FABRÍCIO PEREIRA REIS

TBC-SO/WEB: SOFTWARE EDUCATIVO PARA APRENDIZAGEM DE GERÊNCIA DE PROCESSOS E DE GERÊNCIA DE MEMÓRIA EM SISTEMAS OPERACIONAIS

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação.

LAVRAS
MINAS GERAIS – BRASIL
2009

FABRÍCIO PEREIRA REIS

TBC-SO/WEB: SOFTWARE EDUCATIVO PARA APRENDIZAGEM DE GERÊNCIA DE PROCESSOS E DE GERÊNCIA DE MEMÓRIA EM SISTEMAS OPERACIONAIS

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação.

Área de Concentração:

Informática na Educação

Orientador:

Prof. Dr. Heitor Augustus Xavier Costa

LAVRAS
MINAS GERAIS – BRASIL
2009

Ficha Catalográfica preparada pela Divisão de Processo Técnico da Biblioteca Central da UFLA

Reis, Fabrício Pereira

TBC-SO/WEB: Software Educativo para Aprendizagem de Gerência de Processos e de Gerência da Memória em Sistemas Operacionais / Fabrício Pereira Reis. Lavras – Minas Gerais, 2009. 96 p.

Monografia de Graduação – Universidade Federal de Lavras. Departamento de Ciência da Computação.

1. Informática na Educação. 2. Software Educativo. 3. Sistemas Operacionais. I. REIS, F. P. II. Universidade Federal de Lavras. III. TBC-SO/WEB: Software Educativo para Aprendizagem de Gerência de Processos e de Gerência da Memória em Sistemas Operacionais.

FABRÍCIO PEREIRA REIS

TBC-SO/WEB: SOFTWARE EDUCATIVO PARA APRENDIZAGEM DE GERÊNCIA DE PROCESSOS E DE GERÊNCIA DE MEMÓRIA EM SISTEMAS OPERACIONAIS

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação.

Aprovada em ___/___/_____

_____

Profª. Drª. Marluce Rodrigues Pereira

_____

Prof. Dr. Joaquim Quinteiro Uchôa

_____

Prof. Dr. Antonio Maria Pereira de Resende

_____

Prof. Dr. Heitor Augustus Xavier Costa

(Orientador)

LAVRAS
MINAS GERAIS – BRASIL

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

TBC-SO/WEB: Software Educativo para Aprendizagem de Gerência de Processos e de Gerência de Memória em Sistemas Operacionais

RESUMO

Atualmente, existem diversas linhas de pesquisa e desenvolvimento que buscam melhorar ou criar métodos de ensino. Assim, produtos de software educativos são largamente usados como ferramentas para ilustrar de maneira mais atrativa e detalhada o desenrolar da teoria apresentada aos alunos na sala de aula. Com o aumento de seu uso, a Internet tornou-se um aliado forte do ensino devido ao seu poder de alcance e a sua praticidade. Dessa forma, este trabalho busca unir algumas das políticas de gerência de processos e de gerência de memória em sistemas operacionais presentes na literatura e apresentar o TBC-SO/WEB (Treinamento Baseado em Computador para Sistemas Operacionais via Web), um produto de software educativo que objetiva contribuir no ensino das políticas investigadas. Este produto de software utiliza recursos gráficos animados com interface para a Web empregando a tecnologia Java (JSE – *Java Standard Edition*) para propiciar seu uso por várias pessoas e em qualquer lugar que tenha um computador com acesso a Web e a máquina virtual Java instalada (JVM – *Java Virtual Machine*). Em especial, o TBC-SO/WEB busca contribuir com a qualidade de ensino da disciplina ‘sistemas operacionais’ e outras que englobam tais conteúdos nos cursos de graduação em Computação e Informática.

Palavras-chave: Informática na Educação, Sistemas Operacionais, Gerência de Processos, Gerência de Memória

TBC-SO/WEB: Educational Software to Learn Processes Management and Memory Management in Operating Systems

ABSTRACT

Nowadays, there are many research and development lines that try to improve or create new teaching methods. Thus, educational software is often used as tool to illustrate in a more attractive and detailed way the uncoiling of the theory introduced to the students at the class. Through its raise of use, the Internet became a strong allied to the teaching due to its reach power and practicality. In that way, this paper unites some of the processes management and memory management politics in operating systems found at the literature and introduce the TBC-SO/WEB, educational software which aims to contribute to teaching of the found politics. This software uses animated graphic resources with interface to the Web, applying the Java SE (Standard Edition) technology to provide the software use for anybody who has a computer with Internet access and the JVM (Java Virtual Machine) installed. In special, the TBC-SO/WEB tries to contribute to teaching quality of the ‘operating systems’ discipline and other ones that study the same subject.

Keywords: Computing in Education, Operating Systems, Processes Management, Memory Management

1. INTRODUÇÃO

O constante avanço das tecnologias existentes aliado ao desenvolvimento de novas tecnologias tem tornado o mercado de tecnologia da informação cada vez mais exigente quanto à qualificação profissional das pessoas. A busca por profissionais de caráter persuasivo e inovador, capazes de unir dinamismo e domínio crítico sobre diversos assuntos, está ligada diretamente às metodologias de ensino empregadas durante o período universitário do futuro profissional.

Na intenção de conseguir melhores resultados no processo de aprendizagem dos alunos do curso da área de Computação e Informática, faz-se constante a necessidade de melhoria da didática de ensino em âmbito geral. Para isso, pode-se usar software capaz de transformar processos abstratos em concretos aumentando a interação entre os alunos e o objeto em estudo.

Segundo Araújo (2003) *apud* Rocha *et al*. (2004), o uso de recursos tecnológicos, como o computador e a Internet, desperta o interesse nos alunos em estudar e prepara-os para a integração com uma sociedade altamente tecnológica. Ainda segundo o mesmo autor, com o uso do computador como ferramenta de ensino, o aluno é estimulado a conquistar o conhecimento, ao invés de esperar a sua transferência para si por meio do professor. Outro importante ponto é o poder do software educativo, pois a rápida resposta dada por ele encoraja o aluno a autocorreção, levando-o a experimentação e descoberta.

1.1. Motivação

O Ministério da Educação, por meio da Secretaria de Educação a Distância (Seed), tem atuado como um agente de inovação tecnológica nos processos de ensino e aprendizagem, fomentando a incorporação das tecnologias

de informação e comunicação e das técnicas de educação a distância aos métodos didático-pedagógicos. Além disso, promove a pesquisa e o desenvolvimento voltados para a introdução de novos conceitos e práticas nas escolas públicas brasileiras [Portal MEC, 2009].

Tendo em vista esta vertente de atuação do Governo Federal, iniciou-se o desenvolvimento de uma ferramenta automatizada de ensino e aprendizagem (software educativo), visando contribuir na apresentação mais didática de políticas de gerência de processos e de gerência de memória em Sistemas Operacionais. Esta ferramenta é chamada TBC-SO/WEB (Treinamento Baseado em Computador de Sistemas Operacionais para Web).

É importante salientar que, para os alunos entenderem os conceitos apresentados de forma clara pelo assunto Sistemas Operacionais, deve-se considerar que este assunto deve ser apresentado em forma de aulas teóricas, onde os conceitos são apresentados, e de aulas práticas, onde os ouvintes (no caso os alunos) têm a oportunidade de entender melhor os tópicos abordados nas aulas teóricas.

Nesse contexto, o TBC-SO/WEB pode ser útil como facilitador do processo de aprendizagem dos tópicos abordados por ele, uma vez que os conceitos abstratos poderão ser apresentados de forma mais didática, aprimorando a qualidade do material usado nas aulas. Além disso, o TBC-SO/WEB pode facilitar a transição de educadores, visto que existirá uma base pronta para ser usada. De acordo com Buzin (2001), geralmente, o aluno mostra mais interesse por aulas diferenciadas, nas quais prendem sua atenção. Dessa maneira, além de tornar o conteúdo mais atrativo, a melhoria do rendimento dos alunos nas avaliações pode ser observada.

Um ponto importante deste trabalho é o estudante ter acesso livre ao TBC-SO/WEB para que estude onde quiser. Para isso, ele precisa ter um computador com acesso a Internet e com a máquina virtual Java instalada. Seguindo esta idéia, a disponibilização do TBC-SO/WEB estimula os alunos a estudarem e contribui tecnologicamente para a sociedade com o enriquecimento de fontes de estudos e de pesquisa. Além disso, o TBC-SO/WEB pode ser útil como fonte para consultas futuras por educadores, em caso de necessidade, para abstrair detalhes, conhecer novos assuntos relacionados ou esclarecer dúvidas sobre os assuntos tratados.

1.2. Objetivo

O objetivo geral deste trabalho é apresentar o desenvolvimento de um software educativo gráfico e interativo com usuários para auxiliar no ensino das políticas de gerência de processos e de gerência de memória em Sistemas Operacionais.

Como objetivos específicos, podem ser citados:

- Realizar estudo analítico comparativo da teoria com a prática das políticas de gerência de processos e de gerência de memória existentes na literatura e abordadas pelo software desenvolvido;
- Realizar levantamento de software existente, que tratam o mesmo conteúdo;
- Estimular atualização das metodologias de ensino da área de Computação e Informática, com o desafio de melhorar a forma que os alunos assimilam o conteúdo;
- Manter estrutura de ensino das políticas de gerência de processos e de gerência de memória, mesmo quando há mudança de docente.

1.3. Metodologia de Desenvolvimento

1.3.1. Tipo de Pesquisa

Em observação aos métodos científicos, pode-se classificar este trabalho como:

- de natureza tecnológica, pois objetiva-se aplicar o conhecimento básico em tecnologia Java para a Web e técnicas de desenvolvimento de aplicações educativas no desenvolvimento do TBC-SO/WEB;
- de caráter descritivo, pois busca observar, analisar e registrar, em especial, políticas de gerência de processos e de gerência de memória em Sistemas Operacionais;
- de procedimentos experimentais abalizados em referências bibliográficas, pois a teoria e a análise comparativa das políticas de gerência de processos e de gerência de memória em Sistemas Operacionais estão fundamentados na literatura.

1.3.2. Procedimentos Metodológicos

Para o desenvolvimento deste trabalho, o estudo do uso da informática na educação e de sistemas operacionais, em especial as políticas de gerência de processos e gerência de memória, e o levantamento de ambientes computacionais educacionais, que abordam o mesmo assunto, foram realizados.

Paralelamente ao desenvolvimento, foram estudados recursos da linguagem de programação Java para implementar e disponibilizar o TBC-SO/WEB. Como fonte de estudo, foi feita leitura do livro Deitel; Deitel (2005) em capítulos mais relacionados à pesquisa e do *site* oficial de tutoriais sobre Java da Sun Microsystems (*The Java Tutorials*[1]).

[1] http://java.sun.com/docs/books/tutorial/

Para realizar o levantamento do estado da arte, foram usados textos presentes em bibliotecas físicas e virtuais (Internet) relacionados aos assuntos tratados, por exemplo, livros, artigos em eventos científicos nacionais e internacionais, trabalhos de conclusão de curso de graduação, dissertações de mestrado e teses de doutorado. Além disso, uma pesquisa foi realizada com alunos e ex-alunos que cursaram a disciplina Sistemas Operacionais, ofertada ao curso de Bacharelado em Ciência da Computação pelo Departamento de Ciência da Computação da Universidade Federal de Lavras. O foco desta pesquisa foi coletar sugestões e encontrar eventuais dificuldades que pudessem contribuir com a qualidade do TBC-SO/WEB, bem como a apurar a contribuição TBC-SO/WEB para a sedimentação dos assuntos abordados.

Houve constante preocupação com relação à metodologia de desenvolvimento do código fonte do TBC-SO/WEB para facilitar a sua manutenção e a sua evolução. Além disso, os comentários no código fonte estão no idioma Inglês, visando a sua universalização, e o código fonte está hospedado no Google Code[2]. O paradigma de programação usado foi orientação a objetos e a linguagem de programação foi J2SE (*Java Standard Edition*). O uso de J2SE foi por causa da facilidade de migrar software, escritos nesta linguagem, de ser livre de custos e de possibilitar o desenvolvimento para Web com recursos gráficos animados. Foram usados a IDE (*Integrated Development Enviroment*[3]) NetBeans 6.1 e o *kit* de desenvolvimento Java versão 6 (JDK – *Java Development Kit*[4]). Um *link* foi disponibilizado no *site* http://www.dcc.ufla.br/~heitor/Projetos.html contendo o TBC-SO/WEB.

[2] http://code.google.com/
[3] http://www.netbeans.org/
[4] http://java.sun.com/javase/downloads/index.jsp

O desenvolvimento do TBC-SO/WEB foi realizado usando *applets*, organizados em dois grupos. O primeiro grupo relaciona as seguintes políticas de gerência de memória: i) *First-Fit*; ii) *Next-Fit*; iii) *Best-Fit*; e iv) *Worst-Fit*. O segundo grupo relaciona as seguintes políticas de gerência de processos: i) FIFO (*First In, First Out*); ii) SJF (*Shortest Job First*); iii) SRTF (*Shortest Remaining Time First*); iv) HRRN (*Highest Response Rate Next*); v) Por Prioridade (preemptiva e não-preemptiva) e vi) *Round-Robin*.

Aconteceram reuniões constantes entre orientador e orientado para discussão a respeito do desenvolvimento do projeto, em especial, TBC-SO/WEB, coleta de artigos e de periódicos na área de sistemas operacionais e metodologias de ensino de Computação.

1.4. Estrutura do Trabalho

Além deste capítulo, este trabalho está organizado em mais 7 capítulos.

O Capítulo 2 aborda o uso da informática na educação, apresentando alguns desafios encontrados e possíveis soluções.

O Capítulo 3 discorre sobre definições de sistema operacional e faz um resumo da evolução histórica de sistemas operacionais e seus principais serviços.

O Capítulo 4 apresenta conceitos relacionados à gerência de processos em sistemas operacionais, enfatizando as políticas de escalonamento de processos para uso de processador de um computador.

O Capítulo 5 apresenta conceitos relacionados à gerência de memória em sistemas operacionais, enfatizando as políticas para percorrer a lista de lacunas na memória principal de um computador.

O Capítulo 6 faz breve avaliação do funcionamento de alguns ambientes educacionais que tratam o mesmo assunto proposto por este trabalho.

O Capítulo 7 apresenta o software desenvolvido, o TBC-SO/WEB, com a descrição do processo de elaboração, as ferramentas utilizadas, a funcionalidade e os conceitos envolvidos.

O Capítulo 8 apresenta conclusões, contribuições e sugestão de trabalhos futuros.

2. USO DA INFORMÁTICA NA EDUCAÇÃO

2.1. Considerações Iniciais

A disciplina Sistemas Operacionais, presente na grade curricular dos principais cursos de graduação na área de Computação e Informática, tem como finalidade essencial prover ao aluno conhecimentos básicos relacionados à funcionalidade de um sistema operacional [SBC, 2009]. Embora a disciplina possua semântica prática considerando que parte dos conceitos é empregada em sistemas operacionais reais, ela pode concentrar maior parte do seu conteúdo em aulas teóricas, Dessa forma, seus conceitos podem não ser assimilados adequadamente. Por outro lado, o uso de tecnologias atrativas para o ensino deste assunto pode aumentar a sua assimilação. Uma dessas tecnologias é o computador; o seu uso pode ser benéfico na educação, uma vez que os recursos audiovisuais que podem ser obtidos são atrativos, trazendo uma dinâmica interessante na abordagem de vários assuntos.

Este capítulo apresenta alguns pontos relacionados ao uso da informática na educação na forma de software educativo. A seção 2.2 apresenta alguns desafios encontrados no ensino de computação, considerando algumas soluções. A seção 2.3 apresenta alguns conceitos relacionados a software educativo.

