UNIVERSIDADE ESTADUAL DE FEIRA DE SANTANA
Autorizada pelo Decreto Federal nº 77.496 de 27/04/76
Recredenciamento pelo Decreto n°17.228 de 25/11/2016
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
COORDENAÇÃO DE INICIAÇÃO CIENTÍFICA

XXV SEMINÁRIO DE INICIAÇÃO CIENTÍFICA DA UEFS
SEMANA NACIONAL DE CIÊNCIA E TECNOLOGIA - 2021

Abordagem de Microsserviços Aplicada a Arquitetura do Simulador de Circuitos Elétricos e Eletrônicos

Marcos de Jesus Ramos[1] e Ana Cláudia Fiorin Pianesso [2]
1. Bolsista PROBIC/CNPq, Graduando em Engenharia da Computação, Universidade Estadual de Feira de Santana, e-mail: mjramosfsa@gmail.com
2. Ana Cláudia Fiorin Pianesso, Departamento de Tecnologia (DTEC), Universidade Estadual de Feira de Santana, e-mail: acfpianesso@uefs.br

PALAVRAS-CHAVE: Arquitetura de software de microsserviços; Protocolo de comunicação; Circuitos elétricos

INTRODUÇÃO

A engenharia elétrica possui um histórico de dificuldades quando da realização de testes destinados à comprovação do comportamento de circuitos elétricos e eletrônicos. Para isso, utilizava-se componentes físicos, o que consequentemente acarreta em uma grande elevação nos custos do projeto, além da ausência de segurança e eficiência na análise desses sistemas (MANZONI, 2005). Surgem então, os softwares de simulação, que são aplicações que permitem a criação e manipulação virtual de cenários reais, possibilitando assim a realização de testes sem que haja a necessidade de utilização de componentes físicos.

Existem muitas opções e variações de softwares de simulação disponíveis para uso no mercado, entretanto, mesmo com a evolução destas ferramentas, as pesquisas desenvolvidas na área de engenharia elétrica ainda não estão totalmente amparadas por essas aplicações. Os problemas encontrados são os mais diversos, destacando-se a funcionalidade limitada a circuitos simples, aplicações desenvolvidas em linguagens de programação consideradas defasadas, interfaces pouco intuitivas, alto custo das licenças de uso para softwares proprietários, ausência de manutenção em softwares livres, falta de documentação e a inexistência e flexibilidade para incorporação de modelos matemático elaborados para um cenário específico. Segundo Souza (2011) esses problemas acabam impactando de forma negativa as pesquisas acadêmicas na área.

A motivação do projeto de pesquisa é o desenvolvimento de um simulador de sistemas elétricos e eletrônicos que possa ser capaz de possibilitar a inclusão de modelos já desenvolvidos, sejam em outros sistemas ou como resultados de pesquisas. Pelo que é proposto, surge a necessidade de aplicar técnicas e métodos que visam apoiar a elaboração desse tipo de sistema, garantindo assim a usabilidade e facilidade de manutenção. Esse conjunto de técnicas estão descritas na Engenharia de Software com os aspectos fundamentais para a idealização, estudo, elaboração e concepção de um software e/ou aplicação (SOMMERVILLE, 2007). E, sendo o propósito do projeto a elaboração de um Simulador de Sistemas elétricos e eletrônicos, faz-se necessário projetar e/ou ampliar a proposta de arquitetura de software. Para tanto, utilizou-se a abordagem de microserviços em conjunto com clean architecture na arquitetura já existente, motivando assim a criação de uma nova arquitetura de software que herda características e amplia sua capacidade podendo ser aplicada em diversos cenários.

Além disso, a API de comunicação e interconexão passou por um processo de refatoração. Foram aplicadas técnicas de clean code e em conjunto com técnicas mais enxutas de eficiência de encapsulamento, acarretando assim em um ganho de performance e diminuindo a superfície de ataque e/ou falhas .

