INTERFACE TECNOLÓGICA

AUTOMAÇÃO DE TESTES: uma abordagem comparativa entre ferramentas

TEST AUTOMATION: a comparative approach between tools

Fabio Codo – fabio.codo@fatec.sp.gov.br
Faculdade de Tecnologia de Mogi das Cruzes (Fatec) – Mogi das Cruzes – SP – Brasil

Guilherme Guimarães de Souza – guilherme_guima123@hotmail.com
Faculdade de Tecnologia de Mogi das Cruzes (Fatec) – Mogi das Cruzes – SP – Brasil

Luiz Henrique Pereira Rodrigues – luizhenrique.rodrigues011@gmail.com
Faculdade de Tecnologia de Mogi das Cruzes (Fatec) – Mogi das Cruzes – SP – Brasil

Sarah Agnys Pedroso Batista – sarahagnys18@gmail.com
Faculdade de Tecnologia de Mogi das Cruzes (Fatec) – Mogi das Cruzes – SP – Brasil

DOI: 10.31510/infa.v20i2.1771
Data de submissão: 06/09/2023
Data do aceite: 16/11/2023
Data da publicação: 20/12/2023

RESUMO

Com a evolução de frameworks, bibliotecas comerciais e linguagens de programação ao longo do tempo o processo de desenvolvimento tornou-se mais flexível e simples, facilitando a entrega de um maior número de softwares em menos tempo. No entanto, essa flexibilidade também traz consigo desafios relacionados à qualidade do software. Devido à natureza dinâmica e flexível desses métodos, muitos programas são frequentemente testados quanto à qualidade e, em muitos casos, são encontrados erros e comportamentos inesperados. Com os avanços na área da computação, surgiram diversas plataformas de software amplamente utilizadas, especialmente em aplicações web e mobile. A automação de testes desempenha um papel fundamental ao testar a funcionalidade do sistema, e o uso de ferramentas automatizadas de teste de software pode economizar muito tempo dos testadores. Este estudo tem como objetivo realizar uma comparação entre as ferramentas de automação de teste de software Katalon Studio, Selenium e UFT, analisando as principais características relacionadas ao uso e aprendizado de cada uma delas. Foi desenvolvido um ambiente de testes para explorar e utilizar as funcionalidades de cada uma dessas ferramentas. O estudo busca identificar as vantagens e desvantagens de cada ferramenta, a fim de auxiliar na escolha da melhor opção para a automação de testes de software.

Palavras-chave: Funcionalidade. Automação. Ferramentas.

ABSTRACT

With the evolution of frameworks, commercial libraries and programming languages over time, the development process has become more flexible and simpler, making it easier to deliver more software in less time. However, this flexibility also brings with it challenges related to software quality. Due to the dynamic and flexible nature of these methods, many programs are frequently tested for quality and, in many cases, errors and unexpected behavior are found. With advances in computing, a number of widely used software platforms have emerged, especially in web and mobile applications. Test automation plays a key role when testing system functionality, and the use of automated software testing tools can save testers a lot of time. This study aims to carry out a comparison between the software test automation tools Katalon Studio, Selenium and UFT, analyzing the main characteristics related to the use and learning of each one. A test environment was developed to explore and use the functionalities of each of these tools. The study seeks to identify the advantages and disadvantages of each tool in order to help choose the best option for software test automation.

Keywords: Functionality. Automation. Tools.

1 INTRODUÇÃO

Os processos de desenvolvimento de software que incorporam o uso de metodologia ágil e novos avanços tecnológicos podem concluir projetos com rapidez e eficiência. A criação de software que normalmente levaria meses pode ser concluída em apenas algumas semanas desde que o projeto não seja muito complexo, onde vários frameworks, bibliotecas e linguagens de programação foram criados ao longo do tempo que ajudam a agilizar o processo. Isso torna possível criar mais software em menos tempo (RIBEIRO, 2021).