2.2. Desafios e Soluções no Ensino de Computação

Segundo Valente (2008), nos sistemas educacionais atuais, o computador tem sido utilizado como ferramenta de ensino independente da disciplina lecionada. No ensino de computação, o computador é usado como objeto de estudo, isto é, o estudante usa-o para aprender novos conceitos computacionais, como técnicas de programação ou implicações do computador na sociedade. Porém, o autor afirma que a maior parte dos cursos oferecidos pode ser

caracterizada como de “conscientização para a informática”, ao invés de ensinar o aluno a programar de forma eficiente e inovadora. Assim, para ele, os propósitos são vagos e não determinam o grau de compreensão que o estudante deve ter.

Alguns problemas que ocorrem no processo de ensino das disciplinas de computação [Rodrigues, 2002 *apud* Santos; Costa, 2006]:

- Falta de motivação do aluno criada pelo despreparo e pelo desânimo, principalmente, por causa da crença de algumas disciplinas serem extremamente difíceis de alcançar aprovação;
- Relacionamento do professor com o aluno pode ser um problema quando o primeiro preocupa-se em mostrar o que sabe, desconsiderando o ambiente de aprendizagem colaborativo entre aluno e professor;
- Forma de avaliação pode afetar a tranquilidade do aluno, tornando-o tenso e prejudicando o seu aprendizado;
- Falta de metodologia de ensino adequada dificulta o aprendizado de novos conceitos, associada à falta de maior comunicação entre os professores das diferentes disciplinas, que permitiria identificar conteúdos próximos ou o sombreamento de conteúdo, complementa a extensa lista de problemas que ocorrem no processo de ensino de forma geral.

Sancho (1998) afirma que o ritmo acelerado de inovações tecnológicas exige um sistema educativo capaz de estimular nos estudantes o interesse pela aprendizagem. Além disso, esse interesse diante de novos conhecimentos e técnicas deve ser mantido ao longo da sua vida profissional, que, provavelmente, tenderá a se realizar em áreas diversas de um mercado cada vez mais sujeito ao impacto de novas tecnologias. Nesse contexto, encontra-se um desafio para o ensino de futuros profissionais de qualidade para o exigente mercado atual.

Para Mercado (2002), a computação, quando aplicada no ensino, traz flexibilidade na aprendizagem, une a teoria e a prática, onde os alunos aprendem e sabem como, por que, onde e quando eles aprendem. Porém, a computação não deve ser vista como redentora da educação, mas como um elemento a mais para contribuir na construção de uma estrutura de ensino que pode desenvolver mecanismos que contribuam na superação de suas limitações.

O computador deve ser utilizado como ferramenta auxiliar do professor, cuja postura passa para mediador do processo de apreensão, produção e difusão do conhecimento. O professor se coloca como um sujeito em outro nível de conhecimento que interage e trabalha com informações juntamente com o aluno, contribuindo na elaboração de conceitos mais avançados [Neitzel, 1999].

Existem princípios fundamentais para apoiar a interação da tecnologia com os métodos de ensino. Alguns deles são [Sandholtz *et al*., 1997]:

- Tecnologia é considerada como uma ferramenta que revitaliza a atração de professores e de alunos pelo aprendizado;
- Tecnologia deve ser agregada à estrutura curricular de ensino ao invés de ser ensinada separadamente;
- Aprendizagem de tecnologias é maior em tarefas mais significativas;
- Aprendizagem é um processo ativo e social que acontece melhor em ambientes centrados nos alunos, nos quais os professores assumem o papel de facilitador para orientar os alunos em indagações significativas, nos quais as atividades construtoras de conhecimento são balanceadas com o uso sensato da prática orientada e da instrução direta;
- Constante desenvolvimento acontece quando os professores criam equipes de colaboradores para discutir a prática regularmente.

Com isso, ressalta-se que o uso da informática na educação age como solução e complemento das atividades desafiadoras para os alunos. Desta forma, este uso contribui para a superação das suas limitações para que eles possam adquirir melhor maneira de pesquisar, manipular, raciocinar, enxergar processos de forma mais atrativa e desenvolver as atividades propostas. Deve-se observar que o computador não deve ser utilizado como único meio de transmissão de conhecimento. O professor não controla, mas auxilia na aprendizagem. Entretanto, não basta apenas colocar a disposição recursos computacionais para os professores, é preciso prepará-los ou prepararem-se, respeitar o seu tempo e fazer com que eles entendam o porquê e o poder de uma nova ferramenta de trabalho.

2.3. Software Educativo

Um software educativo é um conjunto de recursos computacionais projetados com a intenção de serem usados em contexto de aprendizagem [Cano, 1998]. Segundo Mercado (2002), alguns programas de multimídia enfatizam as tarefas que trabalham, principalmente, a percepção, trazendo para o monitor do computador significativa quantidade de estímulos (como imagens, textos e animações) e eliminando as tarefas que exigem do aluno o exercício de processos cognitivos mais complexos. Assim, o aluno é estimulado a dar maior atenção ao “Por quê?”, “Como funciona?”, “O que é?” de conceitos e de processos de maneira projetada visando à facilidade de aprendizado.

Ainda segundo Mercado (2002), para que um software promova aprendizagem, ele precisa seguir alguns princípios, tais como:

- Objetivo geral da instrução é promover a aquisição de conhecimento que possa ser facilmente acessado e aplicado em novas situações;

- Objetivo da aprendizagem deve ser significativo; caso contrário, ele seria mera perda de tempo;
- Instruções devem ser centradas no aluno, isto é, a sua inteligência deve ser explorada ao máximo a partir de estímulos, principalmente, visuais e interativos;
- Aprendizagem deve estar em um contexto realista e significativo, sem utopias.

Bom software educativo não deve ser aquele cheio de recursos apenas tecnicamente úteis, mas aquele que permita rápida interação do aluno à sua utilização e faça-o preocupar-se mais em exercer suas indagações, mesmo que elas aconteçam sem esforço [Cysneiros, 1998].

Um software educativo deve [Coscarelli, 1998]: i) propiciar suporte para a reflexão; ii) estimular e criar oportunidades para que o estudante pense em idéias sob várias perspectivas; iii) fornecer *feedback* rico e explicativo; iv) explorar erros como oportunidades para desenvolver o aprendizado; v) explorar diferenças individuais de conhecimento e habilidades; e vi) fornecer medidas significativas de avaliação, por exemplo, um relatório do uso de uma instância do software educativo. Desta forma, Sancho (1998) organiza os softwares educativos em seis categorias:

- Demonstrativo. Este tipo de software educativo possibilita a demonstração de conceitos da disciplina em questão. Este tipo de software educativo usa recursos gráficos animados coloridos e sons, porém a interatividade com o usuário é baixa, pois ele simplesmente demonstra conceitos e processos, desconsiderando variáveis que possam ser introduzidas pelo o usuário;
- Jogo. Este tipo de software educativo apresenta um ambiente onde o jogador que a priori conhece as regras ensaia estratégias para conquistar o objetivo

predeterminado, onde as etapas percorridas até o objetivo final apresentam os conceitos tratados pelo software;

- Simulação. Este tipo de software educativo tem a finalidade simular a modelagem de um sistema ou situação real usando recursos gráficos animados. Além disso, ele garante o teste de situações em que não é possível ter em experiência real e são largamente usados em aulas de química, onde o software simula os acontecimentos de uma reação química;
- Monitoramento. Este tipo de software educativo possui como função o acompanhamento do processo de aprendizagem dos alunos, auxiliando e decidindo quais itens devem ser tratados naquele momento. Além disso, ele mostra o conteúdo, apresenta algumas explicações e propõe exercícios. Após submissão das respostas, o software efetua uma avaliação sob elas e, por fim, o aluno recebe uma mensagem avaliativa;
- Tutoriais. Este tipo de software educativo permite o acesso a conteúdos didáticos usando ícones e trabalha de forma totalmente interativa. Esta interação consiste em, para cada conteúdo apresentado, o software faz perguntas a serem respondidas pelo usuário para que ele possa prosseguir no processo de aprendizagem. As lições podem ser repetidas quantas vezes o aluno quiser. Quanto ao processo de avaliação do aluno é baixa, pois esse tipo de software não interpreta os dados fornecidos pelos usuários fora dos limites pré-estabelecidos;
- Exercício. Este tipo de software educativo permite aos professores apresentarem conceitos em sala de aula e, em seguida, exercícios de tais conceitos utilizando o computador. O software deve corrigir os erros e pode dar exemplos para ajudar. Para que seja coerente, esse tipo de software deve apresentar problemas de forma gradual e sistemática, de acordo com o nível de cada aluno.

2.4. Considerações Finais

Este capítulo apresentou uma visão geral do uso da informática na educação, considerando alguns desafios e algumas soluções e o emprego de software educativo neste âmbito. Entende-se que o uso da informática no processo de ensino está associado às mudanças tecnológicas e sociais.

A informática na educação está associada à modificação do como aprender, da inovação entre o aluno e o professor e da maneira como se transmite o conhecimento. Porém, é preciso levar em conta que não somente a introdução de recursos computacionais trará mudanças na aprendizagem dos alunos. O software educativo e a Internet devem ser vistos como ferramentas interessantes em possibilitar colaborações no nível de aprendizado dos métodos de ensino.

Assim, o uso da informática na educação é uma alternativa para melhorar o nível de aprendizagem dos alunos, mas deve-se ter uma preocupação constante dos profissionais da educação de tentar manter o estímulo à inovação e à cobrança sob os alunos, aumentando a produtividade e a qualidade do conteúdo das disciplinas.

3. SISTEMAS OPERACIONAIS

3.1. Considerações Iniciais

Um computador, sem um software instalado é inútil, ou seja, não recebe dados de entrada, não exibe caracteres na tela; enfim, não é capaz de executar um programa. Dessa forma, com uma máquina sem um sistema operacional que gerencie o hardware instalado, os programadores não são capazes de executar seus programas, muito menos um usuário sem formação técnica. Neste contexto, foi elaborado um programa de computador (software) que permite o seu uso de forma simples e atrativa. Este programa é o sistema operacional. O sistema operacional tem evoluído com o passar do tempo, assimilando características para obter o máximo de recursos que os modernos computadores podem oferecer e aumentando a atratividade aos usuários por utilizar recursos audiovisuais de modo a facilitar o uso do computador.

Este capítulo apresenta o progresso e a melhoria dos sistemas operacionais ao longo do tempo. A seção 3.2 relaciona algumas definições de sistema operacional sob a visão dos principais autores da literatura. A seção 3.3 mostra breve histórico da evolução dos sistemas operacionais. A seção 3.4 trata os principais serviços oferecidos por um sistema operacional.

3.2. Definição

Há muitos autores que abordam o tema Sistemas Operacionais. Em igual proporção, na literatura, podem ser encontradas definições de sistemas operacionais. A seguir, são apresentadas algumas dessas definições:

- Um sistema operacional é um programa de computador que permite aos usuários usufruir o hardware, coordenando o hardware a realizar as tarefas

desejadas. Um sistema operacional bem desenvolvido não é somente fácil de usar, mas agradável de usar [Shay, 1996];

- Um sistema operacional coordena o uso e o funcionamento do hardware e é responsável pela gerência dos recursos básicos do computador, que consiste em: hardware, software e dados. Estes recursos devem ser tratados pelos sistemas operacionais modernos [Davis, 1990];
- Um sistema operacional é um software que habilita as aplicações a interagirem com o hardware de um computador e, para obter o máximo da capacidade do hardware, as aplicações são executadas concorrentemente. Além disso, ele deve fornecer serviços que permitam as aplicações serem executadas com segurança e eficácia [Deitel *et al*., 2005];
- Um sistema operacional pode ser visto como uma máquina estendida (uma camada de software que abstrai o uso do hardware ocultando a "verdade" sobre ele, apresentando uma interface orientada a arquivos simples e agradável) ou como um gerenciador de recursos (responsável por controlar quais aplicações podem ser executadas e quando e quais recursos podem ser usados) [Tanenbaum, 2003b];
- Um sistema operacional explora os recursos de hardware de um ou mais processadores para proporcionar um conjunto de serviços aos seus usuários; além disso, ele gerencia a memória secundária e os dispositivos de entrada/saída [Stallings, 2005];
- Um sistema operacional é um programa que gerencia o hardware do computador, oferecendo base para programas e atuando como um intermediário entre o usuário e o hardware [Silberchatz *et al*., 2004a];
- Um sistema operacional é um gerente executivo, ou seja, aquela parte de um sistema de computação que administra os componentes de hardware e de software. Em termos mais específicos, o sistema operacional controla cada

arquivo, dispositivo, seção de memória principal e nanossegundo do tempo de processamento [Flynn; Mchoes, 2002].

Pode-se perceber que as definições apresentadas levam a uma idéia comum de definição de sistema operacional: ele é responsável por permitir ao usuário usar os vários recursos disponíveis no computador de maneira adequada e mais fácil.

3.3. Evolução

Nos últimos 60 anos, os sistemas operacionais passaram por diversas fases distintas. Esta seção apresenta a evolução dos sistemas operacionais desde a sua primeira proposta de desenvolvimento ocorrida na década de 40 (século XX) até as mais recentes propostas dos dias atuais (século XXI).

3.3.1. Décadas de 1940 e 1950

Na década de 40, os primeiros computadores não possuíam sistemas operacionais. Os programadores daquela época muitas vezes precisavam submeter seus programas em linguagem de máquina.

De acordo com Deitel *et al*. (2005), no final da mesma década e no início da década seguinte, após o desenvolvimento dos cartões perfurados e da linguagem de montagem (*assembly*[5]), o primeiro sistema operacional foi desenvolvido para o computador IBM 701 do Laboratório de Pesquisa da General Motors. O seu objetivo era controlar a transição de *jobs*[6] (término de um *job* e início de outro *job*), de forma que os recursos do computador estivessem dedicados a apenas um programa de cada vez.

[5] É uma notação legível pelas pessoas para o código de máquina usado por uma arquitetura de computador específica (Hayes, 1988).
[6] Conjunto de instruções de programas correspondente a uma tarefa computacional específica (Deitel *et al*., 2005).

Ainda segundo Deitel *et al*. (2005), o programador escrevia o código fonte, primeiramente, em papel (FORTRAN ou *assembly*). Em seguida, o código fonte era traduzido em cartões perfurados e, depois, eles eram levados a um operador que os colocavam na unidade de leitura do computador. Ao terminar o *job*, o operador entregava a saída ao programador. Além disso, os autores afirmam que, naquela época, era comum solicitar ao programador que controlasse explicitamente os recursos usados, tais como memória e dispositivos de entrada/saída.

3.3.2. Década de 1960

De acordo com Shay (1996), o próximo estágio no desenvolvimento foi a capacidade de armazenamento de vários programas em memória simultaneamente. Isto é, os programas poderiam compartilhar os recursos do computador ocupando diferentes turnos ao invés de serem executados em sequência. Cada turno significava uma fatia de tempo que o sistema operacional liberava para o programa usar o processador. Com a implantação da divisão do tempo de processador, os programas menores puderam ser iniciados mais rapidamente e, como os recursos eram compartilhados, terminados mais rapidamente [Willrich, 2003].

Nesse contexto, Oliveira (2006) nota que os projetistas da época desenvolveram sistemas multiprogramados que coordenavam vários *jobs* ao mesmo tempo. Existiam *jobs* que utilizavam principalmente o processador e *jobs* que utilizavam mais os dispositivos periféricos. Aproveitando essa inovação, os ambientes multiprogramados comutavam o processador de *job* em *job*, mantendo vários deles em andamento e, ao mesmo tempo, em uso com os dispositivos periféricos.