MATERIAL E MÉTODOS OU METODOLOGIA

O desenvolvimento deste trabalho começou através de uma análise e revisão bibliográfica da arquitetura de software que foi proposta. Mapeamento dos novos requisitos e funcionalidades para a arquitetura já existente. A API de comunicação e interconexão foi revisada buscando identificar funcionalidades e características que necessitassem de atualização ou refatoração.

Um levantamento bibliográfico da arquitetura software, abordagem de arquitetura de microservices e clean architecture foi realizado. Com isso, iniciou-se o processo de refatoração da arquitetura de software que foi proposta, e das abordagens que foram aplicadas em sua concepção, que acabou acarretando em nova proposta de arquitetura de software. Técnicas de clean code, encapsulamento e compressão foram aplicadas. Contudo, visando não quebrar soluções de requisitos que já foram atendidos através da proposta arquitetural anterior, utilizou-se a linguagem de programação ecmascript (Javascript), juntamente com node js, que é um ambiente de execução ecmascript (Javascript), e também usamos o formato de troca de informações JSON permitindo a comunicação e interação com diferentes aplicações.

RESULTADOS E/OU DISCUSSÃO

O processo de refatoração da Arquitetura do Simulador Digital de circuitos elétricos e eletrônicos, com a aplicação de técnicas de clean architecture e de abordagem de microsserviços possibilitou a utilização da arquitetura resultante para elaboração de diferentes aplicações em diferentes domínios do conhecimento.

A API de comunicação e interconexão e do Protocolo de comunicação pertencentes à arquitetura resultaram em um conjunto de serviços (mini aplicações) que trabalham de forma independente, porém produzindo uma única saída possibilitando assim uma maior escalabilidade e maior compatibilidade das aplicações que venham a ser desenvolvidas a partir do seu uso (Redhat, 2020).

A nova proposta de arquitetura de *software* utiliza o conceito de arquitetura de microsserviços híbrida, permitindo abstrair a complexidade no desenvolvimento de novas aplicações (C.M.U., 2011). Fornece regras de consumo e produção dos *endpoints*.

As partes que compõem a proposta de arquitetura são apresentadas na Figura 1.

