft, e lá tem vários cursos de teste unitário, falando sobre o AAA: Arange, act e o assert. Eu fiz alguns cursos lá de XUnit, NUnit e também li alguns livros na safari books sobre teste de software. Não vou te recordar o nome pois tem anos que li esses livros.

P. Em quais linguagens você cria testes?

E. Meu escopo, sou desenvolvedor .NET, C#. Então eu crio testes em .NET e C# porque eu trabalho só com elas.

P. Qual Linguagem do seu projeto atual?

E. Respondido anteriormente

P. Como é seu processo de criação de teste?

E. Tem o TLD, que é o Test Last Development e o TDD que é o Test Driven Development. Eu já experimentei o TDD hoje eu pratico o TLD. Então eu me sinto mais a vontade deixar a criatividade fluir no código e fazer o teste unitário no final, mas o uncle Bob, outros caras renomados recomendam o contrário, mas eu me adaptei fazendo depois, então eu faço o Test Last Development

P. Você falou que não usa o TDD, usa o TLD. Mas isso e algo seu? Se algum colega quiser usar o TDD ele pode?

E. Que eu saiba do time, o pessoal faz mais TLD. É a prática que mais prevalece apesar dos catedráticos, os grandes âncoras falar que o ideal é ser TDD.

P. Existe na empresa ou equipe algum fluxograma, template documentado que padronize esse processo?

E. Não. Nesse nível não, a gente tem a nível de pull request. O termo cunhado de verificação de código. Se não tiver teste unitário, se tiver alguma parte do código sem cobertura de teste unitário, aparece aquele verdinho o Visual Studio Enterprise dá essa característica, ele vai pintando de verde o que está coberto, o código não vai para produção. Então não existe nenhum documento formal, papel que diga, mas faz parte da cultura do time ter o teste unitário. Sem teste unitário não vai para produção.

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

E. A gente usa o Visual Studio, dentro do Visual Studio a gente usa um framework que é o XUnit. A gente também usa o Fluent Assertion, que você usa muita palavra should para fazer o assert e a gente usa na descrição dos métodos um padrão meio de BDD. A gente coloca quando fazer isso, deve ser aquilo, na descrição. Isso ajuda bastante. A gente usa mock, a gente usa de mock o Alto Substitute. Basicamente isso. Deixa eu pesquisar aqui. É isso: Auto fixture, Nsubstitute, fluent assertion e o Xunit.

P. Como é o processo de verificação de qualidade do código de teste criado, por exemplo: verificação de escrita, há um processo de code review, compatibilidade com o requisito, como é que esses e outros fatores voltados para a qualidade do código de teste são tratados?

E. O code review é a principal âncora. O code review que vai patentear se o teste está bem feito ou não. Esse é o viés, e em alguns cenários a gente tem o teste de integração. Que não é o foco da pesquisa que você levantou, mas a gente em alguns cenários usa o teste integração ele é vinculado ao deploy também. Então tem esse double check com o teste de integração.

P. Quando você fala no code review, seria um desenvolvedor analisa o código produzido por outro, não é isso?

E. isso. Um desenvolvedor pega no Visual Studio, no VLTS que ele mostra o código, baixa o código para sua máquina e vê as que foram alteradas e verifica se está verdinho no Visual Studio se aquele if, aquele for each, aquele while está sendo perpassado por um teste unitário. E a gente também garante, a gente faz muito teste de caixa branca com teste unitário, indicando determinada classe foi chamada com tal parâmetro, a gente até nesse ponto. Se alguém naquele ponto passar aquele parâmetro, o teste unitário não passa.

P. Nesse processo, quantos desenvolvedores fazem a verificação de um código submetido?

E. 2, sempre 2.

P. Passando para o processo de manutenção do código de teste, como esse processo ocorre dentro da empresa?

E. A pessoa fez o desenvolvimento do código dela e sempre que ela está desenvolvendo ela aplica o teste unitário. Na hora que ela aplica o teste unitário, ele vai apontar alguma inconsistência e ela arruma. Caso ela faça checkin, um push no git, com um teste unitário zuado. No processo de build do DevOps da Azure, é detectado que o teste unitário não passou e o código não, ou o pacote de build não é liberado.

P. Nesse caso é muito mais voltado para cobertura que uma análise desse código, uma refatoração, que possa ser realizada nesse processo?

E. A refatoração muitas vezes acaba sendo detectada, o próprio desenvolvedor chega a refatorar, quando ele julga que o código não está muito adequado, está com a complexidade ciclomática muito alta, está bem trabalhoso o código, leitura, o próprio desenvolvedor faz esse primeiro refactory, porém no pull request o time aponta bastante coisa. E aí onde o teste unitário ajuda muito, porque as vezes a gente aponta para um colega de trabalho que ele precisa refatorar determinado cenário, ele refatora e vê se o código de teste ainda está funcionando. isso também é bem interessante, isso acaba acontecendo bastante. Aí o teste unitário funciona como um cão de guarda que não vai deixar o requisito ser quebrado.