Os usuários de computador daquela época não aguardavam o termino do *job* submetido no local. Os cartões perfurados ou as fitas de computador ficavam em mesas de entrada até o operador pegá-los e levá-los até o computador para execução. Frequentemente, o *job* de um usuário ficava parado por horas ou dias antes de ser processado e, muitas vezes, quando ocorria algum erro no programa, o usuário tinha que buscar o programa, corrigir o erro e entrar na fila novamente.

Em 1964, a IBM lançou a série de computadores 360 com maior capacidade de computação e, com eles, o sistema operacional OS/360. Com o passar do tempo, esta linha evoluiu para a família de computadores zSeries e, junto com esta evolução, os sistemas operacionais de processamento em lote evoluíram para sistemas operacionais de tempo compartilhado. Estes foram desenvolvidos para atender a diversos usuários interativos ao mesmo tempo em que interagiam com o sistema via "terminais burros"[7] *on-line*.

Sistemas como o CTSS (*Compatible Time Sharing System*) desenvolvido pelo MIT (*Massachusetts Institute of Technology*), o TSS (*Time Sharing System*) desenvolvido pela IBM e o CP/CMS (*Control Program/Conversational Monitor System*), que posteriormente evoluiu para o sistema operacional VM (*Virtual Machine*) da IBM, mostraram a importância da computação interativa em ambientes de desenvolvimento de software.

Neste novo contexto, o tempo gasto da submissão ao retorno de resultados de um *job* diminuiu para minutos e segundos. Assim, o programador podia entrar com o programa, compilá-lo, receber a lista de erros (caso ela existisse), corrigi-los e compilá-lo novamente, acelerando o processo de desenvolvimento de software.

[7] Dispositivos que fornecem uma interface do sistema ao usuário, mas não possuem capacidade de processamento (Iizuca, 1987).

3.3.3. Década de 1970

De acordo com Shay (1996), os programas que as pessoas usavam na década de 60 mudaram de maneira significativa. Os computadores, que antes eram usados basicamente para efetuarem cálculos monstruosos, neste momento, passam a serem vistos por homens de negócios como ferramentas para gerenciar informações.

Assim, como as informações deveriam ser acessíveis por vários usuários, a necessidade de estabelecer comunicação entre os computadores surgiu e o desenvolvimento das redes de computadores aconteceu a partir dessa época. Comunicações entre sistemas de computadores pelos Estados Unidos da América aumentaram quando os padrões de comunicações TCP/IP (*Transmission Control Protocol/Internet Protocol*) do Departamento de Defesa (DoD – *Depatment of Defense*) foram amplamente usados em redes de menor porte. A comunicação em redes locais (LANs – *Local Area Networks*) ficou prática e econômica com o padrão Ethernet.

Nesta década, os sistemas experimentais da década anterior se tornaram produtos comerciais sólidos. Os sistemas operacionais dessa época eram multiprogramados que suportavam processamento em lote, aplicações de tempo real e tempo compartilhado. Assim como os produtos da década anterior, os lançamentos desta década reafirmaram este novo mercado de tecnologia da informação.

3.3.4. Década de 1980

Na década da evolução dos computadores pessoais e das estações de trabalho, o *Personal Computer* (PC) da IBM, lançado em 1982, e o modelo concorrente da Apple Macintosh, lançado em 1984, permitiram que pessoas e empresas tivessem seus próprios computadores. Sistemas operacionais mais

trabalhados, desenvolvimento de interfaces gráficas com o usuário e aplicativos (editores de texto, planilhas de cálculo, pacotes gráficos e banco de dados) ajudaram a impulsionar a revolução da computação pessoal.

À medida que os custos das tecnologias caíam, a troca de informações entre computadores conectados em rede tornava-se mais prática e econômica [Machado; Maia, 2007]. Aplicações de vários tipos (correio eletrônico, transferência de arquivos e acesso a banco de dados remotos) proliferaram no novo mercado que surgia.

Nesta década, cresceu a computação distribuída com o desenvolvimento do modelo cliente/servidor.

3.3.5. Década de 1990 aos Dias Atuais

Originalmente, os sistemas operacionais executavam o gerenciamento isolado de recursos em um único computador. Porém, com o surgimento da *World Wide Web* na década de 90 e o avanço das tecnologias de redes de computadores, a computação distribuída tornou-se trivial nos computadores pessoais e o suporte a sistemas operacionais com serviços de rede tornava-se um padrão [Brookshear, 2002].

Contudo, ao mesmo tempo em que a revolução dos computadores pessoais na década de 80, o desenvolvimento das redes de computadores e o desenvolvimento de sistemas operacionais (de rede e distribuídos) aconteciam, as ameaças à segurança dos computadores cresciam de forma paralela a estes acontecimentos. Nesta mesma década, surgiram os sistemas operacionais conhecidos até hoje.

A Microsoft, usando conceitos popularizados anteriormente pela Macintosh, como ícones, *menus* e janelas, desenvolveu o Windows, uma

interface gráfica sobreposta ao sistema operacional DOS (*Disk Operating System*). Seguida a evolução do Windows 3.0 ao Windows 98, no final da década de 90, este sistema havia praticamente dominado o mercado de sistemas operacionais. Ainda nos anos 90, outra variante de sistema operacional surgiu como alternativa a usuários que buscavam mais flexibilidade e segurança. Esta variante foi os sistemas operacionais de código aberto, como Linux, FreeBSD e OpenBSD, que tornaram-se concorrentes diretos das soluções proprietárias.

Com o desenvolvimento dos computadores dotados de multiprocessadores, as linguagens de programação sequencial são complementadas por linguagens de programação concorrente que habilitam a especificação de computação paralela [Cosnard; Trystram, 1995]. Um número cada vez maior de sistemas exibe paralelismo maciço, ou seja, possuem grandes quantidades de processadores de modo que muitas partes independentes das computações podem ser executadas em paralelo. Esse é um conceito de computação drasticamente diferente da computação sequencial dos últimos 60 anos.

Os sistemas operacionais atuais padronizam interfaces com o usuário e de aplicação para facilitarem o uso e suportarem maior número de programas [Deitel *et al*., 2005]. Outro ramo atual é o desenvolvimento de aplicativos para sistemas móveis (PDAs – *Personal Digital Assistants* e aparelhos celulares), cujas aplicações são, entre outras: enviar e receber *e-mails*, navegar na Web e capturar e editar imagens. Desta maneira, para o controle do hardware e a gerência dos recursos do dispositivo, sistemas operacionais especiais, em versões enxutas, são desenvolvidos para eles [Lee *et al*., 2005], tais como, Windows Mobile, PalmOS, Linux e Symbian.

3.4. Serviços

A parte do sistema operacional que contém os principais componentes responsáveis pelos serviços prestados chama-se núcleo (*kernel*). O *kernel* difere de sistema operacional para sistema operacional, porém, entre os principais componentes, estão [Deitel *et al*., 2005; Brookshea, 2002]: i) Gerenciador de processos (*scheduler*); ii) Gerenciador de memória; iii) Gerenciador de dispositivos de entrada/saída; e iv) Gerenciador de sistema de arquivos. Estes componentes são apresentados resumidamente nas próximas seções.

3.4.1. Gerenciador de Processos

3.4.1.1. Processos e *Threads*

O conceito de processo é um dos mais fundamentais dos sistemas operacionais modernos. Enquanto um programa é apenas um conjunto estático de comandos, a sua execução é uma atividade mais dinâmica, cujas propriedades mudam à medida que o tempo avança. Esta atividade é denominada processo [Bookshear, 2002].

Os processos existentes em um sistema operacional representam entidades independentes executáveis e competem por recursos lógicos e de hardware [Shay, 1996]. O sistema operacional não está preocupado diretamente com o usuário ou com o programa, sua responsabilidade básica é com os processos que devem ser executados. É importante ressaltar a variação do número de programas por usuário e o número de processos por programa. Os recursos que os processos podem usar são [Shay, 1996]: i) memória; ii) dispositivos de entrada/saída; iii) processos; iv) CPU (*Central Processing Unit*); e v) arquivos.

Em computação, um processo é uma instância em execução de um programa, incluindo as suas variáveis e estados (memória, código e dados,

descritores de arquivos, semáforos). Na maioria dos sistemas operacionais do mercado, a unidade básica de utilização do processador não é o processo, mas o *thread*. [Flynn; Mchoes, 2002].

Thread é a unidade básica de uso do processador que pertence a um processo e compartilha código, dados e outros recursos que pertençam ao processo; no entanto, um *thread* possui seu próprio contador de programa, conjunto de registradores e pilhas e pode ser executado separadamente caso um processo tenha mais de um *thread* [Starke, 2005].

O que os *threads* acrescentam ao modelo de processo é permitir múltiplas execuções ocorrerem no mesmo ambiente de processo com alto grau de independência um do outro. Ter múltiplos *threads* executando em um único processo é análogo a múltiplos processos executando em paralelo em único computador [Tanenbaum, 2003b].

3.4.1.2. Gestão do Processador

O gerenciamento de processador é a base de sistemas operacionais multiprogramados. Com o chaveamento do processador entre os processos, os sistemas operacionais podem tornar o computador mais produtivo. Desta forma, o gerenciador de processos é responsável por determinar quando e por quanto tempo um processo é executado.

O gerenciador do processador verifica se o processador está executando um processo ou esperando um comando de leitura ou de escrita para finalizar a execução. Depois de alocar os recursos ao processador, o gerenciador cria os registradores e as tabelas necessárias e, assim que a tarefa for finalizada ou o tempo tiver terminado, ele libera o processador novamente [Flynn; Mchoes, 2002]. As tarefas associadas à coordenação de processos são manuseadas pelo

escalonador (*scheduler*) e pelo despachante (*dispatcher*) no interior do *kernel* do sistema operacional [Holcombe; Holcombe, 2003].

O *scheduler* é responsável por manter um registro dos processos presentes no sistema, por incluir novos registros nesse conjunto e por remover os registros quando os processos terminarem [Willrich, 2008]. Para realizar essa tarefa, ele mantém na memória principal uma estrutura denominada tabela de processos. Esta estrutura contém os dados (área de memória reservada ao processo, obtida pelo gerenciador de memória, indicador de seu estado e sua prioridade) relacionados a cada processo. Para cada nova tarefa, o *scheduler* cria um processo acrescentando nova linha na tabela de processos. À medida que o processo progride para o estado finalizado, o *scheduler* mantém atualizadas as informações desse processo na tabela de processos.

O *dispatcher* assegura que os processos escalados pelo *scheduler* sejam de fato executados [Willrich, 2008]. Para isso, em sistemas de tempo partilhado, o *dispatcher* divide o tempo de uso do processador em fatias chamadas de *quantum* e alterna a disponibilidade do processador entre os processos da fila de processos dentro dessas fatias de tempo. Quando um processo inicia seu *quantum*, o *dispatcher* inicia um circuito temporizador responsável por medir quando será o início do próximo *quantum*. Ao término desse *quantum*, o relógio do computador (*clock*) gera um sinal denominado interrupção. Quando o processador recebe uma interrupção, ele termina seu ciclo corrente, guarda sua posição e inicia a execução da rotina de tratamento de interrupção. Neste momento, o *dispatcher* permite que o *scheduler* atualize a tabela de processos. Em seguida, o *dispatcher* seleciona o processo de maior prioridade na fila de processos e reinicia o temporizador. O processo de alternar a disponibilidade do processador entre os processo é chamado chaveamento de processos. Para decidir quais processos executar em determinado instante, políticas de

escalonamento são desenvolvidas para satisfazer alguns critérios de desempenho para maximizar o uso do processador.

3.4.2. Gerenciador de Memória

A memória é destinada a armazenar dados e códigos de programas temporariamente para o uso de outro dispositivo, por exemplo, um processador. Na arquitetura atual dos computadores, um processo precisa estar em memória principal para ser executado, pois é nela que o processador busca as instruções para execução. Cada processo obtém uma quantidade finita de memória de acordo com sua necessidade. Assim sendo, quanto maior a quantidade de memória, mais processos podem ser alocados simultaneamente tornando um sistema multiprogramado mais eficiente.

A função do gerenciador de memória é manter o controle de quais partes da memória estão ou não estão em uso, alocando memória aos processos quando eles precisam e liberando memória quando esses processos terminam, além de gerenciar a troca de processos (*swapping*) entre a memória e o disco quando a memória principal está cheia [Oliveira *et al*., 2001].

Em um sistema monoprogramado, a memória principal é dividida em duas partes [Stallings, 2005]: i) parte reservada para o sistema operacional; e ii) parte reservada para outros programas. Enquanto em sistemas multiprogramados, além de reservar uma parte para o sistema operacional, a memória reservada para cada processo é subdividida dinamicamente em várias partições para acomodarem múltiplas instâncias desse processo (*threads*).

Independente do esquema de organização de memória que um sistema em particular adota, estratégias de uso devem ser decididas para obter melhor desempenho da memória. Assim, o gerenciador de memória é um componente

do sistema operacional que se preocupa com o esquema de organização da memória e com as estratégias de gerenciamento [Deitel *et al*., 2005].

A Figura 3-1 mostra uma estrutura hierárquica típica de memória e está estruturada em níveis diferenciados pelo tempo de acesso e capacidade física. A memória de armazenamento secundário está no nível mais baixo, enquanto os registradores (“memória de processador”) estão no nível mais alto.

Figura 3-1 – Típica Hierarquia de Memória (Fonte: Adaptação Tanenbaum (2003b))

Sendo a memória um componente de tamanho fixo, não é possível armazenar infinitos processos, tampouco processos muito grandes que possam se ajustar à memória. Caso o espaço livre da memória não seja suficientemente grande para alocar os processos, o sistema pode usar a memória secundária para armazenar o código e os dados sobre o processo enquanto eles não estão em execução. Este espaço é conhecido como memória virtual [Starke, 2005].

Os endereços gerados pelos processos (endereços virtuais) formam o espaço de endereçamento virtual que é dividido em páginas [Tanenbaum; Woodhull, 1997]. As páginas são unidades de memória onde os processos são armazenados. A memória física é dividida em quadros que possuem o mesmo tamanho das páginas e podem armazenar o seu conteúdo.

O mapeamento dos endereços virtuais para os endereços físicos é feito pela unidade de gerenciamento de memória [Silberschatz *et al*., 2004a]. Para

realizá-lo, existem vários métodos (alocação contígua, paginação, segmentação e segmentação com paginação). À medida que os processos entram no sistema, eles são colocados em uma fila de entrada. O sistema operacional leva em consideração as exigências de memória de cada processo e a quantidade de memória livre para definir quais processos deve receber espaço na memória. Quando um processo recebe espaço, ele é carregado e pode concorrer pela CPU. Assim, têm-se uma lista de blocos disponíveis e a fila de entrada. A memória é alocada aos processos até que as exigências de memória do processo seguinte não possam ser satisfeitas. Dessa forma, tem-se o problema de espaços ociosos de vários tamanhos espalhados pela memória.

Para resolver o problema de alocação dinâmica, ou seja, para percorrer a lista de blocos disponíveis na memória e preenchê-los, existem três algoritmos que podem ser considerados [Stallings, 2005]: *Best-Fit*, *First-Fit* e *Next-Fit*. Estas políticas, entre outras, são tratadas mais detalhadamente no Capítulo 5.

3.4.3. Gerenciador de Dispositivos de Entrada/Saída

O gerenciador de dispositivos de entrada/saída monitora os dispositivos, os canais e as unidades de controle. Sua tarefa é selecionar a forma mais conveniente para a alocação dos dispositivos (*scanners*, impressoras, memórias portáteis, entre outros) de acordo com uma regra de programação de execução definida anteriormente pelos projetistas. Resumidamente, o gerenciador de dispositivos aloca, realiza a operação e, em seguida, desaloca o dispositivo [Flynn; Mchoes, 2002].

