UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
PROJETO DE GRADUAÇÃO

ANDREI CARLOS BASTOS

FERRAMENTA WEB PARA APRENDIZADO DE SISTEMAS ELÉTRICOS DE POTÊNCIA

VITÓRIA – ES
JULHO/2018

ANDREI CARLOS BASTOS

FERRAMENTA WEB PARA APRENDIZADO DE SISTEMAS ELÉTRICOS DE POTÊNCIA

Parte manuscrita do Projeto de Graduação do aluno Andrei Carlos Bastos, apresentada ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Orientador: Prof Dr. Oureste Elias Batista

VITÓRIA – ES
JULHO/2018

ANDREI CARLOS BASTOS

FERRAMENTA WEB PARA APRENDIZADO DE SISTEMAS ELÉTRICOS DE POTÊNCIA

Parte manuscrita do Projeto de Graduação do aluno Andrei Carlos Bastos, apresentada ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Aprovado em 11 de Julho de 2018.

COMISSÃO EXAMINADORA:

_____
Prof. Dr. Oureste Elias Batista
Universidade Federal do Espírito Santo
Orientador

_____
Prof. Dr. Augusto César Rueda Medina
Universidade Federal do Espírito Santo
Examinador

_____
Prof. Dr. Lucas Frizera Encarnação
Universidade Federal do Espírito Santo
Examinador

RESUMO

Neste trabalho apresentam-se os modelos dos principais componentes do diagrama de um Sistema Elétrico de Potência (SEP), a fim de representá-lo em uma interface gráfica web, desenvolvida com o *framework Angular 6*, o qual permite a criação de diagramas de SEP, bem como a realização dos cálculos matemáticos do Fluxo de Potência e Curto Circuito. O presente projeto é uma complementação da dissertação de mestrado de Luiz G. R. Tonini da Universidade Federal do Espírito Santo (UFES), que aborda com mais detalhes a manipulação de dados e operação numérica para realizar os cálculos citados.

Palavras chaves: Sistemas elétricos de potência, Plataforma educacional, Angular 6, Fluxo de Potência, Curto Circuito.

ABSTRACT

This work consists of modeling the main diagram components of an electrical power system in order to represent it through a graphical web interface, developed with Angular 6 framework, which enables the creation of electrical power system diagrams, as well as executing mathematical calculations related to Power Flow and Short Circuit. The present project is a complement of Master dissertation of Luiz G. R. Tonini from Federal University of Espírito Santo, which manipulation of data and numerical operation for accomplishing the calculation previously mentioned are approached in more detail.

Keywords: Electrical Power System, Educational platform, Angular 6, Power flow, Short Circuit

AGRADECIMENTOS

Agradeço, ao meu Orientador e Coorientador por todo trabalho ao longo do desenvolvimento desse projeto.

Aos meus queridos pais Leandro e Andrea, pelo amor e carinho durante toda a trajetória da minha vida, pelo incentivo, suporte e sacrifícios que tiveram para me proporcionar conforto e conhecimento.

Agradeço, em especial, a minha namorada Lilian pelo convívio durante os últimos semestres do curso e pelo apoio no decorrer da elaboração deste trabalho. Grato por toda a força, carinho e palavras de incentivo que me deram energia para continuar seguindo em frente e nunca desistir.

Agradeço aos meus amigos e colegas da faculdade pela companhia durante as aulas, pelos momentos de descontração nos intervalos, pela união e ajuda durante os dias, as noites e as e madrugadas estudando para provas e fazendo trabalhos.

Ao Labic, Laboratório de Imagens e Cibercultura que participei durante quase todo curso, no qual desenvolvi minhas habilidades na área de desenvolvimento de software.

A Sinales, pelo conhecimento adquirido e pela colaboração durante a produção desse projeto.

Ao Gustavo Salzmann pela força e colaboração fornecida quanto ao estilo da página.

A todos os professores que tive contato, não só pelo conteúdo da disciplina, mas por cada ensinamento de vida, ética e conselho pessoal e profissional que cada um transmitiu durante e fora das aulas.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

CC – Curto Circuito

FP – Fluxo de Potência

SEP – Sistema Elétrico de Potência

PESEP – Programa para Ensino de Sistemas Elétricos de Potência

KVA – *Knowlenge Virtual Academy*

SVG - Scalable Vector Graphics

API – *Application Programming Interface*

IDE – Integrated Development Environment

UML – *Unified Modeling Language*

SUMÁRIO

1 INTRODUÇÃO

Os programas de simulação de Sistemas Elétricos de Potência (SEP) possuem como público alvo, em sua maioria, os profissionais que operam o sistema. Devido a esta característica, os programas apresentam interface não didática e preços de licença muitas vezes, inacessíveis para alunos e instituições de ensino. Existe também versões educacionais que apresentam limitações em termos quantitativos quanto ao número de elementos no diagrama entre outras variáveis. Visto tal situação, surgiu o interesse de desenvolver uma plataforma *web* de simulação de SEP de livre acesso, voltado ao ambiente de sala de aula, com a portabilidade do interessado acessar o programa a partir de um navegador de internet (Tonini, Batista, & Rueda, 2017).

A plataforma deve permite a construção do sistema com interface inspirada em livros didáticos que tratam desta temática, e assim, realizar o cálculo do Fluxo de Potência (FP) e Curto-Circuito (CC) do SEP criado pelo usuário.

O projeto é feito juntamente com um aluno de mestrado, cujo objetivo é desenvolver os algoritmos responsáveis pela manipulação e execução dos cálculos acima, utilizando técnicas de minimização em voga na literatura para resolução de problemas de equações não lineares. Assim, o trabalho tem como escopo criar a plataforma *web* e a interface de comunicação entre o usuário e a infraestrutura formada pelos os algoritmos matemáticos. Assim será o ambiente de criação dos SEP para montagem e análise de FP e CC.

1.1 Organização do Documento

Este documento está organizado da seguinte forma: A Seção 1 contém uma introdução e objetivos deste trabalho. Nas seções 2 e 3 é descrito um resumo básico nas áreas de SEP e de engenharia de *software*, respectivamente. Nas seções 4 e 5 detalham o desenvolvimento do projeto em si, desde os requisitos da plataforma, estruturação e diagramas dos componentes, expõe a arquitetura e os *frameworks* utilizados e exibe a interface gráfica, denotando, também, os *framework* e metodologia aplicada. Já nas seções 6 e 7 retratam os resultados, a implantação e manutenção da plataforma. Na sessão 8 apresenta as considerações finais.

1.2 Justificativa

Esse trabalho permitirá aos estudantes, professores e interessados a prática o conhecimento em SEP. Na área de aprendizagem, o estudante poderá simular sistemas visto em livros e/ou sala de aula como Kagan, Oliveira, Robba, & Schmidt, (1996), Zanetta Jr (2006) e Monticelli (1983), entre outros. Já na área de ensino, os professores poderão formular questões, preparar material de aula e explicar o conteúdo através do uso da plataforma.

Em um SEP existem diversas variáveis de análise, porém a plataforma se limita em obter os presentes no cálculo do FP e CC. Devido a esta ser a principal demanda das disciplinas de SEP, o estudo e a ampliação do programa serão contínuos para que esta suporte uma gama maior de assuntos relacionados ao SEP como, por exemplo, proteção de sistemas, estabilidade, modelos desbalanceados de linha de transmissão e cálculo de transitório.

Portanto, este trabalho tem como principal foco o desenvolvimento da arquitetura de *software* e as principais funcionalidades de desenho da plataforma, além de apresentar, o elementar sobre o SEP e algumas das suas principais características como os conceitos básicos para análises de FP e CC.

1.3 Objetivos

1.3.1 Objetivo Geral

O objetivo principal deste trabalho é o desenvolvimento de uma plataforma *web,* na qual o usuário do sistema pode facilmente desenhar e analisar um SEP, estudando o FP e/ou CC por meio de uma interface gráfica simplificada em termos de quantidade de ações de interação com o usuário de forma a apresentar os principais componentes do SEP.

A plataforma pode ser dividida em processar os cálculos e métodos numéricos para resolução das equações do sistema em questão e a interface gráfica de apresentação, das entradas e saídas e a montagem dos SEP.

Atenta-se que, conforme informado anteriormente, não é de escopo deste trabalho montagem dos algoritmos de métodos numéricos, e sim a comunicação com estes.

1.3.2 Objetivos Específicos

Os objetivos específicos são: Modelagem dos principais elementos do SEP para representação no ambiente computacional (orientado a objetos); Estudo e aplicação de tecnologias no desenvolvimento de aplicações *web*, usando arquitetura de cliente-servidor; Desenvolvimento de uma plataforma web simples, visando facilidade de uso para análises básicas de Fluxo de Potência e Curto-Circuito; Desenvolvimento da *Application Programming Interface (*API) de comunicação com os algoritmos.

2 CONCEITOS BÁSICOS DO SISTEMA ELÉTRICO DE POTÊNCIA

Nesta seção apresentam-se os elementos básicos do SEP de uma maneira geral, iniciando pelas barras e linhas. Em seguida, serão abordados os problemas de FP e CC.

2.1 Barras

As barras, também denominadas de barramento, são os nós do circuito. Existem quatro grandezas básicas para uma barra e são descritas no Quadro 2.1. (Zanetta Jr, 2006)

Quadro 2.1. Grandezas elétricas de uma barra.

Grandeza	Descrição
V	Magnitude da tensão
θ	Ângulo de fase da tensão
P	Potência Ativa
Q	Potência Reativa

Fonte: Próprio autor.

As barras podem ser classificadas de acordo com as variáveis conhecidas e as que se deseja calcular. Existem três classificações para as barras: Barras de carga, barras de geração e barra de referência.

Barras de carga (PQ) – São conhecidas a potência ativa e reativa consumidas. E deve ser calculado a tensão; sua magnitude e ângulo. Estas barras não possuem potência gerada.

Barras de geração (PV) – São conhecidas as grandezas de potência ativa gerada e a magnitude da tensão. Deve-se calcular o ângulo da tensão e a potência reativa gerada (ou consumida). Normalmente são usadas em usinas hidrelétricas, onde se comanda recursos capazes de controlar a potência ativa através de controle sobre a turbina e na tensão, por meio do sistema de excitação dos geradores síncronos. (Zanetta Jr, 2006, p. 245)

Barra de referência ($V\theta$, também chamadas de *slack ou swing*) – A tensão (magnitude e ângulo de fase) é conhecida. É desconhecido a potência ativa e reativa injetada.
Além disso, a barra de referência possui duas funções: Fornecer uma referência angular para a rede (a referência da magnitude de tensão é o próprio nó terra); e garantir o equilíbrio elétrico da rede, que toda potência demandada seja consumida.

No Quadro 2.2 apresentam-se as relações com os nomes das barras e seus parâmetros.

Quadro 2.2. Relações de barras e seus parâmetros

	V	θ	P	Q
PQ	Desconhecido	Desconhecido	Conhecido	Conhecido
PV	Conhecido	Desconhecido	Conhecido	Desconhecido
$V\theta$	Conhecido	Conhecido	Desconhecido	Desconhecido

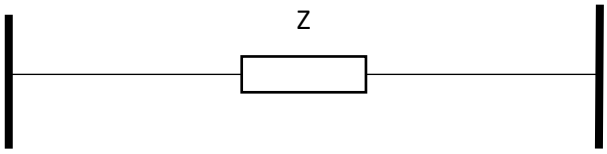
Fonte: Stevenson & Grainger, (1984) adaptado.

2.2 Linhas

Conhecidas também como ligações, ramos ou trechos. São elementos que conectam duas barras. Definem a topologia da rede e fornecem os elementos para a formação da matriz de admitância, que é usada para os cálculos de FP e CC. As linhas são caracterizadas de acordo com seus parâmetros, a saber: impedância da linha, tomada central (tap[1]) e ângulo de defasamento de transformadores, componentes de sequência simétrica ou de fase, comprimento, entre outros.
Podem ser classificadas como bipolos ou quadripolos:
Bipolos são linhas que não possuem ligações para a terra, sua representação é apresentada na Figura 2.1.

[1] tap - Os $taps$ dos transformadores são controles utilizados para variar a relação de espiras, por consequência, a tensão.

Figura 2.1. Linha bipolar.



Fonte: (Zanetta Jr, 2006), adaptado.

Seu parâmetro é a impedância série da linha Z, dada em Ω.

$$Z = R + jX \tag{2.1}$$

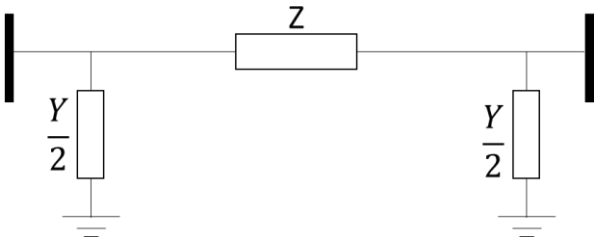
Z – Impedância série, dada em Ω.

r – Resistência série, dada em Ω.

x – Reatância série, dada em Ω.

Quadripolos são, normalmente, representados por circuitos π; possuem elementos para a terra, alterando a corrente de ligação e, consequentemente, influenciando no cálculo de FP. Sua representação é apresentada na Figura 2.2.

Figura 2.2. Ligação quadripolo.



Fonte: (Zanetta Jr, 2006), adaptado.

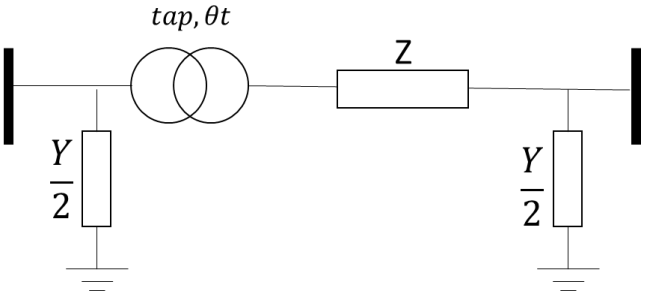
Seus parâmetros são:

Z – A impedância série da linha, dada em Ω.

Y – Admitância em derivação, dada em S.

As ligações podem ser complementadas com outros parâmetros de acordo com o estudo em questão. Outro elemento utilizado é o uso de transformador com tap variável e/ou defasador, sua representação é apresentada na Figura 2.3.

Figura 2.3. Linha com transformador.



Fonte: (Zanetta Jr, 2006), adaptado.

Seus parâmetros são:

Z – A impedância série da linha, dada em Ω ou Ω/m.

Y – Admitância em derivação

tap – Relação de transformação

θt – Ângulo de defasamento

A utilização do transformador pode ser feita com uma linha quadripolo ou bilopo.

2.3 Modelo de Carga

É necessário representar as cargas em função de sua tensão de fornecimento e sua representação da carga no sistema.

Em relação à tensão de fornecimento, existem alguns modelos para representação do comportamento da carga em função da tensão (Kagan, Oliveira, Robba, & Schmidt, 1996), dentre eles, destacam-se:

P_{cte} – Potência constante com a tensão

I_{cte} – Corrente constante com a tensão

Z_{cte}– Impedância constante com a tensão

Nesse projeto foram consideradas cargas P_{cte}., na quais considera-se que a potência da carga é independente da tensão e a corrente varia de modo inversamente proporcional à tensão aplicada.

