IDENTIFICANDO FATORES DE SUCESSO NA APLICAÇÃO DE TESTES AUTOMATIZADOS EM UM SISTEMA LEGADO

Lucas de Azevedo Nonato

Alexandra Aparecida de Souza

Instituto Federal de Educação, Ciência e Tecnologia de São Paulo - Câmpus Guarulhos

Resumo

O artigo tem como objetivo analisar os fatores de sucesso que influenciaram na qualidade do software após a aplicação de testes automatizados e como as metodologias de testes podem influenciar nos resultados. Foi utilizado um projeto real em uma empresa de desenvolvimento de software da cidade de São Paulo que teve ganhos de qualidade e produtividade após a aplicação de testes automatizados em seu processo de desenvolvimento. Os resultados do projeto foram confrontados com números de manutenção, inovação e as boas práticas propostas em embasamento teórico para entender onde a literatura se confirma na prática.

Palavras-chave: Automação de testes, Metodologia de testes, Refatoração de código.

1. Introdução

Muitas empresas utilizam sistemas legados em sua operação pois estes sistemas atendem sua regra de negócio. No entanto, estes sistemas legados podem possuir um alto de esforço de evolução ou adequação de legislações vigentes, devido a sua estrutura ou tecnologias antigas. (KHODABANDEHLOO et al., 2021). Além disto, há diversos fatores que podem aumentar o custo de manutenção ou expansão de um sistema, mas dentre eles, destacam-se problemas relacionados à arquitetura, documentação, projeto ou granularidade dos componentes. (MAJTHOUB; QUTQUT; ODEH, 2018). Tendo todas estas dificuldades o processo de testes se torna essencial para assegurar a qualidade do sistema legado.

No processo de testes é possível obter informações sobre a diferença entre o comportamento real e o esperado do software quando ele é entregue. Dependendo do tamanho e complexidade do software os custos envolvidos no desenvolvimento podem variar entre 30% a 80%, no entanto este custo pode ser diminuído por meio da

automação de testes, onde a execução dos testes é realizada por um software e não por um ser humano (WIKLUND et al., 2017).

A automação de testes é utilizada para aprimorar a eficiência do processo de desenvolvimento, reduzir risco de falhas operacionais, tornar os testes mais repetitivos e possibilita a alocação da equipe em outras demandas (WIKLUND et al., 2017).

Com o objetivo de melhorar a qualidade de um sistema legado de um software de gestão de planos de saúde, uma empresa de desenvolvimento de software de São Paulo optou pelo desenvolvimento e aplicação de testes automatizados em seu processo de qualidade.

O artigo usará o projeto como estudo de caso para entender como a aplicação de testes automatizados e metodologias de teste propostas pela literatura afetaram o sistema e o processo de desenvolvimento. Foi realizada uma revisão da literatura com o objetivo de identificar boas práticas e fatores de sucesso para o processo de desenvolvimento e aplicação de testes automatizados. A seção 2 aborda o embasamento teórico do estudo, explorando projetos de aplicação de testes automatizados em sistemas legados. Para compreender a aplicação das práticas propostas pela literatura, a seção 3 apresenta detalhes do projeto, por fim a seção 4 mostrará os dados documentais analisados junto as conclusões do estudo.

2. Materiais e Métodos

No processo de desenvolvimento de software a etapa de testes evoluiu como uma das atividades principais para garantir a sua qualidade. O processo de testes deve ser feito desde a parte inicial do desenvolvimento contemplando cada pedaço do levantamento de requisitos e se complementando com a cada nova funcionalidade disponibilizada para evolução do produto. Na medida que o software evolui com novas implementações, as antigas também devem continuar a serem testadas pois fazem parte do escopo de requisito, tornando assim um número exponencial de casos de testes que devem ser realizados a cada nova liberação. (KUMAR; KUMAR, 2015) Por esses motivos o teste manual acaba se tornando inviável e a automação de testes acaba surgindo para suprir essa necessidade de maximizar a cobertura e garantir a qualidade de um software.

