Pesquisador. Obrigado por participar dessa 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. Está bom.

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. Ok

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

E. Comecei a trabalhar já no meu primeiro emprego, no meu primeiro estágio em janeiro de 2003, já comecei a trabalhar com desenvolvimento de software, especificamente em PHP. Na época, por ser uma empresa antiga, vamos dizer trabalhava no setor que desenvolvia intranet, então era um sistema web interno. Eles não tinham, eram um pessoal antigo, que não tinha experiência com teste automatizado então até aquele momento eu trabalhava com bastante teste manual e poucos testes unitários, testes automatizados. E foi assim que eu comecei. Comecei como estagiário e aí eu fui me interessando mais e estudando mais sobre teste unitário, camada de teste, os níveis de teste, a importância deles, sobre a pirâmide de teste, os anti-padrões, como cone crew anti pattern, enfim, fui embora nessa área, mas começou por volta de janeiro de 2013.

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

E. Eu sempre me interessei muito sobre qualidade de software, desde o início eu sempre quis desenvolver algo com qualidade. Um pouco antes, na faculdade, eu fazia alguns cursos de PHP, da linguagem que eu programava na época e o pessoal sempre comentava um pouco sobre testes e a partir disso eu fui me interessando mais e fui me aprofundando mais depois que tive um contato com o pessoal da comunidade de PHP, PHPSP aqui da região, e o pessoal falava sempre bastante de testes unitários, a importância e eu fui procurar estudar mais. Aí eu fiz curso, especificamente sobre testes e fui me aprofundando. Foi por comentário de cursos online e pela comunidade que eu costumava frequentar bastante, a comunidade PHPSP. E foi por causa dela que eu me tornei um dos fundadores do PHP Vale que é uma comunidade também da região.

P. Em quais linguagens você cria testes?

E. Eu já trabalhei com Javascript, Java, Ruby e principalmente com PHP.

P. Quais linguagens fazem parte dos projetos que você atua atualmente?

E. PHP e Javascript com React Native para mobile.

P. Como é seu processo de criação de teste?

E. Algumas empresas que eu trabalhei, que a gente consegue implementar um padrão de desenvolvimento, a gente sugere, vou falar um pouco do processo operacional, que é basicamente, eu praticamente gosto e sempre tento passar para a equipe de desenvolver utilizando a metodologia de desenvolvimento guiado por testes, que é o TDD, então escreve os testes primeiro e depois escrever o código de implementação. Pensando mais no operacional e nos testes unitários para criação de um código de qualidade ao nível mais baixo. Pensando num nível mais alto a nível de negócio, eu gosto de conversar com os analistas de negócio para tirar aquelas técnicas de BDD: Given, When, Then. Basicamente os analistas de negócio, escrevem os testes de aceite baseados nessa metodologia e aí a gente, da parte de desenvolvimento consegue transformar isso em testes de aceitação de uma forma mais fácil através de ferramentas de bess em BDD, como por exemplo no PHP a gente tem o Behatch, no Ruby tem o Cucumber e outras ferramentas parecidas, então dependendo do nível que a gente está no projeto, eu gosto de utilizar essas metodologias, mas no operacional é seguindo o TDD

P. Existe em alguma das empresas que você trabalha algum fluxograma, template que padronize esse processo de criação do código de teste de unidade?

E. Na última empresa que eu trabalhei existia uma documentação interna. Hoje eu presto consultoria só, mas basicamente nessa documentação interna tinha como funciona o fluxo de desenvolvimento da empresa desde o momento que o O.P.O. faz o levantamento de requisitos até o momento que o desenvolvedor pega a tarefa, começa o desenvolvimento e manda para a equipe de testes, que era responsável por fazer alguns testes manuais, que as ferramentas automatizadas não conseguiam cobrir. Algumas empresas que eu trabalhei tinha outras não.

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

E. Basicamente aquela família do XUnit, PHPUnit para PHP, JUnit para Java, para Javascript tem o Jest, basicamente essas são as principais.

