ÁGIL OU TARDIO:
A ADOÇÃO DE TESTES AUTOMATIZADOS NAS EMPRESAS

Sablina Alves de Carvalho[1]
Thiago Vasconcelos Costa Freire[2]

RESUMO

Este trabalho se destina a entender a qualidade de software, juntamente com os testes automatizados. Ao longo será abordado os tipos de testes e os seus benefícios, o presente trabalho surgiu como uma necessidade de se ter conhecimento de como as automações de testes estão sendo aceito pelas empresas e as ferramentas que mais estão sendo utilizadas na atualidade. Validando se é ou não vantajoso fazer essa adoção, e refletir com as opiniões dos entrevistados sobre os testes automatizados.

Palavras-chaves: Software; Qualidade de Software; Testes; Automação de testes; Empresa.

ABSTRACT

This work is intended to understand software quality along with automated testing. Throughout the types of tests and their benefits will be addressed, the present work emerged as a need to be aware of how test automations are being accepted by companies and the tools that are being used the most day. Validating whether or not it is advantageous to make this adoption, and reflecting with the pinions of respondents about automated tests.

Keywords: Software; Software quality; Tests; Test automation; Company.

1 INTRODUÇÃO

O desenvolvimento de software nunca esteve tão presente quanto no tempo atual, e esse crescimento se dá a um dos grandes fatos da procura por otimização de processos manuais.

Nem sempre tivemos a tecnologia como aliada no gerenciamento de atividades do cotidiano. Nos dias de hoje, encontramos uma gama de softwares que são voltadas para diversas áreas, seja na área da saúde, engenharias, ciências sociais e outras.

O desenvolvimento de software nem todo tempo foi “perfeito”. Na década de 70, houve um acontecimento que ficou conhecido como a crise do software, pois foi quando surgiu uma série de problemas relacionados ao processo de desenvolvimento na época (REZENDE, 2006), isso se deu a grande procura por softwares, já que cada vez mais sua popularidade crescia.

[1] Graduanda do Curso de Sistemas para Internet. E-mail: sablinaalves13@gmail.com
[2] Professor Orientador, Graduado em Sistemas de Informação, pela Universidade IESP. E-mail: Thiago.freire@iesp.edu.br

Segundo Sisson (2007), a complexidade dos softwares que surgiam na época era muito maior do que a que os desenvolvedores estavam acostumados, e a falta de técnicas e ferramentas como as que conhecemos hoje em dia, foi o um dos principais motivos para o surgimento desta crise.

De acordo com Rezende (2006, p. 11) assegurar a manutenção de um software é uma tarefa difícil, já que em boa parte dos casos este custo chega ao valor de 60% a 80%. Agora imaginemos isso em um software que está no mercado e apresenta falhas constantes, o valor para manter esse software pode ser muito maior do que ao valor que foi a sua construção.

Para evitar este e outros tipos de problemas, é o que a área de qualidade de software está disposta a nos proporcionar, podemos dizer que ela busca a garantia de qualidade do software que está sendo construído, verificando se ele funciona da maneira correta. Essa verificação é feita a partir de testes, onde esses testes são definidos em níveis e fases, sendo assim, nos ofertando uma diversidade de opções para um determinado momento durante a construção. Sem os testes nossa aplicação tem maiores chances de nos retornar determinados prejuízos, e dependendo da área para qual esse software irá ser projetado, pode causar até mesmo danos físicos.

Por exemplo, o caso Therac-25 ocorreu entre junho de 1985 e janeiro de 1987, o acelerador médico de elétrons Therac-25 esteve envolvido em seis overdoses maciças de radiação (LEVENSON; TURNER, 1993). Como resultado, várias pessoas morreram e outras ficaram gravemente feridas. E isso aconteceu por falta de um teste de integração.

Nos dias atuais, há uma variedade de materiais sobre a área de software, as técnicas utilizadas, e ferramentas. É eminente se compararmos a época da crise do software. Mas mesmo diante desses artefatos, muitos softwares ainda apresenta a baixa qualidade inferior à que hoje o mercado exige.

Com base nesses fatos expostos, podemos indagar “Na garantia de qualidade de software, que relevância tem os testes automatizados no desenvolvimento de aplicações?”. Tendo a hipótese de que as empresas não custeiam os testes automatizados com receio de não encontrar profissionais qualificados, e isso lhe gerar como retorno um prejuízo financeiro. Assim, o objetivo deste artigo é expor o processo de desenvolvimento de software e a validação dessas aplicações nas empresas através dos testes automatizados.