P. O que você conhece sobre test smell?

E. Não conheço essa sigla. Talvez eu conheça de uma outra forma, mas inicialmente não me vem a memória

P. Eu vou te passar o conceito sobre o assunto, vou te passar uns exemplos, a gente alinha informação até porque as perguntas a seguir vão estar relacionadas a esse tópico, tá?

E. Beleza.

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. Uhum

P. Eu te mandar 2 exemplos de test smells, um é o chamado Empty Test. Ele é considerado um smell porque não tem nenhum script executável. Então esse teste não vai fazer a verificação que ele deveria fazer no código propriamente dito e não vai ter nenhuma informação de porque ele está comentado ao invés de estar sendo executado ou porque ele não foi excluído e está apenas comentado. Então tem essa questão de gerenciamento de teste que acaba sendo impactada. O outro exemplo é o Conditional Test Logic smell. Ele é considerado um smell pois tem uma estrutura de repetição, condicional dentro dele. Então uma vez que um teste dessa forma falhe vai ser mais difícil identificar em qual momento considerando essa estrutura esse teste falhou. Ou dependendo da experiência do programador ele pode ter uma dificuldade para entender a lógica desse teste, por exemplo. São questões que acabam impactando na independência, estabilidade e repetibilidade dos testes

E. Referente a comentário, quando chega no pull request, não pode ter comentário. Então chegou qualquer código no pull request, mesmo que não seja do teste, é pontuado para deletar. A gente não trabalha nesse nível de comentário no código, de código comentado. Salvo tem algum end point que se queira criar ou design que está fazendo, pode até acabar ficando, mas geralmente no pull request a gente manda deletar. Referente a estrutura de repetição, no framework Xunit ele tem um atributo que coloca em cima do teste chamado Theory e lá você consegue colocar online data, então cada atributo em cima do método vai fazer com que aquele método seja chamado várias vezes, mas sem looping. Então a gente não tem for, não tem for each dentro de teste unitário. Enquanto a if também, se a gente for ter algum if a gente cria um outro teste unitário específico para aquele cenário. O teste unitário é específico por cenário.

P. Existem outros tipos de smells relacionados a asserts, a valor de parâmetro, a dependência entre testes, por exemplo 2 testes que estão relacionados a um mesmo recurso, dependendo da ordem que eles são executados um passa e outro falha. Teste duplicado, asserts duplicados são coisas que acabam impactando na sustentabilidade da atividade de teste. Em relação as próximas perguntas, não se fixe a esses dois exemplos que eu compartilhei contigo, foca mais no conceito, na questão de coisas relacionadas a estrutura que possam impactar na atividade de teste em si, ok?

E. Ok

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

E. O pull request pela experiência do time é o grande divisor de águas. Quando um teste unitário não está legal, a gente pontua para o código ser alterado. Hoje a gente deixa uma sugestão no pull request. Na ferramenta tem uma lampadazinha que é a pessoa só aceitar, que ela troca o código, ou a gente escreve lá e a pessoa faz a alteração. É no pull request que a gente detecta algum smell no teste. Alguma falha alguma coisa que deixou de ser testada.

P. Quando a gente estava comentando sobre manutenção do código de teste, você comentou que no momento de uma alteração, o desenvolvedor pode fazer uma refatoração no código de teste. Esse processo inclui a verificação de smell, a retirada de smell, Como test smells são tratados no processo de manutenção do código de teste?

E. A gente sempre busca aprimorar. Eu posso falar mais sobre o que eu desenvolvo. eu desenvolvo, consigo alcançar o requisito funcional e aí eu faço o teste unitário. Depois eu dou uma olhada no que eu posso refatorar. Então muitas vezes eu faço o teste e depois eu refatoro, depende do momento. Nesse momento a gente arruma o teste se ele tiver com alguma problemática. Como a gente já tem alguns padrões de teste é muito difícil a gente acabar fazendo um teste que tenha um smell. Porque a gente já tem alguns frameworks, já tem uma serie de problemáticas resolvidas, então a gente tem uma arquitetura que apoia a gente. Por exemplo, na figura que você passou, o caso do for, a gente já tem resolvido na arquitetura. a gente acaba muitas coisas já nascendo bem madura.

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

E. Existe uma coisa no Visual Studio, no .NET que se chama code roles. Você consegue configurar uma regra de código lá. Eu nunca trabalhei com code role, mas o Visual Studio permite você criar uma série de regras para o seu código, então pode se detectar algum tipo de problemática ali, mas eu nunca parei para fazer, teria que pesquisar e fazer uns experimentos. é uma forma de reduzir o code smell ao longo do desenvolvimento.

