Pesquisador. Obrigado por participar dessa entrevista. Ela tem como objetivo compreender como funciona os processos de criação e de manutenção de código de testes e como ocorre a verificação da 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. 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.

E. Sem problemas

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

E. No início um amigo, ele era meu superior em outra empresa que trabalhei, ele pedia que a gente testasse as funções que processassem algum tipo informação para garantir que elas estavam sendo feitas da forma correta e, algumas funções da aplicação, algumas rotinas que dependessem de outras rotinas para garantir que essa aplicação funcionasse, ou se alguém desse manutenção futuramente, que não tivesse quebrado ou impactado dessas determinadas rotinas.

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

E. Google, Slackoverflow e o próprio network mesmo, o pessoal na empresa que a gente vai trabalhando.

P. Em quais linguagens você cria testes?

E. Eu trabalho com Javasript, Typescript, C#, Java, Phyton também. Acredito que toda linguagem que eu conheço, eu trabalho com teste desde que a rotina seja necessária ser testada, traga algum ganho para a gente conseguir a qualidade da aplicação, manter a manutenibilidade, etc.

P. Qual plataforma do seu projeto atual?

E. Hoje eu trabalho com Javascript, e Typescript no client, na parte de processamento é COBOL.

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

E. Meu processo hoje inicialmente consiste, no primeiro passo, o usuário homologar a rotina que vai ser testada e depois que eu começo a trabalhar nos testes. Hoje eu dou mais preferência, como eu trabalho com a parte do client, eu trabalho com testes automatizados de interface, então eu simulo como se o usuário estivesse interagindo com o sistema e vendo o que ele retorna. Eu testo a aplicação como um todo e não mais como o unitário. Mas na parte de processamento de dados, que a gente fala que é o back end, aí algumas determinadas funções que são mais complexas que dependem de outras rotinas, acabo fazendo teste para garantir que elas não quebrem quando a gente faz um build novo ou quando tem alteração na aplicação.

P. Hoje você trabalha mais com teste de interface que teste de unidade?

E. Isso, eu dou preferência para teste de interface porque eles testam a aplicação como um todo, eles testam a entrada e do usuário e a saída dela.

P. Eu vou focar mais no teste de unidade, que é o foco da minha pesquisa. É o mesmo processo que você utiliza para criação de teste de unidade ou é outro processo?

E. O teste unitário, eu costumo trabalhar, como ele é mais funcional, mais código, é mais entrada e saída de processamento de dados, você vai escrever uma função e esperar que ela faça determinada ação. Ela tem alguma entrada e te retorna alguma coisa. Quando eu sei o retorno, quando eu sei o que a função vai fazer, especificada já, o usuário não vai chegar e alterar essa função, então eu faço teste unitário para garantir, mas sempre depois que o cliente homologa, pois não faz sentido a gente fazer teste em cima de algo que ele ainda não homologou. Muitas das empresas hoje ao meu ponto de vista eu acho que elas acabam comprando, por exemplo o TDD do desenvolvedor, sendo que eles não têm o escopo do sistema definido, então o desenvolvedor acaba perdendo tempo testando alguma rotina que provavelmente nem fique no sistema futuramente.

P. Quais ferramentas são utilizadas para criar teste de unidade?

E. Com C# trabalhava com o Spring test, no Java era JUnit, no Javascript e Typescript já trabalhei com Karma, Mocha+Chai, Jest e Jasmine.

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

E. Geralmente é o do framework que a gente utiliza, o bootstrap. Você lança o comando e ele cria a estrutura.

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

E. O processo genérico da empresa ele basicamente consiste em analisar todos os seus códigos e verificar se os testes que você desenvolveu não estão quebrando. Esse é o processo da qualidade do código. Infelizmente é só baseado em código do teste. Algumas empresas com que eu já trabalhei elas tinham um processo melhor, que consiste no code review, que vários desenvolvedores analisavam o código da pessoa para ver se o código foi feito da melhor forma possível ou não, se ele está com código clean ou se o desenvolvedor está fazendo várias interações em cima de uma rotina que deveria ser simples.

P. Nesse caso, atualmente é só verificada a estrutura do código de teste, vendo se ele não está quebrando e uma vez que ele não quebra...

E. É verificada a entrada e saída, por exemplo, eu tenho 100 testes programados na aplicação, ele vai rodar os 100 testes e vai olhar se tem algum teste quebrando. Se tiver algum teste quebrando, não entra para produção.

P. Passando para o processo de manutenção do código de teste, me conte como é este processo na empresa?

E. Geralmente o processo padrão é você alterou alguma rotina e ela começou a impactar no teste, o teste que havia sido programado anteriormente. Se esse teste estiver quebrando é válido analisar a rotina, ver se mudou a entrada ou a saída e se deveria quebrar esse teste. Se o teste estiver quebrando de fato e ele deveria estar quebrando, a gente altera a rotina de testes para suportar, atualiza essa rotina de testes, senão é repassado para o desenvolvedor que fez a alteração para ele olhar no fluxo que ele mexeu que não era para ter impactado naquele teste. E aí ele vai lá e refaz.

P. O que você conhece sobre test smell?

E. Disso daí eu nunca ouvi o termo.

P. Você já ouviu falar de code smells?

E. Sim. Code smell eu já ouvi.

P. É o mesmo princípio de code smell só que voltado para o teste de unidade: “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. Entendo. Como eu te falei, os testes eles devem ser feitos, no meu ponto de vista, eles devem ser feitos nas rotinas da aplicação que necessitem ser testada. Não dá para testar a aplicação inteira, por exemplo, como eu cheguei a te falar, as empresas cobram que você faça teste unitário para toda a parte do código de front end hoje em dia, que é só camada gráfica de apresentação da aplicação. Elas requerem que essa aplicação tenha 100% de cobertura de código, você acaba perdendo tempo mais para fazer teste do que para entregar a aplicação. O teste que não vai agregar nada na aplicação.

P. Você possui alguma dúvida sobre esse tema?

E. Pelo que você disse são testes necessários e simples.

P. Eu vou te mandar um exemplo de test smell e a gente conversa um pouco mais, pois as perguntas seguintes tem a ver com test smell, então é interessante que se você tiver alguma dúvida a gente consiga tirar aqui.

E. Sem problemas.

P. Esse primeiro exemplo que eu te mandei é chamado "Conditional Test Logic", ele está relacionado a testes que são criados com estrutura de repetição, estrutura condicional. Nesse caso, ele é considerado um smell, pois uma vez que ele falhe fica mais difícil identificar em qual momento dessa estrutura de repetição ou da condição que ele falhou. O ideal é que essas condições sejam separadas em testes diferentes para que uma vez que ele falhe, caso falhe, seja possível identificar em qual rotina ele falhou.

E. Entendi o conceito.

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

E. Na empresa nós não temos nenhum setor que verifique esse tipo de problemas no teste.

P. Isso tanto na criação com na manutenção?

E. Exatamente.

P. Para você, como seria possível prevenir/evitar a inserção de test smell no código de teste de unidade no momento que esse teste é criado?

E. Eu acredito que esse é um processo cultural. Tem que ser desenvolvida a cultura no seu time, na sua equipe de trabalho e esperar que seja realmente seguida essa cultura. Mas para garantir mesmo é só analisando o código antes dele entrar para homologação ou para produção.

P. Você tem alguma pergunta para mim ou quer acrescentar alguma informação ou ainda tem alguma sugestão para melhorar nossa entrevista?

E. Nenhuma, acho que está legal.