Nos próximos tópicos serão abordados o desenvolvimento de softwares, dando sequência sobre qualidade de software e os tipos de testes.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 SOFTWARE

A área da tecnologia da informação tem ganhado gradativamente mais espaço no mercado de trabalho e na nossa vida, tarefas que não exigem muito esforço, se tornaram ainda mais simples. Nem sempre é mais necessário sair de casa para pedir algo, por exemplo, um táxi, comida e até mesmo fazer compras no mercado, e tudo isso por conta de softwares que foram desenvolvidos para atender essas necessidades.

A construção desses softwares pode parecer simples, segundo Delmaro et al. (2013) essa construção pode se tornar muito complexa, de acordo com as características do sistema a ser criado. O software passa por uma série de etapas durante o seu desenvolvimento até chegarao estágio final (utilização pelo cliente/usuário). Cada uma dessas etapas é de extrema importância para não se ter um prejuízo durante o processo, e como afirma Sisson (2007 *apud* PRESSMAN, 1995) as três fases que sempre se encontram em um projeto são: definição, desenvolvimento e manutenção.

Podemos dizer que a definição do software é a parte que se deve perguntar do porque aquele sistema deverá existir, quais necessidades ele vai buscar atender e qual será o caminho para chegarmos a essa solução. Se temos isso bem concreto na acepção do conteúdo, minimizamos a porcentagem de riscos que poderia ocorrer na etapa de desenvolvimento. O desenvolvimento vai tratar totalmente da construção do software, toda a parte lógica e as regras de negócio, para que no fim do processo a única preocupação seja a manutenção do sistema.

Existem alguns modelos de processos que podem ser escolhidos dependendo do software, podemos citar entre eles, o modelo cascata se quisermos seguir um método sequencial de construção, ou o modelo espiral para uma melhor análise dos riscos durante o processo.

Também existem normas que surgiram pela necessidade de regulamentar algumas obrigações contratuais entre fornecedores e os compradores, para que houvesse garantia na manutenção sem deixar de lado a qualidade do produto (VASCONCELOS et al., 2006). Ressaltando que:

> a ISO tem trabalhado na definição de várias normas que podem ser utilizadas como guias e padrões para diversas áreas de atuação dentro do contexto da ciência da computação. Dentre esses, vale ressaltar a importância da norma ISO9000-3, que estabelece um guia para facilitar a aplicação da ISO9001 para desenvolvimento, suporte e manutenção de software (VASCONCELOS et al., 2006).

Agora que entendemos melhor como funciona o desenvolvimento de aplicações, podemos partir para entender como que durante todo esse processo garantimos que o software esteja apto para o uso.

2.2 QUALIDADE DE SOFTWARE

Quando adquirimos um produto novo, esperamos que ele cumpra o propósito para qual foi construído. Com o software não é diferente, a área de qualidade de software propõe garantir que tudo esteja dentro dos requisitos esperados.

Podemos ver melhor com a seguinte definição:

> O planejamento e o gerenciamento da qualidade têm representado um papel cada dia mais forte no contexto do desenvolvimento de software. Desde o início de um projeto, a qualidade deve ser vista como um fator crítico para o sucesso do software e deve ser considerada no planejamento e gerenciamento do mesmo (VASCONCELOS et al.,2006).

Não devemos apenas nos preocupar com o software apenas no final quando rodarmos a aplicação como um todo, mas sim desde a fase inicial por mais mínima que ela seja. A qualidade não está apenas no funcionamento correto da ferramenta, mas também desde a sua documentação. A documentação é considerada por alguns engenheiros uma tarefa bastante burocrática, mas de extrema relevância (ROCHA et al., 2001).

Quanto mais detalhada for a documentação do sistema, mais fácil será a manutenção do mesmo. Um código limpo, bem estruturado e com nomes de métodos bem definidos, também traz consigo qualidade, pois sempre que surgir um requisito novo, por exemplo, de um cliente, será muito mais simples fazer a implementação da nova funcionalidade.

A maneira de assegurar que há qualidade no sistema, é realizar testes. Um analista de garantia de qualidade (QA) é responsável por algumas atribuições nessa área, para o QA é atribuído a tarefa de criação de um plano de teste, plano esse que detalha cenários que ocasionalmente acontecem no dia a dia.

O detalhamento desses cenários fica em um plano de teste, que especifica as ferramentas, o objetivo do plano de teste, qual o ambiente precisamos para fazer os testes, determinada versão do software, entre outros. Todas as informações que ele precisa saber para ter familiaridade com o sistema, é justamente na documentação que é desenvolvida, pois ela deve retratar fielmente o código. Quanto mais documentado for a aplicação, melhor vai ser a execução dos testes.

Digamos que um desses cenários de testes é preencher determinado campo de um formulário, o QA vai detalhar como fazer o teste. O primeiro caso é tentar enviar esse formulário vazio, o retorno que devemos receber é justamente o qual está especificado no plano, caso o retorno seja diferente devemos relatar o bug. De acordo com Galvani, bug é o oposto do que era esperado, coisas opostas e incompatíveis (2019 *apud* KOSCIANSKI; SOARES, 2007).

