Gerência de Processos no Simulador PampaOS

LUAN MARK DA SILVA BORELA[1]; GABRIEL LEITE BESSA[1]; HECTOR HUDSON DINIZ FERNANDES[1]; JOAO ANTONIO NEVES SOARES[1]; VINICIUS GARCIA PERUZZI[1]; RAFAEL BURLAMAQUI AMARAL[1]

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

1. INTRODUÇÃO

Conforme as diretrizes curriculares nacionais para os cursos de graduação em computação (MEC, 2012), a disciplina de sistemas operacionais é fundamental em cursos de graduação em computação. O estudo desta disciplina desempenha um papel fundamental, proporcionando uma compreensão sobre as estruturas que governam a interação entre o hardware de um sistema computacional e os programas que nele são executados.

Um sistema operacional é um software que atua como uma camada intermediária entre o hardware e os programas do usuário, desempenhando um papel essencial na gerência de recursos do sistema (MAZIERO, 2019).

Sistemas operacionais podem ser softwares bastante complexos e entendê-los por completo não é uma tarefa trivial, por este motivo, existem diversas ferramentas educacionais que buscam expressar esses conceitos de uma forma mais simplificada, como o SOsim (MAIA, 2001), que permite uma visualização intuitiva de gerência de processos e de gerência de memória, ou o YASS (MUSTAFA, 2013), que simula tanto um processador quanto um sistema operacional, permitindo que programas sejam executados nesta arquitetura beneficiando uma compreensão mais completa sobre sistemas operacionais.

Embora os simuladores destacados pareçam ser muito eficientes em seus objetivos, ainda falta a eles a capacidade de combinar dois conceitos importantes para um simulador: a simplicidade para exibir os conceitos desejados e a complexidade para simular detalhes importantes. O SOsim é um simulador bastante simples, o que é bom para o entendimento, porém não possui um processador simulado. Isso limita os conceitos que podem ser explorados em um sistema operacional, como chamadas de sistema ou alocação dinâmica de memória, por exemplo. Por outro lado, o YASS é um simulador bastante complexo, o que permite a simulação de diversos conceitos, mas não é nada intuitivo e demanda um grande esforço do usuário para compreender o funcionamento de cada ponto de sua interface. Um exemplo disso é que o YASS necessita que todos os processos desejados estejam rodando em seu processador para ver a gerência de processos em ação. Isso se torna bastante incômodo, visto que, para criar processos, é necessário entender a linguagem de programação do compilador incorporado, escrever cada um dos programas nesta nova linguagem, compilá-los individualmente, carregá-los para a memória e, por fim, criar uma instância destes processos no sistema operacional simulado.

Este trabalho apresentará o andamento da gerência de processos do projeto "Desenvolvimento de um ambiente para a simulação de Sistemas Operacionais com o propósito de uso educacional" (AMARAL, 2023), que irá simular diversos conceitos complexos de sistemas operacionais sendo simples o suficiente para que graduandos em computação sejam capazes de utilizá-lo de forma eficiente, permitindo-os inclusive reescrever políticas de escalonamento de processos.

2. METODOLOGIA

A primeira etapa do projeto consistiu em realizar uma revisão bibliográfica sobre simuladores de sistemas operacionais já existentes (MUSTAFA, 2013; MAIA, 2001; REIS, 2009; CHRISTOPHER, 1993), analisando suas principais contribuições, além de um estudo sobre a gerência de processos do sistema operacional MINIX (TANENBAUM, 1997). É importante destacar que, durante essa revisão, foi identificado que apenas um dos simuladores em funcionamento era capaz de simular tanto um processador quanto um sistema operacional, conforme também relatado no artigo do simulador YASS (MUSTAFA, 2013). Nesta etapa de aprofundamento teórico, foi criada uma tabela de requisitos funcionais que servirá como base para o desenvolvimento do simulador. A Tabela 1 apresenta os principais requisitos funcionais da gerência de processos.

