

Relatório Técnico

Documentação da Aplicação | Simulador Visual de Sistema Operacional

Vinicius Wandembruck Kureishi

28 de outubro de 2025



UTFPR-CT

1 Resumo

Este documento apresenta a documentação visual da interface da aplicação *Simulador Interativo de Escalonamento de Processos*. A aplicação foi desenvolvida para apoiar o estudo de algoritmos de escalonamento de processos em ambientes multitarefa. A interface gráfica oferece recursos para carregar arquivos de configuração, executar a simulação passo a passo ou de forma contínua, visualizar dados detalhados e salvar o gráfico resultante para análise. O desenvolvimento deste simulador também está alinhado aos conteúdos abordados na disciplina de Sistemas Operacionais, ministrada pelo Prof. Dr. Marco Aurélio Wehrmeister, reforçando aspectos teóricos através de uma abordagem prática.

2 Introdução

A aplicação foi desenvolvida para apoiar o estudo de algoritmos de escalonamento de processos. A interface gráfica apresenta controles para carregar arquivos de configuração, executar a simulação passo a passo ou de forma contínua, visualizar dados e salvar o gráfico resultante. Este relatório descreve a interface e seus componentes, bem como a relação entre os elementos exibidos e as estruturas de dados internas.

3 Visão Geral da Aplicação

A aplicação possui uma janela principal com cabeçalho superior, painel de controle, área de desenho para o gráfico de Gantt, legenda e rodapé. O tema visual pode ser alternado entre claro e escuro. Ícones de atalho direcionam para materiais de referência e para o perfil do desenvolvedor.

4 Documentação Visual da Interface

Esta seção apresenta as telas principais e descreve cada elemento. Substitua os nomes dos arquivos de imagem pelos seus, mantendo as mesmas referências no projeto.

4.1 Janela Principal no Tema Claro

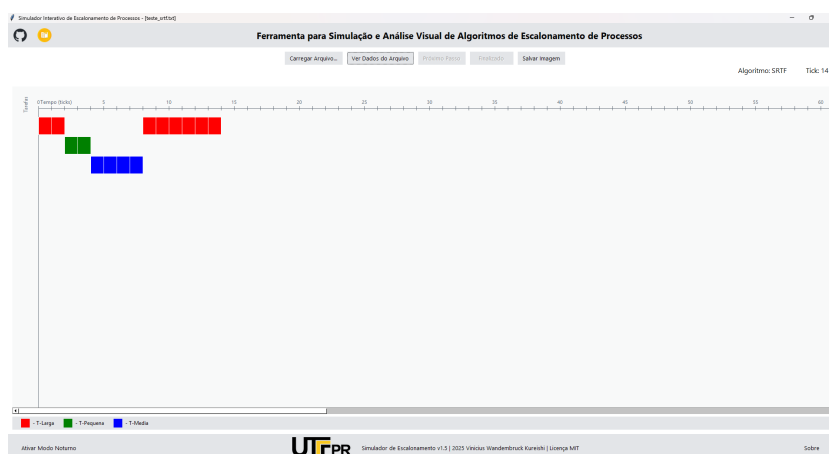


Figura 1: Visão geral da janela principal no tema claro.

Descrição sintética:

- **Barra de título:** Exibe o nome da aplicação e, após carregar um arquivo, o nome do arquivo entre colchetes, por exemplo `[teste_fifo.txt]`. A cor é ajustada pela API do sistema.
- **Cabeçalho superior:** Ícones para GitHub e material de referência, com abertura de links em nova aba.
- **Painel de controle:** Botões para carregar arquivo, ver dados, avançar um passo, executar a simulação completa e salvar imagem. Labels informam algoritmo e *tick* global.

- **Área de gráfico:** Eixos e barras do gráfico de Gantt são desenhados dinamicamente conforme a simulação avança.
- **Legenda inferior:** Relaciona cores das barras aos IDs das tarefas.
- **Rodapé:** Alternância de tema, exibição de logomarca e informações de versão e licença.

4.2 Janela Principal no Tema Escuro



Figura 2: Visão geral da janela principal no tema escuro.

Notas:

- Elementos funcionam de forma equivalente ao tema claro.
- Paleta de cores, ícones e barra de título são ajustados de acordo com o tema ativo.

4.3 Janela Ver Dados

Elementos:

- **Tabela de tarefas:** Exibe ID, cor, tempo de chegada, duração e prioridade, conforme arquivo de configuração.

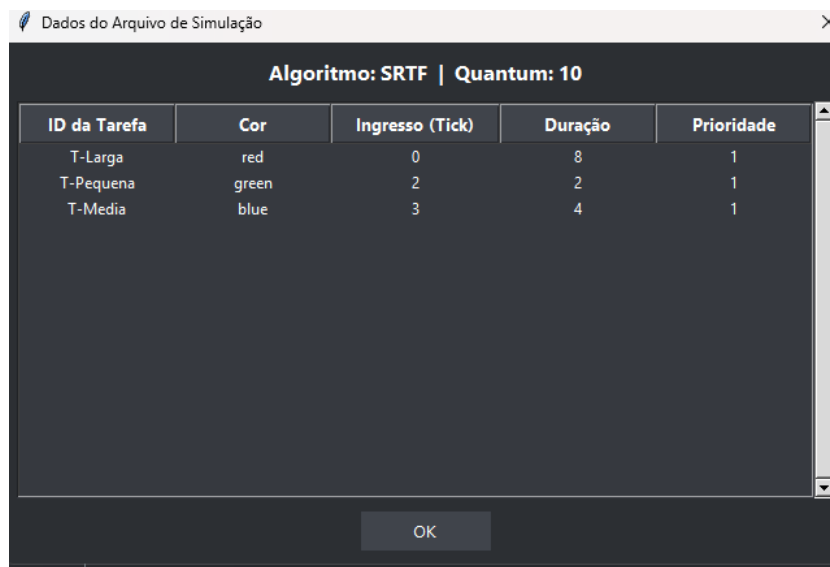


Figura 3: Janela de visualização dos dados carregados.

