PAMPAOS: INTERFACE GRÁFICA DO AMBIENTE DE SIMULAÇÃO

VINÍCIUS GARCIA PERUZZI; GABRIEL LEITE BESSA; HECTOR HUDSON DINIZ FERNANDES; JOÃO ANTÔNIO NEVES SOARES; LUAN MARK DA SILVA BORELA; RAFAEL BURLAMAQUI AMARAL

Universidade Federal de Pelotas – {vgperuzzi, gabriel.lb, hhdfernandes, lmdsborela, jansoares, rafael.amaral}@inf.ufpel.edu.br

1. INTRODUÇÃO

A disciplina de Sistemas Operacionais (SO) nos cursos superiores de computação tem papel fundamental na progressão do aprendizado de um aluno nesta área, isso porque ela faz a síntese do conhecimento de hardware basilar adquirido até o momento, e prepara para aprendizados mais avançados no futuro, como programação paralela e distribuída.

Entretanto, o estudo de SO é bastante desafiador, por tratar de assuntos conceitualmente complexos, como gerenciamento de processos, memória e dispositivos de hardware. Conceitos esses que são abstratos e não diretamente visíveis, o que torna ainda mais difícil compreender o que acontece à nível de SO. Portanto, para esse fim, surge o projeto que propõe a criação de um simulador, com o objetivo principal de mostrar os componentes de SO, bem como permitir que o usuário acompanhe as rotinas em tempo real, aprendendo como funciona na prática um SO, permitindo modificar algumas de suas estruturas. Esse trabalho trata da contextualização da criação da interface gráfica associada.

O principal simulador utilizado nessas disciplinas é o SOsim (MAIA, 2001), porém, ainda que o aplicativo seja bastante usado e tenha grande utilidade para o fim ao qual se propõe, como apresentado na Figura 1, sua versão de lançamento contém alguns erros e *bugs* que impedem que algumas funcionalidades executem bem, assim como se desatualizou em questão de conteúdo, por exemplo, hoje os SOs usam uma arquitetura multiprocessada, com o conceito de *multithreading*, ou ainda características modernas de disco e memória.

