Sistema Operacional Simulado: Ferramenta para o Ensino de Graduação

Mauro Mulati[*], Rogério Gonçalves[*], Valdemir Silva, Ronaldo Gonçalves

Departamento de Informática - Universidade Estadual de Maringá (UEM)
Zona 07 - Av. Colombo 5790 - CEP 87020-900 - Maringá – PR

{mhmulati,rogerio,valdps,ronaldo}@din. uem.br

Abstract: *This work presents SOS, a simulation tool for operating systems directed to undergraduate courses. SOS has four integrated components: a virtual processor, an assembler, a simulated operating system and a graphical user interface. SOS allows developing, assembling and executing several programs on the virtual processor in multiprogramming fashion, where the operating system manages concurrent processes, virtual memory, file system and I/O. SOS allows visualizing graphically the internal structures and actions of the system, as well as performance statistics. Besides, it has configuration facilities, being a fundamental tool for the undergraduate teaching.*

Resumo: *Este trabalho apresenta SOS, uma ferramenta de simulação de sistemas operacionais direcionada ao ensino de graduação. SOS possui quatro componentes integrados: um processador virtual, um montador, um sistema operacional simulado e uma interface gráfica com o usuário. SOS permite desenvolver, montar e executar programas sobre o processador virtual de forma multiprogramada, onde o sistema operacional gerencia processos concorrentes, memória virtual, sistema de arquivos e E/S. SOS permite visualizar graficamente as estruturas e ações internas do sistema, bem como estatísticas de desempenho. Além disso, ele possui facilidades de configuração, sendo uma ferramenta fundamental para o ensino de graduação.*

1. INTRODUÇÃO

A disciplina de Sistemas Operacionais faz parte do currículo mínimo defend ido para os cursos de graduação da área de ciência da computação, devido a sua importância na integração dos conhecimentos de software e hardware. Além disso, o conhecimento prático da mesma se torna uma necessidade inerente ao desenvolvimento de sistemas computacionais, sendo difícil imaginar atualmente um profissional da área desconhecedor desta temática.

[*] Alunos Bolsistas PET/SESU

Uma das dificuldades no ensino desta disciplina está no fato da mesma ter uma grande carga teórica, que deixa muitos alunos desmotivados. Embora em muitos cursos ela seja ministrada também em laboratório, alguns problemas ainda persistem: 1) nem sempre os cursos estão equipadas adequadamente; 2) nem sempre os laboratórios podem ser reservados para a disciplina, de forma a preservar as configurações necessárias; 3) nem sempre os professores têm o perfil prático necessário e 4) a disciplina continua sempre precisando de teoria antes da prática.

Com o objetivo de auxiliar no entendimento da disciplina de Sistemas Operacionais, permitindo que os alunos possam visualizar as estruturas internas e os efeitos das suas funções de gerenciamento, bem como desenvolver aplicativos que possam ser escalonados e executados de forma multiprogramada, o SOS (Sistema Operacional Simulado) foi desenvolvido baseado nos conceitos de sistemas operacionais modernos [1] e é apresentado neste trabalho, o qual apresenta características visuais e funcionais apropriadas ao ensino de graduação.

Este trabalho está organizado da seguinte forma. A sessão 2 apresenta uma breve descrição de trabalhos relacionados. A sessão 3 apresenta o projeto do simulador SOS e da Arquitetura de Hardware. A sessão 4 apresenta a implementação da arquitetura de Hardware Virtual. A sessão 5 apresenta a implementação do Simulador SOS e na sessão 6 é apresentada uma parte de experimentação. As conclusões aparecem na sessão 7 e finalmente, as referências bibliográficas aparecem na última sessão.

2. TRABALHOS RELACIONADOS

Vários trabalhos relacionados ao ensino de sistemas operacionais são encontrados na literatura. Alguns possuem facilidades instrucionais para utilização real em laboratório enquanto outros são simulados e apresentam facilidades visuais. Em [2] Mark Claypool e outros trabalharam com o Fossil Lab, um laboratório de máquinas clientes dedicadas a executarem código aberto do sistema operacional Linux. As máquinas clientes são conectadas a um servidor que fornece "boot" remoto e download do código aberto do Linux. Os estudantes podem baixar localmente o código fonte do Linux, modifica-lo, compila-lo e executa-lo localmente para verificar na prática as conseqüências. Localmente, os estudantes podem desenvolver projetos de sistemas operacionais com permissão de supervisor.