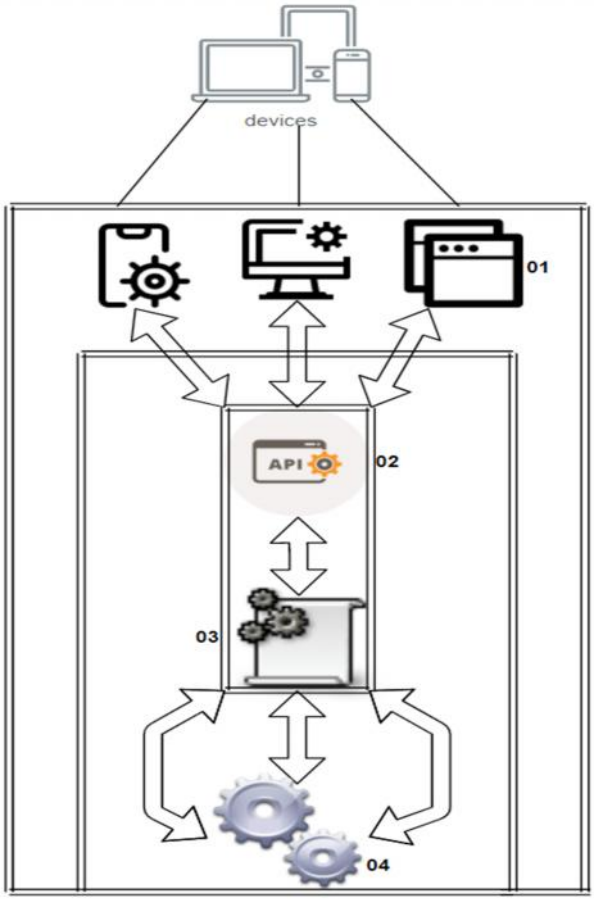

Figura 1: Arquitetura de *software* proposta

A interface principal do sistema pode ser desenvolvida em qualquer linguagem. A implementação desta interface no projeto em questão recomenda-se a utilização de *frameworks e* linguagens que proporcionem a criação de aplicações multiplataforma. Na arquitetura, esta interface é representada pelo elemento 01 da Figura 1. Nela são elaborados os circuitos, as simulações e as verificações dos resultados. Possui funcionalidades diversas destinadas à manipulação dos circuitos elétricos e eletrônicos.
A API de comunicação é representada pelo elemento 02 da Figura 1. Tem como principal objetivo processar as informações da interface e enviar para os módulos de cálculos. As informações gerais do circuito são organizadas em um pacote JSON (J*avaScript Object Notation*), que é enviado para a API de comunicação. O elemento 03 do diagrama da nova proposta arquitetural, representa um *WebService* que um implementa microsserviços que compõem a API de comunicação e interconexão, este, recebe dados da interface já processados por outros microsserviços e faz interconexão com os módulos de cálculo ou matemáticos, que são representados pelos elementos 04 da Figura1.

CONSIDERAÇÕES FINAIS

A arquitetura passou por um processo de refatoração onde buscou-se aplicar técnicas de clean architecture, e ao mesmo tempo mantivesse o conceito de arquitetura de microsserviços visando propor uma maior escalabilidade das aplicações que venham a ser desenvolvidas a partir do seu uso. Aliado a isso buscamos tornar os microsserviços que compõem a API de comunicação que implementa a proposta de arquitetura, que também passou pelo processo de refatoração onde aplicamos os conceitos de clean architecture em conjunto de clean code, provocando assim uma reescrita parcial que se quer completa do core, onde foi buscou-se aplicar no código nomes mais intuitivos, para

deixar mais simples a compreensão, eliminar a repetição internas , diminuir o excesso de comentários, melhorar o tratamento de erros tudo isso acarretou em mais flexibilidade para usou de outras ferramentas externas da API de comunicação e interconexão e na otimização do desempenho em conjunto com o processo de atualização e/ou substituição de algumas libs e/ou frameworks.

Além disso o protocolo de comunicação passou por um processo de refatoração, onde ocorreu alguns updates e também a substituições de algumas técnicas de encapsulamento de dados em JSON, buscando corrigir falhas de segurança nos endpoints, ao mesmo tempo possibilitar uma interação em tempo real das ferramentas que venha a ser desenvolvida e viabilizando uma maior compatibilidade com outras linguagens deixando mais simples a integração com ferramentas elaboradas qualquer linguagem.

Como trabalhos futuros já identificados e em andamento, destacam-se a criação de uma plataforma de integração de APIs e gerenciamento de simulações e análises de circuitos elétricos para o fornecimento de métricas.

REFERÊNCIAS

AMAZON. 2013. Escolher um tipo de endpoint para configurar uma API do API Gateway.

C.M.U. ,Carnegie Mellon University: A.B.L.E. Architecture Based Languages and Environments. Publicado em: 04 de janeiro de 2011. Disponível em: https://www.cs.cmu.edu/~acme/. Acesso em 23, outubro 2020.

ECMA, ECMA-262,ECMAScript® 2021 language specification Disponível em:< https://www.ecma-international.org/publications-and-standards/standards/ecma-262/>.Acesso em: 17, dezembro. 2020.

FOWLER, M., Refactoring :Aperfeiçoando o Design de Códigos Existentes,.2. ed., São Paulo: Pearson Addison Wesley,2018.

MANZONI, A. 2005 .Desenvolvimento de um sistema computacional orientado a objetos para sistemas elétricos de potência : aplicação a simulação rápida e análise da estabilidade de tensão. Rio de Janeiro.

REDHAT. 2013. O que são os microsserviços? . Homepage: https://www.redhat.com/pt-br/topics/microservices/what-are-microservices

Robert C. Martin.Clean Architecture: A Craftsman's Guide to Software Structure and Design, Pearson; Illustrated edição,2017

SOMMERVILLE, I. 2007. Engenharia de Software, 8.ed.São Paulo : Pearson Addison Wesley.

SOUZA, D. M. 2011.Simulador de circuitos elétricos de pequeno porte utilizando modelagem orientada a objetos. Universidade Federal do Rio de Janeiro. Rio de Janeiro