Modelo de sistema operacional básico para ensino na Ciência da Computação

Dallan Augusto Toledo Reis, dallan@rosana.unesp.br, Rosana/UNESP
Maurício Araújo Dias, madias@fct.unesp.br, FCT/UNESP
Raphael Garcia, raphael@fct.unesp.br, FCT/UNESP
Paula Tavares Pinto, paula@ibilce.unesp.br, IBILCE/UNESP
Andreia Cristiane Silva Wiezzel, andreia@fct.unesp.br, FCT/UNESP

Resumo: A inclusão de atividades práticas no ensino de Sistemas Operacionais facilita a aprendizagem, porém incorpora novos desafios à disciplina, por exemplo, desenvolver novos sistemas operacionais. Propõe-se neste artigo um modelo de referência de núcleo de sistema operacional básico, que contribui para estudantes compreenderem mais claramente aspectos relacionados ao desenvolvimento prático de sistemas operacionais. Para tanto apresenta-se o Modelo de Sistema Operacional Básico (MSOB), desenvolvido em linguagem de alto nível, que serve como modelo de referência para o desenvolvimento de sistemas operacionais. Os resultados alcançados com sua aplicação revelaram aumento da compreensão nos estudantes, com impacto positivo no ensino prático de Sistemas Operacionais.

Palavras-chave: sistema operacional, ensino, ciência da computação.

Basic operating system model for teaching in Computer Science

Abstract: Although the inclusion of practical activities in teaching operating systems facilitates learning, it also incorporates new challenges to the discipline, for instance, when developing new operating systems. For that reason, in this article we suggest a reference model for a basic operating system´s core which helps students understand more clearly issues related to the practical development of operating systems. In order to do so, this article presents the Basic Operating System Model (BOSM), which developed in high-level language, serves as a reference model for the development of operating systems. The achieved results show an improvement of understanding from the students, followed by a positive impact on the practical teaching of operating systems.

Keywords: operating system, teaching, computer science.

1. Introdução

A disciplina Sistemas Operacionais, vinculada ao Curso de Bacharelado em Ciência da Computação, geralmente é ministrada por meio de aulas nas quais se expõem conteúdos teóricos, tornando uma tarefa desafiadora para os estudantes de graduação compreenderem como, de fato, é o funcionamento real de um sistema operacional. Maziero (2002) afirma que os conceitos ministrados em tal disciplina dificilmente são

compreendidos pelos alunos sem a utilização de uma abordagem mais prática durante as aulas.

Alguns trabalhos foram desenvolvidos para facilitar a aprendizagem de Sistemas Operacionais. Tanenbaum e Woodhul (2000) citam o sistema operacional *Minix*, desenvolvido com a finalidade de estudo e práticas por meio de operações similares ao Unix. Anderson *et al.* (1993) citam o *Nachos* como um sistema operacional instrucional bem-sucedido em aulas de Sistemas Operacionais na graduação em diversas universidades. No Brasil, o sistema *Banana Kernel* foi proposto por Teigão e Morimoto (2004) como uma maneira de facilitar a aprendizagem por meio de um núcleo de sistema operacional menos complexo. Os trabalhos realizados pelos autores proporcionaram uma visão prática dos conteúdos que, anteriormente, eram desenvolvidos apenas de forma teórica com os alunos.

Em paralelo e conforme a mesma tendência, Dias (2013) propôs uma arquitetura de microcomputador reconfigurável, desenvolvida sobre plataforma *Field Programmable Gate Array* (FPGA), para trabalhar com os alunos do curso de Bacharelado em Ciência da Computação da Faculdade de Ciências e Tecnologia da Universidade Estadual Paulista. Esta arquitetura passou a ser ferramenta principal de ensino na disciplina Sistemas Operacionais. Todavia, a arquitetura não possuía, até dois anos atrás, um sistema operacional compatível, de forma que fosse possível controlar as tarefas a serem processadas, gerenciar a memória e o sistema de arquivos.