Christopher e outros desenvolveram Nachos [3], um sistema operacional simulado que executa no nível do usuário como um processo do Unix. Nachos permite simular a CPU, dispositivos, processos, threads, RPC, hierarquia de memória, programação orientada a objetos e computação distribuída. Possui interface amigável e é direcionado ao ensino de graduação da disciplina de sistemas operacionais, permitindo ilustrar conceitos e princípios importantes. Outro projeto interessante, chamado PortOS [4], foi desenvolvido para complementar o ensino de graduação e de pós-graduação para a disciplina de Sistemas Operacionais. PortOS é uma ferramenta que guia os próprios alunos no desenvolvimento de um sistema operacional para equipamentos móveis do tipo PDA, permitindo trabalhar com threads, concorrência, escalonamento, sistema de arquivos, sincronização e comunicação ponto a ponto. PortOS visa prover uma plataforma de software que permita aos estudantes desenvolver um sistema operacional real, em nível de usuário e com toda a segurança. Recentemente, em 2003, dois trabalhos se destacaram nesta temática.

Wenliang Du [5] desenvolveu o SMINIX, uma extensão do sistema operacional Minix que surgiu

com a adição de módulos que implementam mecanismos de segurança. Esta ferramenta permite aos estudantes praticarem suas habilidades com o projeto, implementação, experim entação e análise das questões de segurança relacionadas a autenticação, controle de acesso, criptografia, sandboxing, vulnerabilidade e privilégios. Os alunos recebem uma versão simplificada do SMINIX, faltando a implementação de determinado mecanismo de segurança, e tentam desenvolve-lo em laboratório. Em [6], Maia e Pacheco apresentam um simulador de sistema operacional com capacidades visuais, chamado SOsim, capaz de gerenciar processos e memória de forma multiprogramada, mostrando a dinâmica interna do sistema operacional. O simulador foi apresentado e avaliado por alunos de graduação, que concluíram que o simulador facilita o aprendizado dos conceitos. A interface é bastante pobre e apenas alguns módulos do simulador foram implementados. Além disso, SO sim não executa aplicações que possam ser desenvolvidas sobre algum tipo de hardware simulado.

3. ESPECIFICAÇÃO DO SIMULADOR SOS

SOS foi especificado em duas grandes etapas. Na primeira etapa foi definido um processador simulado para um conjunto próprio de instruções e um montador de mnemônicos. Na segunda etapa foi definido o sistema operacional simulado propriamente dito, sendo este composto de vários módulos de gerenciamento comumente existentes em um sistema operacional, além de uma interface gráfica. A estrutura em camadas da ferramenta SOS se apresenta tal como na Figura 1, que mostra seus principais componentes e relacionamentos.

Figura 1. SOS em Camadas

De uma forma geral, durante a inicialização do simulador (boot) os processos do sistema são criados e iniciam a execução multiprogramada. Os processos são escalonados da fila de prontos e colocados para execução no processador. Um destes processos, o Gerente de Prompt, verifica por requisições do usuário na linha de comando. A qualquer momento o usuá rio poderá solicitar a execução de seus próprios processos (programas, aplicativos) que são então criados e passam a concorrer pelo processador juntamente com os demais processos do sistema. Os processos do usuário podem então executar operações de E/S e serem bloqueados em filas apropriadas à espera de atendimento. Estas filas são tratadas por processos específicos do sistema, quando estes forem

escalonados para execução, podendo ser o Gerenciador de Memória ou o Gerenciador de Disco. Apesar de aparecer na figura, o Gerenciador de Vídeo ainda não foi implementado.

3.1 Processador Simulado e Montador

Para permitir o desenvolvimento de aplicações que possam ser executadas sobre o sistema operacional simulado, um processador simulado simples foi projetado pa ra um conjunto mínimo de instruções. O processador possui um conjunto pequeno de registradores e pode executar instruções binárias em um formato específico. O ciclo básico de instrução envolve continuamente a busca de uma instrução na memória, a interpreta ção desta instrução, a execução desta instrução e o reinício do ciclo para o próximo endereço. As instruções são de tamanho fixo, compostas de um código de operação e dois operandos. A Tabela 1 mostra o conjunto de instruções SOS.