Sistemas legados possuem um complicador de possuírem poucos ou nenhuns casos de testes e milhares de linhas de código para serem cobertas. O autor

(SHIHAB et al., 2010) avalia algumas abordagens para iniciar o processo de automação, uma delas é isolar as funções e escrever casos de testes específicos para estes, no entanto, para descobrir por quais funções começar o autor sugere a análise do histórico de backlog, para evitar o gasto de esforço desnecessário inicialmente, com funções pouco utilizadas e que agregam pouco valor em qualidade neste momento.

Sem determinar uma ordem, os seguintes fatores devem ser considerados para definir por onde começar:

- Risco: Rotinas críticas do sistema;
- Modificações: Fontes e funções modificados com mais frequência.;
- Correções: Rotinas que possuem alta incidência de bugs.;
- Tamanho: As maiores funções em linhas totais de código, desconsiderando comenta- rios e linhas em branco. (SHIHAB et al., 2010)

Uma das maneiras de desenvolver os scripts é em formato de teste caixa preta, isolando as suas funções e comparando os resultados de saída, para estes testes os dados podem ser gerados no próprio script, modelos pré/pós para comparação ou análise de gravações no banco de e dados. (WIECZOREK; STEFANESCU; SCHIEFERDECKER, 2008)

Além dos scripts pontuais de funções também é necessário realizar os testes mais complexos, se tratando de sistemas ERP se faz necessário também que tenham testes que se aproximam mais da realidade de uso do sistema, testes mais complexos, sendo assim se torna necessário um ambiente com uma base de dados estática para a execução destes testes. Os casos de teste possuem os dados específicos pré-determinados que irão utilizar e talvez manipular. Em casos de sistemas grandes o controle de quais dados estão sendo utilizados para determinados testes acaba se tornando inviável, por isso a tarefa de retornar a base de testes para o seu status original acaba se tornando importante. (WIECZOREK; STEFANESCU; SCHIEFERDECKER, 2008)

Segundo Wieczorek, Stefanescu e Schieferdecker (2008) a execução automática de casos de testes consiste na execução de quatro passos:

- Configuração: Define os dados e parâmetros que o teste irá utilizar.;
- Execução: Executa os scripts em sequência determinada.;

- Análise: Verifica os resultados do teste.;
- Retorno: Retorna a base de testes para seu status original para que os próximos testes possam ser executados.

Além do ganho de estabilidade e qualidade do sistema, com a automação de testes é possível realizar a refatoração de partes do código do sistema, como análise de código redundante, simplificação de código e reusabilidade de classes e métodos. (EBANHESATEN; FüßL; STREITFERDT, 2013) Por mais que exista mudança no código, é primordial que o código novo possua o mesmo funcionamento do código antigo, pois os seus requisitos não podem ser alterados para atender a necessidade do script de automação. (KHODABAN- DEHLOO et al., 2021) Um dos grandes desafios da refatoração de código é que o usuário não seja impactado com nenhuma mudança no sistema causada por estas alterações, sendo assim, o autor (EBANHESATEN; FüßL; STREITFERDT, 2013) recomenda que a automação seja feita antes da refatoração e depois, pois os resultados devem ser iguais.

3. Resultados e Discussão

Visando confrontar as práticas propostas pela literatura bibliográfica com uma aplicação real, foi utilizado um projeto de aplicação de testes automatizados em um sistema legado de uma empresa de desenvolvimento de software da cidade de São Paulo.