Em 2014 e 2015, os estudantes do referido curso desenvolveram, em código de máquina, um núcleo de sistema operacional capaz de gerenciar recursos e processos para a arquitetura de microcomputador proposta em 2013. Isto possibilitou o ensino profundo e consistente não somente dos componentes e do funcionamento de sistemas operacionais, como também da interação desses componentes com cada recurso presente na arquitetura de microcomputador, em tempo real.

Não obstante constituírem importantes instrumentos para o ensino prático de Sistemas Operacionais, o Minix, o Nachos e o Banana Kernel, por terem sido desenvolvidos em linguagens de programação de nível mais elevado que o código de máquina, permitem o ensino apenas de forma mais abstrata, sem abordar a interação do sistema com os recursos disponíveis no computador de maneira mais profunda. Desta forma o ensino de Sistemas Operacionais tem se mantido em um nível de abstração que dificulta atingir a sua própria essência.

Se por um lado trabalhar diretamente com código de máquina traz vantagem aos estudantes, por outro, representa um desafio quanto à compreensão do próprio código e motivação aos estudos. Isto porque é mais complexo e se leva mais tempo para compreender o programa de um sistema operacional quando esse é desenvolvido em código de máquina, do que quando é desenvolvido em linguagem de nível mais elevado.

Diante desses desafios apresenta-se neste artigo o Modelo de Sistema Operacional Básico (MSOB) e a avaliação da aplicação desse sistema ao ensino prático na disciplina Sistemas Operacionais. Para elucidar o processo desenvolvido, este artigo está organizado como descrito a seguir: a Seção II descreve a metodologia e apresenta e discute os resultados alcançados com base na aplicação das atividades relacionadas ao MSOB. Por fim, na Seção III, apresentam-se as conclusões desta pesquisa.

2. Metodologia

A metodologia da pesquisa incluiu duas etapas: a primeira referiu-se ao desenvolvimento do MSOB e a segunda à investigação do impacto da utilização deste modelo em aulas práticas de Sistemas Operacionais.

2.1 Sujeitos

Participaram da pesquisa dois grupos de alunos do curso de Bacharelado em Ciência da Computação da FCT/Unesp, campus de Presidente Prudente. Os alunos de cada grupo estevam regularmente matriculados na disciplina Sistemas Operacionais II, oferecida no primeiro semestre do terceiro ano. O primeiro grupo foi composto por 31 alunos e o segundo por 32, totalizando 63 sujeitos entre os anos 2014 e 2015.

2.2 Procedimento

Esta seção comenta as etapas do desenvolvimento do MSOB e a sua aplicação em aulas práticas de Sistemas Operacionais.

2.2.1 Desenvolvimento do MSOB – Construção do Modelo

O MSOB foi desenvolvido em duas etapas. Na primeira utilizou-se código em linguagem C, visando facilitar a compreensão dos estudantes e, na segunda, realizou-se a tradução deste código para a linguagem *Assembly* – ASM. Por fim, adequou-se o código para realizar a junção a outro, dependente de arquitetura de computador, chegando-se a um código completo em ASM. Para compilar o código em C, foi utilizado o compilador *Borland C Compiler* (BCC), similar ao *Gnu Compiler C* (GCC). Ele foi utilizado para gerar o código objeto dos arquivos em C e para traduzir os códigos C em ASM. Para montar o código ASM, foi usado o NASM, montador *assembly* para família de processadores Intel. Foi necessária a utilização de um ligador, ou *linker*, para fazer a junção de todo o código objeto e gerar o executável do MSOB. A escolha para este trabalho foi o *ld86*, que faz a ligação de código C e ASM.

Todo sistema operacional, para ser carregado, precisa de um arquivo especial conhecido como “imagem” do sistema, segundo Tanenbaum (2003). No MSOB, existe o arquivo *generator.asm*, que cria uma imagem do disco de *boot no* arquivo *a.img*. Este arquivo contém o setor inicial de *boot* e inclui os processos-exemplo e o MSOB em suas posições no sistema de arquivos. A escolha para disco de *boot* foi o disquete de 3,5” - *floppy disk* - pela simplicidade para realização de testes. Todavia, a execução do sistema MSOB não se limita apenas a um computador com unidade de disquete. É possível também utilizar o Bochs e também sistemas de virtualização como o Virtual Box ou o Vmware.