Tabela 1. Conjunto de Instruções SOS

Conjunto de Instruções		
Opcode	***Mnemônico***	**Descrição**
1	addd reg1 dado	acrescenta o dado ao conteúdo de reg1.
2	addr reg1 reg2	acrescenta o conteúdo de reg2 ao conteúdo de reg1.
3	subd reg1 dado	subtrai o dado do conteúdo de reg1.
4	subr reg1 reg2	subtrai o conteúdo de reg2 do conteúdo de reg1.
5	loadd reg1 end	carrega o conteúdo da posição da memória end e o coloca em reg1.
6	loadr reg1 reg2	carrega o conteúdo da posição de memória apontada por reg2 em reg1.
7	stored reg1 end	armazena o conteúdo de reg1 na posição de memória end.
8	storer reg1 reg2	armazena o conteúdo de reg1 na posição de memória apontada por reg2.
9	movd reg1 dado	copia o dado em reg1.
10	movr reg1 reg2	copia o conteúdo de reg2 em reg1.
11	jumpd end1	altera regPC para o valor end1.
12	jumpr reg1	altera regPC para o valor contido em reg1.
13	jumpnd end1 reg1	altera regPC para end1 caso o conteúdo de reg1 seja negativo
14	jumpnr reg1 reg2	altera regPC para o conteúdo de reg1 se o conteúdo de reg2 for negativo
15	jumpzd end1 reg1	altera regPC para end1 se o conteúdo de reg1 for zero
16	jumpzr reg1 reg2	altera regPC para o valor contido em reg1 caso reg2 seja zero
17	read reg1	lê um valor numérico da entrada padrão (disco) e o coloca em reg1.
18	write reg1	escreve o valor numérico contido em reg1 na saída padrão (disco).
19	writes literal	escreve o literal contido em reg1 na saída padrão.
20	Exit	informa término normal do programa.

A escrita de programas é viabilizada por um montador que foi também projetado para converter arquivos escritos em linguagem de mnemônicos para arquivos executáveis no ambiente do simulador. O montador lê o arquivo fonte em texto e o converte para o código executável binário correspondente.

3.2 Sistema Operacional Simulado

O Sistema Operacional simulado é composto por um conjunto de processos gerenciadores disparados durante a inicialização do sistema, os quais são controlados pelo Gerenciador de Processador. Todos os processos gerenciadores são escalonados da fila de prontos, pelo Escalonador de Prontos, e concorrem ao uso do processador. O Escalonador de Prontos pode ser

configurado para atuar segundo uma entre três políticas bem conhecidas: *round-robin*, prioridade ou *shortest job first*. Outros escalonadores existentes atuam sobre outras filas que o sistema operacional gerencia, entre eles, o Escalonador de Disco, que atua sobre a fila de bloqueados para disco e o Escalonador de Memória, que atua sobre a fila de bloqueados para memória.

O Gerenciador de Prompt ao entrar em execução procura por algo digitado pelo usuário na linha de comando (que estará no buffer do teclado), captura esta informação, interpreta o seu significado e executa uma ação associada. Basicamente, esta ação é relacionada ao gerenciamento de processos ou de arquivos, tal como a execução de aplicações, que leva a criação de processos do usuário, ou a listagem de diretório, que permite visualizar o diretório do ambiente do simulador.

O Gerenciador de Memória usa a técnica de paginação para gerenciar a memória de forma virtual. Ele basicamente atua sobre uma fila de processos bloqueados para memória e sobre a própria memória, gerenciando os espaços livres/ocupados, segundo a política de substituição de páginas definida (FIFO, LRU e LFU), e controlando as tabelas de páginas dos processos. O Gerenciador de Memória é composto pelo Escalonador de Memória, que decide qual processo da fila de bloqueados para memória será atendido. Um processo do usuário pode ser colocado na fila de bloqueados para memória quando o processador tentar executar uma instrução do processo cuja página ainda não está carregada na memória.

O Gerenciador de Disco basicamente atua sobre uma fila de processos bloqueados para disco e sobre a estrutura organizacional do disco, gerenciando os espaços livres e ocupados segundo uma estrutura própria de diretório. O Gerenciador de Disco é composto pelo Escalonador de Disco, que decide qual processo da fila de bloqueados para disco será atendido. Um processo do usuário pode ser colocado na fila de bloqueados para disco quando o processador tentar executar uma instrução do processo de leitura ou gravação em arquivo.