Dessa forma, com o desejo de aperfeiçoar seus programas, alguns desenvolvedores de software não conseguem terminar seu trabalho devido a problemas e falhas inesperadas. Antes de terminar um programa, é melhor adicionar fases extras de teste ao seu processo de desenvolvimento. Essas fases adicionais podem ajudar a garantir a integridade do programa e evitar perdas financeiras, como perda de tempo ou dinheiro. Testes de automação adicionais são possíveis com sistemas definidos como aplicativos WEB. Esses sistemas geralmente exigem um processo de teste mais aprofundado devido ao tamanho do sistema e seus muitos componentes diferentes. Para tornar esse processo de teste possível, os programadores podem considerar o uso de ferramentas de automação (RIGBY; ELK; BEREZ, 2020).

As ferramentas permitem que os programadores criem testes automatizados que podem ser aplicados em várias partes do sistema. Isso ajuda os programadores a criarem testes automatizados que replicam seus scripts para qualquer situação que desejem testar, automação de teste de software vêm em muitas variedades diferentes (SILVA, 2021). Algumas ferramentas

facilitam todo o processo de teste, enquanto outras são usadas apenas para partes específicas dele. Isso pode causar problemas para quem deseja criar um conjunto de testes automatizados. A falta de documentação torna difícil para alguém decidir qual ferramenta é mais adequada.

No entanto, as ferramentas de automação de teste de software oferecem muitos benefícios. Comparando as características de cada ferramenta, o aluno pode fazer um estudo sobre suas vantagens e limitações. As ferramentas de teste reduzem a intervenção humana nos resultados do teste e aumentam a confiabilidade do teste (JESUS, 2022).

Então, as ferramentas de teste influenciam a confiabilidade do software testado influenciando qual software testar. Ferramentas de teste com escopos semelhantes — por exemplo, automação — influenciam as decisões de teste de software dos testadores. Uma comparação entre diferentes ferramentas de teste com escopos semelhantes influencia essa escolha. A comparação de ferramentas automatizadas de teste de software para a criação de aplicativos da Web é mais eficaz com ferramentas que mostram seus pontos fortes em relação a um determinado assunto de teste. Isso porque mostra a importância dos aspectos positivos da ferramenta em sua totalidade (PRADO, 2018).

2 FUNDAMENTAÇÃO TEÓRICA

O artigo sugere adotar uma abordagem qualitativa baseada em Gil (2008). Essa abordagem visa desenvolver uma compreensão mais aprofundada do fenômeno em questão por meio de uma análise qualitativa e tirar conclusões consistentes com esse problema. A pesquisa pode ser categorizada de acordo com seu propósito, conforme afirmado por Brizola (2020), que menciona que a pesquisa pode ser exploratória, descritiva e explicativa.

No contexto específico dos testes de software, os testes automatizados desempenham um papel importante, onde se referem ao uso de ferramentas e técnicas para automatizar a execução e verificação de casos de teste. Essas ferramentas oferecem várias vantagens, como a redução do tempo necessário para executar os testes, a melhoria da cobertura de testes e a detecção precoce de defeitos.

No caso do assunto em questão, há uma escassez de estudos, além da observação de sua aplicação prática. Portanto, a abordagem descritiva também é relevante, pois busca descrever as características do objeto de estudo, estabelecer relações entre variáveis e observar o desempenho real. Isso implica em uma análise descritiva qualitativa, em que os dados coletados são organizados e vinculados, culminando em uma análise qualitativa dos resultados obtidos.

3 MATERIAL E MÉTODOS

A coleta de dados, será para comparação das ferramentas Katalon, Selenium e UFT, são obtidas executando testes automatizados usando todas as ferramentas no mesmo cenário de teste. Os métodos de análise e interpretação, dos dados coletados serão comparados, analisados e avaliados de acordo com o usuário que realiza a pesquisa, onde as conclusões serão com base na análise realizada, onde pode-se prever quais ferramentas geralmente são melhores ou não no que fazem. Para medir as propriedades de cada ferramenta para comparação, parâmetros foram definidos e monitorados durante o uso para que ambos pudessem ser comparados lado a lado, onde a facilidade de uso, será mostrado através de dados baseados na experiência do usuário por usuários sem experiência com este tipo de ferramenta, levando em consideração a performance e os dados obtidos no tempo de execução de cada teste analisado e mostrando se é adaptável através do parâmetro que aumenta com a capacidade de executar várias tarefas não pré-programadas pelo aplicativo.