Nos ambientes de virtualização, a execução do MSOB se faz por meio da criação de uma máquina virtual, com pouca memória – cerca de 2 MB – e apenas 1 MB de disco. A unidade de armazenamento do disco deverá apontar para um disco "virtual" de 3,5", por meio da indicação da localização do arquivo *a.img* do MSOB.

2.2.1.1 Arquitetura do MSOB

O núcleo do sistema operacional MSOB é do tipo *kernel* monolítico, com sua arquitetura constituída em quatro módulos integrados em um único arquivo binário. Os módulos podem ser desenvolvidos em arquivos diferentes, mas necessariamente são compilados em um único programa e se comunicam diretamente por chamadas de funções. A vantagem está na eficiência da execução, já que os módulos formam um único programa e se referem diretamente uns aos outros, sem depender da comunicação interprocessos, segundo Oliveira *et al.* (2001). A desvantagem é que, a cada alteração em um dos módulos, o *kernel* inteiro precisa ser recompilado. Devido à eficiência, essa arquitetura é a mesma usada nos sistemas operacionais *Windows*, *Linux* e *Unix*.

Para executar programas é necessário editar o arquivo do processo *init* – localizado no diretório */src/processos*. O processo *init* do MSOB é semelhante ao do *Minix* e *Linux*, pois nestes dois sistemas o *init* é o primeiro processo a executar e invoca outros processos primordiais. Numa futura implementação, *init* poderá incluir a implementação de um *shell*, abrindo ao usuário a possibilidade de executar processos em tempo de execução por meio de um interpretador de comandos.

2.2.1.2 Gerenciador de Processos

O MSOB implementa a multiprogramação e o tempo compartilhado. Devido à velocidade da CPU, tem-se a ilusão de que os processos são todos executados ao mesmo tempo, o que é uma característica de sistemas multitarefas. O objetivo principal da multitarefa é fornecer um ambiente em que múltiplos processos possam executar de forma concorrente e evitar que um interfira na execução e nas informações dos outros. Tais objetivos são conflitantes, já que os processos concorrem pelo(s) mesmo(s) processador(es) e áreas de memória em comum. Se as alterações não forem controladas, um processo poderia alterar dados alheios e interferir na execução de outros processos.

Para resolver este problema, o MSOB possui uma tabela de processos. Cada registro tem informações necessárias à execução do processo correspondente, por exemplo o estado atual, os limites de memória onde código e dados do processo estão armazenados e o conteúdo dos registradores do processador. Em sistemas multitarefas, a execução dos processos ocorre de forma escalonada, ou seja, algoritmos decidem qual deles será executado a cada momento na CPU. Quando a fatia de tempo concedida ao processo for insuficiente para ele terminar sua execução por completo, outro processo deverá "ganhar" a oportunidade de ser executado. Desta forma, o processo que teve seu tempo expirado deverá ter suas informações armazenadas para que consiga voltar a ser processado exatamente no mesmo ponto em que parou. Este procedimento é

denominado troca de contexto. Quando outro processo é escolhido, os registradores da CPU são atualizados conforme as instruções do novo processo. Assim, quando o primeiro voltar a ter a CPU disponível, os valores de registradores estarão com conteúdos diferentes. Por isto, as informações salvas na tabela de processos permite recuperar o estado anterior do processo para continuar a execução no mesmo ponto onde havia parado.

Os algoritmos de agendamento de processos se dividem entre aqueles em que os processos possuem igual importância e aqueles em que eles têm diferentes prioridades. O algoritmo mais conhecido, no qual não há prioridades, é o *Round-Robin*, ou Espera Circular. A ideia é que os processos estão organizados em uma fila circular e que cada um deles recebe uma quantidade de tempo igual para executar. O MSOB usa o algoritmo *Round Robin* pela simplicidade da implementação, pois não há atribuição de prioridades. O laço executa até encontrar um processo pronto, ou chegar novamente ao que estava rodando, pois é um laço circular. Se não encontrar um processo pronto, o *kernel* é escolhido para executar.