3.2.1. Processos

No ambiente SOS os processos são representados por um descritor chamado BCP (Bloco de Controle de Processo), o qual é uma tabela alocada durante a criação dos processos e inserida na fila de prontos. Neste BCP estão armazenadas informações tais como o nome do processo, o estado do processador (registradores), o ponteiro para sua tabela de páginas e outras informações. Os processos do sistema são criados automaticamente durante o boot e os processos do usuário são criados quando o usuário solicita a execução de uma aplicação na linha de comando. Os processos do usuário são escritos em linguagem de mnemônico pelo próprio usuário, para finalidades que interessam ao usuário, e depois são montados para a linguagem de máquina do simulador. Já os processos do sistema são procedimentos já definidos e que desempenham funções de gerenciamento pré-determinadas.

Os BCPs dos processos do usuário transitam entre as diferentes filas do sistema, dependendo das políticas do sistema operacional e das solicitações de recursos feitas pelos processos. Quando o processo é retirado da execução pelo processador ocorre a troca de contexto. A troca de contexto pode ocorrer devido à execução de uma instrução de entrada/saída que bloqueia o processo ou devido ao término de sua fatia de tempo de execução (*time slice*) ou a chegada de um processo

com maior prioridade ou a chegada de um processo com menor tempo de execução previsto, situações estas que dependem do algoritmo de escalonamento utilizado.

Durante a troca de contexto, os conteúdos dos registradores do processador são salvos no BCP do processo que está sendo trocado e este é movido para uma outra fila. Então, um outro processo é escalonado da fila de prontos para execução e os conteúdos dos registradores do processador são restabelecidos com os valores armazenados neste novo BCP, para que o novo processo reinicie a sua execução, que havia sido interrompida em sua última troca de contexto, como se nada houvesse acontecido. Os processos gerentes são executados sem interrupção até que completem o atendimento a um processo da respectiva fila de bloqueados, não precisando assim de troca de contexto.

3.2.2. Memória Virtual

O Gerenciador de Memória é responsável pelo controle da memória e das alocações/liberações de páginas. Quando um processo é criado, uma tabela de páginas é alocada para ele e um ponteiro para ela é inserido no BCP do processo. Esta tabela de páginas é manipulada pelo Gerenciador de Memória. Sempre que o processador tenta executar a próxima instrução de um processo, ele primeiramente verifica na tabela de páginas do processo se a página virtual que contém a referida instrução está alocada na memória física. Caso esteja, o processador obtém da tabela o número da página real, calcula o endereço efetivo da memória física, busca a instrução e a executa. Caso não esteja, ocorre uma interrupção por falta de página. Neste caso, o processo é trocado para a fila de bloqueados para memória, para ser atendido em momento oportuno pelo Gerenciado de Memória.

Quando o Gerenciador de Memória é escalonado para execução, ele percorre a fila de processos bloqueados para memória e seleciona um processo segundo o algoritmo de escalonamento vigente, o qual é tratado sem interrupção. Observando as informações do BCP do processo escolhido, o Gerenciador de Memória detecta qual página está sendo solicitada, providencia a sua carga na memória e atualiza a tabela de pág inas do processo para refletir esta nova situação. Após este tratamento, o Gerenciador de Memória move o BCP do processo tratado para a fila de prontos (para que o mesmo seja novamente escalonado para a execução, tendo agora sua página já carregada na memória) quando então volta também para a fila de prontos para ser escalonado futuramente para o tratamento de outro processo bloqueado para memória.

Durante a alocação de páginas da memória, caso não exista página livre, um algoritmo de substituição de págin as é usado liberar uma página ocupada por um outro processo, que poder ser configurado em FIFO, LRU ou LFU. Este procedimento de escolha da página de memória é feito em dois níveis: no primeiro nível (nível de processo) o gerenciador verifica qual processo possui o maior número de página carregadas na memória e em segundo nível (nível de tabela de páginas) a tabela de páginas do processo selecionado é acessada e uma das páginas carregadas é eleita, de acordo com a política de substituição. No primeiro nível , caso todos os processos tenham o mesmo número de páginas carregadas na memória, o processo a ser escolhido é o próprio processo que está solicitando uma página. Assim, o gerenciador carrega a página solicitada e o coloca o BCP do processo requisitante na fila de prontos, para esperar a sua vez de executar novamente.

3.2.3. Estrutura de Diretório

O Gerenciador de Disco atua sobre a fila de processos bloqueados para disco e atende as operações de leitura e escrita feitas pelos processos. Ao ser escalonado para a execução o Gerenciador de Disco verifica quais processos estão aguardando atendimento na referida fila e seleciona um processo segundo o algoritmo de escalonamento vigente, o qual é tratado sem interrupção. Observando as informações do BCP do processo escolhido, o Gerenciador de Disco detecta qual operação está sendo solicitada e providencia a sua execução, transferindo dados entre a memória do processo e o buffer associado ao dispositivo de disco. Após este tratamento, o Gerenciador de Disco move o BCP do processo tratado para a fila de prontos, quando então volta também para a fila de prontos para ser escalonado futuramente para o tratamento de outro processo bloqueado para disco.