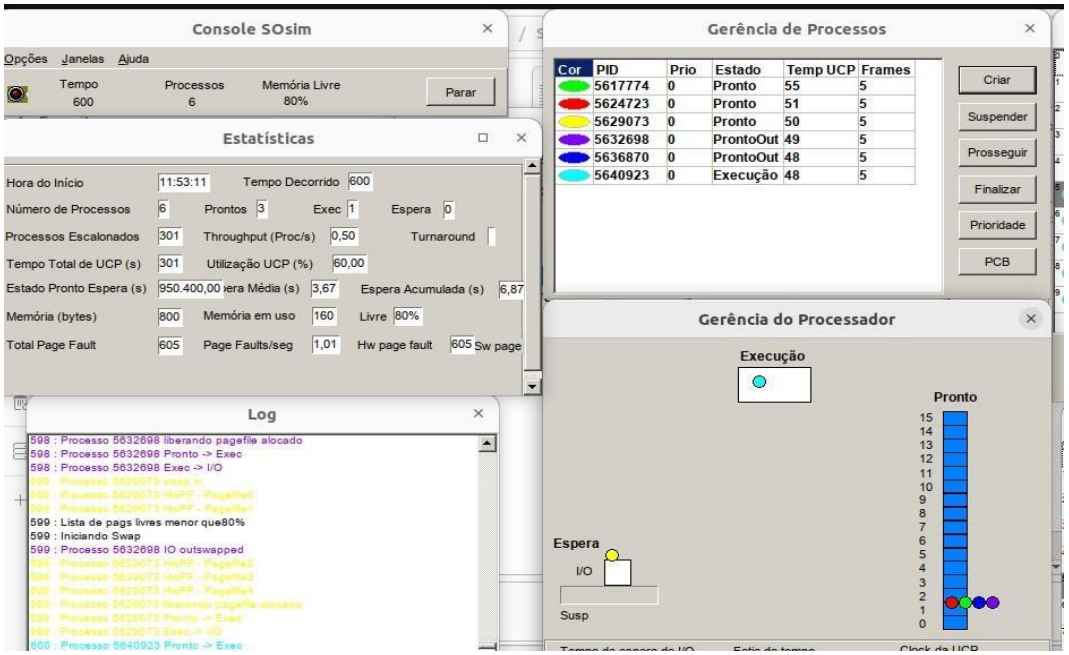

Figura 1 - Layout do SOsim

Existem ainda outros simuladores mais modernos, como o YASS (MUSTAFA, 2013), o SimulateOS (FREITAS, 2023) e o SWSO (OLIVEIRA, 2015), cujos layouts dos dois primeiros são mostrados abaixo nas Figuras 2 e 3. Entretanto, eles não comungam dos nossos objetivos de possuir uma interface gráfica mais intuitiva e amigável, como deixa a desejar no Yass, ou ainda, que sejam detalhados em questão de teoria e elementos, como carece no SimulateOS. Além disso, que possibilite permitir um maior desacoplamento com o *backend* do simulador, visando facilidade de manutenção e atualização deste.