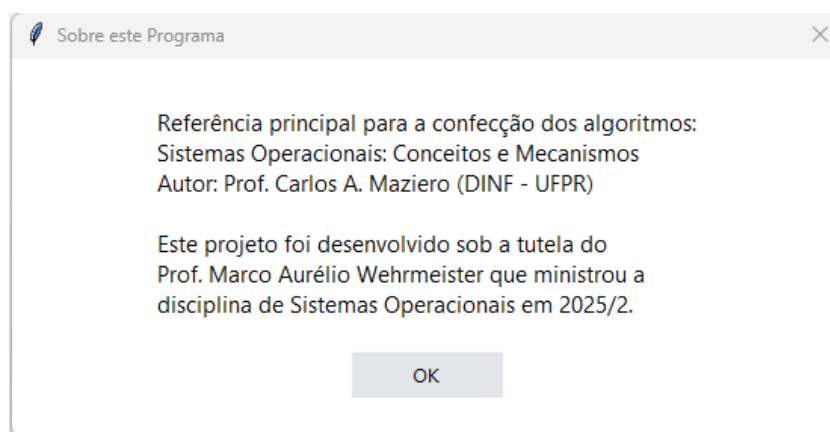


Figura 4: Janela com informações institucionais e referências.

4.4 Janela Sobre

Conteúdo:

- Referência bibliográfica principal e contexto acadêmico.
- Botão para fechar a janela.

5 Diagrama UML de Classes

O diagrama UML de classes resume as principais entidades do simulador, seus atributos e métodos, além das relações entre componentes da interface e do núcleo da simulação.

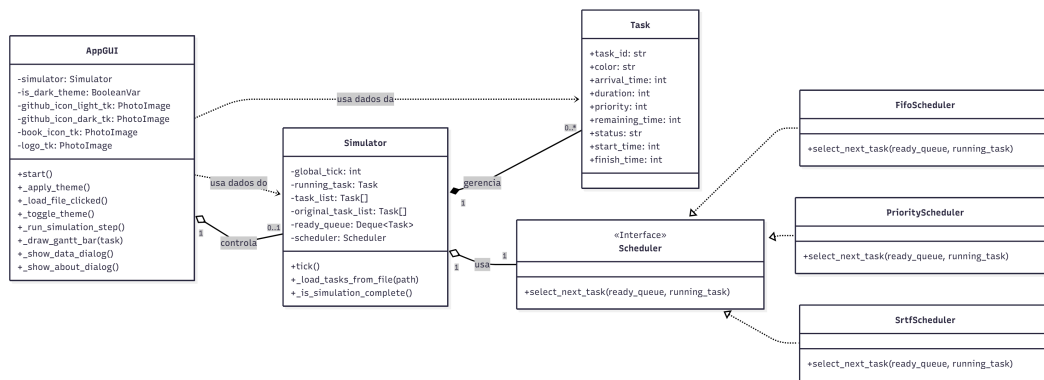


Figura 5: Diagrama UML de classes do sistema.

6 Diagrama de Casos de Uso

O diagrama de casos de uso apresenta os atores e suas interações principais com a aplicação. O foco é ilustrar tarefas do usuário e objetivos do sistema.

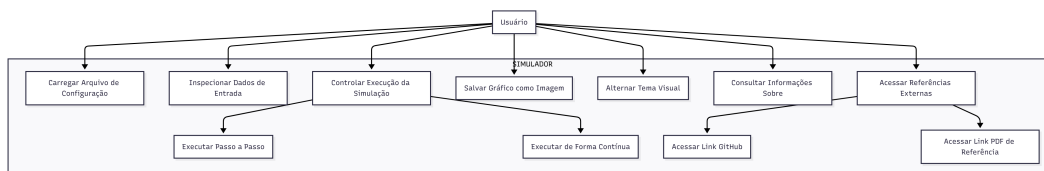


Figura 6: Diagrama de casos de uso da aplicação.

7 Considerações Finais

A documentação visual apresentada permite compreender a organização da interface, a dinâmica de interação e a origem dos dados exibidos. Superando desafios iniciais de layout, como a centralização de títulos e ícones no cabeçalho e rodapé, a estrutura final foi implementada utilizando o gerenciador grid do Tkinter para garantir um posicionamento responsivo dos elementos. A dinâmica de interação, incluindo a implementação da execução contínua ("Executar Tudo") sem congelamento da interface, foi resolvida com o uso de `master.after()` em vez de um loop bloqueante, permitindo a visualização animada do processo. A origem dos dados exibidos é detalhada, assegurando a precisão do Gráfico de Gantt ao corrigir um bug crucial na lógica de preempção que causava um atraso de um tick na representação gráfica da troca de tarefas. Desafios como o desaparecimento de imagens (causado pelo coletor de lixo do Python) foram solucionados ancorando as referências das imagens (`widget.image = photo`) aos widgets correspondentes. Os diagramas UML de Classes e Casos de Uso fornecem uma visão de alto nível da arquitetura, apoiando a manutenção e evolução, talvez, do software. A função `resourcepath` foi importante para garantir que os recursos visuais (ícones, logo) fossem corretamente localizados e empacotados pelo PyInstaller, funcionando tanto no ambiente de desenvolvimento quanto no executável final distribuível. A estrutura adotada facilitou a atualização de imagens e a inclusão de novas telas, mantendo a consistência e coerência.