O Gerenciador de Disco também é o responsável pela organização e controle da estrutura de diretório, a qual é implementada como uma árvore n-ária, conforme mostra a Figura 2, onde cada nó pode ser um arquivo ou diretório. Essa árvore contém os diretórios, subdiretórios e arquivos organizados hierarquicamente. As entradas que correspondem aos diretórios apontam para entradas que são seus subdiretórios e para as entradas dos arquivos do seu nível. Em cada entrada correspondente a um arquivo existem ponteiros ou endereços que apontam para os blocos que formam o arquivo e que estão armazenados no disco virtual. Existe uma árvore oculta ao usuário, que contém os números dos blocos livres no disco virtual. Tanto a árvore de diretório quanto a árvore de blocos livres são gravada com arquivos do disco real. Na medida que os blocos são requeridos, o sistema procura por bloco livre na árvore de blocos livres, retirando esse bloco desta árvore e inserindo-o em sua respectiva referência na árvore n-ária que contém os blocos ocupados.

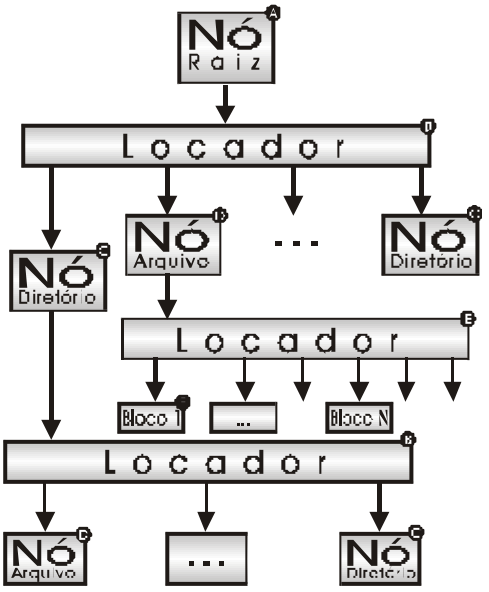

Figura 2. Estrutura Geral do Sistema de Arquivos

4. IMPLEMENTAÇÃO DO SIMULADOR SOS

O simulador SOS foi implementado em C++. Apesar de executar programas binários, todo o ambiente de execução é simulado. O hardware é estruturado em classes e objetos, sendo de fato representado por variáveis, registros, vetores e tabelas, além de estruturas mais elaboradas de programação tais como pilhas e filas dinâmicas. Da mesma forma, as funções de gerenciamento do sistema operacional são implementadas por métodos. A interface geral do simulador é organizada de forma a mostrar os componentes do sistema, conforme mostra a Figura 3.