P. Como é o processo de verificação de qualidade do código de teste criado?

E. Nas empresas que eu trabalhei a gente sempre buscou ter o processo de code review, porque a gente tentava seguir os padrões Solid, que são o Respoonsability, open-closed principal, Liskov substitution principle, Interface segregation e Dependency inversion, que asssim, esse é o principal padrão para criar teste com qualidade e aí a gente usava algumas ferramentas para poder verificar isso. No php no caso tem por exemplo uma ferramenta chamada phpmess detector que ele verifica a qualidade de código, tem umas outras, php stan, enfim. Aí a gente no processo de code review a gente verificava se tanto a classe de implementação quanto a classe de teste ela ficou legível, bem coesa, para poder estar fazendo o merge desse código nas branchs de tanto de homologação quanto de produção.

P. esse code review, essa pessoa que submetia o código para revisão, essa revisão era feita por quantos profissionais?

E. No caso ela marcava os outros membros da equipe que estavam aptos a fazer o code review e a gente meio que tinha um chat central, tipo o slack, e os desenvolvedores mais experientes que estavam marcados nesse code review, olhavam esse código. Teve empresa que tinham 3 pessoas sempre disponíveis para fazer esse code review, teve empresas que teve mais, outras empresas, outros momentos numa mesma empresa era só eu, então variava bastante. Numa equipe de 5 pessoas, duas ou 3 faziam isso.

P. Como é realizada a manutenção do código de teste de unidade?

E. Uma forma que a gente utilizava para manter uma cobertura de testes boa na aplicação, é no processo de integração contínua. Você marcar um percentual de cobertura mínima que você exigia no projeto. Então a gente sempre exigia 80%. Então todo o código que subisse verificava se estava com teste, ou uma parte dele para o nível de cobertura da aplicação não caísse como um todo. Para que o processo de integração contínua não rejeitasse o código automaticamente. Então essa era uma forma de manter o projeto com testes. Agora manter o teste, de uma classe, como eu disse eu sempre gostei de seguir o solid, então uma vez criada uma classe com uma única responsabilidade dificilmente a gente gostaria de altera-la, a gente só gostaria de altera-la só se alterou a regra de negócio e realmente fizesse sentido alterar aquela classe. Do contrário uma vez criada uma classe, o teste para ela e entregue, o melhor seria ou entender ela para fazer uma alteração específica ou ela deixar de ser usada. Basicamente é isso, uma vez entregue, o ideal é que o teste não seja alterado mais.

P. O que você conhece sobre test smell?

E. Test smell, esse termo específico eu nunca ouvi falar. Eu já ouvi falar de code smell, mas test smell não.

P. O test smell ele descende do code smell. Eu vou te passar o conceito sobre o assunto, vou te mandar 2 exemplos de test smells, eles estão em Java mas o importante é muito mais em relação a estrutura do teste do que necessariamente a linguagem. Se você tiver alguma pergunta você me faz porque as próximas perguntas a partir de agora vão estar relacionadas a esse tema, ok?

E. Está bom, pode mandar.

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. É basicamente um clean code para o teste.

P. Tem a ver com essa questão de clean code. Vai ter test mell que vai estar relacionado a construtor, a assert, a um valor específico para parâmetro que eventualmente um teste pode passar porque está com um valor específico, dependência de testes, então ao depender da ordem que o teste é feito, ou ele vai falhar ou ele vai passar porque algum teste anterior depende dele diretamente, então os smells de teste estão relacionados a essas questões. Vou compartilhar contigo os exemplos e te explico. Esse primeiro exemplo é chamado de 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 falhe é difícil de identificar o porque esse teste falhou considerando sua estrutura. Ou também dependendo do profissional fica mais dificil dele entender a lógica desse teste por exemplo.

E. Aí por exemplo a quantidade de encadeamento, de complexidade ciclomática inclui? Um if dentro do outro depois um for.

P. Isso interfere, mas principalmente a existência dos condicionantes e das repetições. A quantidade necessariamente não interfere, mas sim a existência.

E. Num caso desse por exemplo, Java também aquele data provider que pode ser substituído por um loop desse aí.

P. Você pode dividir, para cada condicional, um teste específico, por exemplo. Esse outro exemplo é o chamado Empty test smell. Ele é considerado um smell porque não tem nenhum script executável. Então fica difícil de entender porque esse teste está comentado ao invés de executado ou de ter sido excluído, por exemplo. Então tem a questão de não estar relacionada a execução ou entendimento do que está escrito, mas no entendimento de porque esse teste está se apresentando daquela forma.