Existem duas categorias de dispositivos [Silberchatz *et al*., 2004b]: i) dispositivos de bloco, eles armazenam informação em blocos de tamanho fixo cada um com seu endereço próprio para proporcionar independência entre eles; e

ii) dispositivos de caracteres, eles enviam ou recebem um fluxo de caracteres sem considerar qualquer estrutura de bloco.

Quanto aos objetivos dos dispositivos de entrada/saída dos sistemas operacionais, devem ser consideradas a eficiência e a generalidade [Hayes, 1988]. A eficiência é importante, pois operações de entrada/saída frequentemente formam um gargalo no consumo dos recursos computacionais. Assim, uma forma de resolver este problema é com multiprogramação. A generalidade é desejável para ter dispositivos com estilo uniforme, isto é, padronização. Uma maneira de resolver este problema é usar uma hierarquia modular que aproxima ao *design* dos recursos de entrada/saída. Esta aproximação abstrai parte dos detalhes das rotinas de baixo nível dos dispositivos. Dessa forma, processos e camadas superiores do sistema operacional vêem os dispositivos em termos de funções gerais.

3.4.4. Gerenciador de Sistema de Arquivos

Nos últimos 40 anos, o incremento na velocidade dos processadores e das memórias principais tem sido aproximadamente duas vezes maior do que o incremento na velocidade dos discos rígidos (*hard disk*). Assim, o desempenho no acesso aos subsistemas de armazenamento em disco é de vital relevância e deve ser tratado com bastante cuidado pelo sistema operacional [Stallings, 2005].

As aplicações precisam armazenar e recuperar informações. Dessa forma, alguns problemas podem acontecer durante a execução dos programas. Por exemplo, um processo pode precisar armazenar uma quantidade de dados maior do que o espaço alocado para ele; quando o processo termina, as informações processadas e armazenadas na memória são perdidas e, muitas vezes, múltiplos processos precisam ter acesso a informação ao mesmo tempo. Assim, a solução

usual encontrada para resolver esses tipos de problemas é armazenar as informações em discos ou em outros meios persistentes. A unidade de armazenamento nesses tipos de meios persistentes é chamada arquivo [Tanenbaum; Woohull, 1997].

Quando um arquivo é criado, pode-se definir qual estrutura de organização será adotada e suportada pelo sistema operacional. A forma mais simples de organização dos arquivos é usando sequências de *bytes*, onde a estrutura lógica para os dados não é imposta; neste caso, a aplicação deve definir a organização. Esse modelo apresenta como vantagem a flexibilidade para criar estruturas de dados, porém o controle sobre eles passa a ser de responsabilidade da aplicação. O sistema de arquivos pode recuperar os registros de diferentes maneiras [Laureano, 2008]: i) acesso sequencial (a leitura dos arquivos é restrita na ordem em que as informações foram gravadas, sendo a gravação possível somente no fim do arquivo); ii) acesso direto (a leitura/gravação de uma informação é feita diretamente na sua posição); ou iii) acesso indexado (há uma área de índice no arquivo onde são armazenados ponteiros para diversas informações e, a partir desse índice, é realizado o acesso direto).

Os arquivos possuem os seguintes atributos básicos [Menezes, 2008]: nome, tipo, localização, tamanho, proteção, hora, data e identificação do usuário que o criou. Essas informações são armazenadas na forma de diretórios e mantidas no disco. As operações básicas disponíveis sobre os arquivos são: criar, escrever, ler, remover, truncar, abrir e fechar.

Magalhães *et al*. (1992) e Tanenbaum (2003b) explicam que o acesso a disco é mais lento que o acesso a memória principal. Assim, a técnica mais comum para reduzir o tempo de acesso ao disco é *buffer cache*. Uma *cache* é uma coleção de dados que pertencem ao disco, mas são mantidas na memória

para melhorar o desempenho em decorrência da diferença de velocidade de acesso. Existem vários algoritmos para gerenciar a *cache*, o mais usado verifica as requisições de leitura para saber se o bloco referido está na *cache*. Se estiver, a requisição simplesmente lê os dados, senão ele é inicialmente lido para a *cache* e copiado para a área de processo que requisitou o acesso. Caso um bloco seja carregado para a *cache* e ela esteja cheia, algum bloco terá que ser removido; para isso, são usados os mesmos algoritmos de paginação de memória [Tanenbaum, 2003b].

3.5. Considerações Finais

Este capítulo apresentou uma visão geral sobre sistemas operacionais, mostrando a sua definição para alguns autores da literatura, um breve histórico de sua evolução, sua importância quanto ao uso de um computador e alguns de seus serviços básicos, como: i) gerenciador de processos; ii) gerenciador de memória; iii) gerenciador de dispositivos de entrada/saída; e iv) gerenciador de sistema de arquivos.

Nos próximos dois capítulos, são abordados de forma mais detalhada os serviços de gerência de memória e os serviços de gerência de processos. Para cada gerência, são apresentadas diversas políticas encontradas na literatura.

4. GERÊNCIA DE PROCESSOS

4.1. Considerações Iniciais

Informalmente, processo pode ser visto como um programa em execução, porém esta é uma idéia incompleta de como um sistema operacional o trata. Atualmente, os computadores são capazes de realizar várias tarefas ao mesmo tempo. Ou seja, enquanto executa um programa, um computador pode, por exemplo, realizar operações de leitura/escrita em memória secundária ou executar outro programa. Em sistemas monoprocessados, as tarefas não são executadas ao mesmo tempo, existe apenas a impressão delas estarem sendo executadas ao mesmo tempo, pois o processador é capaz de realizar uma tarefa por vez. Nesse sentido, existe o conceito de multiprogramação que consiste na capacidade de armazenar o código de muitos processos (programas) na memória simultaneamente para que eles possam ser executados [Shay, 1996]. Assim, os programas podem acessar os recursos do computador e serem executados em um tempo razoável tendo a sensação de multiprocessamento.

Neste contexto, basicamente o gerenciador de processos garante que cada processo receba uma fatia de tempo suficiente para funcionar corretamente e escalona os processos presentes na fila de processos para que eles sejam executados. A seção 4.2 apresenta algumas definições relacionadas ao assunto. A seção 4.3 aborda sucintamente algumas políticas de escalonamento de processos usadas pelos sistemas operacionais citadas pelos principais autores na literatura.

4.2. Definições

4.2.1. Processo

Um programa por si só não é um processo, pois ele é uma entidade passiva, enquanto processo é uma entidade ativa composta por atributos como [Silberchatz *et al*. 2004b]: i) contador de programa (especifica a próxima instrução a ser executada e um conjunto de recursos associados); ii) conteúdo dos registradores do processador; iii) pilha de execução (contém dados temporários como: argumentos de métodos, variáveis locais, etc.); iv) seção de dados (contém variáveis globais); e v) pilha de *heap* (memória alocada dinamicamente durante tempo de execução).

No decorrer da história, a palavra "processo", no contexto de sistemas operacionais, recebeu várias definições como [Deitel *et al*., 2005]: i) uma atividade assíncrona; ii) um programa em execução; iii) uma entidade a qual o processador é designado; e iv) uma unidade de despacho.

4.2.2. Estados dos Processos

Ainda que cada processo seja independente de outras entidades, tendo seu próprio contador de programa e estado interno, ele pode precisar interagir com outros processos. Por exemplo, um processo pode gerar uma saída para ser usada como entrada por outro processo. Nesta situação, o processo que recebe a informação poderia estar em estado de espera. Assim, um processo possui um estado e a definição desses possíveis estados varia entre os autores:

- Um processo pode estar nos seguintes estados: i) em execução (usando o processador naquele instante); ii) pronto (temporariamente parado para dar lugar a outro processo); e iii) bloqueado (aguardando o término de um evento externo para que possa continuar a execução) [Tanenbaum, 2003b];

- Um processo pode estar nos seguintes estados: i) novo (criação de processo); ii) executando (execução de instruções pertencentes ao processo); iii) esperando (espera do processo por algum evento externo); iv) pronto (espera do processo para ser atribuído a um processador); e v) terminado (término do processo) [Silberchatz *et al*. 2004b].

Tais definições são arbitrárias e, além de variar entre os autores, variam entre os sistemas operacionais. Na próxima seção, é apresentado o modelo de implementação de processo utilizado pelos sistemas operacionais.

4.2.3. Tabela de Processos e Bloco de Controle de Processo

Os sistemas operacionais implementam o modelo baseado em processos usando uma tabela denominada tabela de processos, onde são armazenados os atributos de cada processo. Estes atributos agrupados formam um bloco de controle de processo [Willrich, 2008].

O sistema operacional deve manter várias informações relativas aos processos na tabela de processos, tratando-os como um registro denominado bloco de controle de processo ou descritor de processo [Oliveira *et al*., 2001]. Este registro une as seguintes informações que o sistema operacional precisa saber sobre o processo [Oliveira *et al*., 2001; Tanenbaum, 2003b]:

- Identificador do processo;
- Prioridade do processo (usado para definir a ordem na qual os processos serão encaminhados ao processador);
- Estado do processo;
- Localização e quantidade de memória principal usada pelo processo;
- Ponteiros para o encadeamento dos blocos de controle dos próximos processos;

- Informações de contabilidade (tempo de processador gasto, espaço de memória ocupado, momento em que o processo começou, etc.);
- Identificador do processo pai;
- Contador de programa (valor que determina qual instrução o processo deve executar em seguida);
- Credenciais (dados que determinam os recursos que esse processo pode acessar);
- Ponteiro de pilha de processos.

4.2.4. Operações de Processo

O sistema operacional deve ser capaz de realizar certas operações com relação a processos [Deitel *et al*., 2005]: i) criar; ii) destruir; iii) suspender; iv) retomar; v) alterar prioridade; vi) bloquear; vii) acordar; viii) despachar e ix) habilitar um processo para se comunicar com o outro. Um processo é capaz de criar outro processo. Assim, o processo criador é chamado de processo-pai e o processo criado é chamado de processo-filho. Estas criações levam a uma estrutura hierárquica de processos semelhante a uma árvore de descendência [Silberchatz *et al*., 2004b].

A partir das operações básicas relacionadas a processos, o sistema operacional realiza a troca de contexto para interromper um processo em execução e começar a execução de outro que esteja na fila de processos. Para fazer esta troca de contexto, o sistema operacional salva o contexto de execução do processo no bloco de controle de processos correspondente (fotografia do estado atual do processo) e carrega o processo a ser executado. Para realizar o chaveamento em sentido contrário, simplesmente são feitas as mesmas operações, porém no sentido contrário [Scama, 2008].

Nos sistemas que adotam a multiprogramação, os vários processos disputam os recursos disponíveis a cada momento. Logo, o sistema operacional precisa gerenciar esta disputa por recursos para que os processos sejam atendidos de forma eficiente. Em outras palavras, o sistema operacional divide o recurso “tempo de processador” entre os processos [Anderson, 1981].

4.3. Políticas de Escalonamento

Esta seção trata as várias políticas de escalonamento de processos citadas pelos principais autores na literatura. A tarefa dessas políticas é escolher qual dos processos presentes na fila de processos será o próximo a ser executado.

Como critério básico é usado o objetivo de aumentar ao máximo a produção (*throughput* = número de processos concluídos por unidade de tempo) do sistema operacional, diminuindo, ao mesmo tempo, o tempo de resposta (*turnaround time* = intervalo de tempo da submissão de um processo até o seu termino) ao usuário Oliveira *et al*., 2001]. Para conseguir um aumento de eficiência no processador, busca-se reduzir o tempo médio de espera na fila de processos prontos. O tempo de espera é a soma dos períodos gastos por um processo aguardando na fila de espera [Silberchatz *et al*., 2004b].

Existem duas categorias principais de políticas de escalonamento [Shay, 1996]: preemptiva e não-preemptiva. Nas políticas de escalonamento não-preemptivas, quando um processo assume o controle do processador, ele o mantém até terminar. Sua principal vantagem em relação às políticas preemptivas é a simplicidade. Sua desvantagem é a falta de resposta ao comportamento do sistema operacional quando, por exemplo, um processo com alto tempo de processamento detém o controle do processador, os demais processos têm de esperar. Nas políticas preemptivas, o sistema operacional pode

retomar o controle do processador, por algum motivo que não seja o término do processo em execução, e passá-lo para outro.

4.3.1. FIFO (*First In First Out*)

Também denominada escalonamento FCFS (*First Come First Served*) (Figura 4-1), esta política é provavelmente a mais simples de ser entendida e implementada [Shay 1996]. Esta política funciona da seguinte forma: quando um processo é iniciado, o sistema operacional armazena seu bloco de controle de processo no fim de uma fila. Esta fila é denominada fila de processos prontos. Quando o processador estiver disponível, o sistema operacional passa o controle para o primeiro processo dessa fila. O processo utiliza o processador até terminar e o sistema operacional passa o controle do processador ao processo seguinte. Em outras palavras, os processos são executados na mesma ordem em chegaram na fila.

O sistema operacional retoma o controle do processador somente quando o processo em execução realiza chamada de sistema ou quando ocorre erro na execução [Oliveira *et al.*, 2001].

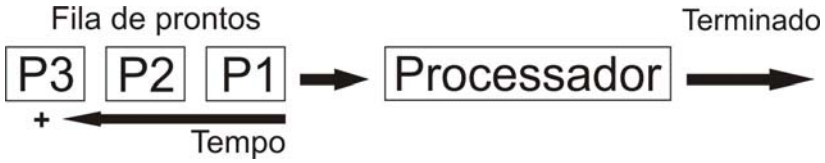

Figura 4-1 – Escalonamento FIFO

4.3.2. SJF (*Shortest Job First*)

A idéia desta política é passar o controle do processador para o processo que tenha o menor tempo de execução [Rocha *et al.*, 2004]. Tal política possui algumas características: i) não-preempção; ii) processos pequenos têm tempo de espera médio menor que os processos grandes; iii) pode causar postergação

indefinida nos processos grandes; e iv) minimiza o tempo médio de espera de um conjunto de processos, pois os processos menores são executados mais rapidamente.

O algoritmo desta política poderia ser implementado usando uma lista ordenada na ordem crescente dos tempos de vida dos processos [Oliveira *et al.*, 2001]. Os processos são inseridos ordenadamente na lista e no escalonamento, basta pegar o primeiro da lista.

O problema para implementar esta política é prever o futuro [Oliveira *et al.*, 2001]. A duração exata de um processo é desconhecida, pois depende, entre outras coisas, dos dados de entrada do programa em execução. Assim, mesmo não sendo possível implementar essa política, ela é útil pois oferece um limite teórico para o tempo médio de espera e a partir dela, pode-se implementar aproximações, usando, por exemplo, cálculos estatísticos sobre dados dos últimos processos executados.

A Figura 4-2 mostra um esquema representativo do uso da política SJF. Neste esquema, P1, P2, P3 e P4 possuem, respectivamente, tempos de criação iguais a 0, 2, 4 e 5 unidades de tempo e tempos de execução iguais a 5, 3, 1 e 3 unidades de tempo.