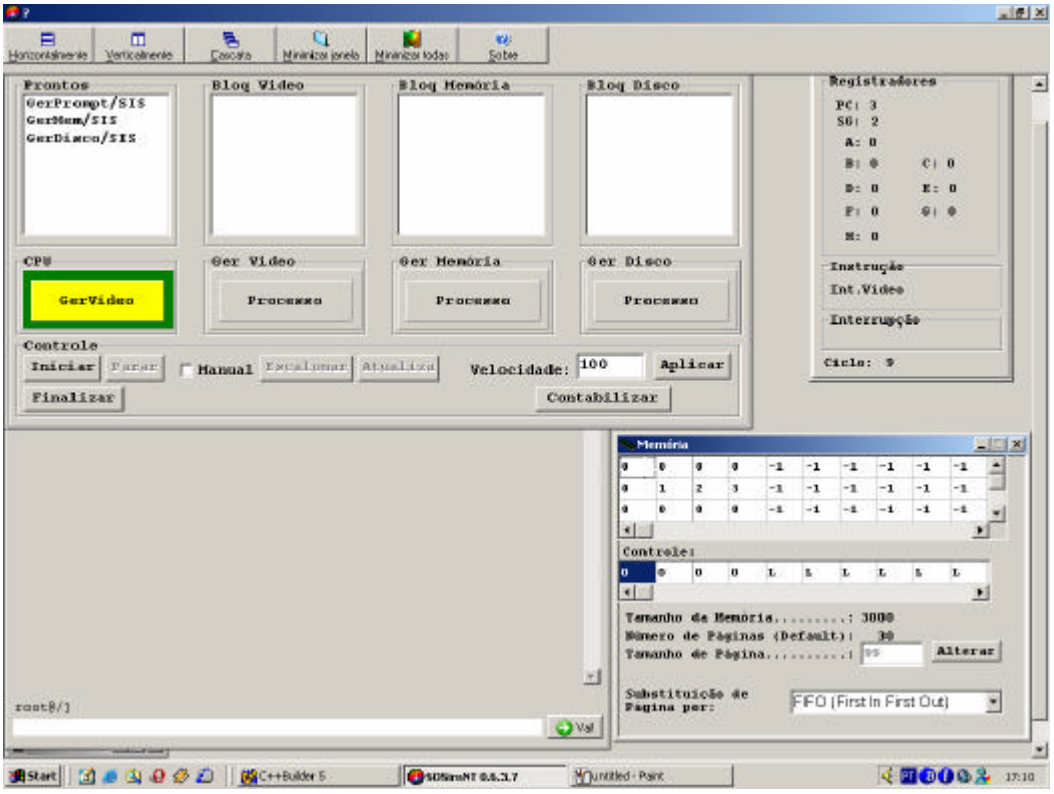

Figura 3. Tela com os Componentes do Simulador

A memória do sistema foi implementada como um vetor (*array*) que contém uma configuração inicial de 3000 posições. Sua manipulação é feita pelos métodos implementados para sua classe. A Figura 4 mostra alguns detalhes da interface apresentada no simulador, a qual permite ao usuário visualizar a situação da memória.

Esta figura permite visualizar as colunas na parte superior que representam as páginas de memória. Na área central, há o controle de páginas livres e ocupadas (O - ocupada , L - livre). Na parte inferior aparecem as configurações para o tamanho da memória, o número de páginas e o tamanho de cada página. Com a alteração do tamanho da página o número de páginas é automaticamente readequado, não sendo este reajuste permitida em tempo de execução. A Figura 5 mostra detalhes para a configuração da política de substituição de páginas.

Figura 4. Representação da Memória e Suas Propriedades

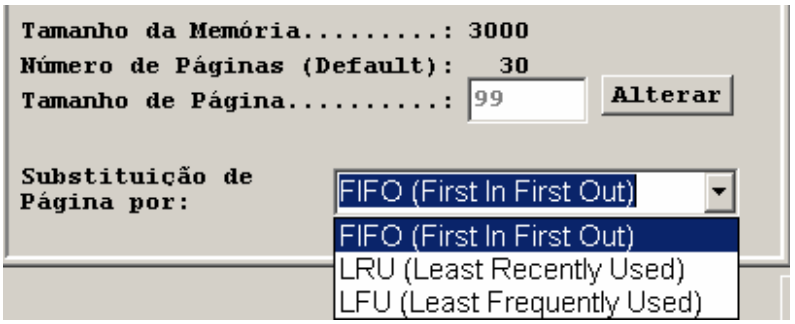

Figura 5. Configuração da Técnica de descarte de Página

A tela de prompt (Figura 6) permite a execução de todos os comandos do sistema, tais como criar diretórios, listar arquivos no sistema, executar programas, entre outros. Os comandos são digitados na parte inferior da tela. Na parte superior são visualizados todos os comandos executados, mensagens de erros e mensagens do sistema. Por exemplo, se for executado o comando LISTAR, veremos os arquivos e diretórios no sistema. O disco virtual é um arquivo binário, tratado de forma simulada como um disco rígido real.

O módulo de interface de apresentação da execução, mostrado na Figura 7, possibilita o acompanhamento das transições de estados de todos os processos (sistema e usuário), bem como as transições entre as filas de prontos, de bloqueados para a memória e de bloqueados para acesso ao disco. A tela está dividida em 4 partes, mostrando as filas de processos, cada uma referente a uma situação de processo. No topo de cada parte, estão os processos que estão na fila aguardando o atendimento, e na parte inferior, se encontra o processo que está sendo atendido. No caso da CPU, é apresentado o rótulo do processo que está em execução.