2.3.1 Representação da carga

Além da tensão aplicada à carga, a carga pode ser representada de acordo com sua distribuição espacial ao longo da rede, dentre elas destaca-se (Kagan, Oliveira, Robba, & Schmidt, 1996): Cargas concentradas e Cargas uniformemente distribuídas.

Neste projeto foram consideradas cargas concentradas em um determinado ponto da rede.

2.3.2 Representação da Rede

A representação da rede é dada em relação a simetria de impedância de linha (rede), a tensão de alimentação e simetria de carga. O Quadro 2.3 ilustra algumas das possibilidades.

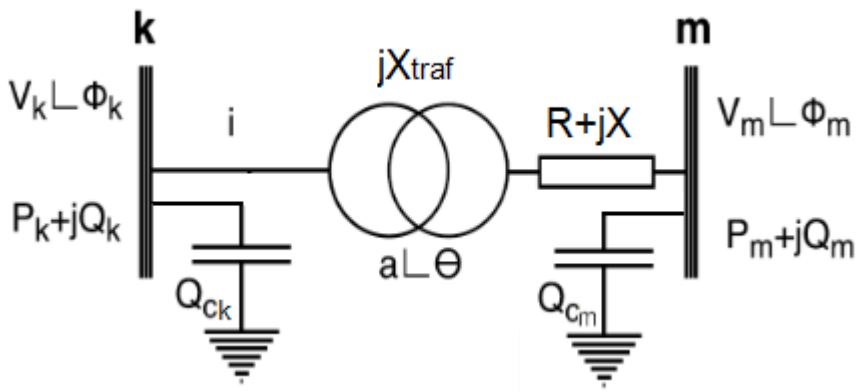
Quadro 2.3. Simetria de redes

	Alimentação	Rede	Carga
Rede Trifásica Simétrica com Carga Equilibrada	Equilibrada	Simétrica	Equilibrada
Rede Trifásica Simétrica com Carga Desequilibrada	Equilibrada	Simétrica	Desequilibrada
Rede Trifásica Assimétrica com Carga Desequilibrada	Equilibrada	Assimétrica	Desequilibrada

Fonte: (Kagan, Oliveira, Robba, & Schmidt, 1996), adaptado.

A representação de um SEP para cálculo do FP considerou o diagrama exposto na Figura 2.4, os elementos de barra e linha.

Figura 2.4. Exemplo de duas barras de um SEP para cálculo do FP.



Fonte: (Monticelli, 1983), adaptado.

O cálculo do FP considera o equacionamento de 2 subsistemas, onde o primeiro trata das barras de geração e carga, para a potência ativa, e apenas carga, para reativa:

$$P_k^{esp} - V_k \sum_{m\epsilon k} V_m(G_{km}(a,\Theta)cos\Phi_{km} + B_{km}(a,\Theta)sen\Phi_{km}) = 0 \tag{2.2}$$

$$Q_k^{esp} - V_k \sum_{m\epsilon k} V_m(G_{km}(a,\Theta)sen\Phi_{km} + B_{km}(a,\Theta)cos\Phi_{km}) + Q_{c_k} = 0 \tag{2.3}$$

Onde,

P_k^{esp} – Potência ativa.

Q_k^{esp} – Potência reativa.

V_k – Tensão da barra k.

V_m – Tensão da barra m.

G_{km} –Condutância na linha (entre as barras k e m).

Φ_{km} – Diferença entre os ângulos de defasamento das tensões das barras k e m

B_{km} – Susceptância na linha (entre as barras k e m).

Q_{c_k} – Carga shunt (equivalente a um banco de capacitores ou banco de indutores).

Enquanto o segundo equacionamento engloba a potência ativa da barra de referência e reativa, para esta e de geração.

$$P_k = V_k \sum_{m\epsilon k} V_m(G_{km}(a,\Theta)cos\Phi_{km} + B_{km}(a,\Theta)sen\Phi_{km}) \tag{2.4}$$

$$Q_k = V_k \sum_{m\epsilon k} V_m(G_{km}(a,\Theta)sen\Phi_{km} + B_{km}(a,\Theta)cos\Phi_{km}) - Q_{c_k} \tag{2.5}$$

2.3.3 Métodos Iterativos

Uma vez definido os parâmetros das barras, das linhas, modelos de carga e da rede, é necessário utilizar algum método iterativo para obter a solução de equações não lineares.

Existem alguns métodos sugeridos pela literatura (Zanetta Jr, 2006), como o método iterativo de Gauss-Seidel e o método Iterativo de Newton-Raphson. E adaptações desses métodos.

Não é de escopo deste trabalho detalhar métodos utilizados para a solução do FP, sendo sugerido o artigo (Tonini, Batista, & Rueda, Simulador Computacional Para

Proteção Digital De Sistemas Elétricos De Potência, 2017), no qual são mais detalhados os algoritmos utilizados, através de fluxogramas e ilustrações.

2.4 Fluxo de Potência

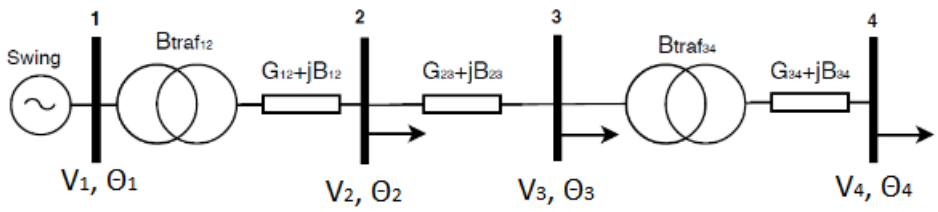
O estudo do FP é de grande relevância no estudo do comportamento de um determinado sistema diante de possíveis mudanças de cenário, seja com o atendimento de uma nova indústria, grupo residencial ou comercial.
A análise do fluxo de potência tem como objetivo a simulação de uma determinada rede e obtenção de tensões e potência, para ter uma constatação do atendimento quanto aos níveis de tensão, limites de carregamento e perdas de potência.
Em circuitos elétricos em regime permanente, as impedâncias são fixas e conhecidas, o que forma um sistema linear de equações. Enquanto para resolver um problema de fluxo de potência a formulação é um pouco diferente, pois considera a contribuição de potência de cada barra, surgindo assim equações com características de não linearidade. (Zanetta Jr, 2006)

Para representar os parâmetros do FP é necessário modelar os elementos a fim de extrair aspectos relevantes que influenciam diretamente com o cálculo. Assim, deve-se modelar as cargas de acordo com a sua representação de carga e a topologia da rede.

2.5 Curto Circuito

Para especificar equipamentos de um sistema elétrico, é relevante determinar a corrente de CC, pois, através dela, é possível ajustar os sistemas de proteção para operar de maneira a evitar problemas elétricos, esforços mecânico e deformações de materiais. Como os sistemas de proteção estão sujeitos a falhas, é necessário que os elementos da rede sejam planejados para resistir as elevadas correntes, provenientes do CC, até que um dispositivo do sistema de proteção atue, eliminando a falta ou parte dela. (Zanetta Jr, 2006)
A representação de um SEP para cálculo do CC considera o diagrama exposto na Figura 2.5, os elementos de barra e linha.

Figura 2.5. Exemplo de quatro barras de um SEP para cálculo do CC.



Fonte: (Tonini, Batista, Rueda, & Bastos, 2018)

Diante de um sistema trifásico, conhecendo condições de pré-falta, divide-se os CC de acordo a ligação que provocou a falta, podendo ser fase-terra, bifásico ou trifásico sendo esses dois últimos, com ou sem o terra. As faltas são provocadas nas barras ou nas linhas do sistema.

O objetivo do cálculo do CC é obter a corrente de falta, as tensões pós falta nas barras e as correntes de pós falta nas linhas.

Estes parâmetros são obtidos conforme as equações de 2.6 a 2.8, para falta simétrica, e dependem da obtenção da matriz de impedância, Equação 2.9. (Tonini, Batista, Rueda, & Bastos, 2018)

$$I_f = \frac{V_f}{Z_{kk} + Z_g} \tag{2.6}$$

$$V_{m_{abc}} = V_f \left(1 - \frac{Z_{mk}}{Z_{kk} + Z_g}\right) V_f \llcorner \varphi_f + \theta_{m_{abc}} \tag{2.7}$$

$$\mathrm{I_{km}} = \frac{V_k - V_m}{jX_{km}} \tag{2.8}$$

$$M_{imp_{km}} = \frac{adj\left(M_{adm_{km}}\right)^t}{|M_{adm_{km}}|} \tag{2.9}$$

Onde,

I_f – Corrente de falta.

V_f – Tensão de falta.

Z_{kk} – Impedância interna da barra k.

Z_g – Impedância à terra.

$V_{m_{abc}}$ – Matriz de magnitude das tensões pré falta.

Z_{mk} – Impedância série entre na linha entre as barras m e k.

φ_f – Ângulo de defasamento da tensão de falta.

$\theta_{m_{abc}}$ – Matriz do ângulo de defasamento das tensões pré falta.

$\mathrm{I_{km}}$- Corrente pré falta na linha entre as barras m e k.

V_k – Tensão pré falta na barra k.

V_m – Tensão pré falta na barra m.

$M_{imp_{km}}$ – Matriz de impedância do sistema na posição da linha entre as barras k e m.

$M_{adm_{km}}$ – Matriz de admitância do sistema na posição da linha entre as barras k e m.

As tensões (módulo e fase) pré falta utilizadas nas equações (2.6 a 2.8) e a matriz de admitância provêm do cálculo do FP.

Assim como o cálculo do FP, a resolução do CC não é de escopo deste trabalho elaborar a maneira dos algoritmos para realização do CC. Para maiores detalhes dessa modelagem, como as equações para faltas assimétricas, é recomendado os livros do (Kagan, Oliveira, Robba, & Schmidt, 1996), (Zanetta Jr, 2006) e até mesmo o artigo (Tonini, Batista, Rueda, & Bastos, Online Platform for learning of Electrical Power Systems, 2018).

3 ENGENHARIA DE *SOFTWARE*

No âmbito do desenvolvimento de *software,* foram adotadas algumas das práticas de engenharia de *software* com o objetivo de melhorar a qualidade do produto. Para isso, foram usados alguns métodos, técnicas e ferramentas que oferecem suporte ao desenvolvimento, que serão citadas no decorrer desta seção.

O ponto de partida para iniciar o processo de desenvolvimento é dividir o projeto em algumas etapas: Análise e Especificação de requisitos, Projeto, Implementação. (Barcellos, 2018, p. 9). Existem outras etapas sugeridas pela literatura que varia de projeto a projeto. Nesse trabalho, foram adotadas as etapas citadas anteriormente.

3.1 Requisitos

A etapa inicial no desenvolvimento do *software* visa estabelecer quais são os requisitos que este deve atender, visando os envolvidos e suas necessidades em relação ao *software*.

Segundo Barcellos (2018, p. 18) “Requisitos são funcionalidades que o sistema deve agregar e as restrições que devem realizadas".

Os requisitos podem ser divididos em três tipos: Requisitos Funcionais, Requisitos Não Funcionais e Regras de Negócio.

Requisitos Funcionais: são aqueles que estão relacionados com as funções contidas no sistema estabelecendo seu comportamento em determinadas situações.

Requisitos Não Funcionais: são aqueles que descrevem determinadas restrições de operação mediante o acesso do *software*, uso de recursos e apresentam aspectos ao sistema como um todo e não a funcionalidades particulares. Comumente, são restrições que levam em conta questões relacionadas a: segurança, portabilidade, desempenho, usabilidade e interoperabilidade.

Regras de Negócio: são regras e/ou restrições específicas as quais o *software* deve cumprir. Servem para complementar os requisitos funcionais. (Barcellos, 2018, p. 18)

O Quadro 3.1 apresenta exemplos dos requisitos citados acima.

Quadro 3.1. Exemplos de requisitos.

Requisito	Exemplo
Funcional	O sistema deve permitir inserção de barras ao sistema elétrico de potência, informando a identificação da barra, nome, tipo, tensão inicial, ângulo de fase, potência ativa, potência reativa.
Regra de negócio	O tipo de barra deve ser: PQ, PV ou $V\theta$.
Não funcional	O sistema deve ser de acesso web.

Fonte: Próprio autor.

A etapa de requisitos ainda pode ser dividida em duas atividades: Levantamento de requisitos e Especificação dos requisitos.

3.1.1 Levantamento de Requisitos

A atividade de levantamento de requisitos tem enfoque em uma perspectiva do cliente e usuário do sistema. Visa investigar quais são as funcionalidades que o sistema deve apresentar ao usuário e as restrições que devem ser satisfeitas. São usadas técnicas de entrevistas, questionários, prototipação, investigação de documentos e literatura, observação de usuários, dinâmicas de grupos, softwares similares entre outras.
Nessa atividade, devem ser considerados alguns aspectos:
Domínio da aplicação: Conhecimento da área de atuação da aplicação desenvolvida.
Problema: Detalhes do problema em questão e como o sistema contribui para a resolução do problema.
Negócio: Como o sistema impacta no âmbito dos usuários e como apoia nos objetivos dos envolvidos no qual deseja atingir.
Necessidades e restrições dos interessados: Procura entender as exigências e demandas do usuário para realizar alguma funcionalidade de uns dos interessados do sistema. Visa entender como é o fluxo de trabalho dos usuários e como são conduzidos. (Barcellos, 2018, p. 21)

3.1.2 Análise de Requisitos

Após estabelecer os requisitos na atividade de levantamento de requisitos, é necessário focar na estrutura interna do sistema. Essa atividade procura definir como e o que o sistema deve ter internamente para atender aos requisitos levantados.

Essa atividade tem como objetivo construir modelos. Um modelo pode ser considerado com uma abstração de algo do mundo real selecionando aspectos, características e métodos que são pertinentes para a finalidade do sistema desenvolvido.

Os modelos dão perspectiva estrutural do sistema e tem como objetivo prover informações de uma visão estática das quais o sistema trata. Normalmente tem como resultado o modelo de classe. (Barcellos, 2018, p. 21)

3.2 Projeto e Arquitetura de *Software*

Na etapa de projeto, procura-se definir a arquitetura de *software*, a plataforma de implementação, o armazenamento dos dados, a interface e detalhe dos componentes.

A arquitetura de *software* está relacionada com a descrição dos relacionamentos e interfaces entre os elementos que compõem o *software*. Existem algumas arquiteturas típicas que são usadas para resolver determinados problemas similares, dentre essas, destacam-se: Divisão de Camadas, Particionamento, Cliente-Servidor, entre outras.

Muitas vezes, um projeto usa combinações de arquitetura para atender o problema em questão (Barcellos, 2018, p. 119).

É nesta etapa que são informadas as tecnologias utilizadas para implantação e operação do sistema.