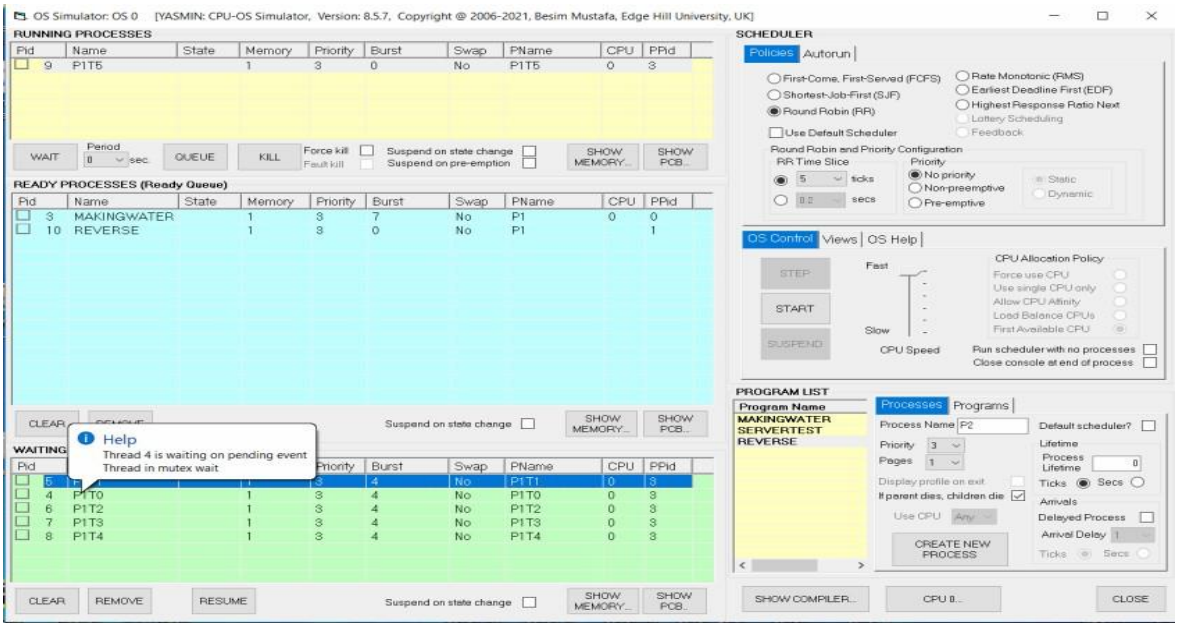

Figura 2 - Layout do Yass

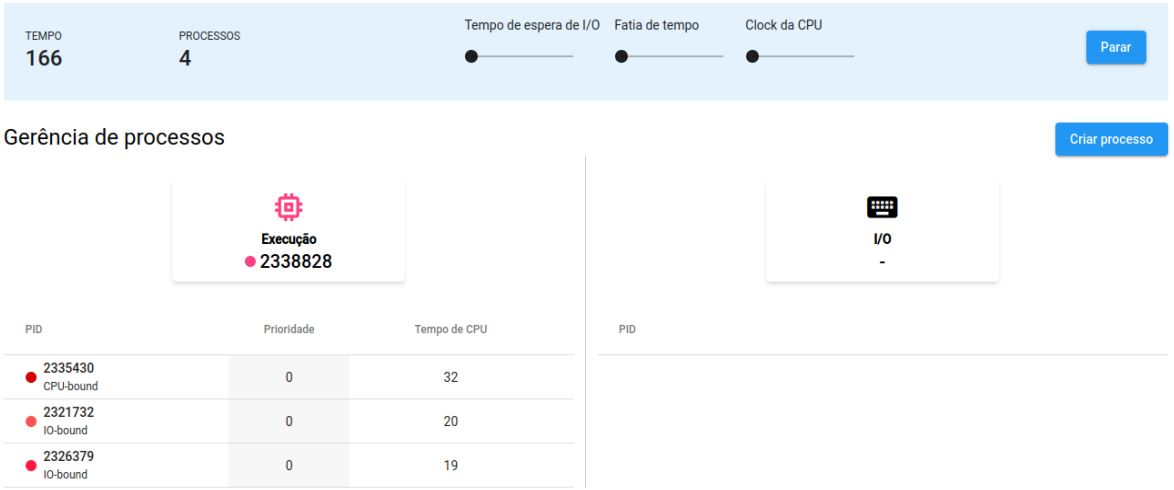

Figura 3 - Layout do SimulateOS[1]

2. METODOLOGIA

A metodologia aplicada para o levantamento dos casos de uso da interface gráfica do nosso simulador, se dá pela análise de outros simuladores, como os citados acima, no intuito de perceber as funcionalidades desejadas e as características a serem evitadas.

[1] SimulateOS: https://fritask.github.io/

Como é possível ver na Figura 2, o Yass já é mais robusto e bastante completo em questão de conteúdo, porém o seu *design* pode parecer complexo demais para um usuário iniciante nos estudos de SO.

Já na Figura 3, pode-se ver o oposto, um simulador mais simples, pois executa em ambiente web, mais intuitivo de entender, porém possui poucas funcionalidades, se afastando muito do funcionamento real de um SO.

No levantamento de requisitos, ficou definido que o objetivo da interface gráfica é mostrar os processadores, processos, rotinas, troca de estados, comportamento da memória, interrupções de sistema, leituras de IO, de forma clara e objetiva, sem poluição do layout, e principalmente, que as transições sejam intuitivas, e não apenas números na tela, como é o caso do Yass.

No entanto, percebe-se a necessidade de apresentar informações mais abrangentes e detalhadas sobre os componentes do SO a fim de evitar a excessiva abstração e simplificação encontrada SimulateOS.

3. RESULTADOS E DISCUSSÃO

Coletando, portanto, as características mais desejáveis de cada simulador conferido na pesquisa bibliográfica, foi possível traçar os requisitos da interface gráfica. Com base neles chegamos ao seguinte layout.