2.2.1.3 Chamadas de Sistema

Para ser possível criar e finalizar processos, fazer leitura em sistemas de arquivos, remover arquivos e demais operações, os SOs usam um mecanismo denominado chamada de sistema, conforme apontado por Oliveira *et al.* (2001) e Silberschatz *et al.* (2001). O SO possui uma tabela de chamadas que os processos podem utilizar para solicitar serviços ao *kernel.*

No MSOB existem duas rotinas responsáveis pela habilitação das chamadas de sistema: *enable_syscalls()* e *syscall_routine()*, que são implementadas na interrupção 80h – semelhante aos SOs da família UNIX, pois interrupções forçam a intervenção imediata do *kernel*, que é desejável em chamadas de sistema.

2.2.1.4 Gerenciador de Memória

A principal função do Gerenciador de Memória é controlar a área de memória reservada aos processos. As atribuições específicas deste são efetuar o carregamento do processo na memória, quando esse é criado, definir a localização desse processo e liberá-lo, quando for terminado. Enquanto o Gerenciador de Processos define a política de execução dos processos, o Gerenciador de Memória fornece o mecanismo para que esses processos possam habitar a memória. É importante ressaltar que o Gerenciador de Memória nada sabe sobre processos, pois enxerga a partição apenas como um armazém de informações, cabe ao Gerenciador de Processos a ciência de que na partição há um processo.

2.2.1.5 Sistema de Arquivos e carregamento do MSOB

Os sistemas operacionais da família UNIX não carregam automaticamente todos os sistemas de arquivos de todos os dispositivos. Quando o usuário necessita de um

arquivo de um dispositivo, ele deve solicitar a montagem do sistema de arquivos do dispositivo para depois fazer a solicitação da leitura. Este é o caso também do MSOB, e há uma chamada de sistema para montagem de sistema de arquivo de dispositivo.

No MSOB cada dispositivo pode ter somente um sistema de arquivos, diferente de outros SOs que permitem o particionamento do dispositivo em vários sistemas de arquivos diferentes. Os dispositivos podem ser representados como sistemas de arquivos não montado (NOT_MOUNTED_FS) ou como sistema de arquivos montado (MOUNTED_FS).

O processo de carregamento do sistema operacional MSOB ocorre após a montagem do sistema de arquivos, que contém o arquivo (de boot) *a.img*, localizado no diretório */bin*. Isto é feito para possibilitar a execução do processo *init* que, posteriormente, faz a chamada dos arquivos *arquivo0* até o *arquivo9*, localizados no diretório */src/processos*. Estes arquivos representam a execução de processos que fazem a escrita de um caractere no monitor de vídeo. O *aquivo0* escreve o caractere "0", o *arquivo1* escreve o caractere "1" e assim por diante. O escalonador de processos, que implementa o algoritmo de escalonamento *Round-Robin*, faz a escolha de qual arquivo-processo será executado. A sequência de caracteres escritos no vídeo indicará a ordem de escolha da execução dos processos, conforme se observa na Figura 1.