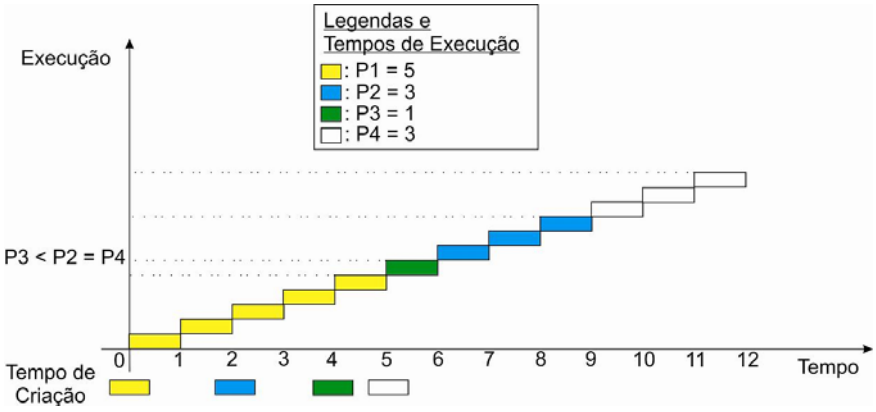

Figura 4-2 – Exemplo de Uso da Política SJF

4.3.3. SRTF (*Shortest Remaining Time First*)

Também conhecida como SRTN (*Shortest Remaining Time Next*), esta política diferencia da anterior somente quanto à preempção. Ou seja, nesta política, um processo pode perder o controle do processador para outro processo que tenha tempo de execução menor que o seu tempo de execução restante. Caso o novo processo tenha tempo de execução igual ao que está executando, o novo é inserido na fila de processos.

A Figura 4-3 mostra um esquema representativo do uso da política SRTF. Neste esquema, P1, P2, P3 e P4 possuem, respectivamente, tempos de criação iguais a 0, 2, 4 e 5 unidades de tempo e tempos de execução iguais a 7, 4, 1 e 4 unidades de tempo.

4.3.4. HRRN (*Highest Response Ratio Next*)

Segundo Deitel *et al*. (2005), também conhecida como política de próxima taxa de resposta mais alta, esta política tenta resolver algumas deficiências da política SJF, especificamente tratando a postergação de processos grandes e o favoritismo excessivo em relação a processos pequenos. Esta é uma política não-preemptiva na qual define dinamicamente a prioridade de cada processo a partir da fórmula:

$$\text{Prioridade} = \frac{tempo\ \ de\ \ espera + tempo\ \ de\ \ vida}{tempo\ \ de\ \ vida}$$

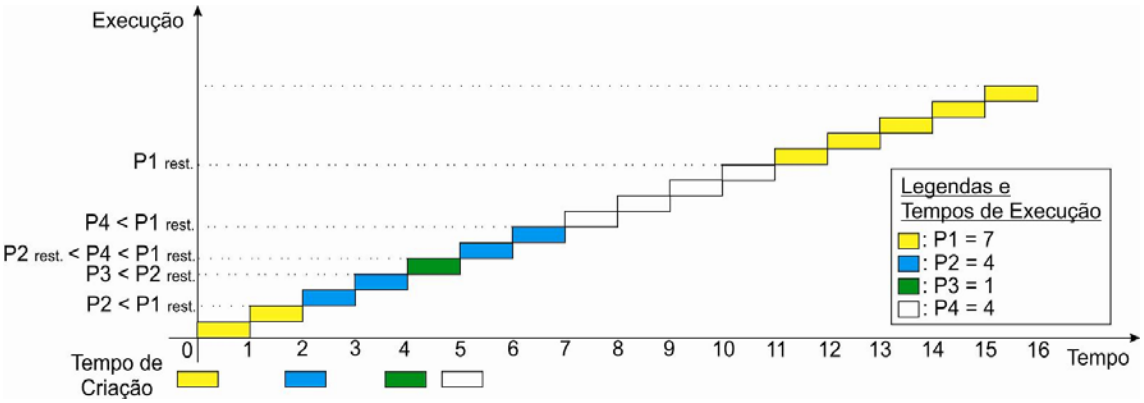

Figura 4-3 – Exemplo de Uso da Política SRTF

Analisando a fórmula, com o tempo de vida no denominador, processos que levam menor tempo para terminar recebem preferência. Entretanto, com o tempo de espera no numerador, processos que levam maior tempo para terminar recebem tratamento preferencial. Para cada escalonamento de processo esta conta é feita e o gerenciador de processos escalona aquele de maior prioridade, impedindo o adiamento indefinido da execução de processos. Quando há igualdade nas prioridades calculadas, esta política age como a política SJF, ou seja, será escalonado o processo com menor tempo de vida.

A Figura 4-4 mostra um esquema representativo do uso da política HRRN. Neste esquema, P1, P2, P3 e P4 possuem, respectivamente, tempos de criação iguais a 0, 0, 2 e 4 unidades de tempo e tempos de execução iguais a 5, 3, 1 e 3 unidades de tempo. Na primeira interação, P1 e P2 possuem prioridade igual a 1 e esta política escalona o menor dos dois processos (no caso, foi escolhido P2). Na segunda interação, P3 possui prioridade (igual a 2) maior do que a prioridade de P1 (igual a 1,6). Na terceira interação, P1 possui prioridade (igual a 1,8) maior do que a prioridade de P4 (igual a 1). Por fim, P4 possui prioridade igual a 2,66.

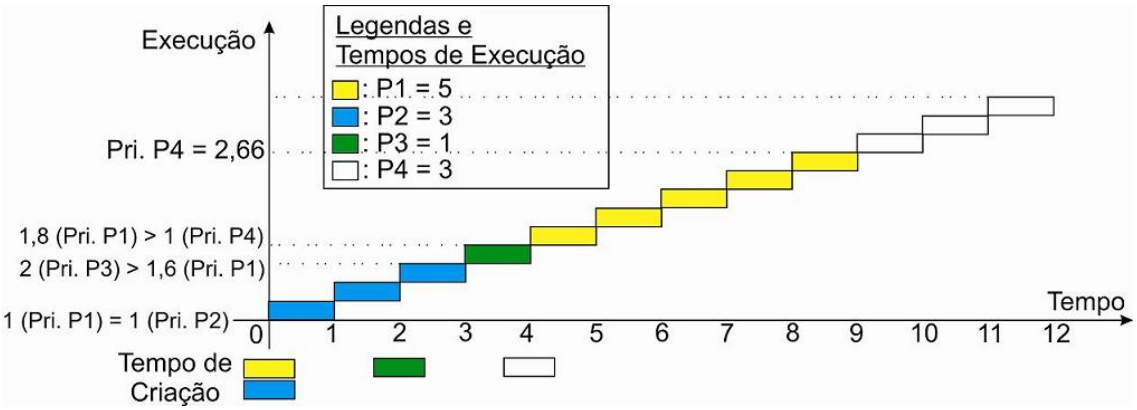

Figura 4-4 – Exemplo de Uso da Política HRRN

4.3.5. Por Prioridade

A idéia básica dessa política é simples [Tanenbaum, 2003b]: para cada processo, é atribuída uma prioridade e o controle do processador é passado ao processo com maior prioridade. Esta política pode ser implementada na forma preemptiva ou não-preemptiva. Alguns sistemas utilizam o zero como a maior prioridade enquanto outros o utilizam como a menor prioridade. Para evitar a execução indefinidamente de processos com prioridades altas, o *scheduler* pode reduzir a prioridade do processo em execução a cada tique do relógio (*clock* do processador) [Tanenbaum, 2003b]. Assim, considerando preempção nesta política, caso a prioridade do processo ativo seja menor do que a prioridade de um processo na fila de processos, então haverá substituição de processo. Se o processo que sofreu redução em sua prioridade volte para a fila de processos, sua prioridade volta para o valor original [Oliveira *et al*., 2001]. Por outro lado, um *quantum* máximo (tempo máximo) pode ser definido para cada processo, correspondendo ao quanto tempo ele pode executar. Quando esgotado esse *quantum*, o controle do processador é passado para o próximo processo com prioridade mais alta, ou seja, haverá substituição de processos. Em caso de

empate entre as prioridades de processos, o desempate é feito recorrendo à política FIFO [Silberchatz *et al*., 2004b].

A prioridade de um processo pode ser definida de duas formas [Silva, 2008]:

- Internamente. Isso significa dizer que o sistema operacional utiliza valores mensuráveis para definir a prioridade do processo. Utiliza-se, por exemplo, requisitos de memória, limites de tempo, etc.;
- Externamente. As prioridades são atribuídas de acordo com critérios externos ao sistema, por exemplo, o departamento responsável pela execução do processo, a importância do processo, etc.

A Figura 4-5 mostra um esquema representativo do uso da política Por Prioridade Não-Preemptiva. Neste esquema, P1, P2, P3, P4 e P5 possuem, respectivamente, tempos de execução iguais a 7, 1, 2, 1 e 5 unidades de tempo e prioridades iguais a 3, 1, 4, 5 e 2. Esses processos possuem tempo de criação igual a 0 e a maior prioridade é a de menor valor.

A Figura 4-6 mostra um esquema representativo após o uso da política Por Prioridade Preemptiva. Neste esquema, P1, P2, P3, P4 e P5 possuem, respectivamente, tempos de execução iguais a 7, 1, 2, 1 e 5 unidades de tempo e prioridades iguais a 3, 1, 2, 5 e 2, tempos de criação iguais a 0, 0, 2, 3 e 8. A maior prioridade é a de menor valor.

Tipicamente, a opção preemptiva é usada, pois, em termos práticos, não faz sentido ter esforço para executar processos com prioridades altas mais cedo do que processos com prioridades baixas e, ao mesmo tempo, permitir que um processo com baixa prioridade tome o tempo de processador indefinidamente, uma vez que ele conseguiu iniciar seu ciclo de execução.

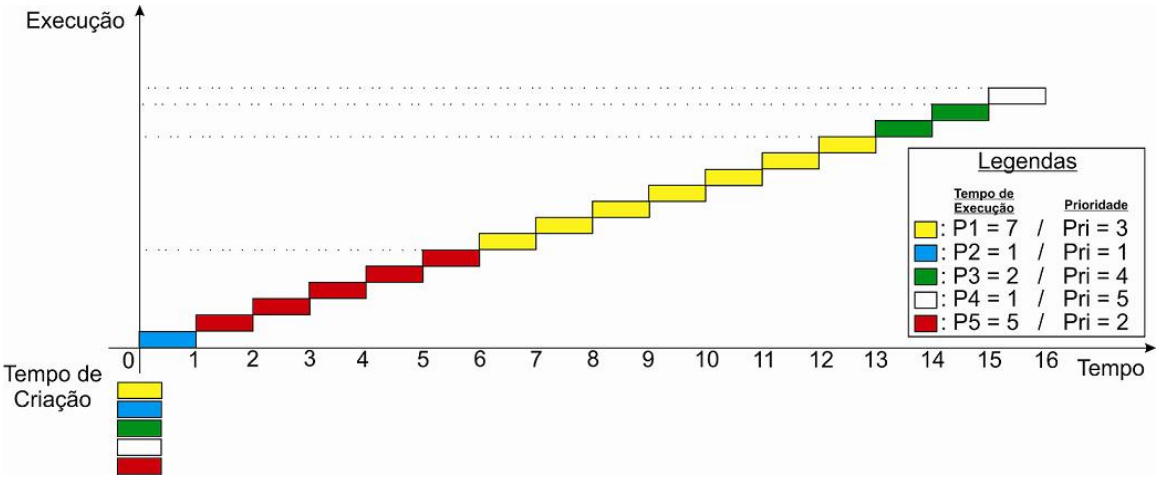

Figura 4-5 – Exemplo de Uso da Política Por Prioridade Não-Preemptiva

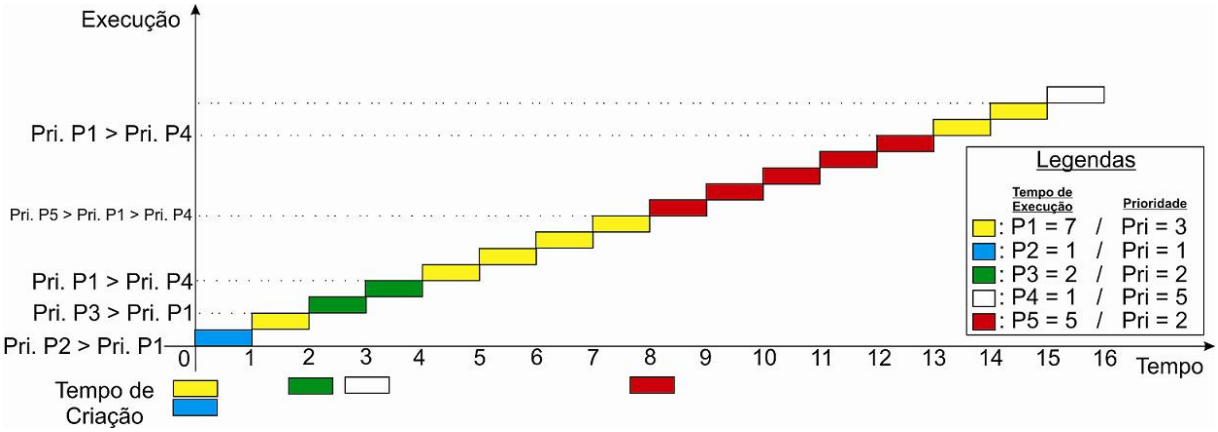

Figura 4-6 – Exemplo de Uso da Política Por Prioridade Preemptiva

4.3.6. *Round-Robin*

Esta política também é citada na literatura como fatia de tempo (*quantum*), por alternância circular ou revezamento [Shay, 1996; Tanenbaum 2003b; Oliveira *et al.*, 2001]. Esta política define uma unidade de tempo denominada *quantum* ou *time slice* e funciona semelhante à política FIFO, porém cada processo recebe uma fatia de tempo de processador para ser

executado [Arruda, 2008]. Assim, o *scheduler* aloca o primeiro processo da fila de processos ao processador durante uma unidade de tempo (*quantum*). Se o processo não terminar a execução após esta unidade de tempo, ocorre a troca de contexto e ele é reinserido no fim da fila de processos. Se ele terminar, o *scheduler* passa o controle do processador para o próximo da lista.

A escolha do tamanho do *quantum* é crítica e deve ser feita com cuidado [Shay, 1996]. Com um *quantum* pequeno, o sistema operacional é forçado a interromper os processos mais frequentemente, afetando desempenho, pois as operações de troca de contexto não são instantâneas, como suposto na Figura 4-7. Porém, com um *quantum* grande, pode-se perder a aparência de paralelismo na execução dos processos.

A Figura 4-7 mostra um esquema representativo após o uso da política *Round-Robin*. Neste esquema, P1, P2, P3 e P4 foram criados juntos e possuem, respectivamente, tempos de execução iguais a 53, 17, 68 e 24 unidades de tempo e a fatia de tempo (*quantum*) é igual a 20 unidades de tempo.

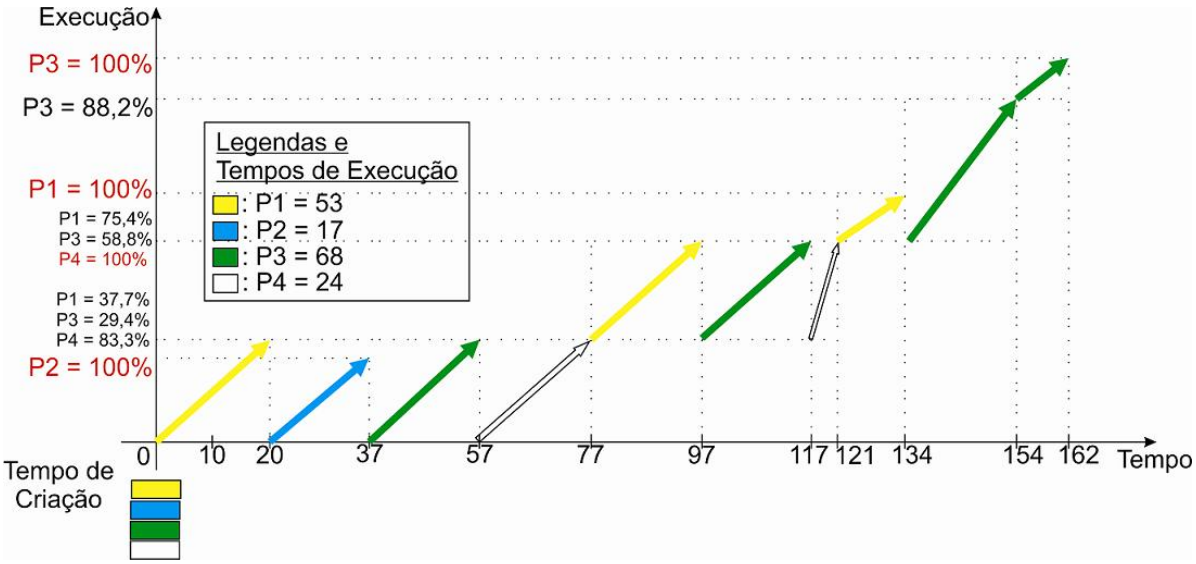

Figura 4-7 – Exemplo de Uso da Política *Round-Robin*

4.4. Considerações Finais

Este capítulo apresentou algumas definições relacionadas a gerência de processos de um sistema operacional, destacando as políticas usadas pelo escalonador de processos (*scheduler*).