GERÊNCIA DE PROCESSOS	
Código	**Requisito Funcional**
RF01	O simulador permitirá a criação de diversos processos, organizando-os em filas
RF02	Cada processo terá seu próprio PCB (Process Control Block), contendo: O PID do processo, os estados do processo, o contador de programa, o ponteiro da pilha, sua alocação de memória, informações sobre o pai do processo e de seus filhos
RF03	Os processos terão os estados: Novo, Executando, Pronto e Esperando
RF04	Um escalonador de processos será responsável por gerenciar o acesso de cada processo ao processador
RF05	O escalonador permitirá que novas políticas de escalonamento sejam criadas
RF06	O escalonamento de processos deverá ser feito para múltiplos núcleos.
RF07	O usuário poderá utilizar políticas personalizadas, explorando diferentes formas de escalonamento
RF08	O usuário poderá criar processos abstratos (que não executam instruções verdadeiras) definindo quantas instruções do mesmo serão CPU-bound e quantas serão IO-bound
RF09	Os processos terão uma hierarquia, onde um processo pode ser pai de outro e processos filhos podem fazer parte de um grupo de processos
RF10	A inicialização do sistema será simulada de forma similar à inicialização do Minix 3 - iniciando serviços
RF11	Os processos podem ser programas reais executando em um processador
RF12	Um processador multinúcleo simulado será o responsável por executar os processos escalonados
RF13	O processador simulado será capaz de executar instruções assembly e terá suporte a chamadas de sistema.

Tabela 1. Requisitos funcionais do simulador PampaOS.

3. RESULTADOS E DISCUSSÃO

Com base na análise dos requisitos, um diagrama de classes preliminar foi gerado, buscando expressar as principais classes do sistema e seus relacionamentos, a Figura 1 apresenta o diagrama de classes obtido, ressaltando em verde as classes envolvidas na gerência de processos.

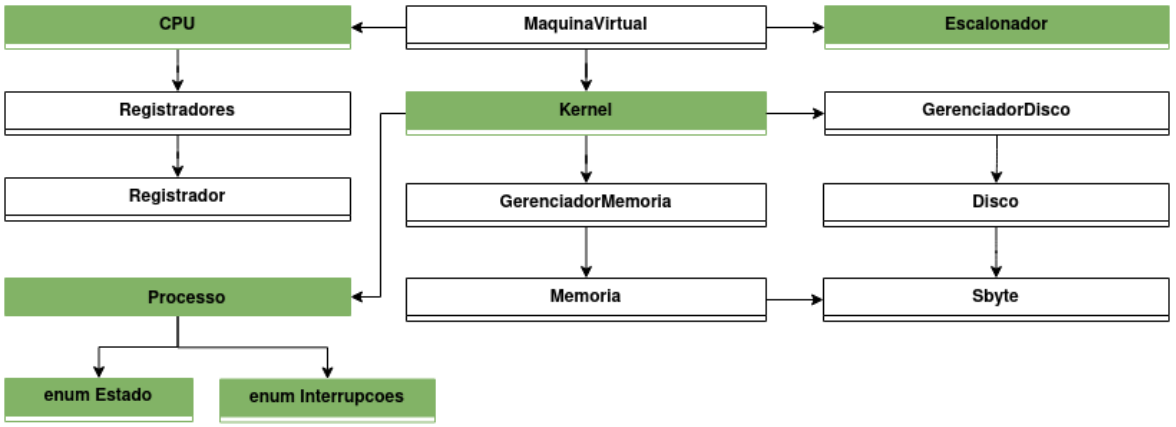

Figura 1. Diagrama de classes simplificado do PampaOS.