3.2.1 Cliente Servidor

A arquitetura Cliente/Servidor possui vários aspectos relevantes para sua definição. Sua concepção pode ser resumida em uma estrutura onde existem interações entre um ou mais clientes com um ou mais servidores executando determinadas tarefas. (NETSOFT, p. 1)

A ideia dessa arquitetura é separar de maneira lógica as operações de clientes dos servidores, de modo que o cliente e o servidor possam até coexistir ou não em uma mesma máquina, desde que tenham suas representações claras e definidas.

Para melhor esclarecimento, é possível definir características básicas sobre cliente e servidor:

Cliente, conhecido como "*front-end*" (traduzido: parte da frente) é um processo que comunica com o usuário do sistema por meio de uma interface, que pode ser gráfica ou não, na qual permite realizar pedidos de determinadas tarefas pré-definidas como consulta de dados, cálculos matemáticos entre outros. O termo "cliente" não é o humano em si, e sim o meio no qual se comunica com um servidor, no qual existem determinadas abstrações para o usuário que pode ser um programa ou um humano. O cliente é um programa que se comunica com o servidor realizando determinadas requisições.

Já, o servidor, também conhecido como "*back-end*" (traduzido: parte de trás) é o meio no qual recebe e responde às solicitações de clientes. Ele é responsável por tratar cada chamada de cliente e possui uma execução contínua

3.2.2 Divisão de camadas

A arquitetura de *software* baseada em divisão de camadas procura dividir, de maneira sistemática, a organização em camadas típicas. São elas: Interface com o Usuário, Domínio do Problema ou Modelos e Gerência de Dados.

A camada de domínio do problema, conhecida também como regra de negócios ou modelos, engloba o conjunto de classes que irão fazer a lógica do sistema, e todas as outras camadas são dependentes desta.

A camada de interface com o usuário é responsável de exibição dos dados e controle das interações, recebendo solicitações de eventos do usuário, sejam de *mouse*, teclado ou toque (arrastar um elemento, cliques) e solicitar da camada de lógica as respectivas solicitações.

A camada de gerência de dados é responsável por persistir os dados da aplicação de maneira a evitar inconsistência de dados.

3.3 Implementação

Uma vez definida a arquitetura de software, bem como as tecnologias utilizadas, a etapa de implementação consiste em codificar o software. Essa etapa contém o ambiente de desenvolvimento com ferramentas de codificar.

Existem os Ambientes de Desenvolvimento Integrado, (em inglês, como é usado pelos desenvolvedores de sistema, denota-se IDE, *Integrated Development Environment*). Esses ambientes oferecem ferramentas de correção de sintaxe, depuração de códigos, compilação, entre outras, e variam de acordo com as IDE.

3.4 Testes

O sistema deve ser testado continuamente a fim de descobrir possíveis defeitos quanto antes aconteçam. Em essência, há dois tipos de testes: Testes funcionais ou Caixa preta e os testes estruturais ou caixa branca. (Barcellos, 2018, p. 149)

3.4.1 Testes Estruturais

São conhecidos também como testes caixa branca e tem perspectiva nas estruturas de controle de um determinado módulo, como, por exemplo, laços de repetição ou condicionais.

Normalmente esses testes são realizados em paralelo ao desenvolvimento do software na etapa de implementação. (Barcellos, 2018, p. 153)

3.4.2 Testes Funcionais

Também conhecidos como testes caixa preta, são testes que enfocam as funcionalidades do sistema, e visa analisar se as funções e regras de negócio foram implementadas corretamente.

Existem diretrizes de estratégias para realização de testes, as mais comuns são:

Teste de Unidade: tem como propósito dividir o sistema em unidades e indicar erros de lógicas na implementação de cada unidade.

Teste de Integração: Tem como foco encontrar erros associados às ligações entre sistemas na etapa de integração de módulos.

Teste de Sistema: Identifica erros relacionados aos requisitos funcionais e a propriedades de comportamento dos requisitos não funcionais que diverge dos acordos da especificação. (Barcellos, 2018, p. 147)

4 MODELAGEM DO *SOFTWARE*

Nessa sessão serão apresentados o levantamento de requisitos e a modelagem dos componentes utilizados no SEP.

4.1 Requisitos

O levantamento de requisito foi produzido através de levantamento da literatura e observações de outros softwares existentes, a saber foram consultados os livros de:

(Monticelli, 1983), (Zanetta Jr, 2006), (Stevenson & Grainger, 1984), (Stevenson & Grainger, 1994), (Kagan, Oliveira, Robba, & Schmidt, 1996)

E pelos softwares:

Power Tools for Windows (PTW), SimPowerSystems (módulo do Simulink/Matlab®)

No projeto foram considerados apenas parâmetros mais comuns para o cálculo de FP e CC.

A lista dos requisitos funcionais é listada no Quadro 4.1.

Quadro 4.1. Requisitos Funcionais

Identificador	Descrição	Requisitos Relacionados
RF01	O sistema deve permitir inserir uma barra, informando: Identificação da barra, Tipo, Nome, Tensão, Angulo, Potência Ativa Gerada, Potência Reativa Gerada, Potência Reativa Mínima, Potência Reativa Máxima, Potência Ativa Mínima, Potência Ativa Máxima, Potência Ativa de Carga, Potência Reativa de Carga, Potência Reativa Shunt, Reatância interna.	-
RF02	O sistema deve permitir inserir uma linha, informando: Barra de origem, Barra de destino, Identificação da Linha, Resistência Série, Impedância Série	RF01
RF03	O sistema deve permitir inserir um Transformador em uma linha, deve-se informado: a linha desejada, tipo de transformador, tap, defasagem, reatância interna, impedância zero	RF02
RF04	O sistema deve permitir inserir uma falta em uma barra, Informando: a barra, tipo de falta, impedância de aterramento, impedância de sequência zero da linha	RF01

RF05	O sistema deve permitir inserir uma falta em uma linha, informando: a linha, barra a montante, barra a jusante, porcentagem da linha, tipo de falta, impedância de aterramento, impedância de sequência zero	RF02
RF06	O sistema deve permitir o cálculo do fluxo de potência informando as linhas e barras do sistema a ser calculado, gerando como resultado: Tensão, ângulo, Potência ativa e potência reativa gerada das barras e as potências ativa e reativa do fluxo.	RF01, RF02
RF07	O sistema deve permitir o cálculo do curto circuito, informando as barras e linhas do sistema e barra ou linha que contém a falta. Como resultado deve ser informado:	RF04, RF05

Fonte: Próprio autor.

E para complementar os requisitos funcionais, o Quadro 4.2 apresenta as regras de negócio do sistema.

Quadro 4.2. Regras de Negócio

Identificador	Descrição	Requisitos Relacionados
RN01	O tipo de barra deve ser: PQ, PV ou Slack	RF01
RN02	Todas as barras devem ter: identificador, sendo um número inteiro e positivo único, nome sendo de no máximo 20 caracteres e o tipo, Potência Reativa Shunt e Reatância Interna	RF01
RN03	Barra do tipo PQ deve ser informado apenas: Potência Ativa de Carga, Potência Reativa de Carga, além das propriedades comuns entre as barras	RF01
RN04	Barra do tipo PV deve ser informado apenas: Tensão, Potência Ativa Gerada, Potência Ativa de Carga, Potência Reativa de Carga, Potência Reativa Gerada Máxima, Potência Reativa Gerada Mínima, além das propriedades comuns entre as barras	RF01
RN05	Barras do tipo slack devem ser informados apenas: Potência Ativa de Carga, Potência Reativa de Carga, Potência Ativa Gerada Mínima, Potência Ativa Gerada Máxima, Potência Reativa Gerada Mínima, Potência Reativa Gerada Máxima, além das propriedades comuns entre as barras	RF01
RN06	A identificação da linha deve ser um número inteiro positivo e único.	RF02
RN07	O cálculo do fluxo de potência só pode ser efetuado se: todas as barras estiverem conectadas e deve	RF06

	existir uma e somente uma barra slack (Barra de Referência)	
RN08	Só pode ter apenas uma falta no sistema, seja na barra ou na linha.	RF04, RN05
RN08	O cálculo do curto circuito só pode ser efetuado se: O fluxo de potência tiver sido executado e exista apenas um curto no sistema, seja na barra ou na linha	RF04, RF05, RF07

Fonte: Próprio autor.

Vale ressaltar que a determinação dessas funcionalidades foi tomada visando uma interface com informações relevantes e necessárias para os cálculos do FP e CC. Para completar são apresentados os requisitos não funcionais no Quadro 4.3.

Quadro 4.3. Requisitos não funcionais

Identificador	Descrição	Categoria
RNF01	O sistema deve ser web e acessível pela internet.	Portabilidade
RNF02	O sistema deve permitir o acesso de múltiplos usuários acessando a ferramenta de forma individual.	Portabilidade
RNF03	A criação dos elementos do SEP deve ser feita usando o conceito de arrastar e soltar, onde o usuário arraste um componente e o posicione no espaço apropriado de desenho.	Usabilidade

Fonte: Próprio autor.

4.2 Modelos Implementados

Diante da determinação dos requisitos, foi construído o modelo de classe para representar os componentes do SEP no ambiente computacional, seguindo o paradigma de orientação a objetos.

4.2.1 Barra

Baseado no levantamento de requisitos foi possível estabelecer os principais atributos do modelo de barra e são apresentados no Quadro 4.4.

Quadro 4.4. Classe de Barra

Nome do Atributo	Tipo	Identificação	Descrição	Restrição
id_barra	Inteiro	Item	Identificação única	inteiro positivo

tipo	enum	Tipo	Tipo de Barra	PQ, PV ou Slack
tensao	float	Tensão	Tensão da Barra	0.9 a 1.2
angulo	float	Ângulo	Ângulo de fase da tensão	-360 a 360
pGerada	float	Potência Ativa Gerada	Potência ativa gerada (fornecida) da Barra	0 a 10
qGerada	float	Potência Reativa Gerada	Potência reativa gerada (fornecida) da Barra	0 a 10
pCarga	float	Potência Ativa Carga	Potência ativa consumida da Barra	0 a 10
qCarga	float	Potência Reativa Carga	Potência reativa consumida da Barra	0 a 10
pGeradaMin	float	Potência Ativada Gerada Mínima	Limite de potência ativa gerada mínima	0 a 10
PGeradaMax	float	Potência Ativada Gerada Máxima	Limite de potência ativa máxima	0 a 10
qGeradaMin	float	Potência reativada Gerada Mínima	Limite de potência reativa gerada mínima	0 a 10
qGeradaMax	float	Potência reativada Gerada Máxima	Limite de potência reativa máxima	0 a 10
qShunt	float	Potência Reativa Shunt	Potência reativa em paralelo com a barra	0 a 10
X	float	Reatância Interna	Reatância Interna	0 a 10

Fonte: Próprio autor.

4.2.2 Linha

Baseado no levantamento de requisitos, foi possível estabelecer os principais atributos do modelo de linha e são apresentados no Quadro 4.5.

Quadro 4.5. Classe de Linha

Nome do Atributo	Tipo	Identificação	Descrição	Restrição
de	Barra	Barra a montante ou barra de origem	Indica a barra de origem	-
para	Barra	Barra a jusante ou	Indica a barra de destino	-

		barra de destino		
R	float	Resistência série	Resistência série da linha	0 a 10
X	float	reatância Série	Reatância série da linha	-10 a 10
Transformador	Transformador	Transformador da linha	Indica informações do transformador da linha	1

Fonte: Próprio autor.

4.2.3 Transformador

Baseado no levantamento de requisitos, foi possível estabelecer os principais atributos do modelo de transformador e são apresentados no Quadro 4.6.

Quadro 4.6. Atributos do transformador

Nome do Atributo	Tipo	Identificação	Restrição	Unidade
tap	float	Tap do transformador	0 a 2	pu
A	float	Ângulo de defasagem	-360 a 360	graus
tipo	enum	Tipo do transformador	Delta Estrela, Delta-Delta ou Estrela-Estrela	-
x	float	Reatância interna	0 a 10	ohm
z0	float	Impedância zero	0 a 10	ohm

Fonte: Próprio autor.

4.2.4 Falta

Baseado no levantamento de requisitos, foi possível estabelecer os principais atributos do modelo de falta e são apresentados no Quadro 4.7.

Quadro 4.7. Atributos da falta

Nome do Atributo	Tipo	Identificação	Restrição	Unidade
xg	float	Reatância de aterramento	positivo	ohm
x0	float	Reatância de sequência zero	positivo	ohm
xf	float	Reatância de falta	positivo	ohm
local	enum	Local da falta	barra ou linha	-
tipo	enum	Tipo de falta	Monofasica, Bifasica ou Trifasica	-
porcentagem	int	Distância da falta em porcentagem	0 a 100	%
id_componente	int	Identificação do componente que contem a falta	positivo	-

Fonte: Próprio autor.

4.2.5 Fluxo de Potência

Baseado no levantamento de requisitos, foi possível estabelecer os principais atributos do modelo de fluxo de potência e são apresentados no Quadro 4.8.

Quadro 4.8. Atributos do fluxo de potência

Nome do Atributo	Tipo	Identificação	Restrição	Unidade
de	Barra	Barra a montante ou barra de origem	1	-
para	Barra	Barra a jusante ou barra de destino	1	-
pFluxo	float	Potência ativa do fluxo	número real	pu
qFluxo	float	Potência reativa do fluxo	número real	pu

Fonte: Próprio autor.

Nesse, as barras contêm os parâmetros atualizados com os novos valores de tensão, ângulo e potências calculadas no FP.

4.2.6 Curto Circuito

Baseado no levantamento de requisitos, foi possível estabelecer os principais atributos do modelo de CC e são apresentados no Quadro 4.9.

Quadro 4.9. Atributos da classe CurtoCircuito

Nome do Atributo	Tipo	Identificação	Restrição	Unidade
if_m	float	Módulo da corrente de falta	real	pu
if_f	float	Fase da corrente de falta	real	graus
tensoes	TensaoPosFalta	Tensões pós falta das barras	-	-
correntes	CorrenteFalta	Correntes de faltas das linhas	-	-

Fonte: Próprio autor.

O CC possui classes auxiliares para complementar seus atributos e são apresentadas no Quadro 4.10.

Quadro 4.10. Atributos da classe TensaoPosFalta

Nome do Atributo	Tipo	Identificação	Restrição	Unidade
barra	Barra	Identificação da barra	-	
va_m	float	Módulo da tensão A na barra	positivo	pu
va_f	float	Fase da tensão A na barra	real	graus
vb_m	float	Módulo da tensão B na barra	positivo	pu
vb_f	float	Fase da tensão B na barra	real	graus
vc_m	float	Módulo da tensão C na barra	positivo	pu
vc_f	float	Fase da tensão C na barra	real	graus
vn_m	float	Módulo da tensão de neutro na barra	positivo	pu
vn_f	float	Fase da tensão de neutro na barra	real	graus

Fonte: Próprio autor.

