

Documento de Planejamento de Software: LinhaMestre

Documento de Planejamento de Software: Software de Métodos de Cálculo de Impedâncias e Capacitâncias de Linha de Transmissão

Sergio Pedro Rodrigues Oliveira Raphael Mello de Oliveira
Paulo Victor de Souza Borges

27 June 2025

1 Visão Geral do Projeto

- **Nome do Projeto:** Software de Métodos de Cálculo de Impedâncias e Capacitâncias de Linha de Transmissão
- **Nome do Software:** LinhaMestre
- **Desenvolvedores:** Sergio Pedro Rodrigues Oliveira e Raphael
- **Propósito:** O principal **propósito** deste software é atuar como uma **ferramenta computacional** dedicada ao **cálculo de impedâncias e capacitâncias de linhas de transmissão**. Ele oferece uma abordagem sistemática e precisa para aplicar os **métodos matemáticos e computacionais** necessários para determinar esses parâmetros cruciais em projetos e análises de engenharia elétrica. Ao automatizar esses cálculos complexos, o software visa **simplificar o trabalho de engenheiros e estudantes, reduzir a ocorrência de erros manuais e acelerar significativamente o processo de análise** de sistemas de transmissão de energia elétrica. Em essência, ele serve como um recurso indispensável para otimizar o design e a avaliação do desempenho de linhas de transmissão.
- **Público-Alvo:** Engenheiros eletricitas (especialmente de sistemas de potência), técnicos em eletrônica, estudantes de engenharia elétrica e pesquisadores na área de transmissão de energia.

2 Funcionalidades Principais (O que o software fará?)

- **Entrada de Parâmetros da Linha:** Interface para inserção de dados como frequência, resistividade do condutor, permissividade do dielétrico e dimensões físicas. O software será projetado para lidar com linhas de transmissão com **3 cabos condutores** e, opcionalmente, **1 cabo para-raios**. Isso implica que a interface e os métodos de cálculo devem acomodar essa configuração específica.
 - **Exibição de Resultados:** Apresentar os resultados dos cálculos de forma clara e organizada, incluindo unidades. As matrizes de resultados serão exibidas no formato **3x3** ou **4x4**, dependendo do método de cálculo e da configuração da linha (presença de cabo para-raios, por exemplo).
 - **Implementação de Métodos de Cálculo Específicos:**
 - **Métodos Longitudinais (Cálculo de Impedância):**
 - * **Método das Imagens:** Para cálculo de impedância própria e mútua, considerando o efeito do solo.
 - * **Método de Carson:**
 - **Transposição:** Cálculos de impedância para linhas transpostas, visando equalizar os parâmetros por fase.
 - **Para-raios:** Consideração da influência do cabo para-raios (se presente) no cálculo da impedância.
 - **Feixe de Condutores:** Determinação da impedância para configurações de condutores em feixe (para 3 condutores por feixe).
 - **Impedância de Sequência:** Cálculo das impedâncias de sequência positiva, negativa e zero, essenciais para análise de faltas.
 - **Métodos Transversais (Cálculo de Capacitância):**
 - * **Método das Imagens:** Para cálculo de capacitância própria e mútua, considerando o efeito do solo.
 - * **Transposição:** Cálculos de capacitância para linhas transpostas.
 - * **Para-raios:** Consideração da influência do cabo para-raios (se presente) na capacitância da linha.
 - * **Feixe de Condutores:** Determinação da capacitância para configurações de condutores em feixe (para 3 condutores por feixe).
 - * **Capacitância de Sequência:** Cálculo das capacitâncias de sequência positiva, negativa e zero.
-

3 Requisitos Não Funcionais (Como o software se comportará?)

- **Precisão:** Os cálculos devem ser altamente precisos, seguindo as fórmulas e métodos padrões da engenharia elétrica, reconhecidos na literatura técnica e normativa. Os resultados numéricos exibidos devem ser apresentados com **duas casas decimais** para clareza e padronização.
 - **Usabilidade:** A interface do usuário deve ser intuitiva e fácil de usar, permitindo que o usuário insira dados e visualize resultados sem dificuldades. A interface deve ser clara ao solicitar os parâmetros dos **3 condutores** e do cabo para-raios (se houver).
 - **Desempenho:** Os cálculos devem ser executados rapidamente, mesmo para os métodos mais complexos. Durante o tempo de processamento, a interface do usuário deve congelar, impedindo a interação até que o cálculo seja concluído. Para um cenário padrão com 3 condutores (e cabo para-raios, se aplicável), se o cálculo exceder 10 segundos, o programa deverá exibir uma mensagem de erro clara, indicando que a operação demorou demais e não pôde ser finalizada.
 - **Confiabilidade:** O software deve ser robusto, apresentar poucos erros ou falhas e fornecer resultados consistentes para as mesmas entradas.
 - **Manutenibilidade:** O código deve ser bem estruturado, modular e documentado para facilitar futuras atualizações, a adição de novos métodos de cálculo ou tipos de linha, e correções de bugs.
 - **Portabilidade:** O software será compatível e deve funcionar corretamente em sistemas operacionais **Windows** e **Linux**, aproveitando a natureza multiplataforma da biblioteca Tkinter.
-