Com o diagrama de classes preliminar e os requisitos funcionais descritos, foi possível partir para a etapa atual do projeto, que é a implementação do simulador. Dito isso, os requisitos atendidos até o momento foram os seguintes:

- **RF02:** A classe Processo contém as informações de PCB (Process Control Block) de um processo.
- **RF01** e **RF03:** A classe Kernel contém as listas de processos e seus estados.
- **RF04** e **RF06:** A classe escalonador possui os objetos das listas de processos, sendo capaz de alterá-las para escalonar os processos em execução.
- **RF06, RF11, RF12** e **RF13:** A máquina virtual é capaz de instanciar múltiplos núcleos.
- **RF14:** A máquina virtual é capaz de detectar a ocorrência de chamadas de sistemas através do enum de interrupções presente no PCB do processo, sendo também a classe responsável por lidar com a chamada de sistema.
- **RF08:** O núcleo (CPU) responsável pela execução do processo abstrato pode executar instruções assembly sem efeito prático, não interferindo em outras partes do sistema, mas permitindo a visualização do escalonamento de processos e da gerência de memória de forma prática

O escalonador atualmente é responsável por gerenciar listas de processos correspondentes aos estados Novo, Executando, Pronto e Esperando. Sua principal responsabilidade é decidir quando e quais processos da lista de Pronto serão transferidos para a lista de Executando. Esta decisão é feita baseada na quantidade de núcleos disponíveis e na política de escalonamento implementada. A política de escalonamento foi concentrada em um único método chamado “Schedule”, o que permite aos usuários alterar o código e as políticas implementadas com facilidade, sendo necessário apenas mudar a lógica de escolha dos processos da lista de Prontos.

Até este ponto do projeto, o simulador é capaz de escalonar processos para múltiplos núcleos e executar instruções vazias ou instruções assembly

pré-carregadas na memória stack (pilha) do processo. No entanto, ainda existem partes importantes para uma boa simulação de processos no PampaOS que ainda serão implementadas, como o carregador e o montador de instruções, assim como mais chamadas de sistema.

4. CONCLUSÕES

O projeto PampaOS ainda está em desenvolvimento e tem como objetivo criar um simulador de sistemas operacionais simples e modular, o qual pode ser facilmente compreendido. Ele é composto por diversos componentes que operam em conjunto, mas que podem ser utilizados de forma independente de acordo com o conceito a ser simulado. Isso permite uma visão abrangente da gerência de processos, desde a interação com a máquina virtual até o escalonamento propriamente dito, proporcionando um conteúdo completo. Espera-se que este simulador contribua significativamente para a educação e a pesquisa na área de sistemas operacionais, permitindo que estudantes compreendam e experimentem conceitos importantes.

5. REFERÊNCIAS BIBLIOGRÁFICAS

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

MAZIERO, C. **Sistemas Operacionais: Conceitos e Mecanismos**. Editora da UFPR, 2019. 456 p. ISBN 978-85-7335-340-2.

AMARAL, R. **Desenvolvimento de um ambiente para a simulação de Sistemas Operacionais com o propósito de uso educacional.** UFPEL. 2023. Acessado em 9 set. 2023. Online. Disponível em: https://institucional.ufpel.edu.br/projetos/id/u6453

Ministério da Educação – MEC. **Diretrizes Curriculares Nacionais para os cursos de graduação em Computação**. 2012. Acessado em 9 set. 2023. Online. Disponível em: http://portal.mec.gov.br/escola-de-gestores-da-educacao-basica/323-secretarias-112877938/orgaos-vinculados-82187207/12991-diretrizes-curriculares-cursos-de-graduacao

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

Reis, F. P., J., P. A. P., C., H. A. X., & Integradas–Outsourcing, S. B. S. 2009. TBC-SO/WEB: Um software educacional para o ensino de políticas de escalonamento de processos e de alocação de memória em sistemas operacionais. **Simpósio Brasileiro de Informática na Educação-SBIE**.

CHRISTOPHER, Wayne A.; PROCTER, Steven J.; ANDERSON, Thomas E. The Nachos instructional operating system. In: **USENIX Winter**. 1993. p. 481-488.