Quadro 4.11. Atributos da classe CorrenteFalta

Nome do Atributo	Tipo	Identificação	Restrição	Unidade
linha	Linha	Identificação da linha	-	-
if_m	float	Módulo da corrente de falta	real	pu
if_f	float	Fase da corrente de falta	real	graus

Fonte: Próprio autor.

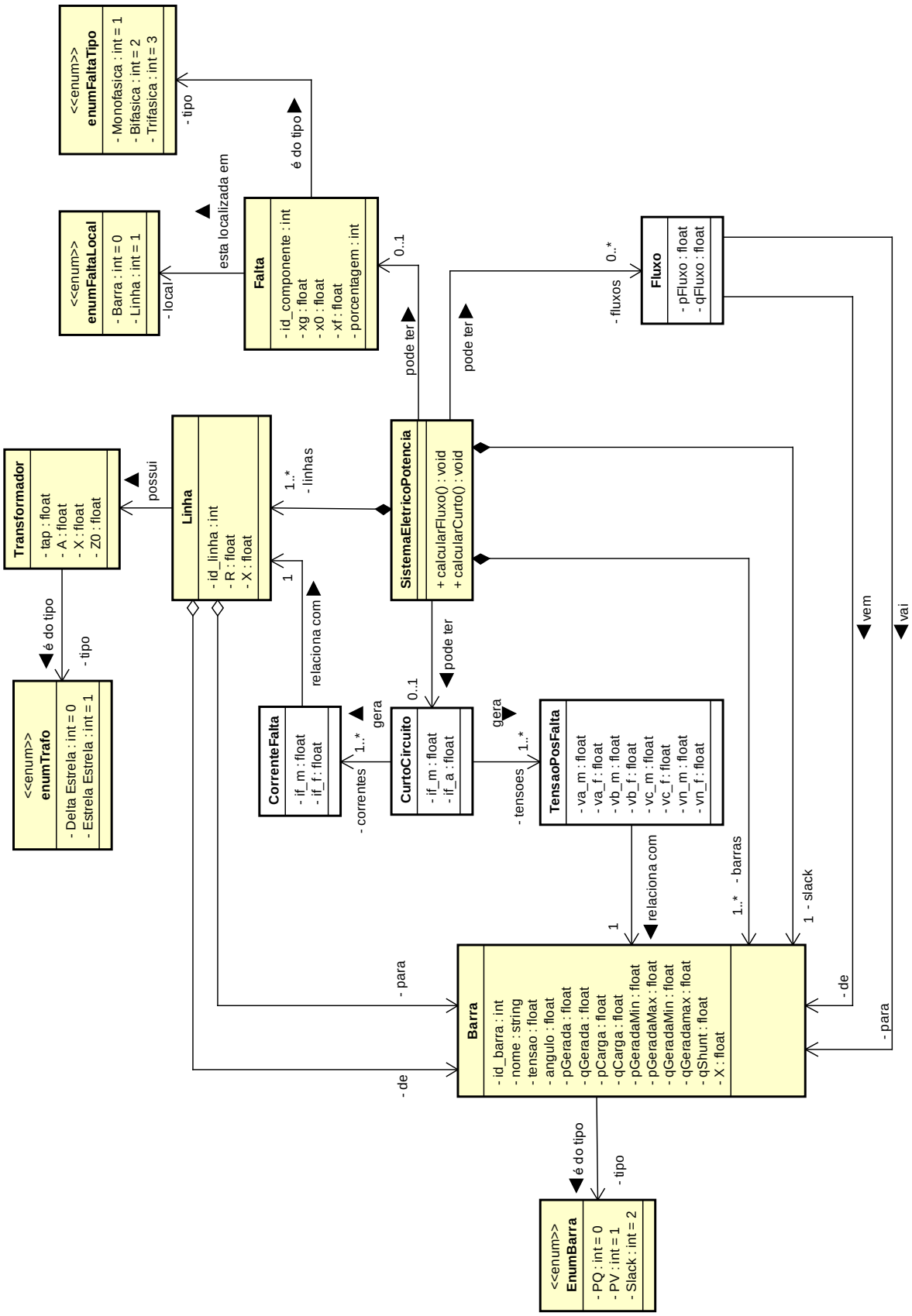
4.3 Diagrama de Classe

O diagrama de classe é uns dos diagramas presente na Linguagem de Modelagem Unificada (do inglês, UML, *Unified Modeling Language*) e enfoca na estrutura estática

do *software*. De maneira geral, ela apoia a elaboração dos modelos de classe e permite listar atributos, métodos, classes e seus relacionamentos. A UML tem como objetivo unificar projetos de *software* independente da linguagem, domínio do problema ou características particulares. (Barcellos, 2018, p. 48)

Neste projeto foi elaborado o diagrama de classes do SEP utilizando a ferramenta *Astah (*Astah, 2018*)*. O diagrama é apresentado na Figura 4.1.

Figura 4.1. Diagrama de classes do Sistema Elétrico de Potência modelado.



Fonte: Próprio autor.

Com esse diagrama de classe é possível identificar os relacionamentos entre as classes e os principais atributos de cada classe identificada na seção anterior.

5 IMPLEMENTAÇÃO

Esta seção apresenta as etapas da fase do projeto e da implementação do sistema.

5.1 Cálculos matemáticos

A implementação dos cálculos matemáticos de FP e CC, desenvolvidas por Luiz G. R. Tonini em (Tonini, Batista, & Rueda, Simulador Computacional Para Proteção Digital De Sistemas Elétricos De Potência, 2017) e (Tonini, Batista, Rueda, & Bastos, Online Platform for learning of Electrical Power Systems, 2018), foram produzidas usando o conceito de programação linear.

Para realizar o cálculo do FP foi utilizado o AMPL que é uma linguagem de programação matemática apropriada para o uso de função objetivo. AMPL é a abreviação das iniciais de *A Mathematical Programming Language* (em tradução para o português Uma Linguagem de Programação Matemática) desenvolvida nos Laboratórios Bell por Robert Fourer, David M. Gay e Brian W. Kernighan com o objetivo de ajudar pessoas a comunicar modelos de otimização para sistemas de computadores com o uso de formulações algébricas (AMPL Optimization Inc., 2017). A linguagem tem suporte nas plataformas de desenvolvimento: Linux, MacOs e Windows. Além disso, o site do AMPL oferece livros com tutoriais introdutórios e disponibiliza os scripts de exemplos básicos.

Para realizar o cálculo do CC foi utilizado a linguagem de programação C++ que pode ser compilada tanto para sistemas operacionais baseados em Unix ou sistemas Windows.

Com base nesse cenário e pela restrição de que o projeto deve ser *web* e acessível pela internet, foi adotado a arquitetura de cliente/servidor que será melhor detalhada nas seções seguintes.

5.2 Arquitetura

Para dividir responsabilidades o sistema foi separado em dois níveis, sendo eles: Cliente e Servidor.

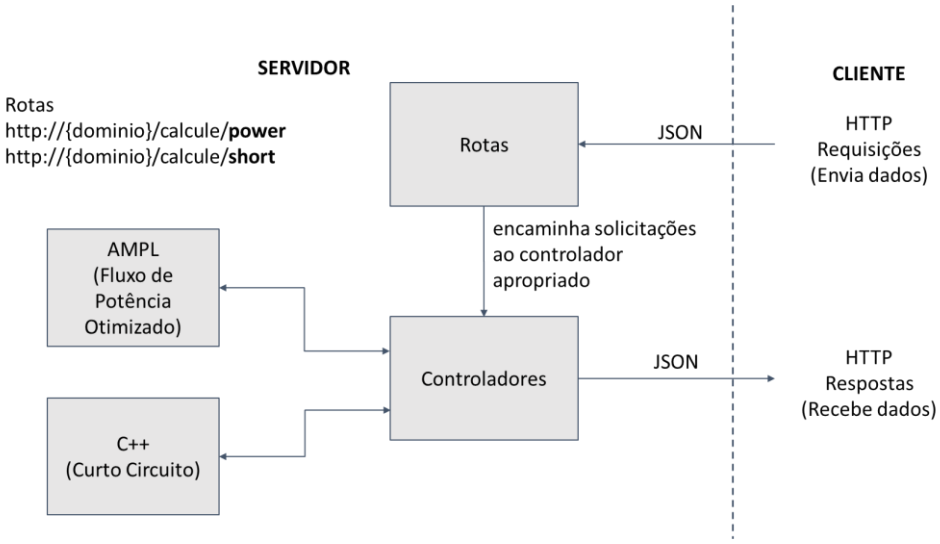
A Figura 5.1 apresenta a visão geral da arquitetura com suas respectivas bibliotecas, *frameworks* e linguagens utilizadas, os quais serão explicados nas próximas seções.

Figura 5.1. Visão geral da arquitetura.

Fonte: Próprio autor.

O fluxo de trabalho com as trocas de mensagens entre os níveis é exibido na Figura 5.2.

Figura 5.2. Fluxo de trabalho entre os níveis de cliente e servidor.



Fonte: Próprio autor.

O funcionamento da plataforma consiste em desenhar o diagrama SEP através da interface web (cliente) do qual pode solicitar cálculos de FP ou CC, enviando dados, no formato JSON[2], através das rotas do servidor. O servidor, por sua vez, recebe as

[2] JSON - JavaScript Object Notation (JSON), traduzido, Notação de Objetos JavaScript é um formato padrão de troca de dados entre linguagens de programação e é baseado em atribuições do tipo chave-valor.

requisições do cliente e encaminha ao controlador apropriado, que responde enviando informações e o resultado do cálculo.

5.3 Servidor

O servidor é uma aplicação usando o *Hypertext Transfer Protocol*, sigla HTTP (traduzido: Protocolo de Transferência de Hipertexto), utilizando a linguagem de programação Python, na qual responderá requisições específicas retornando as informações apropriadas. Em outras palavras, é uma *Application Programming Interface* (API) (traduzido para português: Interface de Programação de Aplicações) que é um conjunto de rotas específicas e padrões de programação e no propósito deste sistema, serve para comunicar os dados do cliente com os cálculos matemáticos desenvolvidos em outras linguagens.

Para o desenvolvimento dessa API foi utilizado o *framework Flask*.

O *Flask* é um *framework* minimalista desenvolvido em Python que segue alguns conceitos fundamentais, dentre eles, para a finalidade desse projeto, destaca-se:

- Baseado na biblioteca *WerkZeug* que é utilizada em desenvolvimento de aplicativos *Web Server Gateway Interface* (*WSGI)*, possui especificação de como deve ser a interface entre aplicativos Python e um *web server*. Além disso, essa biblioteca possui a implementação básica deste padrão para interceptar requisições e lidar com respostas, controle de cache, *cookies*, *status HTTP*, roteamento de *urls* e uma ferramenta de depuração.

A principal função do servidor é executar os cálculos matemáticos solicitados pelo cliente.

Essa API possui rotas específicas que retornam informações requisitadas pelo Cliente. As duas rotas desenvolvidas são: cálculo do Fluxo de Potência e cálculo do Curto-Circuito. Para solicitar um cálculo, deve encaminhar para rota correspondente.

Ambas as rotas necessitam de um cabeçalho, (conhecido como *headers*) com o campo *Content-Type* preenchido com o valor de *application/json*, para que o servidor possa receber os dados no padrão *JSON* podendo interpretar e responder com o resultado do cálculo.

Para solicitar o cálculo do FP o cliente deve enviar para a rota

$$http://\{dominio\}/calcule/power$$

um dicionário de dados (ou objeto) contendo duas chaves: linhas e barras. Essas chaves possuem a lista de linhas e de barras do sistema a ser calculado com os parâmetros indicados no Quadro 5.1.

Quadro 5.1. Parâmetros necessários para cálculo do FP

Linhas	Barras
Identificação da linha	Identificação da barra
Identificação da barra a montante	Tipo
Identificação da barra a jusante	Nome
Resistência interna	Tensão
Reatância interna	Ângulo
Tap	Potência Ativa Gerada
Defasagem do transformador	Potência Reativa Gerada
Tipo de Transformador	Potência Reativa Mínima
Reatância zero	Potência Reativa Máxima
Reatância interna do Transformador	Potência Ativa Carga
	Potência Reativa Carga
	Potência Ativa Gerada Mínima
	Potência Ativa Gerada Máxima

Fonte: Próprio autor.

Para solicitar o cálculo do CC o cliente deve enviar para a rota

$$http://\{dominio\}/calcule/short$$

o mesmo dicionário de dados (ou objeto) descrito no Quadro 5.1 com uma chave adicional que é descrita no Quadro 5.2.

Quadro 5.2. Parâmetros do cálculo do CC.

Local da Falta	Identificação da falta	Barra a montante	Barra a jusante	Porcentagem da falta na linha	Tipo de falta	Impedância de aterramento	Impedância de sequência zero da linha

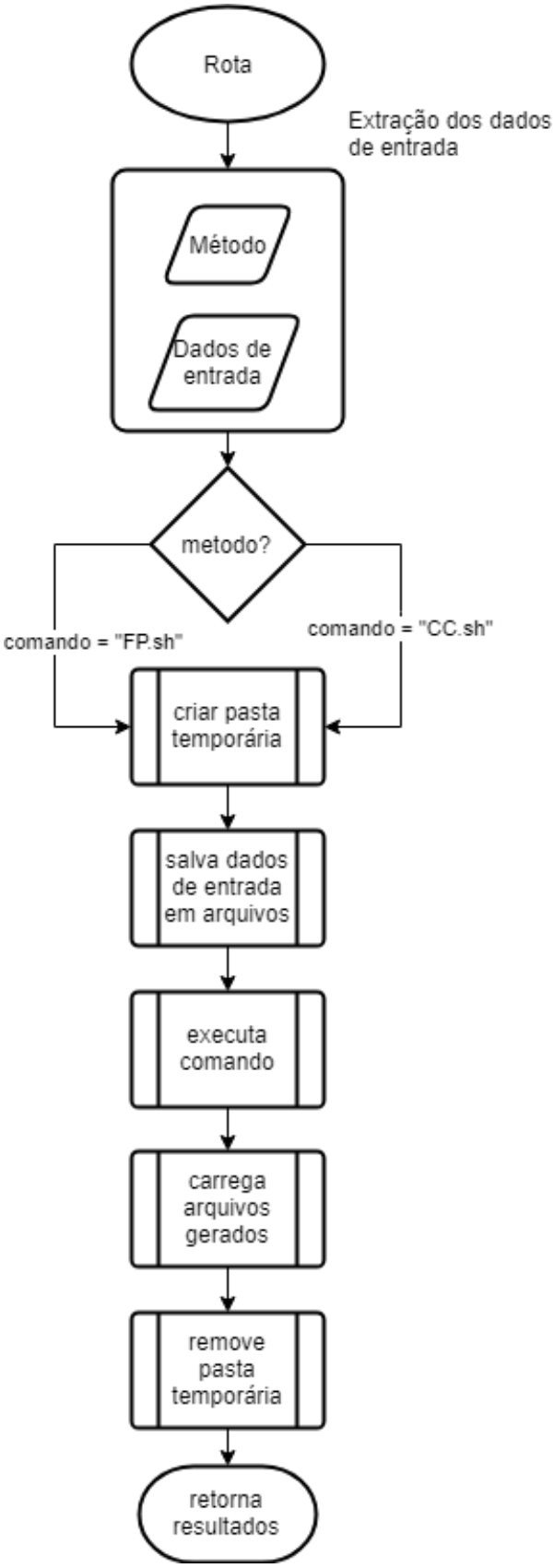
Fonte: Próprio autor.