Com isso, foram definidos os processos metodológicos utilizados para a análise e os passos realizados durante o mesmo.

- **Definição dos critérios de análise:**

O primeiro passo foi estabelecer os critérios que seriam considerados na análise comparativa. Esses critérios foram selecionados com base em fatores relevantes para a avaliação das plataformas, como facilidade de uso, suporte a diferentes tipos de testes, flexibilidade, integração com outras ferramentas e desempenho. Essa definição prévia permitiu uma abordagem sistemática e coerente na análise das plataformas.

- **Seleção dos projetos de teste:**

Com base nos critérios estabelecidos, foram selecionados projetos de teste representativos que abrangiam diferentes aspectos dos testes de software, como testes funcionais, testes de interface do usuário e testes de compatibilidade. Essa seleção foi realizada para garantir que os cenários de teste fossem adequados para avaliar as capacidades das plataformas e permitir uma comparação justa entre elas.

- **Configuração dos ambientes de teste:**

Em seguida, foram configurados os ambientes de teste para cada plataforma. Isso incluiu a instalação e configuração adequada de cada plataforma, bem como a preparação dos recursos e dados necessários para a execução dos testes. Essa etapa foi essencial para garantir que todas

as plataformas estivessem configuradas de forma equivalente, eliminando possíveis vieses causados por diferenças na configuração dos ambientes.

- **Execução dos testes:**

Com os ambientes de teste configurados, os testes foram executados em cada plataforma de acordo com os cenários definidos. Durante essa etapa, foram registrados os resultados obtidos, incluindo métricas de desempenho, erros encontrados e facilidade de implementação dos testes. Essa execução permitiu uma avaliação prática das capacidades das plataformas e a coleta de dados concretos para análise comparativa.

4 RESULTADOS E DISCUSSÃO

A automação de teste oferece a possibilidade de validar softwares de forma instantânea e eficaz. Uma vez que os testes tenham sido automatizados, eles podem ser executados de forma rápida e repetidas vezes. Em quase todos os casos, este é o método mais econômico para produtos de software de teste de regressão que têm uma longa vida útil (ALVES, 2019).

Na verdade, a automação de teste de qualquer aplicativo é a melhor maneira de aumentar a eficácia, eficiência e abrangência do processo de teste (BRITO, 2019). Geralmente, as equipes que configuram a automação de teste estão focadas em selecionar uma ferramenta que lhes permita maximizar os seguintes critérios de sucesso:

- Confiabilidade: altas taxas de detecção de problemas;
- Manutenibilidade: 1) Sensibilidade mínima para alterações no aplicativo e no caso de teste; 2) Definição do caso de teste separada do código de automação;
- Escalabilidade: a capacidade de expandir com eficiência a cobertura de teste e a estrutura de automação, se necessário.

Há ainda mais considerações quando as equipes dão o salto para a automação de teste. Os seguintes fatores como: diversidade de plataformas, diversidade de dispositivos, diversidade de ferramentas e imaturidade da ferramenta, tornam a execução da automação de teste ainda mais complexa.