Relatar um bug não é apenas escrever de qualquer modo o bug encontrado.

> A maneira em que um bug é relatado é de extrema importância para o desenvolvedor designado a corrigi-lo, pois ele possui um grande impacto sobre a facilidade com

> que o mesmo entenderá o problema para posteriormente resolvê-lo (GALVANI, 2019).

E é por causa dessas “pequenas grandes coisas” que a pessoa que irá realizar o plano, tem que ser capacitada para executar a função de testador. Então não basta apenas saber testar, é necessário documentar de maneira adequada que fique entendível para os desenvolvedores.

2.3 TESTES

Visto o processo de software e como se dá o seu desenvolvimento conseguimos entender melhor o seu ciclo e como a qualidade de software se encaixa nesse meio. A maneira como avaliamos a nossa aplicação, pode ser mais custosa dependendo do tipo de teste que se queira realizar, e conseguimos notar as diferenças e vantagens entre eles.

A pirâmide de teste representa alguns testes, tanto quanto seria o tempo de execução e o valor. Se analisarmos a pirâmide, notamos que os testes unitários são os mais rápidos e menos custosos se compararmos com os testes de ponta a ponta. A diferença é que os testes unitários vão verificar a menor unidade do código e isso ocasiona um menor tempo e custo, por isso, esses testes estão na base da pirâmide. Os testes de integração se encontram no centro da pirâmide, isso significa que o tempo e custo são razoáveis, já que esses testes avaliam as interfaces de software e testa algumas unidades funcionando juntos. E por fim, no topo da pirâmide se encontram os testes de ponta a ponta, que simulam um ambiente real, esse teste vai do início até o final da aplicação.

Os testes de ponta a ponta são de fato mais trabalhosos e necessitam de um valor mais alto de capital. Porém, traz uma confiabilidade maior em relação ao sistema, mas isso não significa que o software esteja livre de falhas. É quase impossível testar todas as possibilidades de formas e alternativas de entrada de dados, bem como testar as diversas possibilidades e condições criadas pela lógica do programador (RIOS; FILHO, 2013).

Alguns testadores, verificam o sistema como se fosse o usuário final, mexendo nas funcionalidades sem ter qualquer conhecimento do código e lógica do programa, esse teste se denomina caixa preta (*black box*). Ao contrário do teste de caixa preta, temos o teste que visa avaliar o código, a lógica e outros componentes, esse chama-se caixa branca (*White box*).

Imagine que uma pessoa ficou responsável por fazer todos os testes de ponta a ponta e de caixa preta de um grande sistema, quanto tempo seria que essa pessoa iria precisar. E a

cada requisito novo implementado iria realizar novamente todo o procedimento, e como consequência teríamos um teste exaustivo e falta de produtividade.

Hoje encontramos testes que são totalmente automatizados, e um amontoado de testes que levaria horas, poderiam ser feitos em minutos. Essa é a redução de tempo que os testes automatizados nos garantem.

2.3.1 TESTES AUTOMATIZADOS

Nas décadas de 1960 e 1970, houve um esforço muito maior em relação a codificação e nos testes unitários (RIOS; FILHO, 2013). Isso porque um dos principais motivos era a escassez de ferramentas de automação da época.

Hoje para cada tipo de aplicação, seja ela web, um programa de computador ou uma *Application Programming Interface* (API), se encontra com facilidade uma ferramenta para realizar os testes automatizados. Vamos utilizar o caso da API como exemplo.

Quando solicitamos alguma informação, estamos mandando uma requisição com o objetivo de obter uma resposta, e estas são requisições *Hypertext Transfer Protocol* (HTTP),

que nada mais é que um protocolo de comunicação. Suponha-se que o plano de teste desenvolvido pelo QA tenha 30 casos de testes para diferentes rotas e *endpoint* da API, e toda vez que esses *endpoint* forem atualizados, terá que repetir os casos de testes existentes mais os novos para a funcionalidade implementada, e todo esse processo ocorrendo de maneira manual.

Chega um ponto de que deixa de ser totalmente inviável. E por isso que existem algumas ferramentas que auxiliam a criar essas requisições e automatizá-las conforme a necessidade dos testes. Mas dependendo do tipo que requisição que se deseja automatizar, é necessário realizar criação de scripts com alguma linguagem de programação, no caso do *postman* a linguagem utilizada é JavaScript para a criação das *pré-request*, que é uma solicitação que irá ser executada antes da requisição em si. podendo criar um usuário e passar ele como parâmetro no body da nossa requisição, neste caso não precisaria de uma nova requisição a parte, salvo o caso de o teste realizando seja E2E, pois esses precisam testar o fluxo da aplicação do começo ao fim.