O projeto consistiu na elaboração de scripts de testes automatizados a partir de um software de gestão de operadoras de planos de saúde. O produto atual da empresa consiste em uma aplicação client-server e multiplataforma. A solução pode ser instalada em um ambiente do cliente (On-Premise), em uma aplicação hospedada na nuvem própria fornecida pela empresa desenvolvedora do software (SaaS - Software as a Service) ou em uma infraestrutura de fornecedores terceiros de serviços para nuvem (IaaS - Infrastrutucture as a service). Independentemente da forma que foi instalada, para o seu funcionamento, o sistema utiliza um repositório com todos os fontes da aplicação compilados e criptografados em uma extensão própria da empresa, mas independentemente do formato de instalação contratada do sistema, o compromisso de manter o repositório de fontes atualizado fica com o cliente.

O sistema conta com 3 possibilidades de atualização de fontes nesse repositório, pacotes pontuais que podem ser manutenções para correção de bugs ou pacotes contendo alterações de legislações para manter o sistema de acordo com os órgãos reguladores, pacotes de expedição contínua, que são um compilado de alterações de bugs, legislações ou pequenas

melhorias, e pacotes de virada de versão. A cada expedição oficial de qualquer uma das 3 expedições são realizados testes unitários e testes funcionais manuais.

O primeiro passo do projeto foi realizar o levantamento de rotinas críticas do sistema, e quais funcionalidades específicas possuíam mais demanda de correção de bugs, para isto foi utilizado o histórico de backlog de manutenção do sistema. Possuindo as 3 funcionalidades específicas com mais demandas, foi solicitado ao time de qualidade, responsável por realizar os testes funcionais manualmente, a elaboração de casos de testes, detalhando parametrizações, níveis e funcionalidades por eles acessadas no momento do teste.

Tendo os casos de teste escritos, os desenvolvedores começaram a analisar a rotina em formato de teste caixa preta. A elaboração dos scripts consiste na montagem de um cenário inicial com a inserção de dados de entrada em uma função e definidos os valores de saída esperado, posteriormente na execução deste script os resultados devem coincidir com os dados informados previamente nos valores de saída.

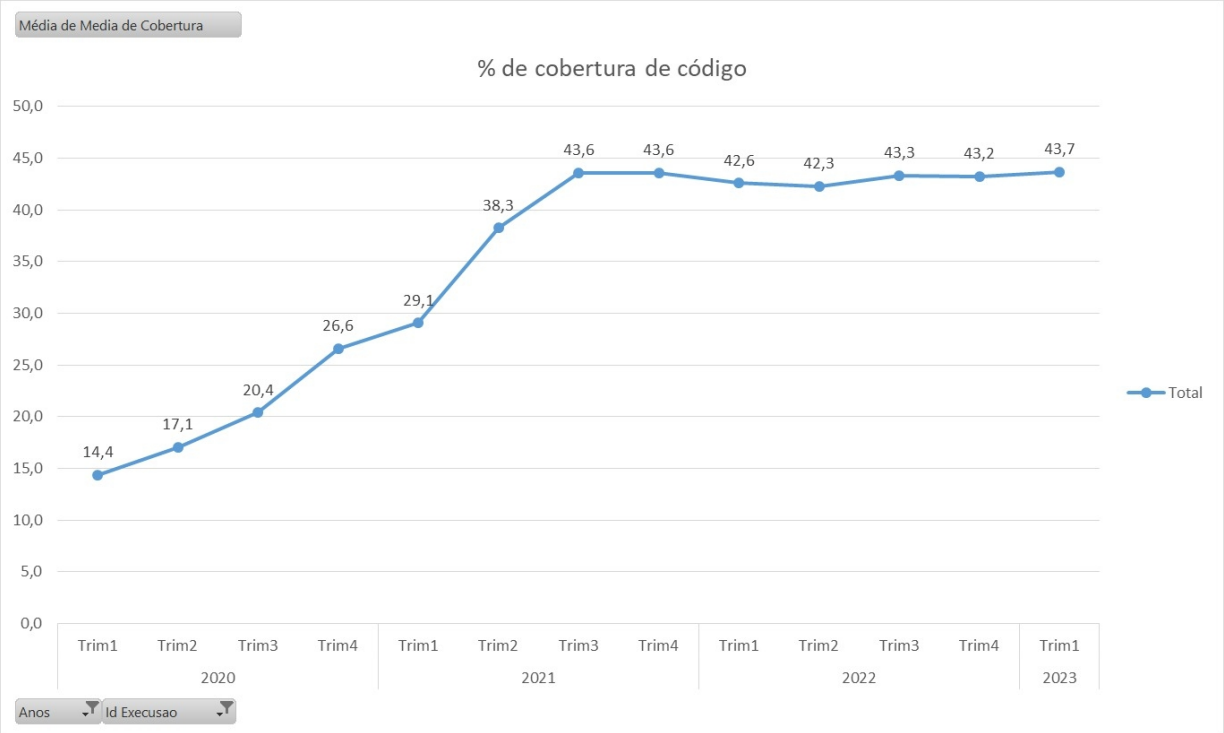
Nem todas as partes do código podem ser automatizadas, como por exemplo, a avaliação de comportamentos da interface visual. O foco do projeto foi em automatizar processos críticos com códigos monolíticos desenvolvidos em arquitetura estruturada e que podem trazer um maior retorno no longo prazo diminuindo tarefas consideradas repetitivas no processo de testes manuais.