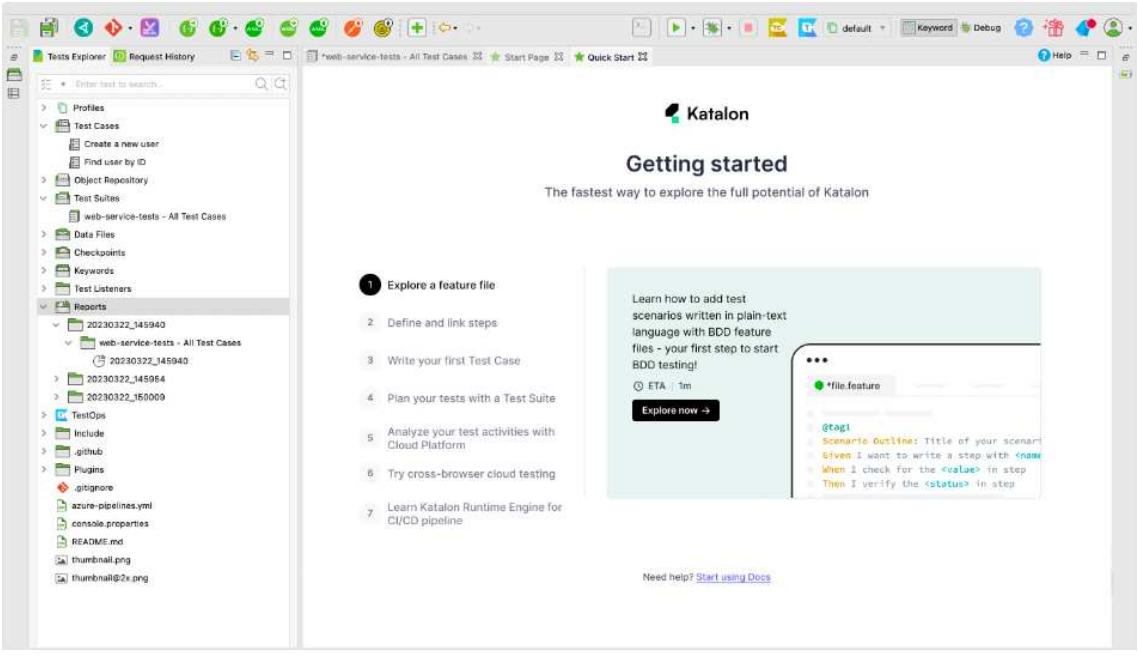
O ciclo de desenvolvimento rápido, onde há uma alta taxa de mudança de tecnologia, ao selecionar uma ferramenta de teste, compatibilidade de plataforma, flexibilidade, suporte disponível e custo devem ser considerados (ARAÚJO, 2022). A compatibilidade de plataforma, onde a capacidade da ferramenta de interagir com todas as plataformas móveis e de desktop suportadas. Flexibilidade, onde a ferramenta considerada suporta o aplicativo mais

importante para suas equipes, cliente, administrador etc. Suporte disponível, onde as equipes de suporte, recursos ou comunidades existentes para ajudar equipes ou clientes a superar obstáculos técnicos. O custo de propriedade, onde o investimento de longo prazo que a compra/configuração ou suporte da ferramenta exige em comparação com o retorno esperado.

Por sua vez, aumenta a necessidade de selecionar e usar o conjunto certo de ferramentas de automação de teste. Dada a ampla gama de opções de ferramentas de teste controladas, com seus próprios pontos fortes e controles, vamos aprender como selecionar a ferramenta de automação certa para o seu projeto, comparando qualitativamente o Katalon Studio com outros kits de ferramentas de teste não automatizados.

O Katalon Studio (figura1) é uma plataforma de teste controlado que fornece um conjunto abrangente de recursos para implementar uma solução de teste totalmente controlada para web, API e dispositivos móveis. Construído sobre as estruturas de código aberto Selenium e Appium, o Katalon Studio permite que as equipes mudem para a automação de teste rapidamente e cumpram o esforço e a experiência necessários para aprender e integrar essas estruturas para suas necessidades de teste.