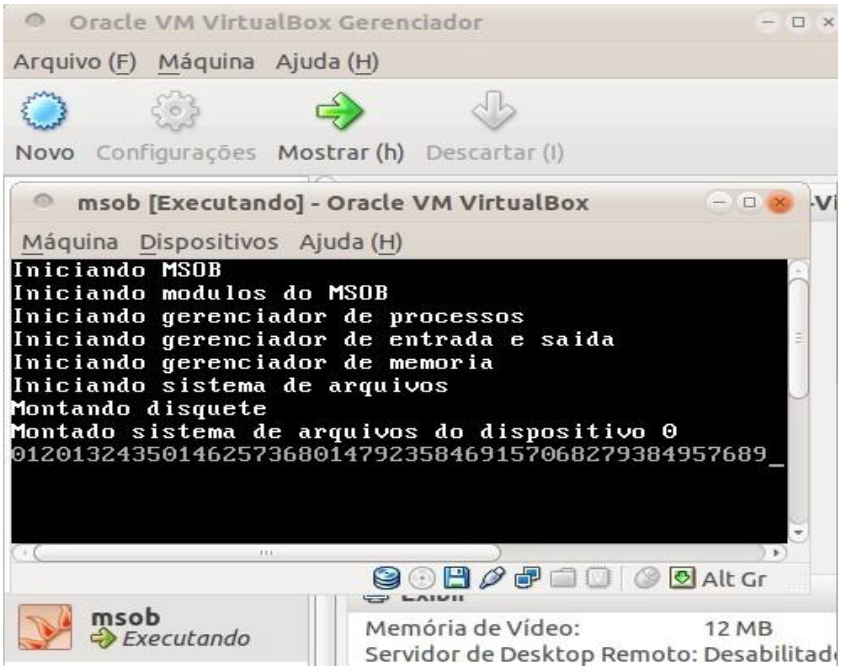

Figura 1 - Carregamento do MSOB e execução de processos.

O MSOB pode ser carregado de 3 formas. A primeira pelo "disquete", a segunda pelo emulador Bochs e a terceira por um sistema de virtualização. A Figura 1 ilustra o carregamento do MSOB por meio do sistema VirtualBox. Observa-se que, após a

montagem do sistema de arquivos, o MSOB faz o escalonamento de 9 processos que escrevem os caracteres de 0 à 9. O sistema permite fazer alterações do conteúdo que será escrito. Todavia, a ordem de carregamento é definida pelo algoritmo de escalonamento *Round-Robin*. Toda vez que houver alterações, faz-se necessária a compilação dos arquivos para geração da imagem do sistema.

Para este trabalho, a Figura 1 ilustra o carregamento do MSOB a partir de um disco de *boot* montado em um disquete de 3,5", o qual é emulado pelo VirtualBox, por ser mais simples para realização de testes e raro encontrar computadores com unidade de disco flexível atualmente.

2.2.1.6 Sistema de Entrada e Saída (E/S)

O Sistema de Entrada e Saída (E/S) é o módulo do SO que lida com os dispositivos de entrada e saída. Isso é feito via *drivers* de dispositivos, ou simplesmente *drivers*. Os *drivers* de dispositivos são trechos de código adicionados ao *kernel*, cujo objetivo é prover uma interface entre o SO e o dispositivo de E/S a ser incorporado ao computador. Quando um novo dispositivo de E/S é criado, ou o fabricante fornece ou disponibiliza *drivers* do dispositivo para os SOs mais conhecidos e utilizados, ou os desenvolvedores de um determinado SO criam o *driver.* Por isso, desenvolver um novo SO envolve suportar o máximo de dispositivos de E/S já existentes e ser capaz de suportar novos *drivers* a qualquer momento.

O MSOB utiliza a E/S mapeada em Espaço Especial da Memória, pois tal decisão é imposta pela arquitetura Intel, já que ela utiliza esse mapeamento de dispositivo de E/S. Este espaço especial de memória inicia-se no segmento de memória A000h e vai até o início de 10000h, onde são mapeados vários dos dispositivos de E/S fundamentais para o funcionamento mínimo do computador. Por exemplo, segundo Santos e Raymundi (1989), a imagem apresentada no monitor de vídeo é mapeada no segmento B800h. A alteração de bytes nessa área significa alterar cores ou caracteres do monitor.

2.2.2 Aplicação do MSOB em aulas práticas de Sistemas Operacionais

A investigação acerca do impacto do MSOB ao processo de ensino e aprendizagem de Sistemas Operacionais ocorreu por meio de experimento aplicado a dois grupos de estudantes. O primeiro grupo desenvolveu um núcleo de sistema operacional, em código de máquina, para uma arquitetura de microcomputador modelo, sem o apoio de modelo de referência desenvolvido em linguagem de programação de alto nível. Ao contrário do primeiro, o segundo grupo de estudantes realizou atividades relacionadas ao MSOB antes do início do desenvolvimento prático do núcleo do sistema operacional em código de máquina.