Ao enviar a requisição e ter o retorno das informações, fazemos o teste validando se a informação retornada é de acordo com a esperada. Com isso, se observa que temos uma melhor tratativa das informações e validações das respostas. Todas essas requisições ficam guardadas em uma *collection* e sempre que precisar testar, basta executar a coleção.

Mas nem sempre esse segmento de melhoria que é ter testes automatizados como uma ferramenta poderosa, é negligenciado por algumas empresas.

2.3.2 PORCENTAGEM DE TESTES AUTOMATIZADOS DENTRO DAS EMPRESAS

As empresas de software estão se expandindo cada vez, buscando o acolhimento que o mercado de tecnologia tem. Mas mesmo toda essa vontade de crescer não é o suficiente.

Progressivamente a tecnologia da informação surge com novas ideias, ferramentas, frameworks e outros, as empresas que querem se destacar precisam ficar atentas ao surgimento dessas novidades e se adaptar a elas, caso contrário acaba ficando para trás. Testes automatizados hoje já é uma realidade que veio se consolidando durante os anos, e mesmo assim algumas empresas não trazem esse otimizador de tempo para dentro delas.

Uma pesquisa realizada sobre testes automatizados mostrou que, 53,3% das empresas pesquisadas, os testes são executados somente de forma manual, e os outros 43,7% utilizam as duas técnicas (FONSECA, 2018). Isso nos traz uma visão de que algumas empresas não conseguem se adequar ao ritmo do mercado, ou isso é consequência da carência de profissionais da área.

Visto até aqui que, as técnicas e ferramentas para garantir a qualidade de software não apenas no ciclo final de construção da aplicação, mas desde o seu início, os tópicos seguintes irão tratar sobre a pesquisa de campo que irá ser realizada.

3 METODOLOGIA

Para contextualizar e fundamentar o presente estudo, foi feito algumas análises para identificar quais pesquisas estavam aptas para a aplicação. Sendo a plataforma utilizada para as pesquisas, o google acadêmico, e o site integrado de busca da USP.

A técnica de pesquisa utilizada é a de observação direta extensiva, onde o formulário para realização da pesquisa é através da ferramenta de Formulários Google, a qual a mesma é online e gratuita, e de fácil compreensão. O objetivo geral é expor o processo de desenvolvimento de software e a validação dessas aplicações nas empresas através dos testes automatizados, e essa avaliação foi feita através de um formulário para obter um levantamento de dados sobre os testes automatizados dentro da empresa onde atua/atuava o respondente da pesquisa.

Como a linha de pesquisa deste artigo é a qualidade de software, os respondentes são da área de tecnologia da informação. E diante das perguntas propostas, foi possível:

- Determinar quais são os principais testes que são utilizados em projetos;

• Especificar se a empresa faz uso de testes automatizados, quais ferramentas ela utiliza;

• Estimar o balanceamento de QA para cada projeto.

Ao apresentar esses dados obtidos, para melhor visualização será apresentado em gráficos as porcentagens alcançadas, e tabelas para as respostas obtidas. As ferramentas utilizadas foram ocanva, excel e o próprio formulário google.

O questionário passou-se por uma análise necessária para identificar se estava apto para ser implementado, o resultado foi aprovado sendo seu número do CCAE – 65258522.1.0000.5184.

4 RESULTADO E DISCUSSÃO

A pesquisa realizada obteve o número de 24 respondentes, superando 60% a mais do planejado. O gráfico abaixo mostra a área de atuação dos participantes e seus respectivos números:

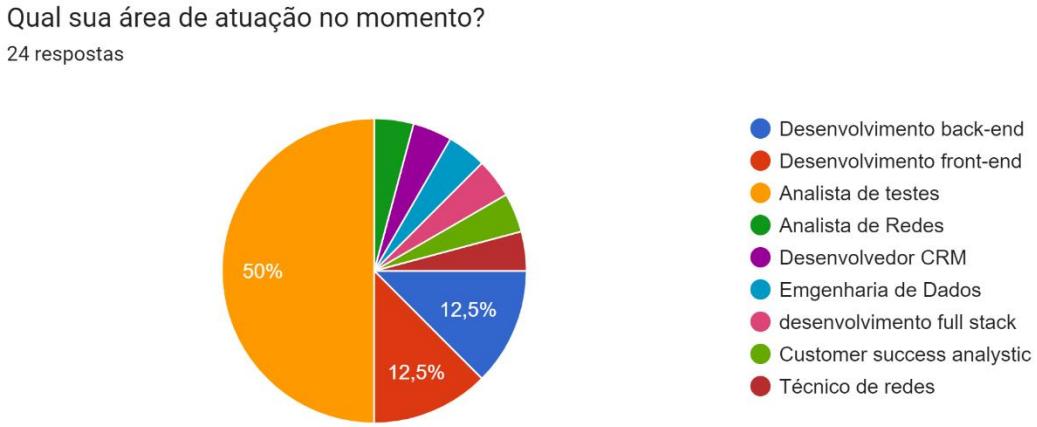

Figura 1. Relação do participante e sua área de atuação

No gráfico é possível visualizar que a área com grande destaque é analista de testes, seguido de desenvolvimento back-end e front-end. Asim, podemos supor que a carreira na área de qualidade está crescendo para o mesmo patamar e destaque que a área de desenvolvimento possui.