P. alguma outra sugestão?

E. A outra é alguém criar uma ferramenta de Inteligiencia Artificial que tivesse um plugin que lesse o que está escrito. Algum link alguma coisa do gênero. Basicamente isso. É uma coisa que você não chegou a perguntar é sobre mock.

P. Não perguntei, mas como a próxima pergunta tem a ver com se você quer acrescentar alguma informação, fique a vontade.

E. Eu ja trabalhei com teste unitário a muitos anos, até hoje no time dos colegas acabam me tendo como referência nessa parte, quando o pessoal tem alguma dúvida ele me procura. Uma coisa bem interessante é que lá a gente trabalha muito com mock sendo gerado automaticamente. Isso gera uma performance muito boa no nosso teste unitário porque você coloca a classe que você quer lá no parâmetro para ser recebido e o próprio sistema já gera aquele objeto para você sem você precisar codar nada. Então a geração automática de mock tenho experimentado de um ano para cá, a performance de execução de teste é muito rápida. quando a gente precisa de ter um mock específico para algum cenário, a gente faz uma configuração em determinado local e sempre que alguém gerar esse mock automático, ele já gera conforme esse padrão que a gente colocou, por exemplo. Imagina que a gente trabalha numa financeira. Então você quer sempre ter um cartão de crédito válido que esteja dentro de uma string. Se você pede para gerar um mock automaticamente, ele ia pegar qualquer string lá dentro e não ia servir como um cartão válido para você, um cartão mascarado. A gente consegue configurar um cartão mascarado para aquela propriedade, quando qualquer pessoa for usar. então isso é bem interessante. uma outra coisa que eu já usei muito, aquele a gente nem usa, mas também é interessante. é usar o padrão de projeto builder para construir objetos também. construir mocks, é uma coisa que agrega bastante o teste unitário. Acho que é basicamente esses pontos mesmos.

P. Você tem alguma pergunta para mim, ou tem alguma sugestão para melhorar a entrevista?

E. num primeiro momento quero te parabenizar pelas perguntas, bem interessante. Umas coisas que eu perguntaria para nível de pesquisa, eu não sei onde você quer chegar com sua pesquisa, qual o objetivo final. Talvez perguntar um pouco sabe o que é de fato o que é e para que serve teste unitário, porque tem muita gente que faz e não sabe, eu era um desses no passado. Ter algum quantitativo na sua pesquisa se o pessoal tem consciência do que está fazendo é interessante, então acho que é algo maneiro. Eu acho que é basicamente isso. Acho que esta maneiro. Acho que seu trabalho está ficando muito legal, acredito que você queira pegar um quantitativo de quantos fazem, quantos não fazem, quantos sabem o que está fazendo o que não sabe...

P. Falando um pouco mais da minha pesquisa para você e depois se você quiser depois acrescentar alguma informação fique a vontade. Minha pesquisa é focada na parte da qualidade do código de teste, especificamente em se tratando de test smell. Por isso que da metade para o final as perguntas têm a ver com esse tópico. Ela tem como base minha própria experiência enquanto profissional. Eu trabalhei como QA durante 8 anos numa empresa e só depois que eu voltei para a academia que eu tive acesso a essa temática, a test smell, tipos e diferença entre test e code smell. Eu quis voltar para a indústria enquanto pesquisador para entender como esse processo está ocorrendo na indústria, se os profissionais conhecem o tópico, se de alguma forma mesmo que não conheçam, mas lidem com isso no dia a dia que era o meu caso, eu não conhecia o tema test smell mas a gente tratava da qualidade do código de teste no dia a dia, identificando duplicata, identificando coisas que poderiam ser melhoradas, mas não conhecia o termo especificamente, então eu queria saber como isso ocorria de fato na indústria num processo geral e ver uma forma de melhorar os processos de qualidade de criação de código de teste.

E. É bem pertinente sua pesquisa. O mercado é muito carente de teste unitário, a empresa que eu trabalho hoje é uma das empresas depois de muitos anos que leva teste unitário a sério. E o pessoal sabe o que está fazendo. Geralmente não é valorizado o teste unitário em si, quanto mais o test smell. O mercado não sei como é lá fora, mas o mercado brasileiro que eu trabalhei, já trabalhei em mais de 8 companhias e é uma carência muito grande de testes, já tive vários embates em alguns locais que trabalhei, por o pessoal não querer, não dar valor a teste unitário. Têm vezes que até reclamavam de cronograma, enfim. Às vezes é bem espinhoso. As vezes valoriza mais teste unitário quem trabalha com produto e tem um zelo grande com o produto. É uma temática bem pertinente. Você pode se posicionar como um especialista nisso aí e escrever sobre isso, palestrar. Acho que tem muito mercado aí.

P. essa é a ideia. Muito obrigado por ter participado da pesquisa. As informações que você me passou elas ajudam muito a compor o cenário e ajudar nas próximas etapas da pesquisa, então eu te agradeço muito.