Para executar os cálculos que são realizados em outras linguagens foi feito um algoritmo genérico para ambos os cálculos que consiste em receber os dados do cliente, criar pasta temporária, executar o método matemático, carregar arquivos

gerados, apagar pasta temporária e retornar com os dados do cálculo. O fluxograma do algoritmo é ilustrado na Figura 5.3.

Os comandos “FP.sh” e “CC.sh”, exibidos na Figura 5.3, são *scripts* feitos em *bash* (interpretador de comandos de várias distribuições do Linux) e tem como objetivo executar os comandos do *AMPL* ou do *C*++.

Figura 5.3. Fluxograma do algoritmo que solicita/recebe os cálculos.



Fonte: Próprio autor.

5.4 Cliente

Essa seção trata a construção do aplicativo do cliente.

Neste trabalho, por se tratar de uma interface simples e intuitiva, a concepção da rede é cargas equilibradas e redes simétricas, fazendo uma representação gráfica monofásica (Tonini, Batista, Rueda, & Bastos, 2018).

Para prover a estrutura foi desenvolvido uma aplicação com base nos frameworks *Angular 6* e *Bootstrap 4.* Para as interações e criação de figuras foram utilizadas as bibliotecas Interact.js e SVGjs.

Nas subseções seguintes serão explicados os conceitos básicos e uso de cada item citado acima.

5.4.1 Angular

Angular é uma plataforma e estrutura para criar aplicativos clientes em HTML[3] e TypeScript[4]. Angular é escrito em TypeScript. Ele implementa a funcionalidade principal e opcional como um conjunto de bibliotecas TypeScript para importação no desenvolvimento de um aplicativo.

Os blocos de construção básica de um aplicativo Angular são: Módulos, Componentes, Serviços e Rotas.

Módulos fornecem um contexto de compilação para os Componentes. Um aplicativo Angular é definido por um conjunto de Módulos. Um aplicativo sempre possui pelo menos um módulo raiz que permite a auto inicialização.

Componentes definem exibições e são conjuntos de elementos de tela que o Angular pode escolher entre exibir e modificar de acordo com a lógica e os dados do programa. Todo aplicativo possui pelo menos um Componente raiz.

Serviços são usados pelos Componentes e fornecem funcionalidade específica não diretamente relacionada a visualizações. Os provedores de serviços podem ser injetados em componentes como dependências, tornando o código modular, reutilizável e eficiente.

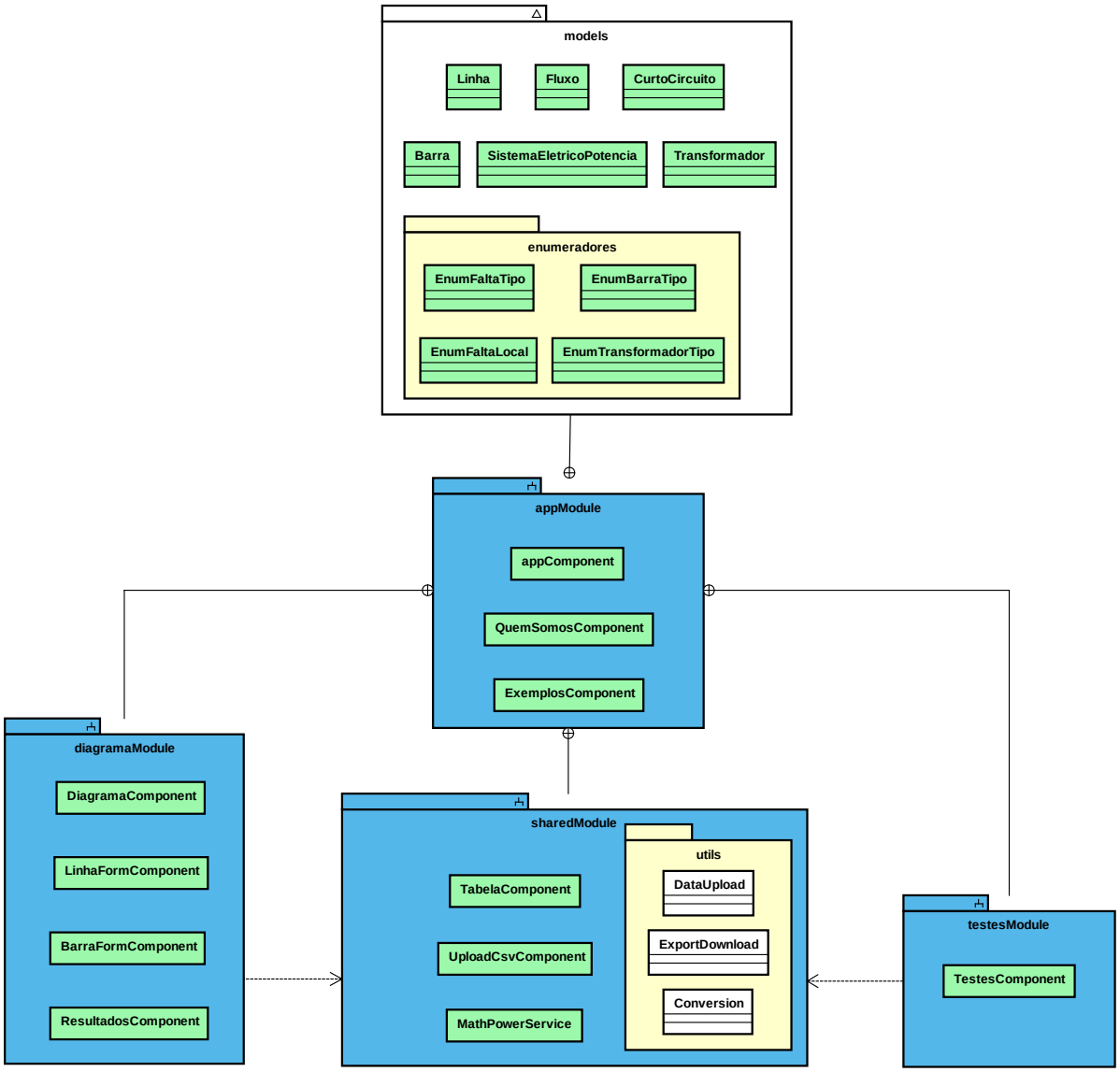
---

[3] HTML - HyperText Markup Language (HTML), (em tradução Linguagem de Marcação de Hipertexto) que é uma linguagem de marcação e é usada na criação de páginas Web.

[4] TypeScript – É uma linguagem de programação desenvolvido pela Microsoft no qual adiciona tipos e outros recursos a linguagem JavaScript (JS).

As rotas definem recursos de navegação para exibir um determinado componente. Nesse projeto foram desenvolvidos quatro módulos: *AppModule, DiagramaModule, TestesModule, SharedModule*. Cada módulo deste possui seus respectivos componentes. A Figura 5.4 apresenta o diagrama dos módulos criados no *framework* Angular.

Figura 5.4. Diagrama de Módulos do Angular.



Fonte: Próprio autor.

O *AppModule* é o módulo raiz e contém todos os outros módulos e o componente raiz *appComponent.*

O módulo *SharedModule* contém componentes que são comuns entre módulos. Além disso, contém o serviço chamado MathPowerService, responsável por fazer a

interface com a API do servidor. Também contém o pacote *utils* com funções de envio de dados, Exportação de tabelas e conversões de tipos.

O componente ExemplosComponent contém informações de exemplos de livros. Ao todo são cinco exemplos. Neste módulo é possível baixar os exemplos para testá-los ou desenhar no diagrama. O QuemSomosComponent contém informações dos autores e contribuintes da plataforma.

O módulo TestesModule contém um componente UploadComponent utilizado para submissão de arquivos em formato ".csv" (*comma-separated values*, em tradução, valores separados por vírgula) de linhas e barras para cálculo do FP ou CC em testes.

O módulo Diagrama é o principal módulo da aplicação cliente e contém os elementos necessários para criação do SEP.

Além dos quatro módulos, no cliente também possui o pacote *model* no qual possui a implementação do diagrama de classe apresentado na Figura 4.1.

O Angular usa o conceito de *Single Page Application* (SPA) (traduzido: Aplicação de Página Única) no qual aparenta uma navegação fluida e rápida pelo fato de usar o conceito de *Lazy Loading* carregamento estratégico que permite carregar somente os componentes que estão sendo visualizados (*component init*).

5.4.2 Bootstrap

Para a criação da estrutura HTML e estilos com o *Cascading Style Sheets* (CSS) (em tradução, Folhas de Estilo em Cascata), que é uma linguagem que tem o objetivo de incluir estilos (tamanho, cores, fontes, espaçamento, etc.) a páginas web, foi utilizado o *framework* Bootstrap na versão 4.1.

O Bootstrap é um conjunto de ferramentas de código aberto para desenvolvimento com HTML, CSS e JS. (Bootstrap, 2018).

Este *framework* contém um conjunto de classes em CSS e códigos em JS que ajudam na criação das páginas.

5.4.3 Bibliotecas

Para fazer a interação do *mouse* para o recurso de "arrastar e soltar" foi utilizado a biblioteca de código aberto *Interact.js* (Interact, 2018) no qual disponibiliza uma documentação seu site para sua utilização.

Para criação e posicionamento das figuras na tela foram testadas três bibliotecas: D3.js, SVG.js e RaphaelJS. Ambas as bibliotecas utilizam gráficos vetoriais escaláveis, conhecido como SVG (do inglês, Scalable Vector Graphics). A biblioteca que ofereceu maior suporte, facilidade de uso, abstração de código e documentação mais detalhada foi a biblioteca SVG.js.

A biblioteca tem como objetivo manipular os objetos SVG e fazer uma representação cartesiana. (Wout Fierens, 2018)

6 RESULTADOS

A primeira versão do programa teve o nome de Programa para Ensino de Sistemas Elétricos de Potência (PESEP) e durante desenvolvimento do projeto, passou a se chamar: Knowlenge Virtual Academy (KVA).

O resultado será exibido através de fotos instantâneas dos módulos citados na Sessão 5.4.1 e das principais funcionalidades que a ferramenta contempla atualmente. Como dito na Seção 5.4, o cliente foi desenvolvido com os *frameworks* Angular e Bootstrap, além de outras bibliotecas de JS que serão citadas a seguir. Nessa sessão será apresentado os módulos e componentes da interface gráfica do cliente.

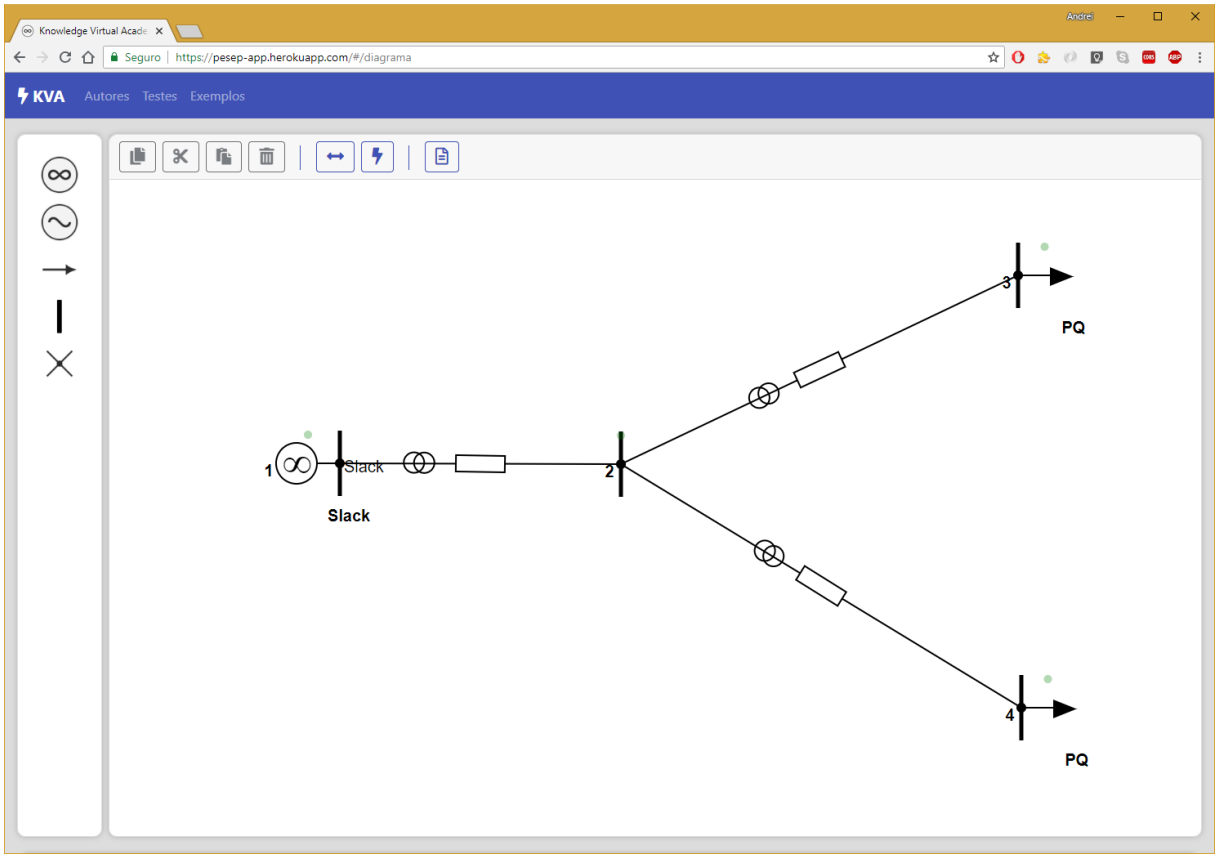
6.1 Interface gráfica

Começando pelo módulo de elaboração dos SEP, módulo Diagrama (*DiagramaModule*), a Figura 6.1 apresenta a visualização de um SEP composto por 4 barras e 3 linhas.

Ao lado esquerdo da Figura 6.1, está localizado os elementos disponíveis para arrastar e soltar no diagrama SEP desenhado. Ao meio, existem duas partes: A área de desenho onde são posicionados os elementos do SEP e a barra de ações, na parte superior a qual possui botões para determinadas ações.