Após o desenvolvimento dos scripts, foi necessária uma forma automática de execução para não ter que contar com a execução manual de cada desenvolvedor, inicialmente a execução automatizada era disparada a cada subida de alteração de código (check-in).

Tendo o cenário de que todas as rotinas críticas mapeadas estavam cobertas por scripts de automação, foi iniciado um processo de análise de cobertura por linhas de código, foi extraído o total de linhas de código executável e após uma execução de todos os scripts foi extraído o número de linhas que os scripts executaram. No gráfico abaixo (Figura 1), na primeira execução em janeiro/2020 pode-se analisar que a automação das rotinas críticas representou um total de apenas 15 porcento das linhas totais do sistema.

As informações do gráfico abaixo (Figura 1) foram coletadas por meio de extrações periódicas realizadas periodicamente nos momentos citados no gráfico.

FIGURA 1 – PERCENTUAL DE COBERTURA POR MÊS/ANO

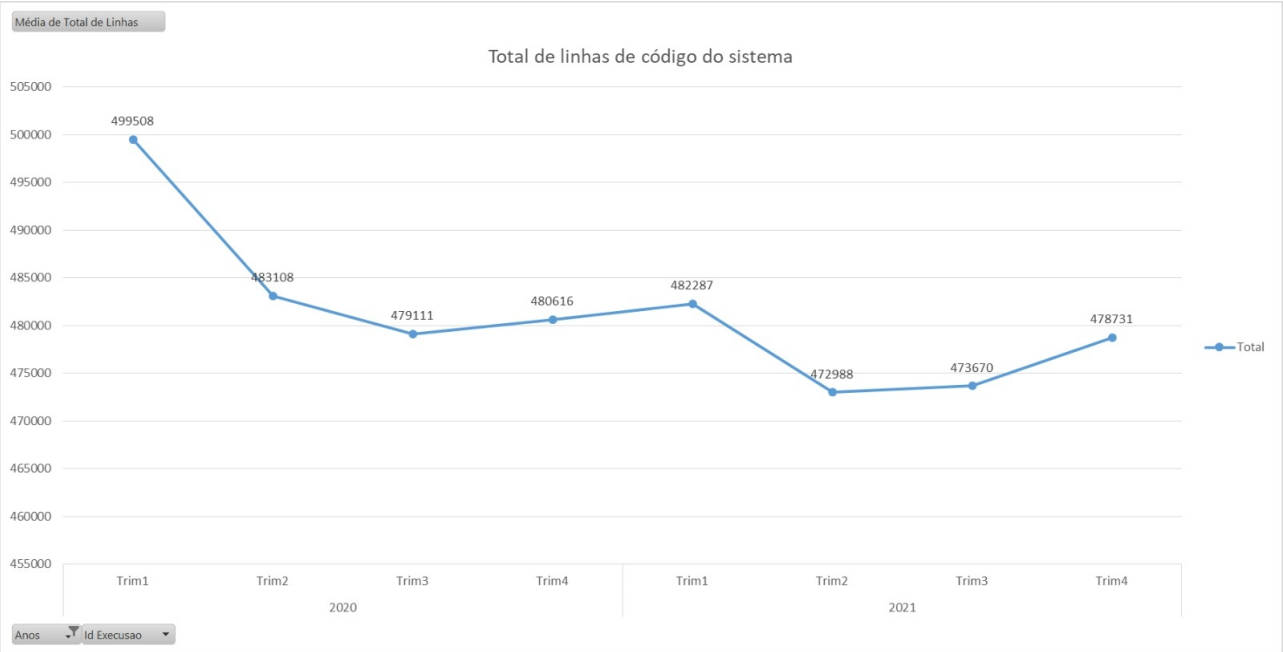


FONTE: ELABORADO PELO AUTOR (2023)

Com a necessidade de realizar o máximo possível de cobertura de linhas e revisitando as fontes do sistema foi possível realizar a refatoração de código, tendo uma diminuição de 25 mil linhas de código após a refatoração de alguns blocos, como pode ser analisado no Gráfico abaixo (Figura 2).

Os dados do gráfico abaixo (Figura 2), foram extraídos em conjunto da Figura 1, pois o percentual de cobertura é calculado a partir do total de linhas do sistema com o total de linhas cobertas.