Pode-se considerar que, a partir das políticas apresentadas, é possível estabelecer outras políticas, por exemplo, a combinação de duas ou mais políticas (políticas híbridas), porém essa não é a finalidade deste trabalho. A escolha de qual(is) política(s) adotar ou o desenvolvimento de uma híbrida no projeto de um sistema operacional depende de qual será a sua finalidade. Mesmo a política FIFO é viável para certas ocasiões, desde que se tenha controle sobre laços infinitos dos programas. Do mesmo modo que em sistemas de tempo real, a prioridade e os limites de tempo de espera, entre outras características dos processos, devem ser requisitos para bom funcionamento.

No próximo capítulo, são abordados alguns conceitos relacionados a gerência de memória em sistemas operacionais, em especial, são apresentadas políticas presentes na literatura para percorrer a lista de espaços livres na memória principal de um computador.

5. GERÊNCIA DE MEMÓRIA

5.1. Considerações Iniciais

Na multiprogramação, vários processos são executados simultaneamente por meio da divisão do tempo do processador. Para que a troca de contexto entre os diferentes processos aconteça de forma eficiente, eles devem estar na memória principal do computador.

A seção 5.2 apresenta algumas definições relevantes ao assunto. A seção 5.3 apresenta resumidamente funções de um gerenciador de memória em um sistema operacional. A seção 5.4 relata as políticas mais citadas pelos principais autores na literatura e adotadas pelo gerenciador para buscar espaços na memória principal.

5.2. Definições

A seguir, são apresentadas algumas definições relacionadas a gerência de memória em sistemas operacionais. Estas definições são importantes para o entendimento do assunto e das políticas aplicadas pelo gerenciador de memória para encontrar espaços (partições) livres na memória principal de um computador:

- Memória Física. A memória física é implementada pelos circuitos integrados de memória. Os endereços físicos apontam para a memória física, ou seja, os circuitos integrados de memória [Oliveira *et al*., 2001];
- Memória Lógica. A memória lógica possibilita ao processo acessá-la usando suas próprias instruções. Os endereços gerados pela CPU são denominados endereços lógicos, ou seja, as instruções de máquina de um processo especificam endereços lógicos, enquanto o conjunto desses endereços

lógicos é denominado espaço de endereçamento lógico [Silberchatz *et al*., 2004b];

- Unidade de Gerência de Memória. A unidade de Gerência de memória é um componente de hardware responsável por prover os mecanismos usados pelo sistema operacional para gerenciar a memória. Entre outras funções, ela mapeia endereço lógico em endereço físico [Oliveira *et al*., 2001].

5.3. Gerenciador de Memória

Gerenciador de memória é um componente do sistema operacional responsável pelo esquema de organização da memória, pelas estratégias de gerenciamento, por determinar como o espaço livre deve ser alocado a processos e por responder a mudanças de uso da memória de um processo [Deitel *et al*., 2005]. Sua função é manter o controle das partes da memória em uso, alocando espaço para processos quando eles precisarem e liberando quando eles terminam. Além disso, ele gerencia a troca de processos (*swapping*) entre a memória principal e a memória secundária quando a primeira não é suficientemente grande para acomodar os processos [Tanenbaum, 2003b].

5.3.1. Funções do Gerenciador de Memória

Um programa está salvo em dispositivos de memória secundária (discos, CDs, DVDs, fitas) na forma de sequência de *bits* executáveis [Machado; Maia, 2007]. Como o processador somente é capaz de executar o que está em memória principal, o programa a ser executado deve ser carregado para ela e adicionado no sistema operacional na forma de processo. O sistema operacional procura minimizar o número de operações de entrada/saída em dispositivos de armazenamento secundário para que o seu desempenho não fique comprometido, pois o tempo de acesso a dados nestes dispositivos é maior do que o tempo de acesso à memória principal. Dessa forma, o conjunto de

processos em memória secundária forma uma fila de entrada, aguardando seu carregamento para a memória principal para a execução.

O gerenciador de memória deve manter na memória principal o maior número de processos possíveis para garantir o compartilhamento dos recursos computacionais. Mesmo que não haja memória livre, o gerenciador deve permitir que novos processos sejam aceitos na fila de entrada. Outra preocupação desse gerenciador é permitir a execução de processos maiores do que a memória física disponível. Para isso, é usado um esquema de memória virtual [Machado; Maia, 2007].

Em sistemas multiprogramados, o gerenciador de memória deve garantir a integridade dos espaços reservados para os processos em execução, além da área que reside o próprio sistema operacional. Caso algum processo tente acessar outra área de memória que não a reservada para ele, o sistema operacional deve bloquear tal tentativa de acesso [Stallings, 2005].

A memória principal de um computador deve ser considerada em termos de organização. Assim, várias questões podem ser levantadas quanto às diferentes possibilidades e à organização da memória, por exemplo [Deitel *et al.*, 2005]:

- Memória deve conter apenas um processo ou vários processos ao mesmo tempo (multiprogramação)?
- Caso a memória tenha vários processos ao mesmo tempo, deve-se dividi-la igualmente para cada processo ou dividi-la em partições de tamanhos diferentes?
- Deve-se exigir que processos executem em partições específicas ou em qualquer lugar onde couberem?

- Deve-se permitir que o sistema operacional coloque cada processo em blocos contíguos de endereçamento ou dividi-los em blocos separados?

Dessa forma, nas próximas seções, são apresentadas duas formas de gerência de memória relativas a multiprogramação que tentam responder tais questões: multiprogramação por partição fixa e multiprogramação por partição variável.

5.3.1.1. Multiprogramação por Partições Fixas

Esta é a forma mais simples de gerência de memória para multiprogramação. Neste caso, a memória é dividida em duas partes, sendo uma para o sistema operacional e a outra para os processos do usuário. Em seguida, a partição do usuário é dividida em várias outras partições de diferentes tamanhos, porém fixos [Oliveira *et al*., 2001].

Os processos presentes na fila de entrada são associados às menores partições capazes de armazená-los. Assim, pelo fato das partições terem tamanho fixo, o restante de espaço disponível não usado pelo processo é perdido (fragmentação interna). Outra possibilidade é quando há memória perdida fora da área ocupada por um processo (fragmentação externa) [Laine, 2008]. Por exemplo: em certo momento, existem dois blocos de memória, 50 e 200 *Kbytes*, respectivamente. Nesse mesmo instante, é criado um processo para executar um programa de 230 *Kbytes* que não poderá ser carregado em memória pela forma que ela é gerenciada, sendo que a memória total livre no momento é de 250 Kbytes.

5.3.1.2. Multiprogramação por Partições Variáveis

Neste esquema de organização, a quantidade, o tamanho e a localização das partições devem variar dinamicamente com o tempo, ou seja, à medida que

os processos entram e saem da memória. Estas partições devem ser ajustadas dinamicamente às necessidades exatas de cada processo [Medeiros, 2008].

Uma das principais vantagens desta estratégia em relação a multiprogramação por partições fixas é a flexibilidade alcançada em aumentar o uso da memória, reduzindo o desperdício de espaço. Entretanto, está gerência é mais difícil e pode gerar fragmentação externa. Para isso, o sistema operacional mantém na memória uma lista de partições livres e a manipula de acordo com a estratégia implementada [Laine, 2008].

Quando um processo precisa de memória, o gerenciador de memória percorre a lista de partições livres em busca de uma partição suficientemente grande para esse processo. Caso a partição encontrada seja muito grande, ela é dividida em duas partes na qual uma é destinada ao processo e a outra volta para a lista de partições livres, assim como acontece quando um processo termina. Caso a partição liberada seja adjacente a uma partição livre, as duas partições são agrupadas para formar uma maior [Silberchatz *et al*., 2004b].

5.3.2. *Swapping*

Há momentos nos quais não é possível manter os processos simultaneamente na memória. Uma solução para esse tipo de problema é denominada *swapping* [Tanenbaum; Woodhull, 1997].

Swapping é uma técnica aplicada na memória principal que visa propiciar maior taxa de utilização a ela, melhorando o seu compartilhamento [Oliveira *et al*., 2001]. Além disso, esta técnica visa resolver o problema da falta de memória principal no sistema operacional. O gerenciador de memória do sistema operacional reserva um espaço no disco para uso próprio. Em determinadas situações, a execução de certos processos é suspensa. Quando um processo é

suspenso, seu descritor de processo é retirado da fila do processador e colocado na fila de processos suspensos (*swap-out*). Após um período de tempo esperando na fila de processos suspensos, ele é copiado novamente para a memória (*swap-in*) e, posteriormente, executado [Oliveira *et al*., 2001].

A técnica de *swapping* provoca aumento no tempo de execução dos programas, pois as operações *swap-out* e *swap-in* são demoradas. Esta técnica pode ser usada em multiprogramação por partições fixas e por partições variáveis [Lima, 2008].

5.4. Políticas de Uso da Memória Principal

A seguir, são apresentadas políticas encontradas na literatura, implementadas pelo gerenciador de memória do sistema operacional e usadas para percorrer e encontrar partições livres na memória principal de um computador [Tanenbaum, 2003b; Stallings, 2005; Magalhães *et al*., 1992; Deitel *et al*., 2005]:

- *First-Fit*. Esta política de gerência de memória, também conhecida como "a primeira partição livre que couber", é a mais simples. O gerenciador de memória percorre a lista de partições livres em busca de uma partição suficientemente grande para o processo recém criado (ou aquele transferido da memória secundária para a memória principal) ser armazenado. Ao encontrar uma partição livre que satisfaça tal condição, a busca é concluída. O tempo de execução desta política é considerado pequeno, pois a busca por partição livre encerra o mais cedo possível;
- *Next-Fit*. Esta política de gerência de memória é muito semelhante a *First-Fit*, diferindo na partição considerada para o início da busca, pois ela continua a busca por uma partição livre começando da posição que parou na

última busca. Esta política segue a mesma idéia da *Circular-Fit* citada por Oliveira *et al*. (2001);

- *Best-Fit*. Esta política de gerência de memória consiste em percorre a lista de partições livres em busca daquela que tem o tamanho mais próximo do tamanho do processo a ser alocado na memória. Esta política possui um algoritmo lento e gera partições pequenas na memória principal que dificilmente serão alocadas. Por outro lado, para processos grandes, este algoritmo aumenta as chances de encontrar uma partição livre de tamanho adequado, minimizando o uso de partições livres grandes por processos pequenos;
- *Worst-Fit*. Esta política de gerência de memória, também conhecida como "a pior partição que couber", é aparentemente esquisita, porém com apelo intuitivo. Ela procura pela maior partição livre para armazenar o processo requerente por memória. A atração intuitiva é, após escolher uma partição livre grande, o seu restante será suficientemente grande para alocar um novo processo.

5.5. Considerações Finais

Este capítulo apresentou algumas definições relacionadas a gerência de memória de um sistema operacional, destacando as políticas utilizadas pelo gerenciador de memória para percorrer a lista de partições presentes na memória principal após alocações e liberações de espaço requisitados por processos.

No próximo capítulo, são abordados alguns ambientes computacionais educacionais para o ensino de políticas de gerenciamento de processos e de gerenciamento de memória. Tais ambientes são avaliados quanto a sua funcionalidade, os seus objetivos e a sua usabilidade.

6. AMBIENTES COMPUTACIONAIS EDUCACIONAIS PARA O ENSINO DE GERÊNCIA DE PROCESSOS E DE GERÊNCIA DE MEMÓRIA

6.1. Considerações Iniciais

Este capítulo apresenta breve avaliação da funcionalidade e das interfaces de alguns ambientes educacionais computacionais que abordam o ensino de gerência de processos e de gerência de memória em sistemas operacionais.

A avaliação de interfaces é um item muito importante no processo de desenvolvimento de software interativo. Basicamente, existem cinco fatores básicos relativos a usabilidade de um software [Zambalde; Alves, 2003]:

- Facilidade de aprendizado. As tarefas do software devem ser de fácil aprendizado;
- Eficiência de uso. O software deve permitir alta produtividade após o seu aprendizado;
- Retenção com o tempo. A forma de usar o software deve ser relembrada facilmente;
- Recuperação de erros. O software deve estar preparado para se recuperar de erros cometidos pelos usuários;
- Satisfação. O software deve corresponder as curiosidades do usuário de forma satisfatória, ou seja, o usuário deve gostar de usá-lo.

A seção 6.2 apresenta o software educacional SOSim, desenvolvido por Luiz Paulo Maia. A seção 6.3 apresenta o software educacional MOSS (*Modern Operating Systems Simulators*), desenvolvido por Ray Ontko e Alexander Reeder orientados por Andrew S. Tanenbaum. A seção 6.4 apresenta o software

educacional wxEscalProc, desenvolvido por um grupo de estudantes do curso de Ciência da Computação da Universidade Federal de Lavras.

6.2. SOSim

Este software auxilia no aprendizado de conceitos e mecanismos de um sistema operacional multiprogramável e/ou multitarefa de forma simples e animada. Ele foi desenvolvido pelo Prof. Luiz Paulo Maia, como parte de sua dissertação de mestrado no Núcleo de Computação Eletrônica da Universidade Federal do Rio de Janeiro (NCE/UFRJ). O SOSim[8] foi desenvolvido utilizando a Ferramenta Borland Delphi 7.0. Na sua configuração básica, ele é composto por quatro janelas: janela principal, janela de gerência de processo, janela de gerência de memória e janela de gerência de processador.

Utilizando o SOSim, um usuário pode simular o funcionamento de um sistema operacional quanto à gerência de processos e à gerência de memória de um computador. O SOSim permite ao usuário configurar parâmetros do sistema relacionados às políticas de escalonamento de processos e à busca de páginas na memória. Acionando os botões e as barras de rolagem, um usuário pode criar e manipular processos na memória do simulador e definir configurações como *clock* da CPU. É possível visualizar graficamente o caminho que os processos inseridos fazem no computador enquanto manipulados pelo sistema operacional, bem como a execução das políticas de escalonamento de processos. A abordagem de animação dos algoritmos possui a vantagem imediata de contribui para a observação dinâmica do funcionamento dos tipos de dados e algoritmos envolvidos, proporcionando maior nível de abstração.