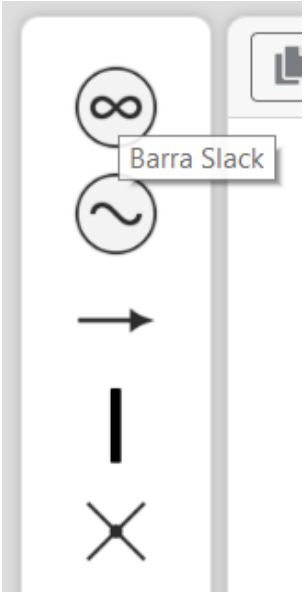
A Figura 6.2 é um recorte da área lateral. Nesta figura é apresentado os elementos do SEP que foram modelados na sessão 4.2. Ao passar o mouse em cima do elemento seu nome é exibido. O primeiro elemento é a barra *slack* sendo formada por um círculo e o símbolo do infinito ao centro. O segundo elemento é a barra de geração sendo formada por um círculo e um símbolo do acento til ao centro. O terceiro e o quarto elemento são barras de cargas, sendo a terceira uma barra com carga (como padrão 1 pu de potência ativa consumida) e a quarta é uma barra de geração com consumo zero. O quinto é o elemento de falta sendo um xis com um círculo ao centro.

Figura 6.1. Diagrama -Visualização da Interface.



Fonte: Próprio autor.

Figura 6.2. Diagrama - Componentes laterais.



Fonte: Próprio autor.

A Figura 6.3 apresenta a barra de ações. Essa barra possui grupos de três separações: Ações do diagrama, realizações de cálculos e informações do SEP atual.

Figura 6.3. Diagrama - Barra de ações.

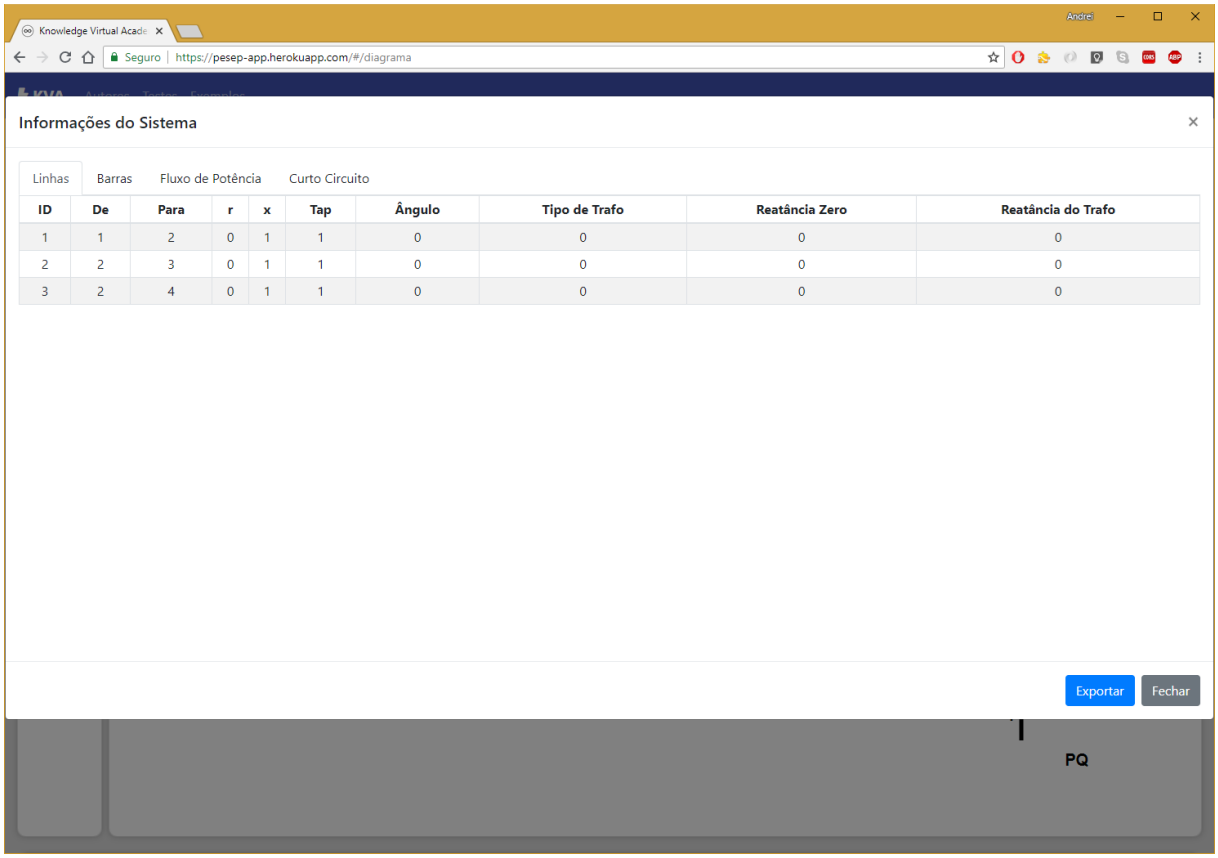
Fonte: Próprio autor.

A ações do diagrama, primeiro grupo da separação, é composta pelos botões de: Copiar, Recortar, Colar e Excluir. Para que esses botões fiquem ativados é necessário que uma ação tenha acontecido anteriormente. Para copiar, recortar ou excluir é necessário que uma ou mais barras estejam selecionadas. Para colar é necessário que uma ou mais barras tenham sido copiadas ou recortadas.

A realizações de cálculos, segundo grupo da separação, é composta pelos botões de: Calcular Fluxo de Potência e Calcular Curto-Circuito. Ambos, para que fiquem ativados é necessário que: Exista uma barra slack no sistema, o número de linhas deve maior que um e a quantidade de linhas for maior ou igual que a quantidade de barras a menos de uma unidade. Ao pressionar algum desses botões, o sistema transforma para o formato esperado pela API, citado na Sessão 5.3 no Quadro 5.1 e no Quadro 5.2. O servidor recebe os dados, executa o algoritmo de encaminhamento de rota para o cálculo apropriado, como descrito na Figura 5.3, e retorna informações e resultados para o cliente.

O terceiro grupo da separação é de apresentações das informações do SEP. Contém o botão de visualizar informações. Esse botão chama o componente *ResultadosComponent*, do módulo do diagrama, no qual é responsável por receber um objeto do tipo SistemaEletricoPotencia (como citado na Figura 4.1) e gerar tabelas, exibido na Figura 6.4.

Figura 6.4 - Diagrama - Informações do sistema.

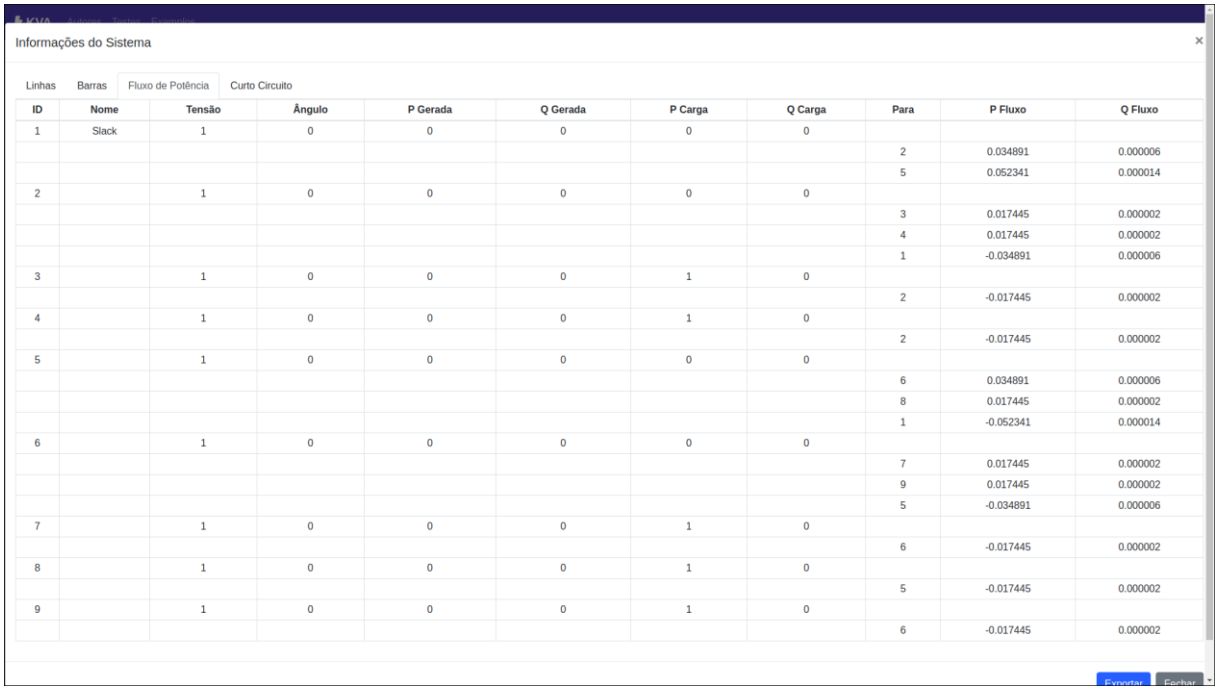


Fonte: Próprio autor.

O componente citado acima utiliza o componente *TabelaComponent* (do módulo compartilhado *SharedModule*) no qual tem a função de receber uma lista de lista (tabela) e exibir o conteúdo. Esse conceito de componentes com responsabilidades únicas permite o reaproveitamento de componentes em várias partes da interface onde ocorre a necessidade de exibir informações tabuladas.

Até o momento, o componente *ResultadosComponent* possui quatro abas: Linhas, Barras, FP e CC. Cada aba, exibe uma tabela com os dados do seu respectivo nome, exibindo o cabeçalho para identificar cada item. As abas de FP e CC só aparecem quando os respectivos cálculos foram efetuados. As tabelas de FP e de CC são exibidas nas figuras Figura 6.5 e Figura 6.6

Figura 6.5. Diagrama - Fluxo de Potência.



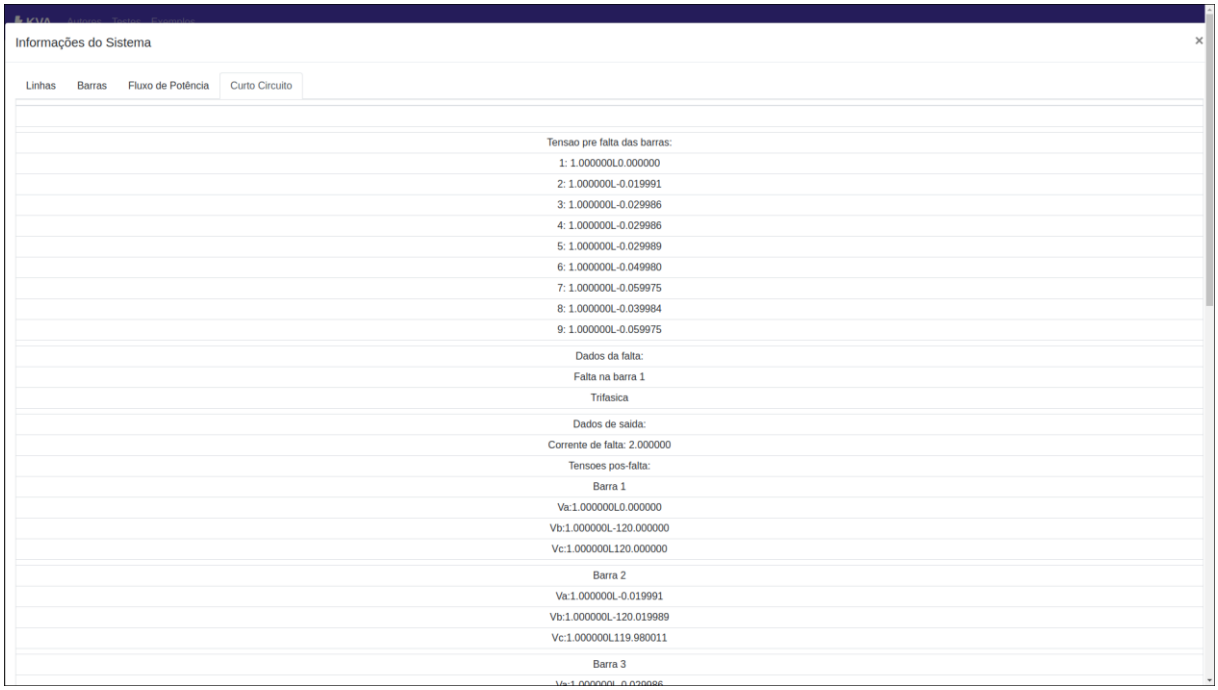
Informações do Sistema

Linhas | Barras | Fluxo de Potência | Curto Circuito

ID	Nome	Tensão	Ângulo	P Gerada	Q Gerada	P Carga	Q Carga	Para	P Fluxo	Q Fluxo
1	Slack	1	0	0	0	0	0			
								2	0.034891	0.000006
								5	0.052341	0.000014
2		1	0	0	0	0	0			
								3	0.017445	0.000002
								4	0.017445	0.000002
								1	-0.034891	0.000006
3		1	0	0	0	1	0			
								2	-0.017445	0.000002
4		1	0	0	0	1	0			
								2	-0.017445	0.000002
5		1	0	0	0	0	0			
								6	0.034891	0.000006
								8	0.017445	0.000002
								1	-0.052341	0.000014
6		1	0	0	0	0	0			
								7	0.017445	0.000002
								9	0.017445	0.000002
								5	-0.034891	0.000006
7		1	0	0	0	1	0			
								6	-0.017445	0.000002
8		1	0	0	0	1	0			
								5	-0.017445	0.000002
9		1	0	0	0	1	0			
								6	-0.017445	0.000002

Fonte: Próprio autor.

Figura 6.6. Diagrama - Curto Circuito.