FIGURA 2 – LINHAS DE CÓDIGO EXECUTÁVEIS POR MÊS/ANO



FONTE: ELABORADO PELO AUTOR (2023)

Com o avanço das automações em larga escala foi encontrado a necessidade de ter um ambiente do sistema com dados fixos para que os scripts possam utilizá-los em determinadas rotinas que necessitam de interação com banco de dados, sendo assim foi formalizado um ambiente do zero onde deveriam ser feitos cadastros apenas para consumo dos scripts de teste e a cada execução de scripts o banco de dados desse ambiente é restaurado para que não quebre outros scripts de testes.

Tendo diversos scripts e com a necessidade de execução continua destes testes se tornou inviável a execução da automação após cada check-in de fonte, sendo assim alterado o processo para uma execução automática uma vez ao dia no período da noite.

Devido a utilização apenas de testes caixa preta, os códigos de interface, integrações e comportamentos visuais acabaram não sendo contabilizados, sendo este um grande fator para diminuição na crescente linha de percentual de cobertura do sistema.

4. Considerações Finais

Para auxiliar o entendimento do processo de desenvolvimento, foi realizada uma análise do histórico de documentos e artefatos produzidos ao longo do projeto.

A equipe de desenvolvimento do produto utilizou como ferramenta de controle de demandas o software JIRA. A partir do tipo de issue foi realizada uma categorização por modalidade, separando entre:

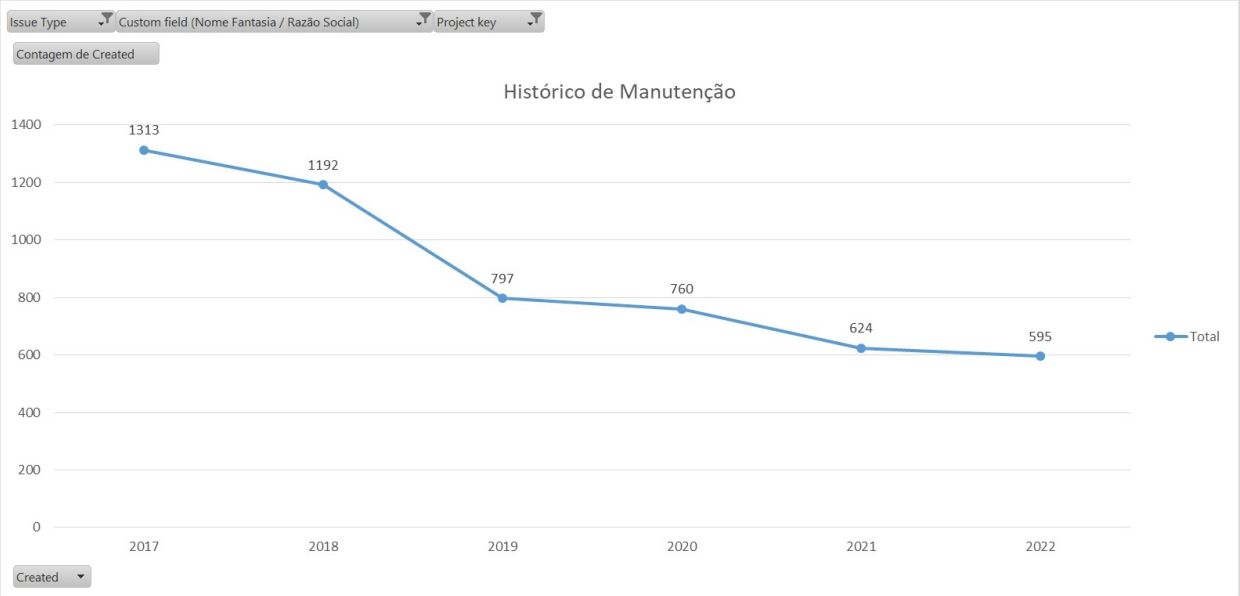
- Manutenção: Sustentação do produto, não conformidade com legislação ou regras de negócio.;
- Inovação: Épicos de desenvolvimento de novas funcionalidades ou incrementação em funcionalidades já existentes.

Pela análise do gráfico de histórico de entradas de manutenções (Figura 3) de uma das squads de desenvolvimento do sistema ERP, pode se dizer que a aplicação de testes automatizados obteve sucesso em qualidade do produto, causando redução de manutenções entrantes.

O gráfico (Figura 3) demonstra o total de bugs abertos por clientes por ano desde o início do projeto no ano de 2017 até o ano 2022. Comparando o gráfico (Figura 3) com o (Figura 1), é notável a queda entre 2017 até 2020, época em que os casos de testes eram focados nas rotinas críticas do sistema, resultando em aproximadamente 15% do total de cobertura, e após isto com os casos de teste focados em subir a cobertura total do sistema, foi demonstrado a estabilização do número, com pequenas variações para baixo.

Os dados do gráfico abaixo (Figura 3) foram extraídos do sistema de controle de demandas utilizado pela equipe, o JIRA. São consideradas como manutenção, tarefas relatadas por clientes que se caracterizem como um erro, não conformidade, falha de regra de negócio ou necessidade de adequação a alguma regra vigente.