A área de tecnologia vem crescendo bastante, como explicado no ínicio deste artigo. Podemos perceber isso com a figura abaixo:

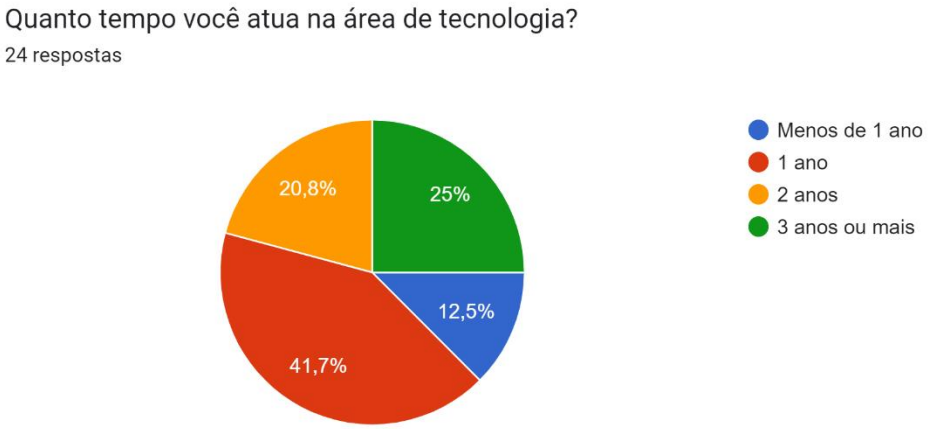

Figura 2. Atuação na área de tecnologia

Nota-se que 41% das pessoas responderam, então recentemente neste mercado. Enfatizando a ideia de que esta área vem tendo um destaque e ganhando mais profissionais, mesmo que a carreira desses indivíduos ainda sejam curta ou longa, nenhum deles declarou não conhecer sobre os testes de softwares, como mostra a figura a seguir:

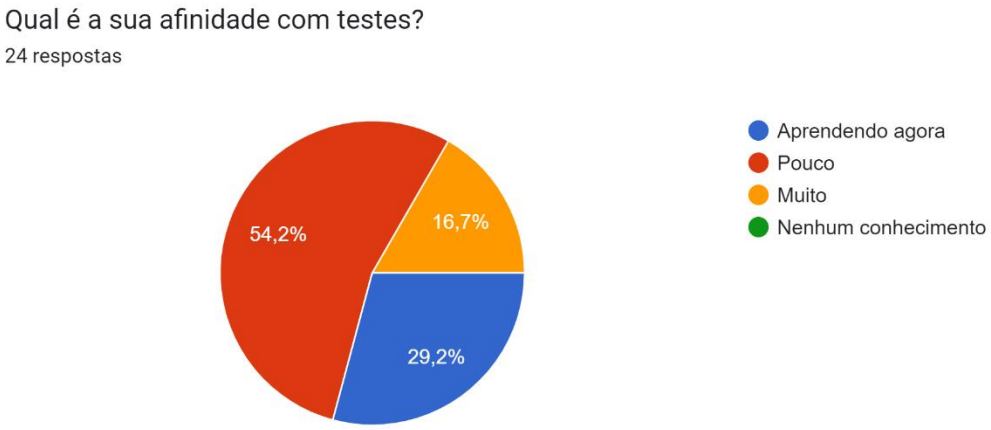

Figura 3. Conhecimento sobre testes

Atualmente, podemos considerar isto como um grande avanço, visto que, o conhecimento sobre testes entre os profissionais de tecnologia era praticamente escasso. Passando a ideia de que apenas quem atuava na área de qualidade deveria ter este conhecimento.