Informações do Sistema

Linhas | Barras | Fluxo de Potência | Curto Circuito

Tensao pre falta das barras:
1: 1.000000L0.000000
2: 1.000000L-0.019991
3: 1.000000L-0.029986
4: 1.000000L-0.029986
5: 1.000000L-0.029989
6: 1.000000L-0.049980
7: 1.000000L-0.059975
8: 1.000000L-0.039984
9: 1.000000L-0.059975

Dados da falta:
Falta na barra 1
Trifasica

Dados de saida:
Corrente de falta: 2.000000
Tensoes pos-falta:
Barra 1
Va:1.000000L0.000000
Vb:1.000000L-120.000000
Vc:1.000000L120.000000

Barra 2
Va:1.000000L-0.019991
Vb:1.000000L-120.019989
Vc:1.000000L119.980011

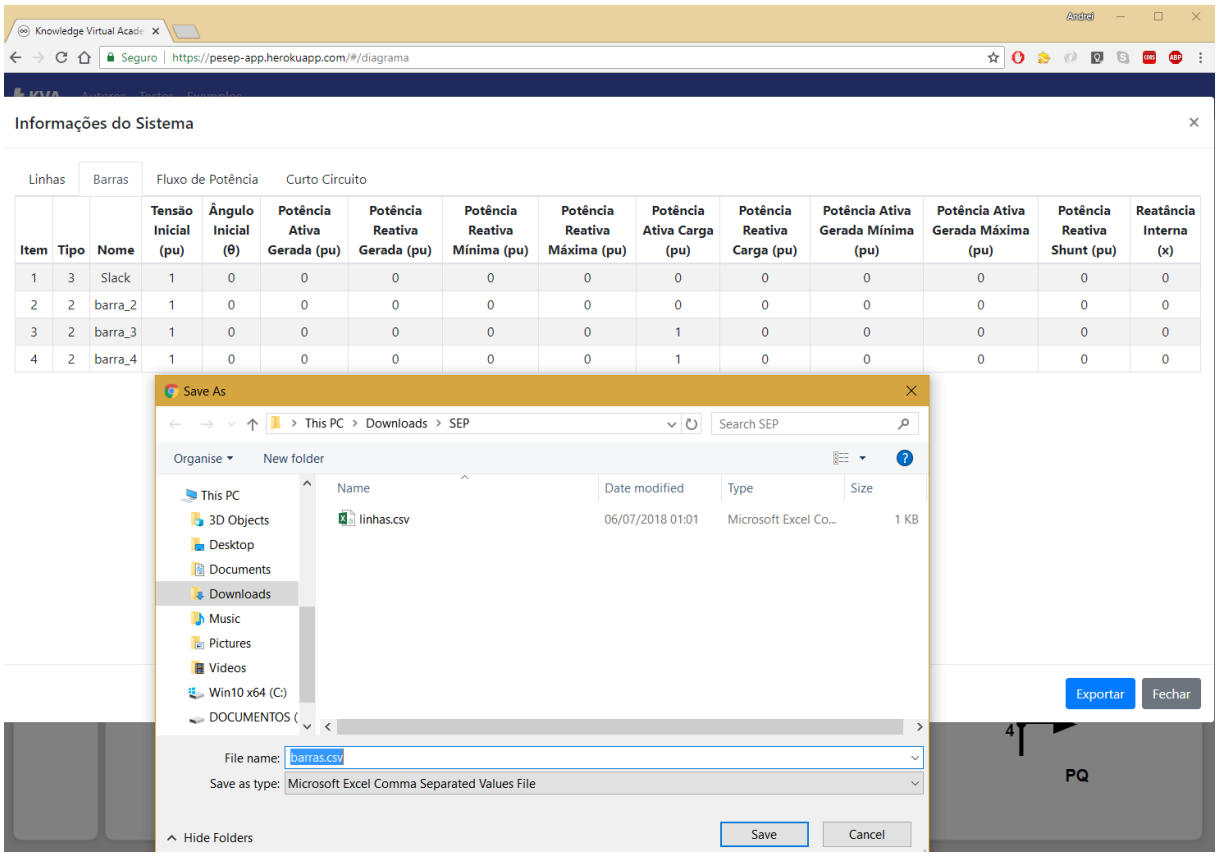
Barra 3

Fonte: Próprio autor.

Além disso, esse mesmo componente possui, no inferior, dois botões: Exportar e Fechar. O botão de exportar transforma a tabela em arquivo no formado CSV e abre

a janela de opção de baixar o arquivo, como é apresentado na Figura 6.7. Já o botão fechar fecha a janela e volta para o desenho do diagrama.

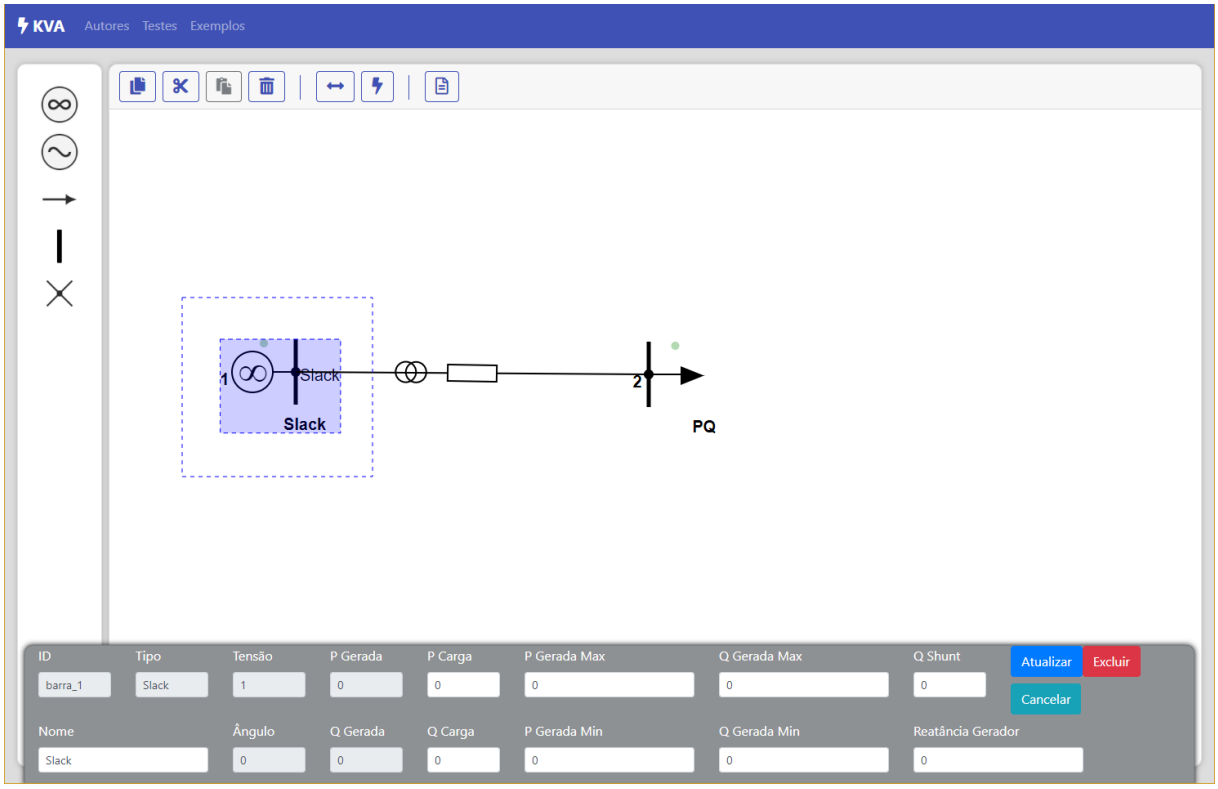
Figura 6.7. Diagrama - Exportando dados.



Fonte: Próprio autor.

Para editar os parâmetros de uma barra, basta selecionar a barra, com o clique em cima da barra ou criando uma área de seleção contendo a barra, como mostrado na Figura 6.8. Quando uma barra é selecionada, a mesma é enviada para o BarraFormComponent que é o componente responsável por exibir o formulário dos parâmetros da barra, no inferior da página, permitindo edita-la. Este componente possui três botões no canto inferior direito: Atualizar, Cancelar ou Excluir. O botão atualizar envia para o diagrama a barra com seus parâmetros modificados e pode atualizar o desenho da barra dependo do parâmetro. O botão de cancelar desfaz a seleção e o botão de excluir, remove a barra do sistema. Quando múltiplas barras são selecionadas o BarraFormComponent é destruído, pois apenas uma barra pode ser editada por vez.

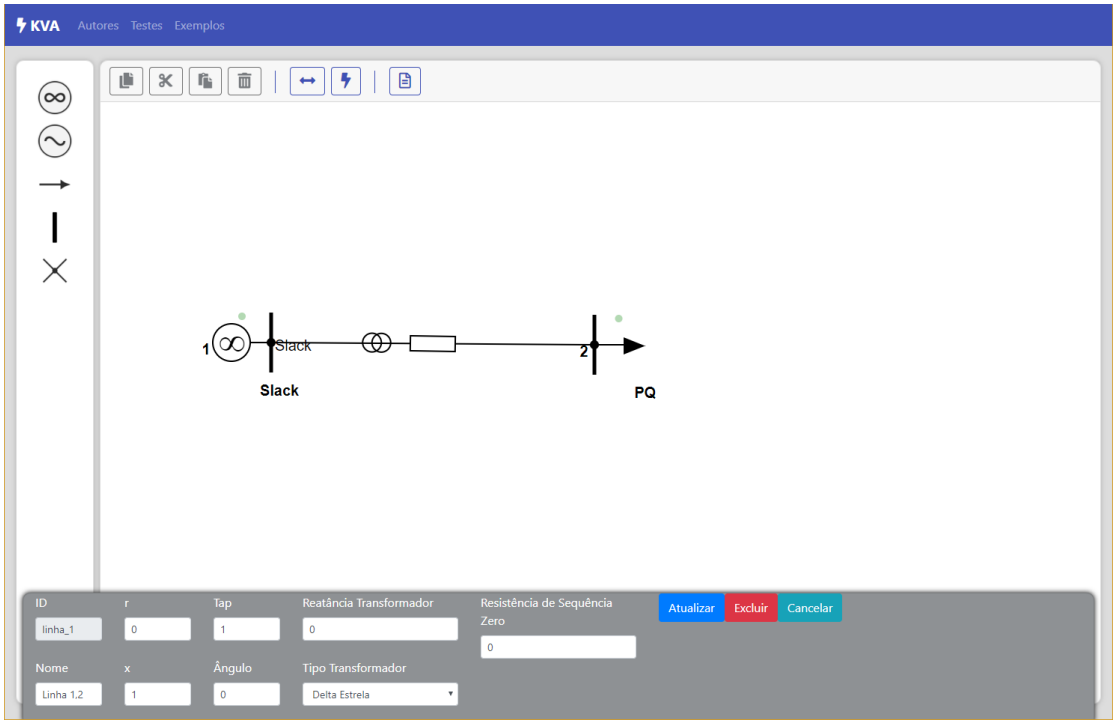
Figura 6.8. Diagrama - Alterando parâmetros da barra.



Fonte: Próprio autor.

Para editar os parâmetros de uma linha, basta selecionar a linha (clique sobre ela). Quando uma barra é selecionada, a mesma é enviada para o LinhaFormComponent que é o componente responsável por exibir o formulário dos parâmetros da linha como mostrado na Figura 6.8, permitindo assim, edita-la. Este componente possui três botões no canto inferior direito: Atualizar, Cancelar ou Excluir. O botão atualizar envia para o diagrama a linha com seus parâmetros modificados. O botão de cancelar desfaz a seleção e o botão de excluir, remove a linha do sistema. Quando múltiplas barras são selecionadas o LinhaFormComponent é destruído, pois apenas uma linha pode ser editada por vez.

Figura 6.9. Diagrama - Alterando parâmetros da linha.

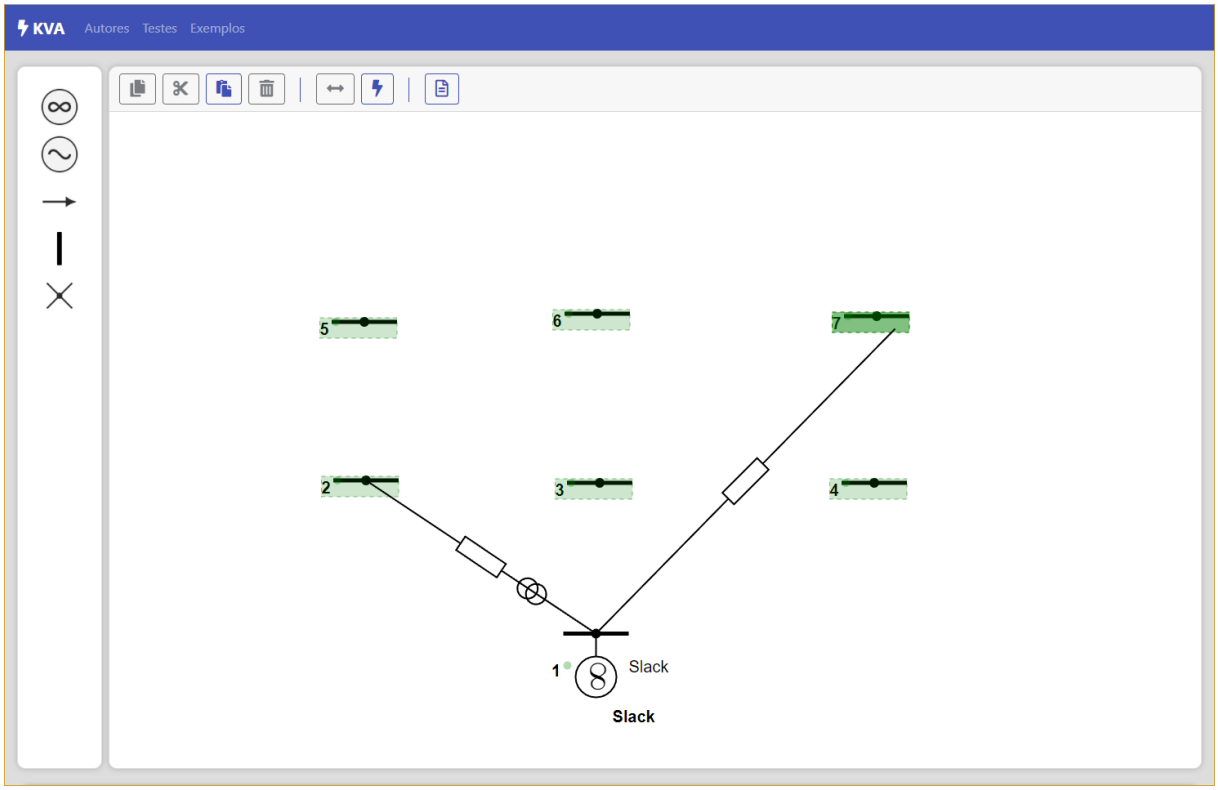


Fonte: Próprio autor.

Além dessas funcionalidades citadas acima, a interface permite rotacionar uma barra. Para isso, é necessário levar o *mouse* até um círculo de cor verde, próximo a barra, clicar sobre o círculo e arrastar para ajustar o ângulo desejado. Quando o mouse está sobre este círculo, o mesmo dobra o tamanho do raio para ficar mais visível e fácil a rotação. A interface também permite selecionar múltiplas barras movendo o conjunto.

Para fazer uma ligação de uma barra a outra, basta puxar o círculo centralizado no barramento da barra até a uma barra desejada. Quando isso acontece, as possíveis barras que podem ser conectadas criam uma zona de cor verde, em torno da barra, para que seja possível soltar e fazer a devida conexão. A Figura 6.10, ilustra o exemplo da criação de uma linha entre duas barras, exibindo as zonas para largar. Ao largar sobre a zona delimitada, a barra é conectada na outra através de uma linha.