FIGURA 3 - PERCENTUAL DE COBERTURA POR MÊS/ANO

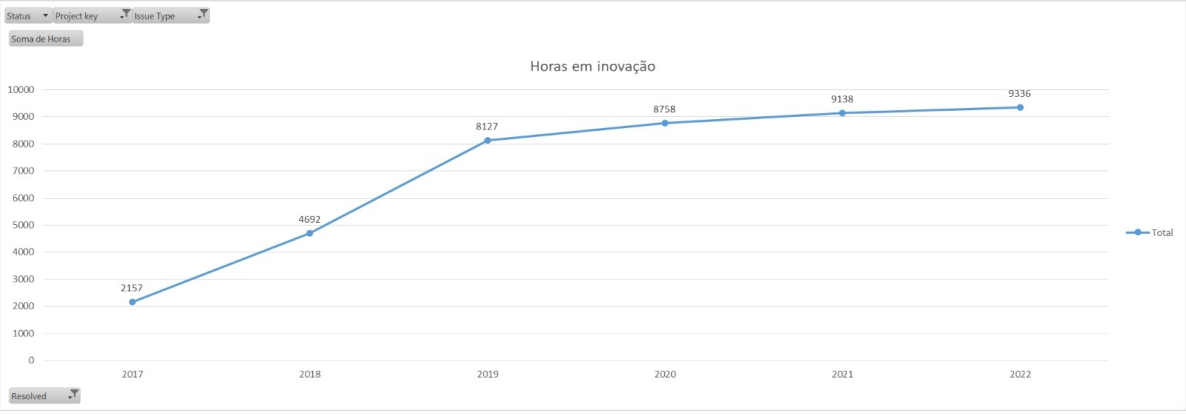


FONTE: ELABORADO PELO AUTOR (2023).

Do ponto de vista do time de desenvolvimento, o projeto também obteve sucesso, analisando o gráfico de horas aplicadas em inovação de produto (Figura 4), pode se dizer que após a aplicação dos testes e com a queda de entradas de manutenção, o tempo gasto em inovações aumentou.

Os dados do gráfico abaixo (Figura 4) foram extraídos do sistema de controle de demandas utilizado pela equipe, o JIRA. São consideradas como inovação, tarefas que se caracterizem como melhorias no produto, tanto como rotinas novas, telas novas, e melhorias em rotinas ou telas existentes que agreguem valor ao produto. Para todas as tarefas, o analista responsável deve apontar as horas gastas naquele dia, na tarefa em questão, o que resulta no gráfico abaixo de inovação por horas trabalhadas.

FIGURA 4 - HORAS GASTAS EM INOVAÇÃO POR ANO



FONTE: ELABORADO PELO AUTOR (2023).

Comparando o processo realizado pela equipe de desenvolvimento com os trabalhos relacionados pela literatura citada na seção 2. Pode se dizer que as estratégias já utilizadas foram aderentes e eficazes, tanto na identificação de como começar, quanto na utilização contínua da metodologia de automação.

Foi realizada uma pesquisa com a equipe do sistema para registrar a percepção de valor que o processo de testes automatizados gera no produto entregue e em suas carreiras. O questionário foi encaminhado aos membros do time de desenvolvimento, engenharia, liderança e suporte.

Foram identificados 4 temas recorrentes nas respostas do questionário:

1- Grau de satisfação com as ferramentas e processos de automação de testes

Esta questão mede o grau de satisfação dos analistas de desenvolvimento com a sua rotina de trabalho com ferramentas e processos. Com base nos resultados, é possível concluir que a maioria dos desenvolvedores está satisfeita com as ferramentas e processos de automação de testes utilizados na empresa, com 4 respondendo ótimo e 12 respondendo bom. Isso indica que uma boa aderência e implementação das ferramentas de automação de testes.

No entanto, é importante observar que 3 dos entrevistados responderam médio, o que sugere que há espaço para melhorias. É interessante notar que esses 3 respondentes são novos na empresa, o que pode indicar que eles ainda estão se adaptando aos processos e ferramentas utilizados.

Com base nisso, é importante realizar uma análise mais detalhada para entender quais aspectos da automação de testes podem ser melhorados para atender melhor às expectativas e aumentar o grau de satisfação geral.

2- Percepção de valor na utilização de automação de testes no produto.

Considerando que a grande maioria (17) dos respondentes avaliaram como "ótimo” a percepção de valor na utilização de automação de testes no produto, podemos inferir que a maioria dos colaboradores que trabalham com automação de testes na empresa estão satisfeitos com os resultados obtidos.

3- Percepção de valor na utilização de automação de testes na carreira.

De maneira geral, podemos dizer que os resultados indicam uma percepção positiva sobre a importância da automação de testes na carreira, a grande maioria (19) dos respondentes acredita que há muito valor na utilização de automação de testes na sua carreira.

4- Segurança na entrega do produto final.