Figura1. Página inicial da ferramenta Katalon

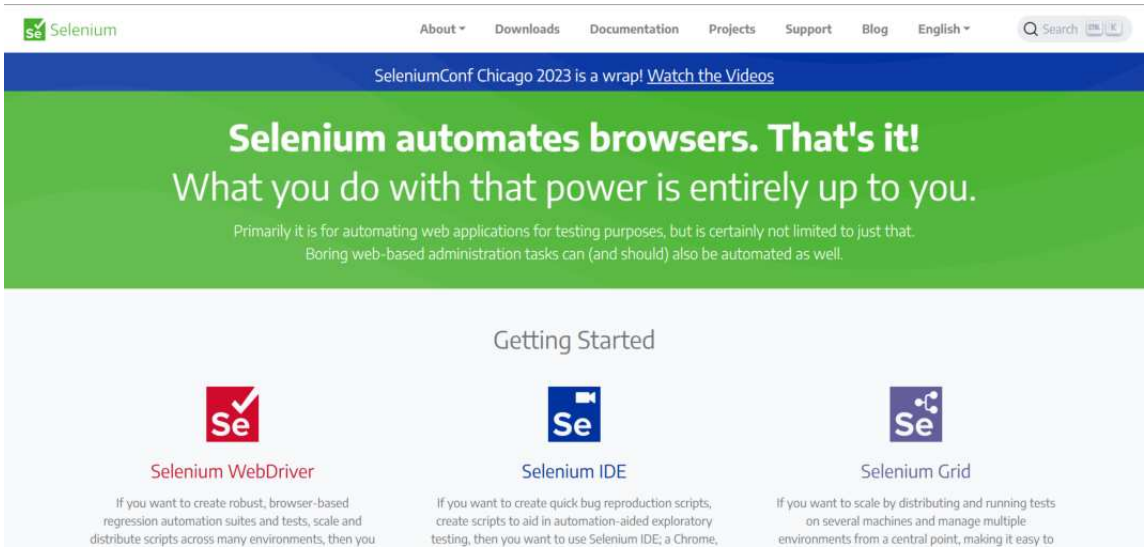


Fonte: Katalon

O Selenium (figura2) é provavelmente a estrutura de automação mais popular e consiste em várias ferramentas e plug-ins para testar aplicativos da web. O Selenium é conhecido por sua poderosa capacidade de oferecer suporte a testes de desempenho de aplicativos da web. O Selenium é altamente valorizado por sua capacidade de integração e compatibilidade com

diferentes navegadores e sistemas operacionais, tornando-se uma escolha confiável para testes de compatibilidade de plataforma.

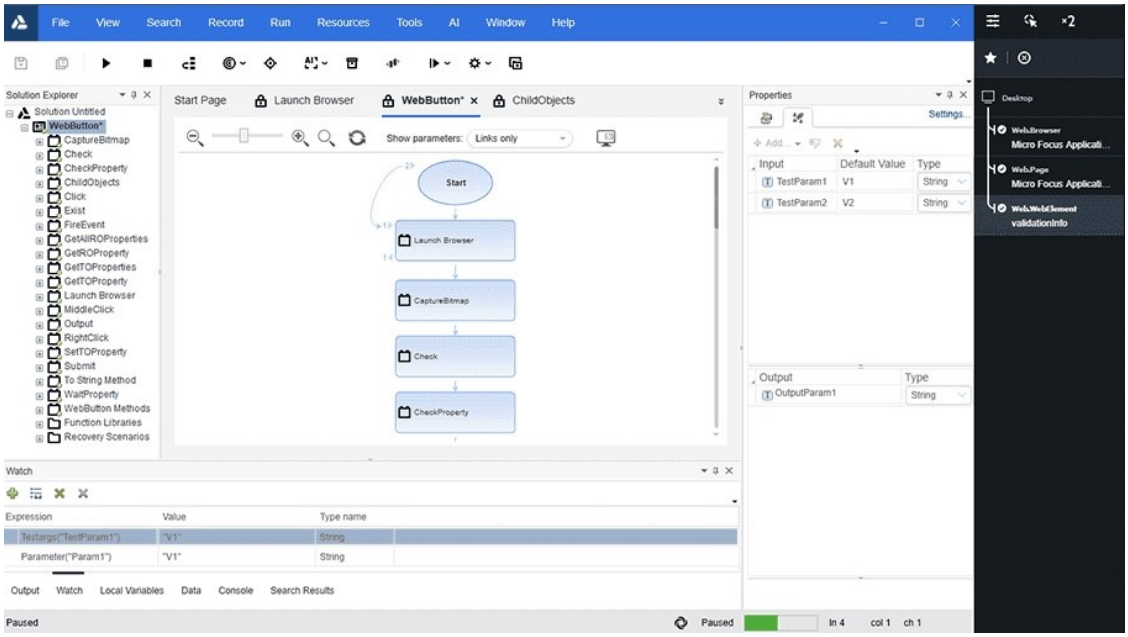
Figura 2. Página inicial do site da ferramenta Selenium