Quanto à sua interface, na janela principal, o SOSim permite acessar *menus* que possibilitam configurar os parâmetros de gerência de processos e de

[8] http://www.training.com.br/sosim

gerência de memória, abrir outras janelas (*log*, dados estatísticos e arquivo de paginação) e exibir o tempo de execução do software, o número de processos e a porcentagem de memória livre. Usando a janela gerência de processos, o SOSim permite criar, suspender, prosseguir a execução e finalizar processos, alterar a prioridade dos processos e exibir o contexto do processo selecionado a partir de botões de ação. A janela gerência de memória exibe apenas um diagrama contendo "bolinhas coloridas" que representam os processos criados pelo usuário. Usando a janela gerência de processo, figuras são exibidas, representando um processador, dispositivos de entrada/saída e uma fila de prontos. Nesta janela, é possível alterar de forma simples o tempo de espera dos processos por operações de entrada/saída, a fatia de tempo de uso do processador e o *clock* do processador.

Na avaliação realizada, foi notado que o SOSim apresenta alguns erros de execução exibidos na forma de janelas de alerta ao usuário. O SOSim é gratuito, porém possui somente versões para o sistema operacional Windows (software proprietário). Como pontos positivos, o SOSim: i) oferece um conjunto de funções configuráveis que revelam ser intuitivos e de fácil manipulação; ii) apresenta constante retorno ao usuário; iii) utiliza a linguagem do usuário; e iv) é fácil de usar.

6.3. MOSS (*Modern Operating Systems Simulators*)

Esta é uma coleção de programas de simulação desenvolvidos em Java. O MOSS[9] foi criado por Ray Ontko e Alexander Reeder orientados por Andrew S. Tanenbaum apoiados pela Prentice-Hall. Esta coleção engloba quatro diferentes aplicativos: i) Simulador de escalonamento; ii) Simulador de *Deadlocking*; iii) Simulador de gerência de memória; e iv) simulador de sistema de arquivo.

[9] http://www.ontko.com/moss

O simulador de escalonamento ilustra o comportamento de algoritmos de escalonamento de processos a partir de um conjunto de processos gerados. O usuário pode especificar o número de processos, a média e a divergência de tempo de processador e tempo de bloqueio de dispositivos de entrada/saída para cada processo e a duração da simulação. Ao término da simulação, é apresentado um resumo estatístico.

Este simulador recebe parâmetros de configuração do arquivo de configuração scheduling.conf. A partir dele, o simulador cria um conjunto de processos com tempos de processamento em milissegundos. Cada processo é executado por um tempo gerado aleatoriamente, com valor forçado a ficar entre a média e a divergência de tempo daquela média. Depois de ler o arquivo de configuração, o simulador executa cada processo, fazendo operações de entrada/saída, caso necessário, até cada processo completar o tempo reservado a ele ou até o tempo limite determinado para a execução do simulador seja atingido. Enquanto a simulação é executada, um arquivo de *log* é gerado para mostrar os passos efetuados pelo algoritmo. Depois de encerrada a execução do simulador, um relatório é gerado para mostrar estatísticas de cada processo e da simulação como um todo.

Quanto à sua interface, os aplicativos da coleção possuem interface textual (não gráfica), indo de encontro à tendência de softwares desenvolvidos atualmente. Desta forma, a interação com o usuário não é apropriada. Para alterar parâmetros de execução (duração da execução do simulador, número de processos, etc.) deste simulador, é preciso alterar e passar como parâmetro de execução um arquivo texto. Assim, o simulador de escalonamento não apresenta interação com o usuário. Para visualizar os resultados obtidos, é preciso abrir o arquivo texto criado pelo simulador de formato semelhante a um *log* de eventos. Dessa forma, a satisfação do usuário ao usar este simulador fica prejudicada.

O simulador de gerência de memória segue a mesma linha do simulador de escalonamento de processos. Este simulador ilustra o comportamento do sistema operacional quando acontece falha de página no sistema de paginação de memória virtual. O programa obtém o estado inicial de uma tabela de páginas e uma sequência de instruções de memória virtual. Por fim, ele escreve um *log* que indica o efeito de cada instrução lida anteriormente. Este simulador inclui interface gráfica com o usuário que pode observar o algoritmo de substituição de página funcionando de forma intuitiva.

A execução do simulador de gerência de memória não é trivial, pois é preciso digitar o comando junto com o nome do arquivo passado como parâmetro. Apesar de possuir interface gráfica, este simulador apresenta interface de difícil utilização e baixa interação com o usuário. Assim como acontece com o simulador de escalonamento de processos, este simulador não é de fácil retenção de uso (baixa apreensibilidade), devido à forma de executá-lo e de verificar o resultado após sua execução.

Como ponto positivo, a coleção de simuladores é multiplataforma (foi desenvolvida usando tecnologia Java), sendo possível executar em qualquer sistema operacional que possua a máquina virtual Java instalada.

6.4. wxEscalProc

Este é um software educativo que simula a aplicação das políticas de escalonamento de processos FCFS (*First Come First Served*), *Round-Robin*, SJF (*Shortest Job First*) e RJF (*Remaining Job First* – citado no presente trabalho como SRTF – *Shortest Remaining Time First*). O wxEscalProc foi desenvolvido por um grupo de estudantes do curso de Ciência da Computação da Universidade Federal de Lavras.

Com wxEscalProc, o usuário pode inserir um conjunto de processos e, em seguida, submetê-los a uma das políticas implementadas. De acordo com o(s) parâmetro(s) necessário(s) da política escolhida pelo o usuário, ele pode inserir valores para tal(is) parâmetro(s). Após a execução do algoritmo, é exibido o resultado do algoritmo em relação ao conjunto de processos submetidos.

O wxEscalProc possui as seguintes opções:

- Configurar Processos. O usuário criar um grupo de processos para serem escalonados. Para a criação de um processo, o usuário deve inserir o tempo de criação e o tempo de processamento do processo, ou seja, quanto tempo de processador o processo necessita para terminar. Esta opção abre uma janela para o usuário fazer tal operação;
- Escolher Política. O usuário escolhe qual política de escalonamento de processos será aplicada sobre o grupo de processos inseridos e define se os passos da simulação serão controlados pelo usuário ou se será automatizada. Esta opção é exibida em uma nova janela (Política de Escalonamento) criada após concluir as inserções de processos;
- Tic. O usuário controla os passos da simulação, caso ele tenha escolhido a opção ‘Passo a Passo’ no painel ‘Escolha o modo de Simulação’ da janela ‘Política de Escalonamento’.

Para a avaliação do wxEscalProc, foi utilizada a versão para o sistema operacional Windows disponível no site http://www.ic.unicamp.br/~rocha/grad/index.html. Nesta avaliação, o gráfico ilustrativo que mostra o resultado final da execução após a submissão da política de escalonamento escolhida sob os processos inseridos não foi exibido corretamente nos testes realizados. Com isso, a versão avaliada não proporciona satisfação ao usuário.

Quanto à interface, o wxEscalProc apresenta facilidade de aprendizado devido, principalmente, à sua simples interface e poucas opções de configurações. Além disso, o software permite o uso de teclas de atalho que aumentam a sua usabilidade. O arquivo disponível no site possui, além dos arquivos binários, os arquivos-fonte do software, *scripts* para compilação em ambientes Linux e Windows e um artigo. Foi tentado compilá-lo novamente usando o *script* de compilação disponibilizado, porém o mesmo não funcionou corretamente.

Como pontos negativos, notou-se que o wxEscalProc não está preparado para recuperação de erros e de acordo com os testes realizados, ele pode apresentar resultados inesperados.

6.5. Considerações Finais

Este capítulo mostrou três ambientes educativos que abordam, entre suas funções, a apresentação do funcionamento de gerência de memória e de gerência de processos em sistemas operacionais. Na pesquisa realizada, foram encontrados citações de outros ambientes que abordam o mesmo assunto, entretanto seus arquivos binários não foram encontrados, apenas artigos: Software Hipermídia para Auxílio no Ensino de Sistemas Operacionais em Guedes e Guedes (2006) e S^2O em Carvalho *et al*. (2006).

Assim, foi constatado que a preocupação com a usabilidade da interface é uma constante no processo de desenvolvimento de ambientes educativos. Os ambientes avaliados apresentam os algoritmos que eles tratam, porém não mostram de forma mais atrativa e intuitiva os passos de sua execução. Para alcançar sucesso, quanto ao objetivo do presente trabalho (desenvolver um software educativo para o ensino das políticas de gerência de processos e de gerência de memória), são considerados os problemas encontrados nos ambientes avaliados, com o intuito de tornar o TBC-SO/WEB mais atrativo.

7. TBC-SO/WEB: UM SIMULADOR DIDÁTICO PARA O ENSINO DE GERÊNCIA DE PROCESSOS E DE GERÊNCIA DE MEMÓRIA VIA WEB

7.1. Considerações Iniciais

Este capítulo apresenta o software educativo **TBC-SO/WEB (Treinamento Baseado em Computador para Sistemas Operacionais via Web)**. O **TBC-SO/WEB** segue a linha de desenvolvimento de softwares educativos propostos, na qual existem **TBC-AED/WEB** e **TBC-GRAFOS/WEB** desenvolvidos por Rodrigo Pereira dos Santos em projeto de iniciação científica [Santos; Costa, 2007; Santos; Costa, 2005a; Santos; Costa, 2005b, Santos; Costa, 2008] e **TBC-GAAL/WEB** desenvolvido por Igor Ribeiro Lima [Lima *et al*., 2006]. Para mais informações, estes softwares podem ser acessados em http://www.dcc.ufla.br/~heitor/Projetos.html.

A seção 7.2 faz breve análise do desenvolvimento do **TBC-SO/WEB**. A seção 7.3 faz breve descrição no nível de modelagem do **TBC-SO/WEB**. A seção 7.4 apresenta a organização e a estrutura do **TBC-SO/WEB**, incluindo alguns recursos usados para sua construção. A seção 7.5 descreve os temas abordados e o uso do **TBC-SO/WEB**. A seção 7.6 apresenta uma comparação entre ambientes educacionais analisados no capítulo anterior e o **TBC-SO/WEB**. A seção 7.7 mostra o relativo desempenho educacional com os alunos da disciplina Sistemas Operacionais do Curso de Bacharelado de Ciência da Computação da Universidade Federal de Lavras no primeiro semestre letivo de 2009.

7.2. Análise do Desenvolvimento

O **TBC-SO/WEB**, um software educativo com interface gráfica para Web, foi desenvolvido para servir como ferramenta de ensino das políticas de gerência de processos e de gerência de memória em sistemas operacionais. O **TBC-SO/WEB** aborda as políticas mais citadas pelos principais autores na literatura, apresentando processo gráfico passo a passo da execução de seus algoritmos, adicionado a conteúdo teórico sintético.

Dessa forma, a visualização e o entendimento de conceitos apresentados pelo professor são facilitados. Além disso, pode-se ganhar tempo nas aulas e, assim, alcançar maior interação entre professores e alunos, no sentido de aumentar o espaço para questionamentos.

Dentre as vantagens da abordagem construtiva, estão mecanismos para facilitar o processo de abstração. O uso de animação gráfica reflete interação com o estudante e facilidades de detecção visual de erros. Com isso, incentiva-se o processo de compreensão e autocorreção por parte do estudante [Garcia *et al.*, 2008].

Assim, verifica-se nas próximas seções que o **TBC-SO/WEB** é amplamente didático e esse fato embasa o seu uso no ensino de disciplinas que apresentam em suas ementas, tópicos relacionados aos tratados por ele. Além disso, o **TBC-SO/WEB** contribui para aprimorar a formação de recursos humanos para a área de Computação e Informática.

7.3. Modelagem

Nesta seção, a modelagem do **TBC-SO/WEB** é apresentada, usando a UML (*Unified Modeling Language*), focando exclusivamente o Diagrama de Casos de Usos e o Diagrama de Navegação. Entretanto, deve-se considerar que

os diagramas tratados pelo presente trabalho não objetivam elucidar o **TBC-SO/WEB** sob todos os pontos, mas detalhar os seus pontos principais, buscando uma visão geral do mesmo, bem como não intenciona apontar que os demais diagramas de UML são menos importantes. A UML é utilizada, pois é uma linguagem de modelagem padrão para descrever software orientação a objeto [UML, 2009].

7.3.1. Diagrama de Casos de Uso

A Figura 7-1 ilustra o Diagrama de Casos de Uso com o Ator Usuário e os casos de uso do **TBC-SO/WEB**: i) Acessar Applet, ii) Inserir Processos, iii) Digitar Tamanho do Processo, iv) Criar Processos, vii) Digitar Tempo de Burst, viii) Digitar Prioridade, ix) Visualizar Animação, x) Inserir Quantum, xi) Exibir Relatório e xii) Reiniciar.

7.3.2. Diagramas de Navegação

A Figura 7-2 e Figura 7-3 ilustram, respectivamente, os Diagramas de Navegação do **TBC-SO/WEB** para os tópicos Gerência de Memória e Gerência de Processos. Estes diagramas visam refletir a dinâmica do software, com relação a mudanças no espaço navegacional.