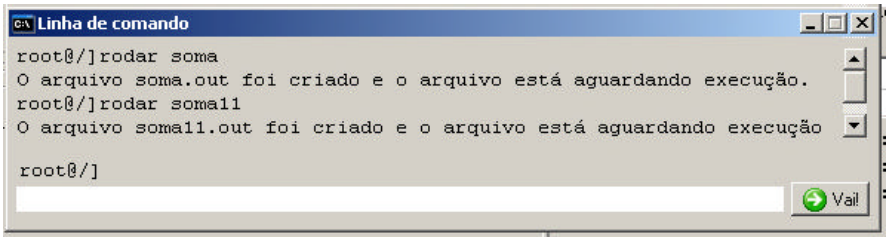

Figura 6. Janela do Prompt

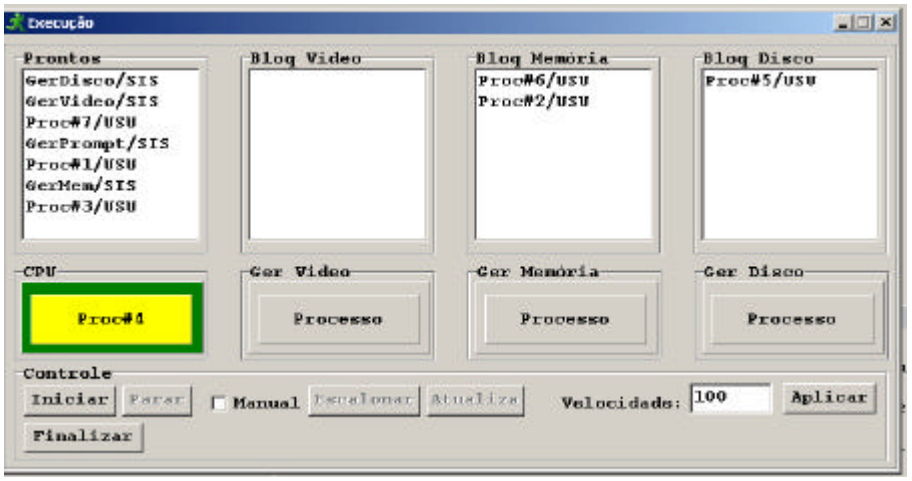

Figura 7. Tela de Execução

Ainda nesta interface, é possível controlar a execução do Simulador. Tem-se o botão INICIAR para iniciar a execução. A velocidade de simulação é controlada para que o usuário possa acompanhar a execução dos processos, podendo ser alterada na opção apresentada na tela. Podemos pausar a execução através do botão PARAR e finalizá-la no botão FINALIZAR. Assim é possível acompanhar passo a passo o funcionamento e o trânsito dos processos entre as filas.

O processador foi implementado através de uma classe. Desta forma é possível utilizar as estruturas como registradores e os métodos definidos para o processador para executar o código dos processos do sistema e do usuário, conforme mostra a Figura 8. Nesta tela é apresentado o conteúdo de todos os registradores do processador e também qual instrução está sendo executada pelo processador. Na parte inferior da tela (INTERRUPÇÃO) é possível acompanhar as interrupções que ocorrem no decorrer da execução de um processo.

SOS possibilita a geração de estatísticas de desempenho das aplicações que estão sendo executadas. Para avaliar esta funcionalidade foi executada uma bateria de testes com aplicações que estressam a memória. Para cada um dos 3 algoritmos de substituição de páginas foi coletado o número de processos bloqueados na fila de memória a cada ciclo até o término das aplicações. As Figuras 9, 10 e 11 mostram estes resultados. Para este exemplo, o algoritmo LRU proporcionou um melhor aproveitamento da memória.

