Framework Integrador para Automação de Testes Desktop

David F. Brandão[1,3], Talita M. Ferreira[1,3], Wylliams Santos[1,2]

[1]Centro de Estudos Avançados do Recife (CESAR School)
Recife – PE – Brazil

[2]Escola Politécnica de Pernambuco (Poli), Universidade de Pernambuco (UPE)
Recife – PE – Brazil

[3] Instituto SiDi de Pesquisa e Desenvolvimento (SiDi)
Recife – PE – Brazil

{d.brandao, talita.m}@sidi.org.br, wbs@upe.br

Abstract. *It is noticed on market a shortage of frameworks focused on desktop tests automation, understanding it as a challenge for IS in the quality technology development field, this paper looks forward to make simpler and more practical the automation process introducing an integrator framework for desktop tests automation.*

Resumo. *Percebe-se no mercado uma escassez de frameworks com foco em automação de testes em aplicações desktop, entendendo como um desafio de SI na área de desenvolvimento de tecnologias de qualidade, este trabalho busca tornar prático e simples o processo de automação apresentando a criação de um framework integrador para automação de testes desktop.*

1. Visão Geral

Tecnologias da informação desempenham um papel cada vez mais importante na forma como os sistemas de informação são desenvolvidos e utilizados. Com avanço constante desta tecnologia, surgem novas possibilidades e inovações que podem ser aplicadas aos sistemas de informação já existentes, agregando significativo valor aos processos de negócio das organizações em termos de produtividade, eficiência e qualidade [Audy, De Andrade e Cidral, 2009].

A indústria de software com foco no desenvolvimento de aplicações web, mobile ou microsserviços provê de diversas tecnologias e *frameworks* para automação de testes, tendo uma comunidade ativa com diversos produtos open source, que sempre está se atualizando e buscando por melhorias. Entretanto, no que se refere à automação de testes para aplicações desktop, surgem alguns desafios. Por mais que as tecnologias relacionadas à automação de testes estejam em alta, em sua maioria as ferramentas gratuitas são predominantemente direcionadas a aplicações web, devido à grande quantidade de aplicações criadas para esse ambiente [Botossi, 2019].

Dessa forma, propõe-se como Grande Desafio de SI, a necessidade de inovar na forma como os testes são automatizados para ambiente desktop, a fim de suprir a escassez de *frameworks* de código aberto disponíveis no mercado atual.

Buscando entregar praticidade no desenvolvimento e diminuir os esforços na fase inicial da implementação de uma solução de automação de testes, foi realizada uma

pesquisa aplicada a fim de entender os comportamentos chave e as principais funcionalidades dos frameworks open source de automação de testes. Foram levantados pontos a fim de facilitar o desenvolvimento de uma estrutura de automação fazendo com que os usuários não precisem se preocupar com itens como a arquitetura da solução, obtenção de evidência de testes, comunicação com a aplicação e geração manual de relatórios de teste.

A criação do *framework* foi baseada em conceitos que estão amplamente difundidos e validados no mercado de automação de testes, promovendo a união de algumas tecnologias para o seu funcionamento. O *Specflow* implementa a camada mais próxima da linguagem humana por meio da criação de um arquivo *.feature* contendo os cenários de teste escritos em *gherkin*. O teste, por sua vez, é traduzido em código C# por meio de um método que implementa a execução dos passos dentro de cada cenário, como demonstrado pela figura 1.