4 Tecnologias Sugeridas (Como o software será construído?)

- **Linguagem de Programação:** **Python**
 - **Biblioteca GUI:** **Tkinter** (para construção da interface gráfica).
 - **Bibliotecas de Cálculo Numérico:** **NumPy** e **SciPy** (essenciais para manipulação de matrizes, números complexos e resolução de sistemas lineares complexos envolvidos nos métodos de Carson e de Sequência).
 - **Controle de Versão:** **Git** (ferramenta de controle de versão distribuído) e **GitHub** (plataforma para hospedagem de repositórios e colaboração).
-

5 Design da Interface do Usuário (UI/UX) e Fluxo de Telas

O software será estruturado em duas telas principais para proporcionar uma experiência de usuário clara e focada:

1. Janela Inicial (Seleção de Métodos):

- Esta será a **tela de entrada** do software.
- Exibirá uma **lista clara e organizada** de todos os métodos de cálculo disponíveis (tanto longitudinais quanto transversais). Pode ser uma lista de opções (`OptionMenu`).
- O usuário **selecionará o método desejado** (clicando em um item da lista ou em um botão).
- Haverá um **botão “OK”** (ou “Prosseguir”, “Selecionar”) que, ao ser clicado, validará a seleção e abrirá a próxima janela.
- A janela anterior (inicial) será **destruída** para dar lugar à janela do método específico.

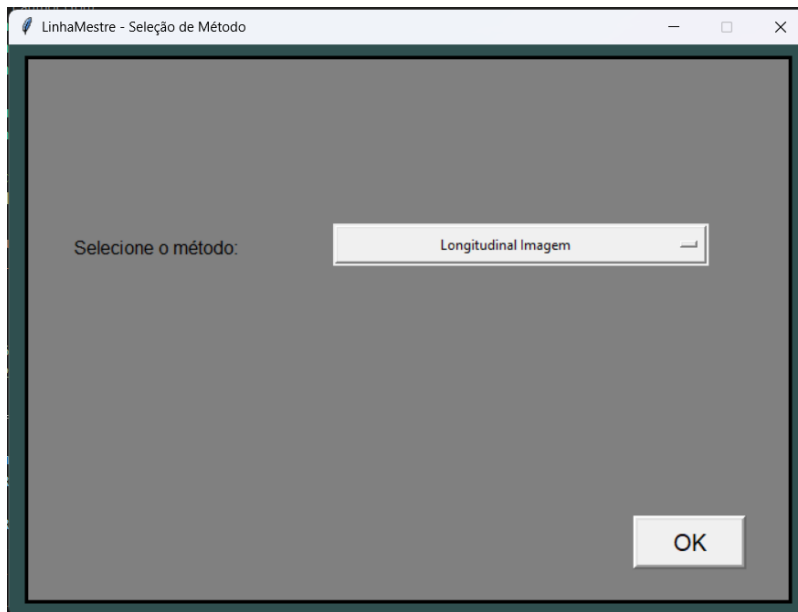


Figure 1: Prototipo da janela principal.

1. Janela do Método Específico (Cálculo):

- Esta janela **substituirá a janela inicial**.
- **Layout:** Será dividida visualmente em duas seções:
 - **2/3 Superiores (Área de Inputs e Controles):**
 - * **Botão “Retornar”:** Localizado na parte superior, permitirá ao usuário voltar à Janela Inicial para selecionar outro método. Ao retornar, a janela atual será fechada e a janela inicial será reexibida ou recriada.
 - * **Lista de Inputs de Variáveis:** Uma área contendo os campos de entrada (Entry widgets do Tkinter) para as variáveis e parâmetros específicos do método selecionado.
 - Os inputs para os **3 condutores** e para o cabo para-raios (se houver) devem ser claramente identificados e organizados. Por exemplo, pode haver rótulos como “Condutor 1 - Diâmetro:”, “Condutor 1 - Coordenada X:”, “Condutor 1 - Coordenada Y:”, e assim por diante, para cada condutor. Para o cabo para-raios, rótulos similares, mas indicando “Cabo Para-raios”.
 - O software deve permitir que o usuário indique se há ou não um cabo para-raios presente.
 - * **Botão “Limpar”:** Ao ser clicado, apagará todos os valores inseridos nos campos de input da janela atual.
 - * **Botão “Calcular”:** Este será o acionador principal. Ao ser pressionado, o software:
 - Coletará os dados dos campos de input.
 - Validará os inputs (verificando tipos de dados, intervalos, etc.).
 - Chamará a função Python correspondente ao método de cálculo selecionado, passando os inputs como parâmetros.
 - Receberá a matriz de resultado dos cálculos.
 - Exibirá essa matriz na área de resultados.
 - **1/3 Inferior (Área de Resultados):**
 - * Um espaço dedicado para exibir a **matriz de resultado** dos cálculos. Pode ser um widget de texto (Text) ou uma estrutura de tabela (TreeView, se mais complexo) que formatará os dados de forma legível. Essas matrizes serão do tipo **3x3 ou 4x4**. Os resultados devem incluir unidades e, se aplicável, as partes real e imaginária de números complexos, **arredondados para duas casas decimais**.