Com base nos resultados da pesquisa, é possível observar que a maioria (16) acredita que a automação de testes pode trazer muita melhora na segurança na entrega do produto final. Isso sugere que a automação de testes é vista como uma estratégia eficaz para garantir a qualidade e confiabilidade do software entregue aos clientes.

No entanto, também é importante notar que 5 respondentes indicaram que a automação de testes tem um efeito moderado na segurança na entrega do produto final. Isso pode indicar que esses respondentes acreditam que outras estratégias, além da automação de testes, também são necessárias para garantir a segurança e qualidade do software.

A partir das respostas obtidas, é possível concluir que a percepção geral dos participantes do projeto é que a implementação de uma estratégia de automação de testes, juntamente com uma metodologia robusta de desenvolvimento de software, é benéfica para a melhoria contínua, desenvolvimento da carreira dos analistas envolvidos e qualidade do produto final entregue.

Com base nos estudos realizados e artefatos analisados, é possível notar que a implementação de sucesso de um processo de automação resulta em:

- Ganho de qualidade no sistema final entregue aos clientes;
- Diminuição no tempo gasto com testes manuais;
- Gastando menos tempo com correções e testes manuais, é notável o ganho de horas em inovações para o produto, gerando novas funcionalidades;
- Diminuição no retorno de correções entregues e aberturas de chamado com problemas colaterais;
- Aumento de confiança dos desenvolvedores ao realizar alterações em um código já existente.

Conclui-se que a implementação de um processo de automação de testes em um sistema legado não é rápida nem simples, requer tempo, investimento, planejamento e capacitação. No entanto, pode trazer benefícios significativos, como um produto mais estável, redução de retrabalho e diminuição das ocorrências de erros, podendo assim auxiliar projetos que estão em busca de um produto mais estável.

É importante ressaltar que a automação de testes não substitui completamente a necessidade de analistas de testes, já que ainda são necessários em

testes que exigem análise de interface e em etapas críticas de lançamento de versões. Em resumo, a automação de testes é uma ferramenta essencial para melhorar a qualidade do software, mas deve ser vista como um complemento importante em qualquer processo de garantia de qualidade de software.

5. Referências

EBANHESATEN, K.; FüßL, F.-F.; STREITFERDT, D. Refactoring and automated testing of distributed erp systems. In: **2013 IEEE 37th Annual Computer Software and Applications Conference Workshops**. [S.l.: s.n.], 2013. p. 681–684.

KHODABANDEHLOO, H. et al. A testing approach while re-engineering legacy systems: An industrial case study. In: **2021 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)**. [S.l.: s.n.], 2021. p. 600–604.

KUMAR, R.; KUMAR, V. Process optimization for testing of domain specific languages in industrial automation. In: **2015 World Congress on Information Technology and Computer Applications (WCITCA)**. [S.l.: s.n.], 2015. p. 1–4.

MAJTHOUB, M.; QUTQUT, M. H.; ODEH, Y. Software re-engineering: An overview. In: **2018 8th International Conference on Computer Science and Information Technology (CSIT)**. [S.l.: s.n.], 2018. p. 266–270.

SHIHAB, E. et al. Prioritizing unit test creation for test-driven maintenance of legacy systems. In: **2010 10th International Conference on Quality Software**. [S.l.: s.n.], 2010. p. 132–141.

WARREN, I.; RANSOM, J. Renaissance: a method to support software system evolution. In: **Proceedings 26th Annual International Computer Software and Applications**. [S.l.: s.n.], 2002. p. 415–420.

WIECZOREK, S.; STEFANESCU, A.; SCHIEFERDECKER, I. Test data provision for erp systems. In: **2008 1st International Conference on Software Testing, Verification, and Validation**. [S.l.: s.n.], 2008. p. 396–403.

WIKLUND, K. et al. Impediments for software test automation: A systematic literature review. **Software Testing, Verification and Reliability**, v. 27, n. 8, p. e1639, 2017. E1639 stvr.1639. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/stvr.1639>.