Pesquisador. 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. Tudo bem?

Entrevistado. Tranquilo

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. Ok.

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

E. Eu nunca trabalhei como testador então o contato que eu tive com a parte de testes é fazendo a automatização de testes, um dos projetos que eu participei na empresa que eu estou agora mesmo era para automatizar testes naquelas máquinas de cartão, P.O.S. Então a ideia era que a gente tinha uma biblioteca que a gente implementou e ela fazia essas transações de cartão de MV e a ideia era otimizar esse teste porque do jeito que era feito era manual, você tinha que inserir o cartão, aí ia imprimir uma rede de dados de bytes lá, tinha que ficar passeando esses dados manualmente, então a gente fez essa otimização desse processo. Essa leitura de dados, para ver se passava automaticamente para ficar mais rápido para quem fosse testar essa biblioteca que a gente implementou conseguisse fazer com mais facilidade. Então esse foi o primeiro contato, mas aí além disso o que eu tive mais contato também foi com testes unitários também. Alguns dos projetos que eu participei, no banco nem todos tinham testes unitários, mas alguns, o que eu estou atualmente tem, foi onde eu tive mais contato, tanto nessa parte de automatizar testes e a parte de testes unitários durante o desenvolvimento mesmo para validar as funções específicas.

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

E. Eu tive uma fonte específica. Foi o Google.

P. Em quais linguagens você cria testes?

E. Em C, C++

P. Qual linguagem do seu projeto atual?

E. linguagem C

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

E. Geralmente a gente trabalha, hoje com o que a gente está fazendo. A gente trabalha com scrum, então quando a gente faz o planejamento da sprint, que a gente define as tarefas, a gente geralmente já define algumas tarefas são necessárias testes unitários, outras não. Então para as que são necessárias o time mesmo já define que vão precisar de testes unitários e quem puxar aquela tarefa fica responsável de desenvolver os testes unitários para aquela funcionalidade que vai ser desenvolvida. Hoje a gente tem trabalhado assim no projeto que estou atualmente.

P. Existe algum fluxograma, template documentado que padronize esse processo e criação do código de teste?

E. Não, o formato que a gente cria os testes unitários hoje, acho que quando iniciou, eu não estava no projeto ainda, mas se eu não me engano, mas tem um modelo do Google, uma biblioteca de testes unitário de C ou C++. Então a gente segue mais ou menos aquele padrão para criar os testes. Depois que fez a primeira parte o pessoal foi só seguindo a estrutura. Não tem um padrão documentado para a gente seguir.

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

E. A IDE que a gente usa não é fixa, mas a maior parte do time usa VS Code, da Microsoft e outros usam o Eclipse. Eu uso o VS Code.

P. Como é o processo de verificação de qualidade do código de teste criado, por exemplo: verificação de escrita, alguém revisar o teste depois de criado, verificar compatibilidade com o requisito, entre outros fatores?

E. A revisão do teste em si, a gente faz junto com a revisão do código, então no processo que a gente tem hoje para cada coisa que a gente implementar, a gente tem um processo de revisão de código. A gente usa o gitlab hoje, então quando a gente implementa um código, uma funcionalidade nova ou implementa um teste, vai junto. Então a gente sobe esse código no git, e lá no gitlab ele tem uma tela que ele tem essa ferramenta de merge request que eles chamam lá e a gente faz a revisão desse código por ali. Aí o time inteiro, quando uma pessoa faz o merge request, para adicionar um código novo da release, geralmente 4 ou 5 pessoas avaliam esse código antes de mergear ele, antes de integrar na release. A gente sempre coloca o arquiteto do time, um desenvolvedor mais experiente, e coloca mais 2 desenvolvedores ou uma pessoa de testes também faz parte nessa revisão de código. Fica aberto para o time todo ver, mas essas pessoas são obrigatórias de ver. Se elas não aprovarem, o código não pode ser mergeado. Então tanto para o código da aplicação quanto para os testes unitários funciona da mesma forma, então se tiver alguma correção nos testes unitários, a pessoa ali mesmo vai comentar que poderia adicionar mais casos de testes, ou o caso de teste adicionado está errado ou coisas desse tipo. Nesse processo de revisão de código que o time vê.

P. Passando para o processo de manutenção do código de teste, como é realizado esse processo dentro da equipe?

E. A gente não tem nenhum processo específico para os testes, hoje a forma que a gente trabalha a gente só mexe nos testes unitários quando a gente vai implementar uma funcionalidade nova, por exemplo. O que acontece, que eu já vi acontecer, é a gente precisar implementar uma funcionalidade nova e os testes unitários referentes a ela, alguns testes legados que já tinham a muito tempo, estão ruins. Então a pessoa que for escrever os testes novos refatorou os testes. Mas não tem um processo, é sobre demanda. Conforme a gente vai precisando a gente vai alterando os testes se forem necessários.

P. O que você conhece sobre test smell?

E. Eu acho que não conheço. Já ouvi falar de code smell, mas não de test smell. Não sei se é a mesma coisa

P. tem a ver, test smell eles descendem dos code smell. Eu vou te passar o conceito, vou te passar 2 exemplos sobre esse tópico, porque as perguntas a seguir estão relacionadas a ele, ok?

E. Legal

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. Certo

P. Eu vou te passar 2 exemplos de tests smells, falar um pouco sobre eles e qualquer dúvida que você a gente alinha para depois seguir, ok?