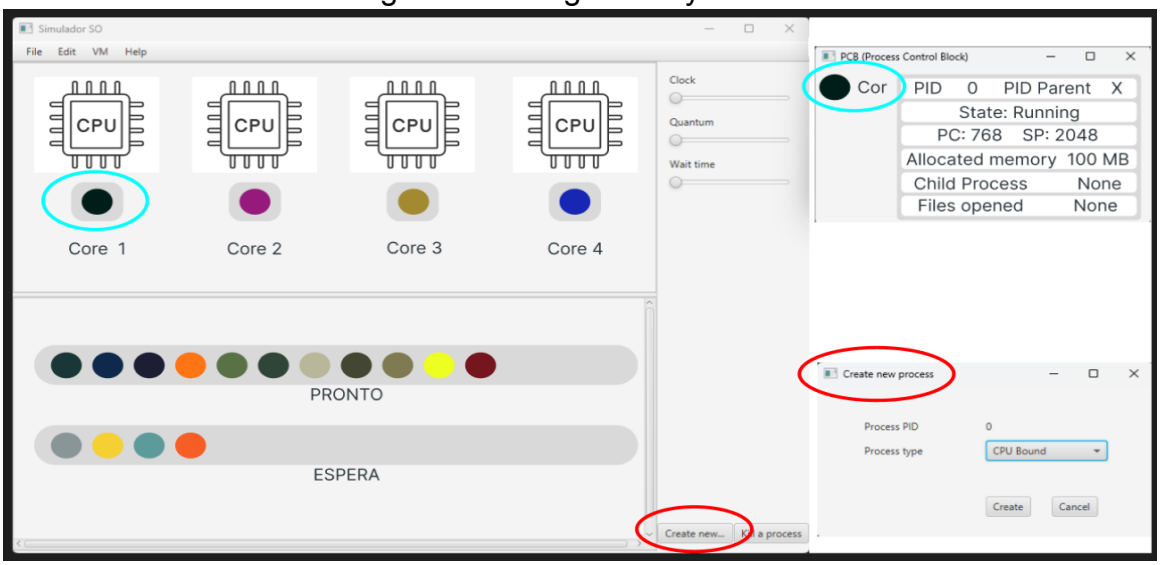

Figura 4 - Layout do PampaOS

A interface irá apresentar suas funcionalidades por abas. Até o momento da escrita desse artigo, já está implementada a aba de processos. Nela podemos ver representações dos *Cores* (Unidades de processamento da CPU - *Central Processing Unit*), cada qual com um espaço para receber uma esfera colorida, as quais representam os processos da simulação. As filas de espera e de pronto vão sendo preenchidas dinamicamente, conforme novos processos forem sendo criados e forem assumindo esses estados. Essa estrutura permite mostrar os fluxos de execução de forma mais clara e objetiva, onde os processos são representados pelas esferas, e os estados são representados pela posição que a esfera ocupa no *layout*. Com essa estrutura pode-se abordar conceitos como preempção, escalonamento, filas de prioridade, interrupções de sistema, e outras rotinas do SO.

Ao iniciar a simulação, não há nenhum processo em execução. Ao clicar no botão “Create new process”, abre-se a janela no canto inferior direito da Figura 4,

onde é possível escolher algumas características do novo processo a ser criado. Ao lado existe o botão que permite encerrar algum processo, fazendo com que saia da rotina de execução.

As informações mais detalhadas dos processos podem ser vistas ao clicar sobre uma esfera, que nesse caso, abrirá a janela do *Process Control Block* (PCB) contendo as informações utilizadas na gerência do processo, o que inclui seu identificador (PID), o identificar de seu processo pai, o seu estado, o ponteiro de pilha (SP), o contador de programa (PC), a quantidade de memória alocada, a quantidade de processos filhos e a quantidade de arquivos abertos.

Com essa abordagem, permite-se que o usuário tenha a possibilidade de escolha sobre a quantidade de informações a serem mostradas, permitindo uma apresentação de dados mais robusta, ao mesmo tempo que a pura execução não se torna um obstáculo para aqueles em estágio mais inicial do aprendizado desses conceitos.

4. CONCLUSÕES

Com base no que foi apresentado, a interface já apresenta um ambiente claro e objetivo, permite a apresentação de informações mais detalhadas, caso o usuário assim deseje, e ainda, permite um alto grau de desacoplamento, visando facilidade em manutenção e atualizações futuras.

Com essas características, percebe-se, que o projeto avança diretamente ao encontro dos objetivos definidos, conforme a abordagem proposta na etapa de planejamento e levantamento de requisitos.

Como proposta de trabalhos futuros, planeja-se implementar as demais abas referentes à memória, ao disco e à entrada e saída de periféricos, com objetivo de explorar de forma ainda mais abrangente as informações envolvidas no entendimento do funcionamento de SO. Ainda espera-se realizar testes de interação humano-computador para coleta de opiniões de usuários da ferramenta.

5. REFERÊNCIAS BIBLIOGRÁFICAS

MAIA, L. P. **SOsim: Simulador para o Ensino de Sistemas Operacionais**. 2001. Dissertação - Informática, Universidade Federal do Rio de Janeiro.

MUSTAFA, B. YASS: A System Simulator for Operating System and Computer Architecture Teaching and Learning. European Journal of Science and Mathematics Education, Mersin Turkey, v.1, n.1, p.34–42, 2013

Oliveira, R.A. SWSO - **Simulador Web de Sistemas Operacionais**. 2015. Monografia - Tecnologia em Análise e Desenvolvimento de Sistemas, Instituto Federal de Educação, Ciência e Tecnologia da Bahia.

FREITAS, G.M.B. **Simulador de gerência de processos para sistemas operacionais**. 2023. Monografia - Sistemas de Informação, Universidade Federal de Uberlândia.