Fonte: Selenium

O Unified Functional Test (UFT)(figura3), é talvez a ferramenta comercial mais popular para testes automatizados autônomos. Oferece um conjunto abrangente de recursos para atender à maioria das necessidades de testes automatizados em plataformas de desktop, móveis e da Web. suporta várias linguagens de programação, como VBScript, JavaScript e C#, permitindo que os testadores gravem scripts interativos ou escrevam manualmente seus próprios scripts.

Figura 3. Página inicial da ferramenta UFT One



Fonte: MicroFocus UFT One

A tabela 1 apresenta a comparação entre as ferramentas e se concentra nos recursos comuns para execução de testes automatizados.

Tabela 1. Comparação das ferramentas de testes automatizados

Características	Estúdio Katalon	Selenium	UFT
Plataforma de desenvolvimento de teste	Plataforma cruzada	Plataforma cruzada	Janelas
Aplicativo em teste	Área de trabalho do Windows, Web, aplicativos móveis, serviços API/Web	Aplicativos da web	Área de trabalho do Windows, Web, aplicativos móveis, serviços API/Web
Linguagens de escrita	Java/GroovyName	Java, C#, Perl, Python, JavaScript, Ruby, PHP	VBScriptGenericName
Habilidades de programação	Não obrigatório. Recomendado para scripts de teste avançado	Habilidades avançadas necessárias para integrar várias ferramentas	Não obrigatório. Recomendado para scripts de teste avançado

Curvas de aprendizado	Médio	Alto	Médio
Facilidade de instalação e uso	Fácil de configurar e executar	Requer instalação e integração de várias ferramentas	Fácil de configurar e executar
Hora de criação do script	Rápido	Lento	Rápido
Guarda e manutenção de objetos	Repositório de objetos integrados, XPath, identificação de objetos	XPath, mapas de interface do usuário	Repositório de objetos integrados, detecção e correção de objetos inteligentes
Teste baseado em imagem	Suporte integrado	Requer uma instalação de bibliotecas adicionais	Suporte integrado, reconhecimento de objeto baseado em imagem
Integrações DevOps/ALM	Muitos	Não (requer bibliotecas adicionais)	Muitos
Integrações contínuas	Ferramentas populares de CI (por exemplo, Jenkins, Teamcity)	Várias ferramentas CI (por exemplo, Jenkins, Cruise Control)	Várias ferramentas CI (por exemplo, Jenkins, HP Quality Center)
Análise de teste	Katalon TestOps	Não	Não
Suporte ao produto	Comunidade, Serviço de apoio empresarial, Equipe dedicada	Comunidade de código aberto	Funcionários dedicados, Comunidade
Tipo de licença	Proprietário	Código aberto (Apache 2.0)	Proprietário

Fonte: Autores (2023).

A tabela 2 apresenta uma outra perspectiva, selecionando e comparando os principais pontos fortes e limitações dessas ferramentas.

Tabela 2.

Ferramentas	Forças	Limitações
Katalon Studio	• Não são necessárias taxas de licenciamento e manutenção (serviços de suporte dedicados pagos estão disponíveis, se necessário). • Integrando estruturas e recursos necessários para criação e execução rápida de casos de teste. • Construído sobre a estrutura do Selenium, mas eliminando a necessidade de habilidades avançadas de programação necessárias para o Selenium.	• Solução emergente com uma comunidade em rápido crescimento. • O conjunto de recursos ainda está evoluindo. • Falta de opções para linguagens de script: apenas Java/Groovy é suportado.
Selenium	• Código aberto, sem taxas de licenciamento e manutenção. • Desenvolvimento grande e ativo e comunidade de usuários para acompanhar o ritmo das tecnologias de software. • Aberto para integração com outras ferramentas e estruturas para aprimorar sua capacidade	• As equipes de teste precisam ter boas habilidades de programação e experiência para configurar e integrar o Selenium com outras ferramentas e estruturas. • Novas equipes precisam investir tempo antecipadamente para configuração e integração. • Apoio lento da comunidade.
UFT	• Recursos de teste concluídos maduros e	• Solução cara: as taxas de licença e

	abrangentes integrados em um sistema único. • Suporte dedicado ao usuário, além de uma grande comunidade de usuários estabelecida. • Requer apenas habilidades básicas de programação para começar a criar e executar testes.	manutenção são consideravelmente altas. • Possíveis altos custos para atualizações e módulos adicionais. • Suportando apenas VBScript.