Cálculo de Impedância - Longitudinal Imagem

Retornar Limpar

Resistências

Ra (Ohm/unidade)

Rb (Ohm/unidade)

Rc (Ohm/unidade)

Raio Condutor (R) [m]

OU RMG [m]

Coordenadas dos Cabos

X_a (m) H_a (m)

X_b (m) H_b (m)

X_c (m) H_c (m)

Calcular

Fase	Condutor A	Condutor B	Condutor C
A	(nan + j0.000000)	(nan + j0.000000)	(nan + j0.000000)
B	(nan + j0.000000)	(nan + j0.000000)	(nan + j0.000000)
C	(nan + j0.000000)	(nan + j0.000000)	(nan + j0.000000)

Figure 2: Prototipo da janela dos métodos específicos.

6 Plano de Desenvolvimento Simplificado (Como será feito?)

- **Prazo Limite para Apresentação:** 10/07/2025
- **Fase 1: Definição, Modelagem e Prototipagem da UI:**
 - Revisão e detalhamento das fórmulas e algoritmos para cada método de cálculo de impedância e capacitância (Método das Imagens, Carson, etc.).
 - Desenho de wireframes (esboços de tela) detalhados das duas janelas, incluindo a disposição exata dos elementos do Tkinter.
 - **Criação de um protótipo básico da interface com Tkinter**, focando na transição entre as janelas e na disposição dos inputs/outputs, especialmente na forma de solicitar e organizar os dados dos **3 condutores** e do cabo para-raios (se houver).
- **Fase 2: Desenvolvimento do Core de Cálculo (Módulos):**
 - Implementação das funções matemáticas e lógicas para cada método de cálculo em Python (separadas da interface, em módulos ou classes dedicadas).
 - Utilização extensiva de NumPy e SciPy para cálculos complexos.
 - Adaptação dos métodos de cálculo para acomodar a configuração de **3 condutores** e, opcionalmente, 1 cabo para-raios, **garantindo que as saídas sejam matrizes 3x3 ou 4x4 conforme a necessidade do método e entradas**.
 - Implementação do arredondamento para duas casas decimais nos resultados finais dos cálculos.
 - Testes unitários rigorosos para cada função de cálculo para garantir a precisão e a robustez dos resultados.
- **Fase 3: Integração da Interface com os Cálculos:**
 - Conectando os botões de “Calcular” da interface às funções de cálculo do core.
 - Implementação da validação de entrada de dados (tratamento de erros para inputs inválidos).
 - Formatação e exibição dos resultados da matriz na área inferior da janela do método, garantindo a apresentação das matrizes 3x3 ou 4x4 com duas casas decimais.
 - Implementação das funcionalidades dos botões “Limpar” e “Retornar”.
- **Fase 4: Testes Completos e Refinamentos:**
 - Testes de integração: verificar se a interface e os módulos de cálculo funcionam bem em conjunto.
 - Testes de sistema: simular cenários de uso real para garantir que o software atenda aos requisitos.
 - Testes de usabilidade com usuários-alvo (engenheiros, estudantes) para coletar feedback e fazer ajustes na interface e fluxo.
 - Correção de bugs e melhorias na experiência do usuário e na precisão dos cálculos.
 - Testes de portabilidade: Verificação do comportamento e desempenho do software nos sistemas operacionais Windows e Linux.

- **Fase 5: Documentação e Implantação:**

- Criação de um manual do usuário detalhado, explicando como usar o software, os inputs esperados (incluindo a configuração dos **3 condutores** e do cabo para-raios) e interpretando os resultados.
 - Elaboração de documentação técnica interna para futuros desenvolvedores.
 - Empacotamento do software para distribuição (ex: com PyInstaller para criar um executável compatível com Windows e Linux).
 - Configuração e manutenção do repositório no GitHub para controle de versão e colaboração.
-

7 Considerações Adicionais (Opcional)

- **Diferencial:** Este software pode se diferenciar por oferecer uma suíte de métodos de cálculo abrangente e integrada, superando calculadoras online que geralmente focam em apenas um tipo de cálculo. A capacidade de considerar detalhes como cabos para-raios e condutores em feixe o tornará uma ferramenta mais completa.
- **Escalabilidade Futura** (Possibilidade de adicionar):
 - Variação flexível do número de cabos de transmissão e de cabos para-raios, oferecendo maior adaptabilidade e precisão para diferentes configurações de linhas.
 - Geração de relatórios ou gráficos de desempenho.
 - Capacidade de salvar e carregar projetos de linhas de transmissão.
- **Requisitos de Conhecimento:** O desenvolvimento exigirá um forte domínio dos princípios de linhas de transmissão, grandezas complexas, álgebra linear e métodos numéricos em engenharia elétrica, além de programação em Python e uso de Tkinter.