Apesar do grande impacto que um produto de baixa qualidade desenvolvido para um cliente, pode causar para uma empresa, como uma má reputação ou uma quebra de contrato, algumas delas ainda não tem um profissional da qualidade atuando nelas:

Figura 4. QAs nas empresas

Cada empresa lida com uma maneira de trabalho diferente, sendo assim, o balanceamento de pessoas em um projeto de uma empresa X pode ser totalmente diferente em relação a empresa Y, a figura abaixo mostra este balanceamento nas empresas que os pesquisados atua/atuavam:

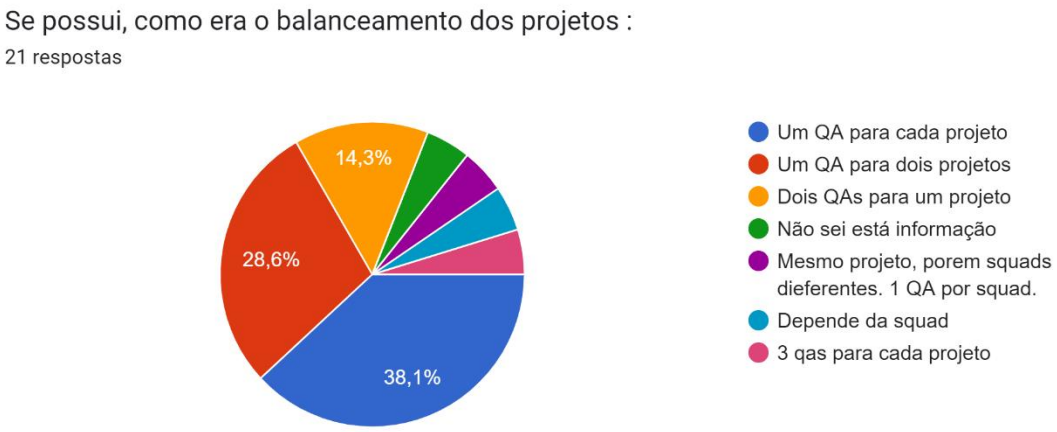

Figura 5. Balanceamento de QAs

38,1% afirmaram que havia um qa para cada projeto, o quê dependendo do projeto, pode se tornar muito mais prático que o foco do profissional seja voltado totalmente para o trabalho que ele foi exposto. Se pensarmos no lado da complexidade, é plausível ter dois qas para um projeto, como afirma 14,3% dos participantes.

Apesar de existir vários tipos de testes, não significa que todos eles serão utilizados em um único projeto, mas sim aqueles que se compactuam com o projeto. Podemos ver isso na figura a seguir, onde apenas uma pessoa afirmou que na empresa onde ela trabalha/trabalhou se utilizava o teste de fumaça:

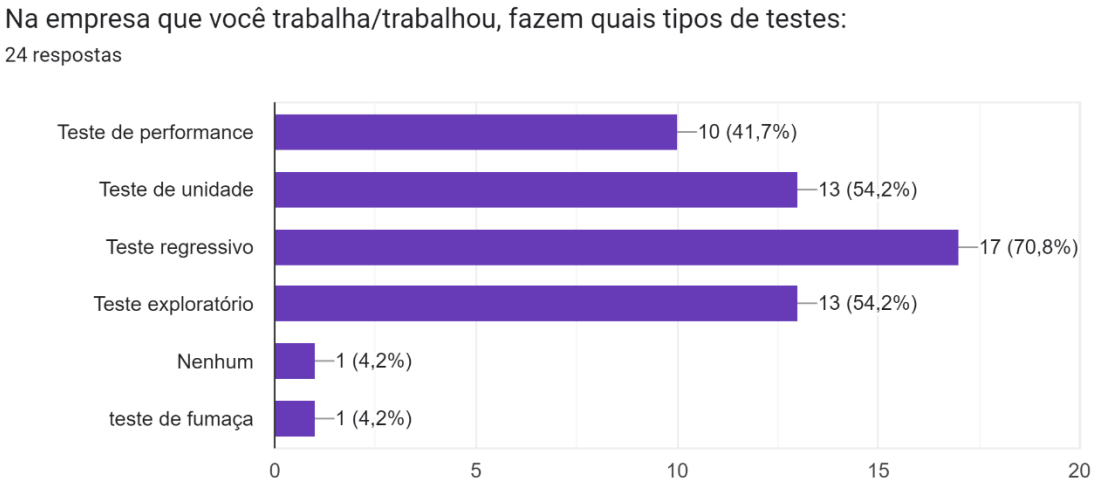

Figura 6. Tipos de testes nas empresas

Nota-se que o teste regressivo é o mais utilizado, visando que, este tipo de teste tem como objetivo verificar toda a aplicação do início ao fim sempre que for implementado uma nova funcionalidade. Mas na medida que o projeto for crescendo isso passa a ser totalmente inviável, se tornando exaustivo, como explicado anteriormente no tópico 2.3, e assim os testes automatizados passam a ser algo necessário.