E. Ok.

P. Esses testes estão em Java, mas é muito mais a questão da lógica do que da linguagem. O primeiro exemplo que vai ser compartilhado é o chamado Empty test smell. Ele é considerado um smell porque não tem nenhum script executável. Então esse teste pode passar sem estar de fato testando o código do software em si. E além disso, você não tem a informação de porque ele está comentado, porque ele não está sendo executado ou porque ele não está sendo excluído. Acaba no momento que alguém detecta o teste dessa forma não vai ter nenhuma informação de porque ele está desse jeito no conjunto de testes da aplicação. O outro exemplo ele é chamado de Conditional Test Logic smell. Ele é considerado um smell porque ele tem uma estrutura de condição, de repetição dentro dele. Então uma vez que sse teste falhe vai ser mais difícil de identificar exatamente o porquê que esse teste falhou e também ao depender do desenvolvedor ele pode ter dificuldade para entender, de acordo com a experiência dele. de compreender o que está sendo testado por conta da estrutura. Existem outros smells relacionados a asserts, a construtores, a valor de parâmetro, a dependência dos testes que estão relacionados com um mesmo recurso e ao depender da ordem que são executados um falha e outro passa e vice-versa, então são formas que esse teste é codificado que acaba prejudicando a questão de independência de estabilidade, repetibilidade desses testes, ok?

E. Ok

P. Posso passar para a próxima pergunta?

E. Sim, consegui entender bem

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

E. Eu acho que esses casos, esses exemplos que você deu, são exemplos que a gente muito provavelmente vai identificar na revisão do código. Porque geralmente a gente tem alguns padrões que a gente segue no código, tanto na aplicação que a gente acaba adotando e também para os testes unitários. Mesmo que ele não tenha uma documentação específica a gente acaba adotando alguns padrões. Então essas coisas, igual a que você mostrou que uma função tem só comentários ali e não sabe porque foi comentado aquele teste, então esse tipo de coisa se a gente vê durante a revisão do código, com certeza alguém do time vai escrever lá "esse teste aqui ou você remove ele ou explica porque ele está comentado. Deixa um comentário explicando porque ele está comentado”. O outro caso que você mostrou de um teste mais complexo que pode ter outros tipos de problema eu acho que cairia no mesmo cenário. Na experiência que eu tenho com o time que eu trabalho, provavelmente na revisão do código esses problemas seriam identificados.

P. Isso aconteceria tanto na criação quanto na manutenção ou na manutenção teria algum outro processo diferenciado para verificar essas questões de complexidade, dependência, assim por diante?

E. Eu acho que na manutenção, no sentido de alguém alterou alguma coisa no código, a revisão do código é feita da mesma forma. Acho que o que a gente não tem hoje nada, pelo menos desde quando eu entrei no time, é ter um processo de alguém avaliar os unit tests. Eu sei que no código que a gente tem hoje tem coisas duplicadas, tem coisa que não precisava estar lá, que precisaria de um refactory, mas não tem um processo para alguém sentar e olhar o código antigo e corrigir isso, mas se alguém precisar fazer manutenção, aquele código que for mexido ele vai ser revisado.

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

E. Para mim, me parece que o processo que eu trabalho hoje parece ser adequado. No sentido de que todo o código é revisado por várias pessoas do time, é muito difícil, não impossível, mas é difícil acontecer de passar alguma coisa, passar várias coisas, vários erros despercebidos, tanto no código da aplicação, quanto no teste unitário. Me parece que esse processo de revisão do código sendo sempre bem feito, o time maduro que consegue sempre avaliar o código reduz significativamente, consegue reduzir bastante esse problema de código, essa questão de test smell ou code smell.

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

E. Não, para mim foi bem tranquilo na verdade. Espero que tenha conseguido esclarecer bem o que você precisava saber e também estou aberto se precisar de mais alguma dúvida, alguma coisa. Porque pessoalmente não tenho conhecimento muito grande da parte de testes, mas essa parte de testes unitários que trabalhei um pouco agora, automação de testes, espero ter ajudado.

P. Sim, ajudou sim. Não tenha nenhuma dúvida disso. Toda a informação que a gente recebe serve para compor o cenário e para a gente também conseguir fazer a pesquisa. Minha pesquisa voltada para test smell, então entender como funciona esse processo dentro da indústria. Se de alguma forma os profissionais conhecem, se está sendo tratado de alguma forma no contexto industrial e se de alguma forma posso estar melhorar esse processo.

E. Legal. Uma pergunta só. Não sei se você já tem esses resultados parciais, mais o que você tem acompanhado na sua pesquisa. Existe bastante esse processo, além do code review tem outras técnicas que você tem pesquisado que ajuda a reduzir esses test smells. Tem outras técnicas?

P. Do que eu tenho de pesquisado, existem ferramentas que fazem essa detecção, e que podem ser utilizadas para auxiliar a refatoração. Inclusive outros pesquisadores do meu grupo de pesquisa, trabalham com base nisso. Identificar ferramentas, melhorar ferramentas que fazem esse tipo de detecção e indicar algum tipo de melhoria no código de teste. Na prática o que a gente tem identificado é mais esse processo de code review mesmo, o código ele é criado da forma que o profissional já tem o hábito de fazer ou é estabelecido algum padrão de equipe e essas questões de escrita tem sido feitos de fato via code review.

E. Beleza.

P. Muito obrigado mais uma vez. Se você tiver algum contato que você possa me indicar para que eu possa realizar a entrevista também eu vou agradecer bastante.