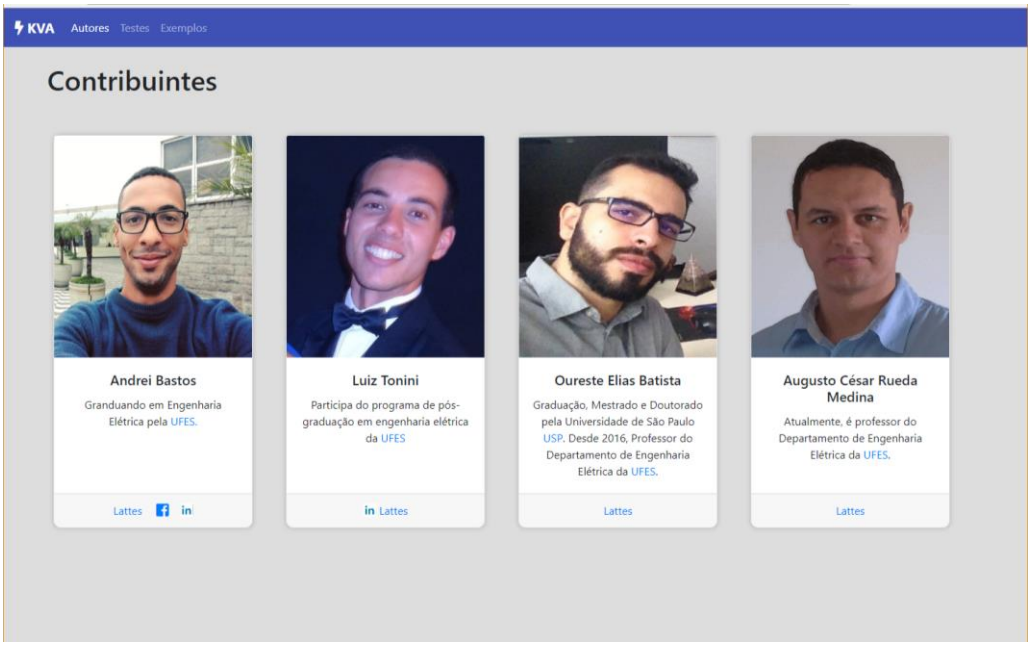
Figura 6.10. Diagrama - Criando linhas.



Fonte: Próprio autor.

O componente QuemSomosComponent apresenta informações dos autores e contribuintes do projeto, sendo exibido na Figura 6.11.

Figura 6.11. Autores - Descrição.



Fonte: Próprio autor.

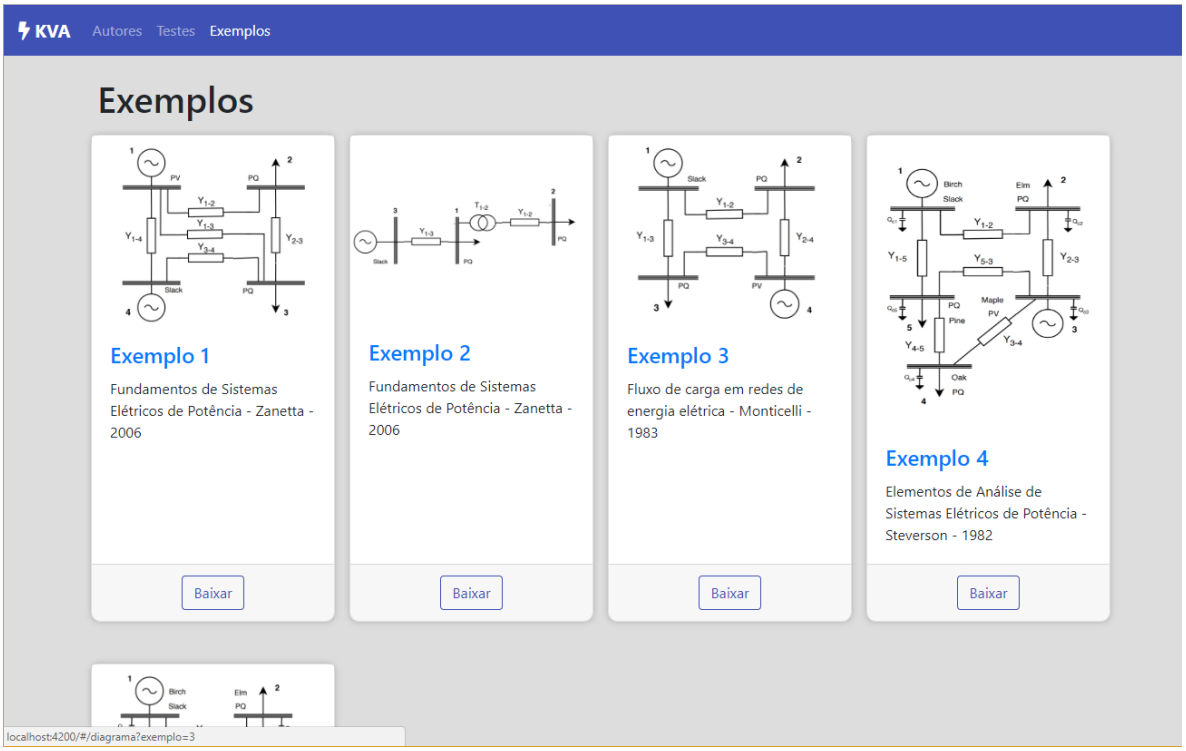
O módulo *TestesModule* tem o objetivo permitir que o usuário importe os dados de linha, barra e entrada_falta.txt (gerados pelo diagrama) (apresentado no Quadro 4.7. Atributos da falta) para realizar o cálculo de FP ou CC. A Figura 6.12 exibe essa interface. Ao importar um arquivo, o mesmo é exibido como uma tabela, igual à que foi mostrada na Figura 6.4.

Figura 6.12. Teste - Submissão de arquivos.

Fonte: Próprio autor.

O componente *ExemplosComponent* tem a reponsabilidade de exibir alguns exemplos. O usuário pode salvar e explorar os arquivos que vêm nesses exemplos, para efetuar testes ou podem redirecionar para o diagrama com o desenho montado. Sua interface é exibida na Figura 6.13.

Figura 6.13. Exemplos - Descrição.



Fonte: Próprio autor.

6.2 Limitações e considerações

Pela interface, a limitação é quantidade de itens que pode ocupar a área limitada. Na Sessão 8.1 Trabalhos Futuros essa funcionalidade é citada.

Em relação aos cálculos de FP e CC, a limitação se dá em termos de quantidade de barra para cálculos de FP. Atualmente, são permitidos no máximo dez barras e é considerado que no SEP as tensões nas barras são equilibradas. No cálculo do CC é permitido somente uma falta por vez, o sistema é sempre balanceado, a consideração é de que as impedâncias de sequência negativa são iguais as positivas, e que as impedâncias dos geradores em regime transitório são iguais em regime permanente.

7 IMPLANTAÇÃO E MANUTENÇÃO

Para que a ferramenta possa ser acessível de forma *online* é necessário que a mesma seja implantada em um servidor com acesso externo. Esse servidor deve possuir pelo menos duas portas, do protocolo TCP/IP, abertas, uma para servir os arquivos gerados para o aplicativo cliente e outra para o servidor web desenvolvido em Python, como descrito Seção 5.3.

A ferramenta tem um propósito de código aberto e para isso foi utilizada a plataforma de hospedagem de código-fonte *Github*.

7.1 Servidor de Implantação

A especificação da configuração do servidor onde deve hospedar as aplicações de cliente e servidor são descritas a seguir.

Para o cliente, necessita-se de um servidor web de arquivos, uma vez que o processamento é realizado do lado do cliente. O tamanho de disco para armazenar esses arquivos do cliente é menor que 5 MB. Independe do sistema operacional, basta ser possível a instalação e configuração de um servidor web de arquivos, como por exemplo, Servidor Apache ou Servidor Nginx, ambos amplamente utilizados pela comunidade de código aberto. (Apiki Blog, 2018)

Para o servidor, desenvolvido em Python, são necessárias mais especificações. Como o servidor é responsável pelo processamento dos cálculos matemáticos, este, precisa ser robusto para atingir uma quantidade maior de usuários. O dimensionamento de um servidor é uma tarefa detalhada e foge do escopo desde projeto.

Vale ressaltar que o cliente pode ficar em servidor diferente do servidor da API, desde haja conexão entre eles.

7.2 Repositórios

Para que haja contribuição de outros desenvolvedores e obter suporte a versionamento da plataforma, foi utilizado o serviço de gerenciamento de versão: *git*, sistema no qual permite que colabores possam acompanhar o versionamento, histórico do desenvolvimento e contribuir para a plataforma.

A hospedagem do código-fonte foi feita na plataforma *Github* no qual oferece suporte ao controle de versão usando o git.

A plataforma possui dois repositórios: pesep-api e pesep-app.

O Pesep-api é o repositório do servidor e pode ser visitado através do link: https://github.com/andreibastos/pesep-api/. Nesse repositório é armazenado os códigos do AMPL e C++ para os cálculos do FP e CC, além do código em Python responsável pelo encaminhamento de rotas, descritas em 5.3.

O Pesep-app é o repositório do cliente e pode ser visitado através do link: https://github.com/andreibastos/pesep-app. Nesse repositório é armazenado todo código-fonte do estilo, estrutura e dinâmica do cliente.

8 CONSIDERAÇÕES FINAIS

O projeto encontra-se na fase de testes e melhorias e tem expectativa para lançamento de uma versão operacional até o fim do ano de 2018, sendo hospedado em um servidor da UFES. Até o momento, o sistema está sendo hospedado na plataforma do Heroku no link https://pesep-app.herokuapp.com.

O projeto desenvolvido contribuiu de forma profissional para o estudante, uma vez que tenha colocado em prática conhecimentos na área da computação e elétrica e aprendido sobre ferramentas atuais do mercado de trabalho e considerações do SEP.

A documentação dos *frameworks* Angular e Bootstrap e das bibliotecas SVG.js e Interact.js estão disponíveis em seus respectivos portais, oferendo uma gama de exemplos introdutórios os quais serviram como base para o desenvolvimento da plataforma.

8.1 Trabalhos Futuros

O uso do controle de versionamento pelo *git* (nome próprio) e hospedagem na plataforma do *Github* permite a entradas de novos colaboradores. Além disso, o uso de frameworks e bibliotecas bem documentados facilita no entendimento desses novos colaboradores (Git,2018).

Para que a plataforma fique mais didática e mais robusta são necessárias implementações de melhorias e novas funcionalidades. Segue abaixo uma lista possíveis melhorias a serem implementadas:

Histórico de comandos: Permitir que seja possível desfazer ou avançar um comando feito. Essa funcionalidade é bastante utilizada em ferramentas de edição de desenho em geral.

Visualização de resultados: Desenhar novas formas de visualização de resultados dos cálculos inserindo, por exemplo setas indicando direção do fluxo, quando calculado.

Melhoria nos desenhos: Exibir o transformador na linha. Hoje a ferramenta não exibe um transformador na linha, apenas indica que existe um transformador e permite editar os parâmetros. Adicionar, no desenho, impedância interna das barras quando é diferente de zero.

Interface gráfica: Permitir funcionalidade de zoom, fazendo ampliar ou reduzir escala dos elementos do diagrama.

Selecionar linhas: Atualmente a plataforma só permite selecionar as barras através de seleção (retângulo de seleção realizado pelo arraste do mouse); a ideia é que seja possível selecionar, além das barras, as linhas e assim, usar as funcionalidades de copiar, recortar e excluir.

Relatório detalhado de erros: E necessário fazer um relatório de erros com maiores detalhes, visto que atualmente a plataforma só indica dois erros, são eles:

- O sistema só pode ter uma barra de referência que acontece quando o usuário tenta arrastar ou copiar uma barra de referência quando já existe uma no sistema.
- E o erro “servidor não encontrado” que acontece quando o servidor tem erro interno ou está fora do ar.

A melhoria seria informar um detalhe contendo mais informações sobre a realização dos cálculos, informando se foi possível realizar ou não, e no caso de não possível, detalhar melhor o acontecido.

Comunicação com outro servidor: Uma funcionalidade para quem deseja usar um servidor de cálculos próprio, permitindo que usuários usem a ferramenta apenas para o desenho e gere os dados de cálculo do seu próprio servidor (desde que atenda as rotas de cálculos).

Essas são apenas algumas das recomendações para trabalhos futuros. Com o uso do programa, usuários podem sugerir novas funcionalidades ou reportar erros.

Em relação ao SEP, poderia ser implementado outros tipos de redes, como por exemplos barras com redes não balanceadas. Permitir o cálculo de FP para distribuição.

Uma aba especial de ajuda e endereço dos repositórios do Github.

O módulo de testes ter uma ajuda com a formatação dos dados de entrada.

REFERÊNCIAS BIBLIOGRÁFICAS

AMPL Optimization Inc. (2017). AMPL .**Streamlined Modeling for Real Optimization**. Disponível em <http://ampl.com/about-us/>. Acesso em: Fevereiro de 2018.

Apiki Blog. Apiki**. Conceitos, semelhanças e diferenças entre NGINX, Apache e WordPress**. Disponivel em <https://blog.apiki.com/2017/06/27/nginx-apache-wordpress/>. Acesso em: Junho de 2018.

ASTAH. Documentação do software astah. Astah: Disponível em < http://astah.net/manual#doc > Acesso: Fevereiro de 2018.

Barcellos, M. P. **Engenharia de Software - Notas de Aula**. Nemo - Núcleo de Estudos em Modelagem Conceitual e Ontologias. Disponível em <https://nemo.inf.ufes.br/wp-content/uploads/Monalessa/ES-EngComp/NotasDeAula-EngSw-EngComp-v2018.pdf>. Acesso em: Março de 2018.

Bootstrap. Bootstrap. ***Get started***. Disponível em <https://getbootstrap.com/>. Acesso em: Janeiro de 2018.

Interact. **Documentation of Interact**. Interact.js. Disponível em: <http://interactjs.io/docs/> . Acesso em: Fevereiro de 2018.

Git. **Documentation.** Git. Disponivel em :< https://git-scm.com/docs/git >. Acesso em: Julho de 2018.

Kagan, N., Oliveira, C. C., Robba, E. J., & Schmidt, H. P. (1996). **INTRODUÇÃO A SISTEMAS ELÉTRICOS DE POTÊNCIA**. São Paulo: Egard Blücher LTDA.

Monticelli, A. J. (1983). Fluxo de Carga em redes de energia elétrica. São Paulo: Edgard Bluücer LTDA.

NETSOFT. **Fundamentos da Arquitetura Cliente/Servidor**: NETSOFT: Disponivel em: <http://www.netsoft.inf.br/aulas/4_SIN_Programacao_Cliente_Servidor/Fundamentos_de_Cliente-Servidor.pdf>. Acesso em: Novembro de 2017.

Stevenson, W. D., & Grainger, J. J. (1984). **Elementos de Analise de Sistemas de Potencia** (4 ed.). McGraw-Hill.

Stevenson, W. D., & Grainger, J. J. (1994). **Power System Analysis**. McGraw-Hill.

Tonini, L. G., Batista, O. E., Rueda, A. C., & Bastos, A. C. **Online Platform for learning of Electrical Power Systems**. ICAEEDu, 2018.

Tonini, L. G., Batista, O. E., & Rueda, A. C. **Simulador Computacional Para Proteção Digital De Sistemas Elétricos De Potência**. COBENGE, 2017.

Wout Fierens. SVG.js. **Getting started**. Disponível em: <http://svgjs.com/getting-started/>. Acesso em: Fevereiro de 2018.

Zanetta Jr, L. C. **Fundamentos de Sistemas Elétricos de Potência**. 2006, São Paulo: Livraria da Física.