Os instrumentos utilizados para a coleta de dados foram atividades didáticas, avaliações somativas e questionário. Os dados obtidos foram analisados e cotejados, de forma a se comparar a produção dos estudantes com e sem o apoio do MSOB e, ao mesmo, analisar suas experiências.

2.2.2.1- Execução da pesquisa com os alunos: resultados e discussão

A pesquisa envolveu as seguintes etapas: 1- preparação dos estudantes de ambos os grupos com atividades práticas envolvendo conceitos fundamentais para o desenvolvimento prático de núcleos de sistemas operacionais; 2- realização de atividades relacionadas ao MSOB (somente para o segundo grupo); 3- desenvolvimento prático de núcleo do sistema operacional em código de máquina; 4- aplicação de avaliações somativas e 5- aplicação de questionário de percepção do estudante.

A preparação abordou diferentes conceitos sobre Sistemas Operacionais, tais como: escalonamento de processo, gerência de memória, entrada e saída, organização do sistema de arquivos, chamadas de sistemas, rotinas para tratamento de interrupções, apenas como breve exemplo de tudo que foi trabalhado com os estudantes durante as atividades práticas em sala de aula e laboratório.

A realização de atividades relacionadas ao MSOB, aplicada somente ao segundo grupo de estudantes, compreendeu aula expositiva teórica, debate, demonstração prática do funcionamento do modelo etc. Esta etapa objetivou preparar os estudantes para melhor compreenderem porque, como e onde deveriam desenvolver um núcleo de sistema operacional em código de máquina para uma arquitetura de microcomputador modelo e como ele funciona ao interagir com o microcomputador.

O desenvolvimento prático do núcleo do sistema operacional em código de máquina se deu de forma equivalente para ambos os grupos. Ao final da tarefa, cada estudante apresentou a sua própria implementação do sistema operacional em funcionamento na arquitetura de microcomputador modelo. O objetivo desta etapa foi proporcionar ensino profundo e consistente não somente dos componentes e do funcionamento de sistemas operacionais, como também da interação desses componentes com cada recurso presente na arquitetura de microcomputador em tempo real.

As avaliações somativas, por sua vez, instigaram os estudantes a identificarem funções, características e componentes de um sistema operacional, a partir da apresentação de trechos em código de máquina e da micromemória da arquitetura de microcomputador modelo. O objetivo foi verificar se os estudantes realmente compreenderam aspectos relacionados ao desenvolvimento prático de núcleo de sistema operacional. À guisa de exemplo, apresenta-se uma das quatro questões que constituíram o instrumento: 1- O código fonte do FCTos01 (núcleo de sistema operacional básico, que desenvolveremos na prática) será escrito em uma linguagem de alto nível, a FCTlang01. De forma similar, o código fonte do MSOB também está escrito em uma linguagem de alto nível, a Linguagem C. Quais recursos de uma linguagem de alto nível você considera mais apropriados para representar e implementar conjuntos de procedimentos que possuem finalidades específicas para o FCTos01? (a)

Variáveis; (b) Constantes; (c) Funções; (d) Comandos de decisão; (e) Alocações dinâmicas.

As análises dos erros e acertos às quatro questões propostas, apontaram que aproximadamente 84% dos estudantes do segundo grupo (o que corresponde a 27 sujeitos) compreenderam aspectos relacionados ao desenvolvimento prático de núcleo de sistema operacional, contra aproximadamente 51% do primeiro grupo, equivalente a 16 sujeitos.

Como última atividade os estudantes do segundo grupo responderam ao questionário de percepção. É importante mencionar que houve um teste piloto antes que os questionários fossem aplicados ao grupo. As perguntas, em um total de três, tiveram as respostas organizadas de acordo com a escala de Likert (1932):

1- O estudo do código fonte do MSOB, enquanto modelo de sistema operacional básico, contribuiu para que eu compreendesse aspectos relacionados a projeto para desenvolvimento prático de núcleo de sistema operacional.
( ) Concordo fortemente, ( ) concordo, ( ) indeciso, ( ) discordo, ( ) discordo fortemente.
Justifique:

2- O uso de modelos de referência no ensino de disciplinas, por exemplo MSOB, FC-Tos, FCTarc, contribuem para eu compreender aspectos relacionados ao desenvolvimento prático de projetos.
( ) Concordo fortemente, ( ) concordo, ( ) indeciso, ( ) discordo, ( ) discordo fortemente.
Justifique:

3- Em relação às questões 1 e 2, caso você concorde em uma delas e discorde na outra, por favor, justifique o motivo das respostas serem contraditórias aqui. Justifique:

Aproximadamente 91% dos estudantes, o que corresponde a 29 sujeitos, reconheceram que o uso de modelos para o ensino de disciplinas contribui para a compreensão de aspectos relacionados ao desenvolvimento de trabalhos práticos. Dentre as justificativas dos aproximadamente 9% que acharam que não contribui, o que corresponde a 2 sujeitos, relativas à questão 3, destaca-se o comentário "...o uso de modelos… auxilia o aprendizado… porém… requer muita dedicação e esforço… (só quero passar na matéria)". Na sua justificativa, o estudante afirma que o uso de modelos auxilia no aprendizado, embora ele tenha discordado da afirmação da questão 2.

3. Conclusões

Propôs-se neste artigo um modelo de referência de núcleo de sistema operacional básico, o qual contribuiu para estudantes compreenderem mais claramente aspectos relacionados ao desenvolvimento prático de sistemas operacionais. Por isso, este artigo apresentou o Modelo de Sistema Operacional Básico (MSOB), o qual possui elementos comuns de qualquer SO e é compacto, o que o torna facilmente compreensível, além de ter sido desenvolvido privilegiando a legibilidade. É possível customizar os módulos,

substituindo estratégias usadas em um módulo por outras, por exemplo modificando o algoritmo de agendamento de processos ou o esquema de partições da memória e dos sistemas de arquivos.

Propostas interessantes para futuras extensões do MSOB seriam as de estender o quadro de chamadas de sistema, ampliando o número de serviços do MSOB aos processos de usuários, desenvolver um *shell*, ou linha de comandos, que possibilite ao usuário uma interação maior com o SO, criar elementos para controlar condições de corrida, como semáforos e monitores, e implementar uma estratégia para comunicação entre processos.

Referências Bibliográficas

ANDERSON, T.; CHRISTOPHER, W.; PROCTER, S. The Nachos instructional operating system. In: **Proceedings of the Winter Usenix Technical Conference**, p. 481–489, 1993.

DIAS, M. A. Microcomputador re-configurável em FPGA para ensino de Arquitetura de Computadores na Ciência da Computação. **Revita RENOTE - Novas Tecnologias na Educação da UFRGS**, Rio Grande do Sul, v.11, n.3, dez. 2013.

LIKERT, R. A Technique for the Measurement of Attitudes. **Archives of Psychology**, v. 140, p. 1-55, 1932.

MAZIERO, C. A. Reflexões sobre o ensino prático de sistemas operacionais. **Anais do X Workshop Sobre Educação em Computação da SBC**, SBC - Sociedade Brasileira de Computação, p. 1–12, 2002.

OLIVEIRA, R. S.; CARISSIMI, A. S.; TOSCANI, S. S. **Sistemas operacionais**. Porto Alegre: Sagra-Luzzatto, 2001.

SANTOS, J. R. D. P. dos; RAYMUNDI JR., E. **Programando em Assembler 8086/8088**. São Paulo: McGraw-Hill, 1989.

SILBERSCHATZ, A.; GAGNE, G.; GALVIN, P. B. **Sistemas operacionais**. Rio de Janeiro: Campus, 2001.

TANENBAUM, A. S. **Sistemas Operacionais Modernos**. São Paulo: Pearson Prentice Hall, 2003.

TANENBAUM, A. S.; WOODHULL, A. S. **Sistemas operacionais: projeto e implementação**. Porto Alegre: Bookman, 2000.

TEIGÃO, R. C.; MORIMOTO, J. H. BananaKernel: **Um sistema operacional para ensino**. Curitiba: Pontifícia Universidade Católica do Paraná, 2004.