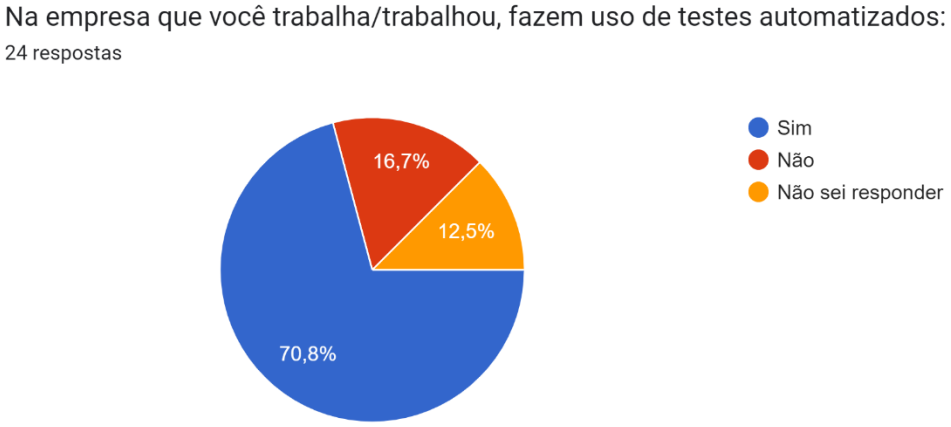

Figura 7. Testes automatizados

Os testes automatizados já estão presentes em 70,8% nas empresas dos entrevistados, mas ainda é preocupante o fato de 16,7% ainda não terem adotado a automação de testes.

Isso pode gerar diversas divergências em relação ao tempo, já que os testes mais complexos levariam até mesmo o triplo do tempo de um teste automatizado. Já que com um simples click no botão, realizaria todo um teste de regressão.

Essa automação não é feita de qualquer forma. Para isso existe ferramentas voltadas para este assunto, e a figura abaixo mostra as mais utilizadas:

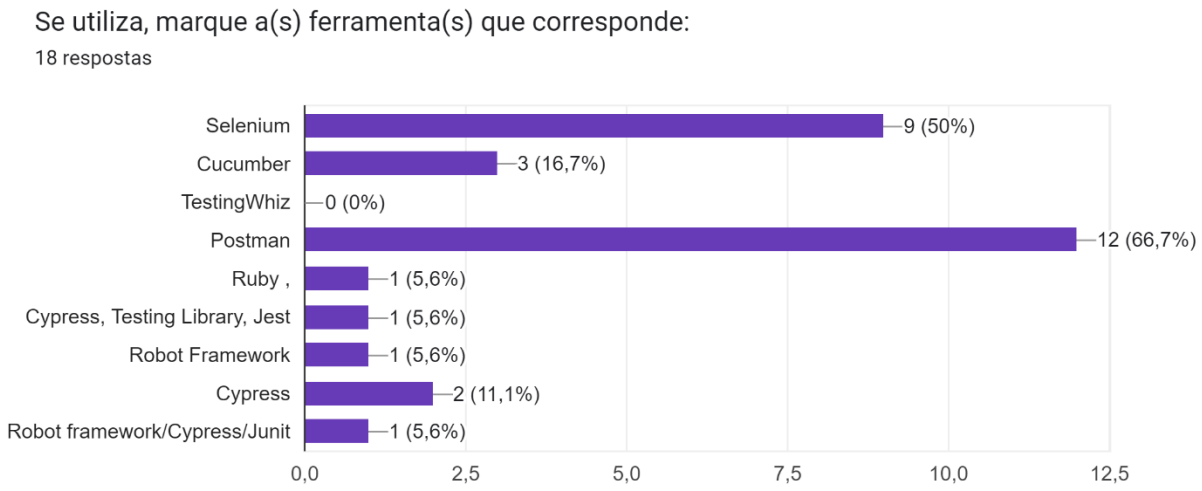

Figura 8. Ferramentas de automação

Tendo em grande destaque o postman, já que é uma ferramenta simples e de fácil manuseio, seguido por uma das mais utilizadas no mercado, a Selenium.

E para finalizar a entrevista, ao fim foi feita uma pergunta (de caráter não obrigatório) sobre qual seria a opinião individual sobre os testes automatizados. As respostas serão expostas nas imagens a seguir:

É uma boa prática

Imagem 1. Opinião A

Acho interessante e acredito que servem para complementar os testes manuais.

Imagem 2. Opinião B

Facilitam bastante a vida nos projetos porém não substituem o analista de QA

Imagem 3. Opinião C

Na minha opinião os testes automatizados vieram para somar e ajudar os profissionais de qualidade/testes. Se for bem aplicado pode auxiliar muito em alguns testes, como é o caso de testes de regressão. Mas não basta somente conhecer uma linguagem e um framework e começar a automatizar, precisamos entender também o processo se quisermos aplicar a automação de maneira eficaz.

Imagem 4. Opinião D

Uma ferramenta importante para a otimização de um software de maneira ágil e precisa.