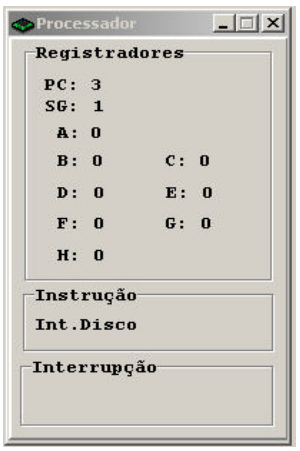

Figura 8. Contexto do Processador

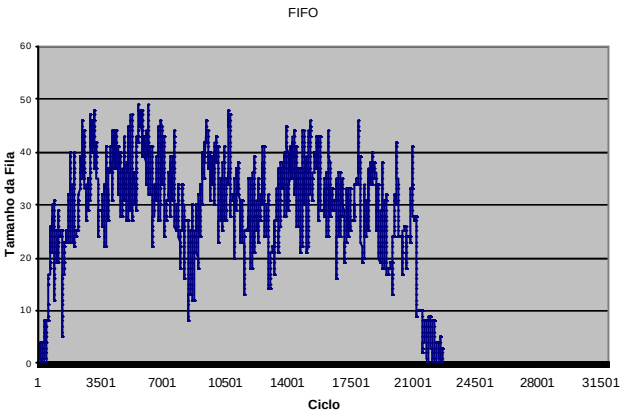

Figura 9. Saturação da Memória Segundo o Algoritmo FIFO

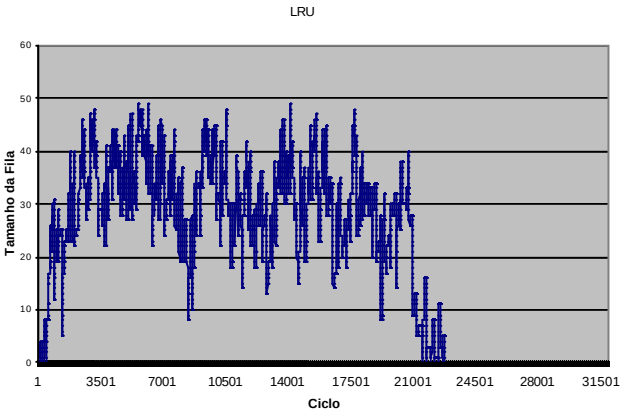

Figura 10. Saturação da Memória Segundo o Algoritmo LRU

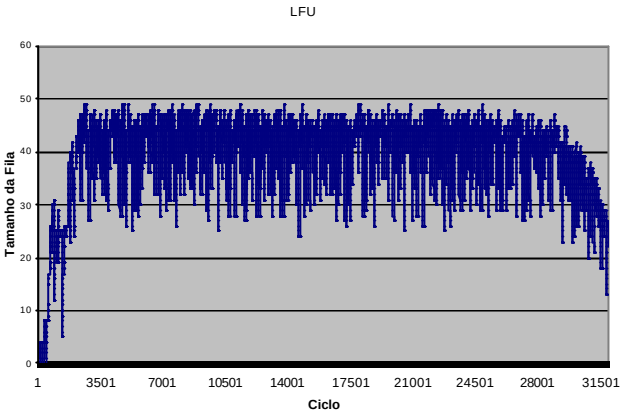

Figura 11. Saturação da Memória Segundo o Algoritmo LFU

5. CONCLUSÃO E TRABALHOS FUTUROS

Este trabalho apresentou SOS, uma ferramenta de simulação de sistemas operacionais que permite aos alunos desenvolver aplicações para a solução de problemas reais, podendo executá-las em um sistema operacional multiprogramado e visualizar o escalonamento de processos, memória, disco e processador. O presente simulador foi apresentado aos alunos do curso de Informática da UEM e o mesmo mostrou-se adequado ao ensino da disciplina de sistemas operacionais. De uma forma geral, as capacidades visuais e funcionais se mostraram altamente eficiente e não temos encontrado nenhum simulador equivalente na literatura. Além disso, SOS possibilita a geração de estatísticas de desempenho, inserindo o presente simulador no contexto de ferramenta de avaliação e desempenho. Na continuidade deste trabalho, será desenvolvido o módulo de gerenciamento de Vídeo e além disso, o conjunto de instruções será substituído pelo conjunto de instruções do processador Intel 8085, trabalho este que já foi iniciado.

6. REFERÊNCIAS BIBLIOGRÁFICAS

[1] Andrew S. Tanenbaum, Sistemas Operacionais Modernos. Prentice-Hall, 2003.
[2] Mark Claypool, David Finkel, Craig Wills, An Open Source Laboratory for Operating Systems Projects, Computer Science Department Worcester Polytechnic Institute 100 Institute Road Worcester, MA 01609
[3] W. A. Christopher and S. J. Procter and T. E. Anderson. The Nachos Instructional Operating System. Technical Report UCB//CSD-93-739, University o California, Berkeley, April 1993.
[4] Benjamin Atkin Emin Gn Sirer, PortOS: An Educational Operating System for the Post-PC Environment. Computer Science Department Cornell University. Proceedings of the 33rd ACM Technical Symposium on Computer Science Education (SIGCSE) .
[5] Wenliang Du. Using An Instructional Operating System In Teaching Computer Security Courses. 7th Colloquium for Information Systems Security Educa tion (CISSE), 2003.
[6] L. P. Maia and A. C. Pacheco. A Simulator Supporting Lectures on Operating Systems. Proceedings of the 33rd ASEE/IEEE Frontiers in Education Conference, November 5-8, Boulder, CO, USA. 2003.