E. Está bom.

P. Alguma dúvida?

E. esse no caso seria separar as responsabilidades mais específicas de cada teste, do que fazer um teste maiorzão

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

E. eu vou falar um pouco da minha experiência, porque eu nunca, já devo ter lido, só que pouco sobre melhoria de qualidade de código de teste, mas no geral, por exemplo, uma estratégia que eu uso bastante, que a gente sempre conversa sempre os desenvolvedores é utilizar as técnicas tanto de clean code, utilizada para o código de implementação comum, quanto pro utilizar as técnicas de refactoring, que tem um livro do Martin Fowler que tem várias técnicas de refatoração que incluem isso aí, de você ter um método mais bem dividido, qual o método que vem primeiro, qual que vem depois, qual a nomenclatura dos métodos, então eu praticamente sempre tentei seguir e sempre tentei passar para as equipes que eu desenvolvi, nos code reviews que eu faço, usa essas técnicas, essas estratégias que são utilizadas no código de implementação comum, para serem utilizados também nos arquivos de teste, tanto para poder manter uma legibilidade boa dos testes, quanto para um reaproveitamento do código de teste. As mesmas técnicas que a gente utiliza para a implementação de um código a gente tenta usar para o código de teste. Claro, uma coisa importante nos testes que é bom ter, que é ter um conhecimento mais abrangente da ferramenta que você está utilizando, por exemplo do phpunit, você conhecer todas as funcionalidades que ele provê porque você consegue ter várias facilidades que ele te dá, por exemplo, do data provider que o java tem, o phpunit também tem, que é uma forma de você colocar uma notation em cima de um método de teste você provê alguns dados para executar vários testes de uma vez só.

P. Esse cuidado em relação a clean code, o uso de ferramenta para melhorar a escrita desse teste ele também ocorre no processo de manutenção?

E. Sim. Tanto na hora de desenvolvimento quanto na hora de manutenção a gente tenta manter essa qualidade. Na primeira entrega as vezes passa alguma coisa então sempre a gente gosta de passar para o pessoal dos times que trabalho, a gente usar uma coisa que o uncle bob fala no livro de clean code que é da regra de escoteiro, naquele lugar que você vai você deixar mais limpo do que você chegou. Se você vai fazer uma manutenção do código, só aquela parte especifica do código que você está mudando, se possível deixar um pouco mais limpo do que você encontrou. Eu acho essa uma boa técnica. Também você não pode querer refatorar tudo para não quebrar outras funcionalidades, mas só exatamente aquilo que você está mexendo, porque as vezes as pessoas se confundem e querem refatorar muita coisa e aí acaba prejudicando o negócio. Saber balancear e fazer a refatoração especificamente da parte que você está mexendo.

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

E. Acho que uma documentação da equipe, um processo especificado ou uma ferramenta automatizada, não sei. Pelo que eu venho trabalhando é mais com um processo definido mesmo. Eu não sei se tem alguma ferramenta, alguma coisa automatizada que ajude a fazer isso ou um razon boiled plate, uma estrutura, um esqueleto de um modelo de um processo que você pode estar copiando ou seguindo, acho que seria uma boa forma de estar melhorando os códigos de teste. Especificamente de teste eu nunca vi, mas acho que poderia ser parecido com o código mesmo, ter algo para consultar e estar verificando se está seguindo as boas práticas.

P. Estou chegando no final da entrevista, você tem alguma pergunta para mim, quer acrescentar alguma informação ou ainda tem alguma sugestão para melhorar na entrevista?

E. Eu acho que foi boa a entrevista, tem bastante coisa, acho que é mais relacionada a essa parte específica de ferramentas, o pessoal tem bastante conteúdo para melhorar a qualidade, mas sempre voltado para o código de implementação, mas para o código de teste eu acho que isso está em falta e talvez você pode. Eu acho que as perguntas que você fez estão boas, mas eu acho que você pode checar isso mesmo, se tem algum conteúdo específico para melhorar a qualidade do código de teste ou se o pessoal está usando as mesmas técnicas de código de implementação. De resto é isso.

P. Muito obrigado pela sua participação. Caso você tenha algum contato que você possa me indicar para que eu possa realizar a entrevista também eu vou agradecer bastante.

E. Está bom.