Imagem 5. Opinião E

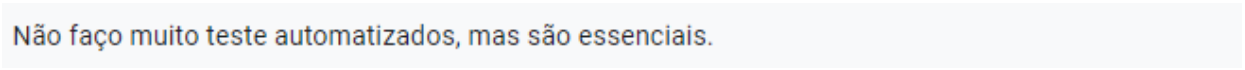

Imagem 6. Opinião F

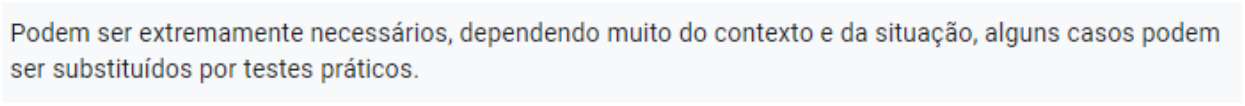

Imagem 7. Opinião G

Com isso concluímos a entrevista e obtemos dados interessantes de um modo geral. Podemos afirmar que os testes automatizados já fazem parte de uma grande parcela das empresas, e mesmo que tenha essa automação ela não substitui o analista (Opinião C), pois ele que escreve e expõe todo o processo dos testes.

5 CONSIDERAÇÕES FINAIS

Em suma, diante dos fatos apresentados no tópico anterior, podemos responder a problemática da questão com um “alto”, os testes automatizado tem um alto nível de relevância no desenvolvimento de aplicações, não apenas pelo fator tempo, mas também por ser indispensável com a sua precisão para encontrar bugs e a sua agilidade na execução dos testes.

Uma entrevista onde todos os participantes disseram ao menos ter um nível mínimo de conhecimento sobre testes, já nos mostra o quão significativo é este assunto. O presente trabalho atingiu um dos objetivos que era a apresentação de dados sobre a automação de testes, onde esse foi um dos problemas para a construção do mesmo. A escassez de dados sobre automação para fundamentar a teoria.

Que esta pesquisa ou trabalho como um todo, sirva para futuros pesquisadores que façam as mesmas perguntas. É possível que este trabalho sofra futuras atualizações, à medida que o mercado de qualidade de software for se expandindo.

Podemos concluir,que o objetivo geral e específico foram alcançados com êxito. E que podemos prever um futuro ainda mais promissor para a área de qualidade.

REFERÊNCIAS

FILHO, E. R. T. M. **TESTE DE SOFTWARE**. 3. ed. RIO DE JANEIRO: ATLAS BOOKS, 2013. p. 9-10.

FONSECA, M. D. G. UM LEVANTAMENTO SOBRE A UTILIZAÇÃO DE TESTES DE SOFTWARE EM EMPRESAS DE MICRO E PEQUENO PORTE NO CENTRO-OESTE MINEIRO. **UM LEVANTAMENTO SOBRE A UTILIZAÇÃO DE TESTES DE SOFTWARE EM EMPRESAS DE MICRO E PEQUENO PORTE NO CENTRO-OESTE MINEIRO**, INSTITUTO FEDERAL DE MINAS GERAIS – Campus Formiga, v. 1, n. 1, p. 31-32, nov./2018. Disponível em: https://www.formiga.ifmg.edu.br/documents/2018/Biblioteca/TCCs_e_Artigos/Marcela_das-_Gracas_Fonseca.pdf. Acesso em: 6 out. 2022.

VASCONCELOS, A. M. L. D. *et al*. INTRODUÇÃO À ENGENHARIA DE SOFTWARE E À QUALIDADE DE SOFTWARE . **INTRODUÇÃOÀ ENGENHARIA DE SOFTWARE E A QUALIDADE DE SOFTWARE,** Lavras - MG , v. 1, n. 1, p. 88-89, ago./2006. Disponível em: http://nti.facape.br/jocelio/es/apostilas/Mod.01.MPS_Engenharia&QualidadeSoftware_V.28.09.06.pdf. Acesso em: 3 out. 2022.

SISSON, Martim Chitto. Avaliação e Melhorias no Processo de Construção de Software. **Avaliação e Melhorias no Processo de Construção de Software**, Florianópolis, v. 1, n. 1, p. 21-21, nov./2007. Disponível em: https://repositorio.ufsc.br/bitstream/handle/123456789/184376/monografia_martim_sisson.pdf?sequence=-1&isAllowed=y. Acesso em: 5 out. 2022.

GALVANI, A. L. F. IMPLEMENTAÇÃO DE UM PROTÓTIPO DE ONTOLOGIA PARA RELATOS DE BUG. **IMPLEMENTAÇÃO DE UM PROTÓTIPO DE ONTOLOGIA PARA RELATOS DE BUG**, MARINGÁ-PR, v. 1, n. 1, p. 7-7, set./2019. Disponível em: https://rdu.unicesumar.edu.br/bitstream/123456789/5201/1/TRABALHO%20DE%20CONCLUSÃO%20DE%20CURSO.pdf. Acesso em: 6 out. 202