Fonte: Autores (2023).

4 CONCLUSÃO

Não existe uma única ferramenta de teste automatizado que seja adequada para todos os cenários. Recomenda-se fortemente que os testadores avaliem diferentes ferramentas para selecionar aquela que melhor se adapta às suas necessidades gerais de teste. À medida que as linguagens de programação e as tecnologias utilizadas no desenvolvimento de software continuam a evoluir, as ferramentas de teste automatizado também acompanham essa evolução.

Os fornecedores comerciais geralmente cobram por atualizações das ferramentas, o que pode ser relevante caso o seu software utilize tecnologias novas e em constante mudança. Por outro lado, as ferramentas de código aberto e não comerciais não exigem custo adicional, mas podem demandar esforço e experiência para integrar novas atualizações. Encontrar o suporte e o conhecimento necessários para integrar diversas ferramentas e frameworks em uma solução de código aberto pode ser desafiador.

Uma alternativa viável para soluções comerciais e de código aberto são as novas ferramentas que se integram a frameworks de código aberto, como o Katalon. Essas ferramentas oferecem uma opção que combina características e suporte adequados para ambientes de teste controlados. Ao considerar as opções disponíveis, é importante avaliar não apenas o custo, mas também a compatibilidade com as tecnologias utilizadas no projeto, a facilidade de integração, a disponibilidade de suporte e a experiência necessária para utilizá-las de forma eficaz.

No final, a escolha da ferramenta de automação de testes deve ser baseada em uma análise criteriosa das necessidades específicas do projeto, levando em consideração fatores

como orçamento, tecnologias envolvidas, suporte disponível e experiência da equipe. O objetivo é selecionar a ferramenta que melhor atenda aos requisitos de teste e proporcione eficiência, precisão e qualidade no processo de teste automatizado.

REFERÊNCIAS

AHR, Miha. **Avtomatizirano testiranje spletnih aplikacij em storitev**. 2019.

ARAÚJO, Fernando Henrique Duarte. **Desenvolvimento de sistemas de informação com tecnologia Low-Code.** 2022. Tese de Doutorado.

ALVES, Leandro Domingues Macedo. **Ambiente ágil de testes automáticos em nuvem para os sistemas web fence dos experimentos do CERN**. 2019.

BRITO, João David Guerreiro de. **Aperfeiçoamento da integração de sensores de dispositivos móveis na domótica**. 2019. Dissertação de Mestrado.

BRIZOLLA, Maria Margarete Baccin et al. **Uma revisão sobre a pesquisa qualitativa em ciências sociais aplicadas**. UFAM Business Review-UFAMBR, v. 2, n. 3, p. 103-130, 2020.

JESUS, Patrícia Isabel Cunha de. **Impactos da Automatização de Testes no Meio Empresarial.** 2022. Tese de Doutorado.

PRADO, Marllos Paiva et al. **Contribuições ao suporte cognitivo em teste de software unitário: um framework de tarefas e uma agenda de pesquisa**. 2018.

RIBEIRO, Márcio Vinicius Machado et al. **Inteligência artificial no Poder Judiciário: ética e eficiência em debate**. 2021.

RIGBY, Darrell; ELK, Sarah; BEREZ, Steve. **Ágil do jeito certo: transformação sem caos.** Saraiva Educação AS. 2020.

SILVA, Lucas Rodrigues. **O Framework ConBaT: suporte a testes baseados em contexto para sistemas desenvolvidos em Arduino.** 2021. Dissertação de Mestrado. Universidade Federal do Rio Grande do Norte.