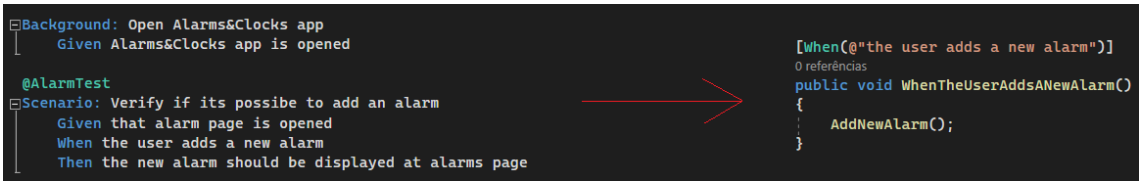

Figura 1. Relação do gherkin com o código C#

A classe que implementa os métodos mencionados acima, segue o padrão de design *page objects*, que de acordo com Romanini e Sotto (2019), é definido como "Um padrão de projeto com a proposta de criar objetos para cada página web, utilizando orientação a objetos". Este padrão que nasceu para web se mostrou útil na abstração de atributos e métodos, reduzindo a duplicação de código e facilitando a manutenção dos testes em aplicações desktop, conforme mostrado na figura 2.

Figura 2. Implementação do modelo *page objects*

Quando um teste é executado, o *Specflow* identifica quais passos compõem o cenário e irá relacionar com o código C#, que por sua vez, é escrito com o suporte da biblioteca *NUnit* do C#. A comunicação do teste com a aplicação se dá pela biblioteca *Appium,* que cria uma ponte de comunicação com o *WinAppDriver* que executa os comandos na aplicação alvo. Ao final dos testes, o plugin *LivingDoc* do *Specflow* irá ler o resultado do teste e exportará um relatório da execução no formato *HTML*, como mostrado na figura 4.

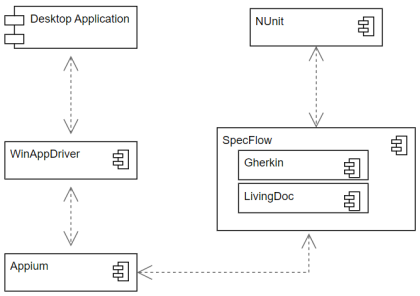

Figura 4. Diagrama de comunicação das ferramentas

Após seu desenvolvimento, este framework foi aplicado em projetos da empresa SiDi que tinham como foco o desenvolvimento de aplicações desktop. A partir da sua implementação, foi notado pelos gerentes de projeto e desenvolvedores de teste a sua efetividade no processo testes, diminuindo os esforços iniciais na construção de uma solução de automação, melhorando a qualidade do desenvolvimento e facilitando a manutenção dos testes. Estes ganhos no processo fizeram com que o framework fosse escalado a mais projetos desta empresa, se tornando a base a ser seguida para este tipo de automação.

Reconhecimento - Este trabalho não teria sido possível sem o apoio do instituto SiDi de Pesquisa e Desenvolvimento.

2. Audiência

Pessoas desenvolvedoras de software, engenheiros de qualidade, analistas de requisitos e gerentes de projeto.

3. Minibiografia das pessoas autoras

David Ferreira Brandão é especialista em testes (Cesar School), engenheiro de computação (Wyden Educacional). Atua com testes de software desde 2019, sendo hoje QA PL no SiDi.

Talita Moraes Ferreira é especialista em Testes (Cesar School), engenheira da computação (Wyden Educacional), técnica em desenvolvimento de sistemas (ETE Miguel Batista). Atua com testes de software desde 2021, sendo hoje QA JR no SiDi.

Wylliams Santos é doutor e mestre em Ciência da Computação (UFPE). Professor Adjunto dos cursos de Graduação, Mestrado e Doutorado em Engenharia da Computação da Escola Politécnica de Pernambuco (Poli/UPE).

Referências

Audy, J. L. N., de Andrade, G. K., & Cidral, A. (2009). Fundamentos de sistemas de informação. Bookman editora.

Botossi, R. (2019). Desenvolvimento de um protótipo de sistema de automação de testes do tipo registro/reprodução para ambientes desktop windows utilizando técnicas de padrões de projeto.

Romanini, I. Z., & Sotto, E. C. S. (2019). SELENIUM WEB DRIVER NA EVOLUÇÃO DOS TESTES MANUAIS. Revista Interface Tecnológica, 16(2), 112-123.