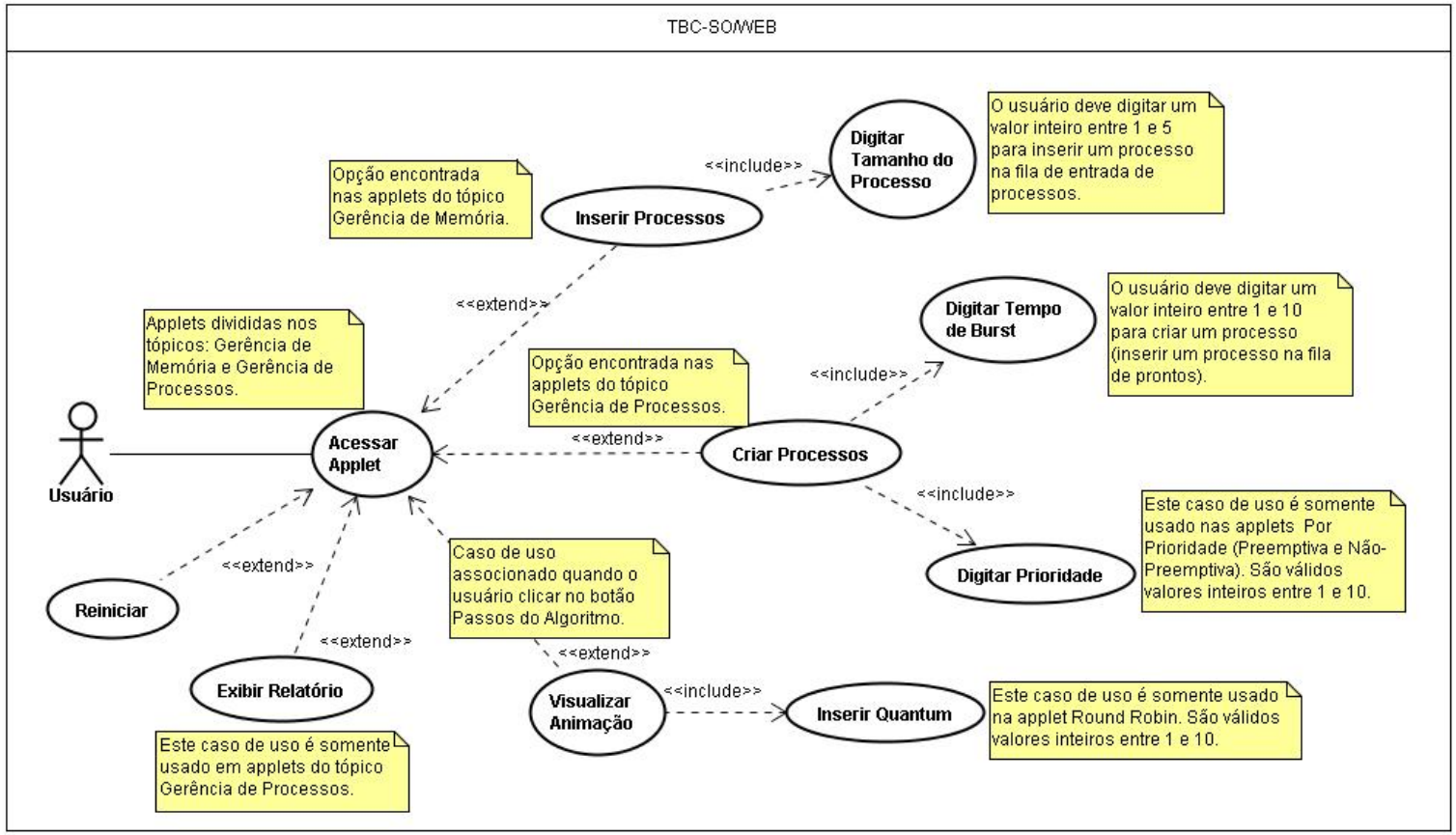

Figura 7-1 – Diagrama de Casos de Uso do TBC-SO/WEB

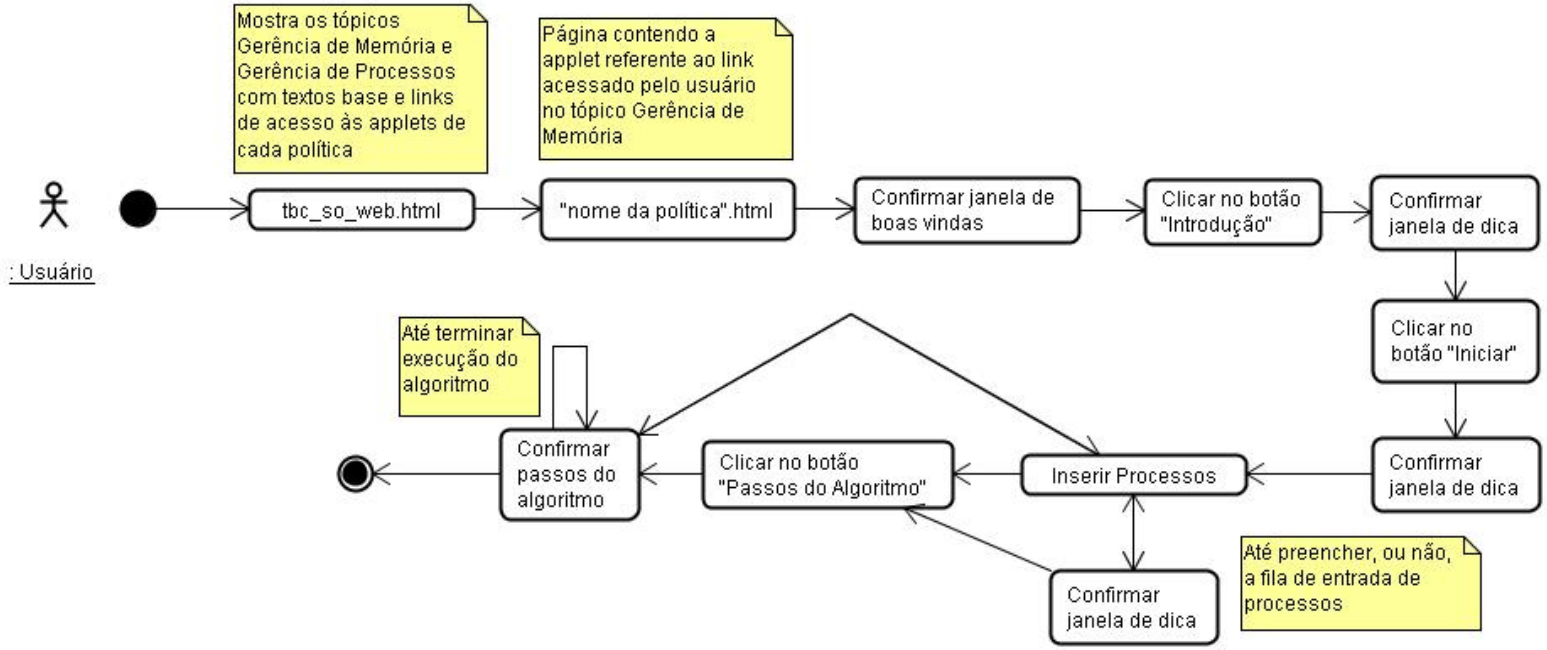

Figura 7-2 – Diagrama de Navegação – Tópico Gerência de Memória

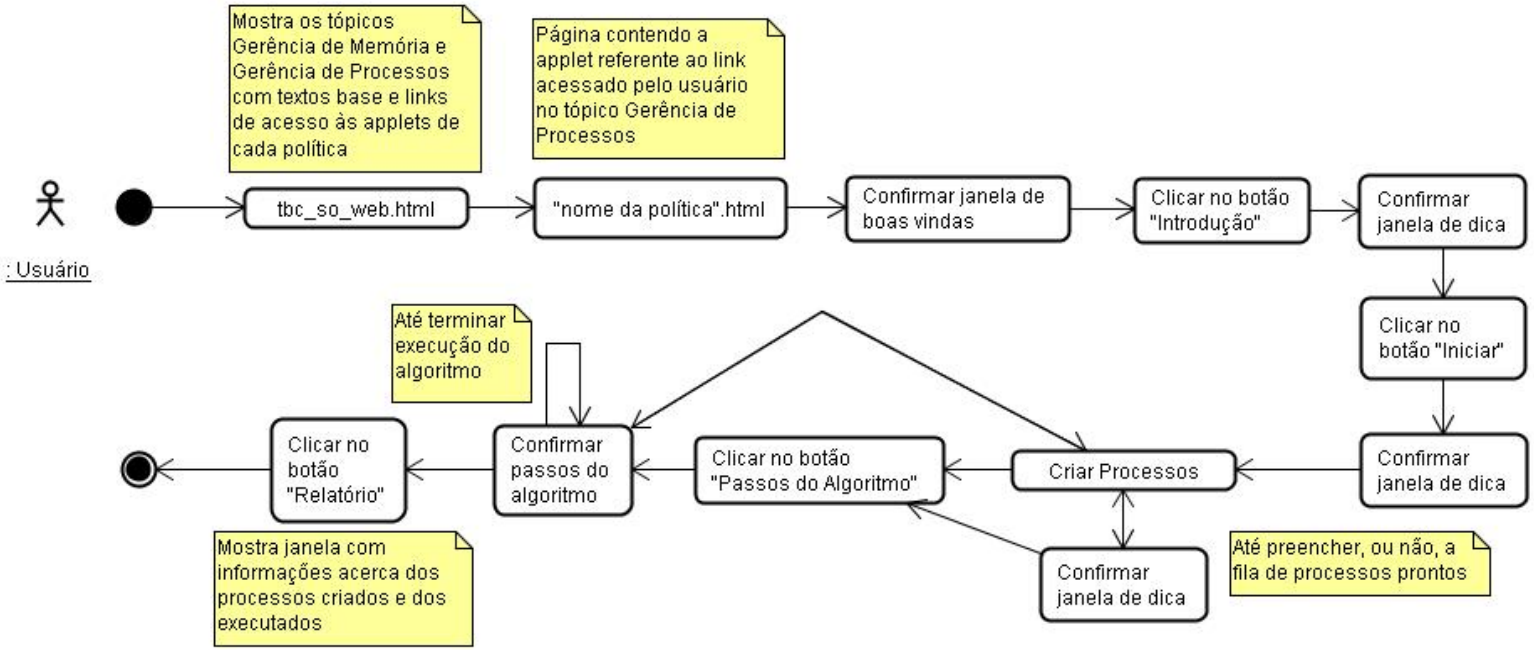

Figura 7-3 – Diagrama de Navegação – Tópico Gerência de Processos

7.4. Organização e Estrutura

Quanto à organização, o **TBC-SO/WEB** está disponível nos *sites* http://www.dcc.ufla.br/~heitor/Projetos e http://alunos.dcc.ufla.br/~fabricio (Figura 7-4), na forma de *links* divididos em dois tópicos: gerência de processos e gerência de memória (Figura 7-5). Cada tópico contém *links* de acesso aos programas de cada política e ao código fonte correspondente. Quanto à estrutura, um *applet* Java foi desenvolvido para cada política de gerenciamento.

Figura 7-4 – Tela Inicial do TBC-SO/WEB

A Figura 7-6 mostra uma tela da política de gerência de memória *Best-Fit*, apresentando (i) breve introdução sobre o tópico tratado (parte superior), (ii) o

algoritmo em *Portugol*[10] (à esquerda), (iii) o painel de animação (memória principal), onde os passos realizados para atingir a solução acontecem graficamente (à direita), (iv) uma legenda (à direita abaixo do painel de animação), (v) um painel para inserção de processos (à direita abaixo da legenda), (vi) um painel para representar processos na fila de processos (à direita abaixo do painel de inserção de processos) e (vii) um conjunto de botões (parte inferior). Esta organização é a mesma para as políticas de gerência de memória abordadas pelo **TBC-SO/WEB**.

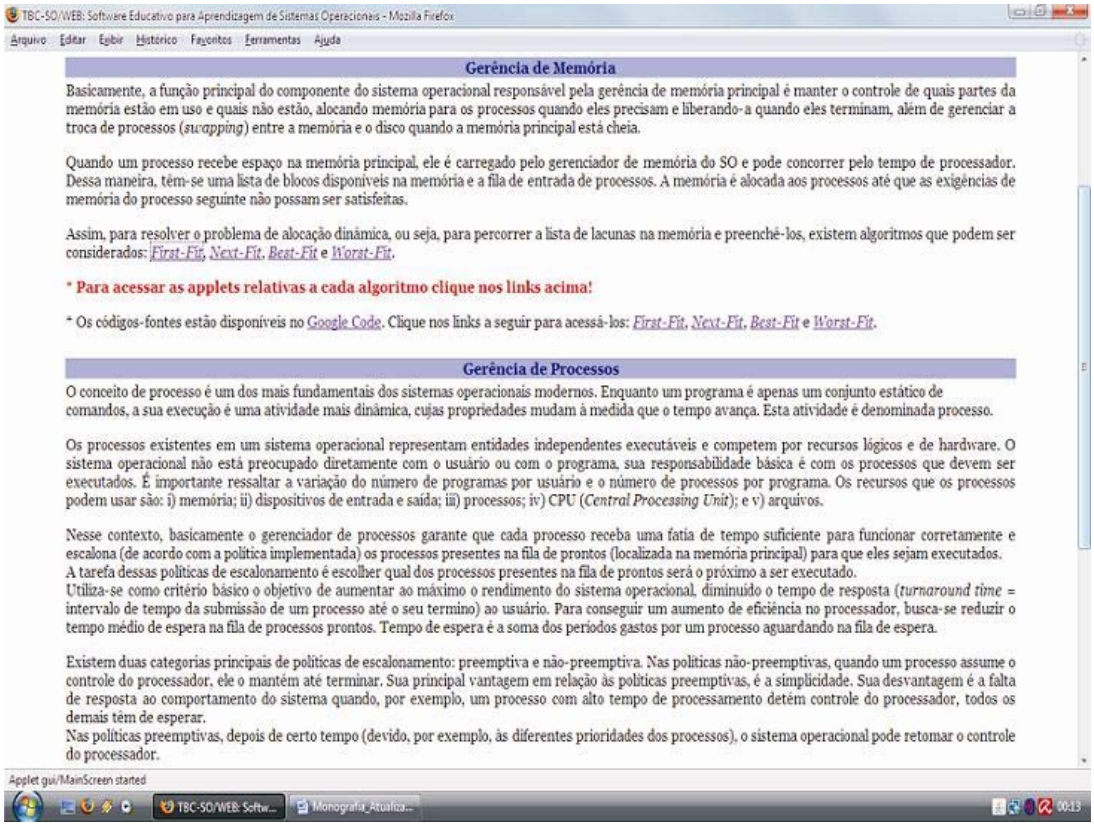

Figura 7-5 – Tela dos Tópicos do TBC-SO/WEB

[10] Portugol é código elaborado com regras bem definidas que descrevem uma sequência de passos para a solução de um dado problema (Campos; Ascencio, 2003).

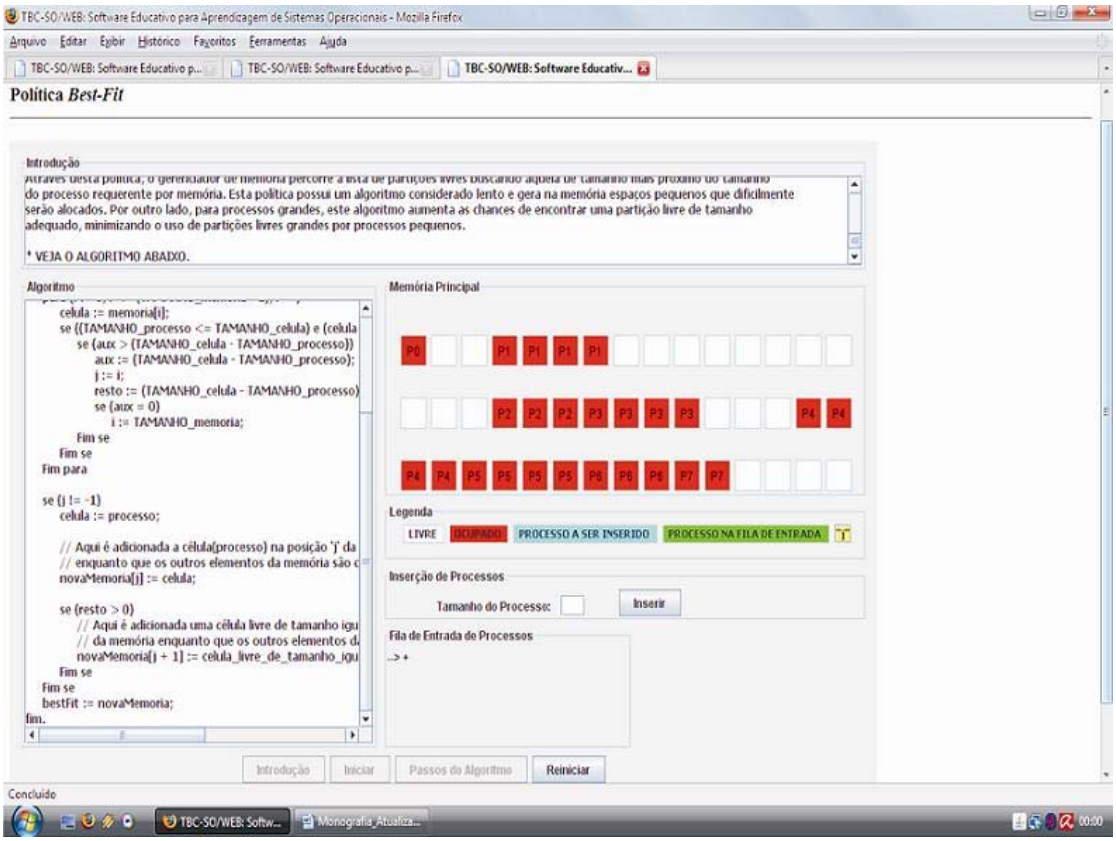

Figura 7-6 – Tela Inicial da Política de Gerência de Memória *Best-Fit*

No tópico gerência de processos, as políticas foram divididas em duas categorias: preemptivas e não-preemptivas. A Figura 7-7 mostra a tela relativa à política de gerência de processos Por Prioridade Não-Preemptiva, apresentando (i) breve introdução sobre o tópico tratado (parte superior), (ii) o algoritmo em *Portugol* (à esquerda), (iii) o painel de animação (à direita) contendo representação de um processador (à esquerda, no painel de animação) e o painel de processos prontos (à direita, no painel de animação), (iv) uma legenda (à direita abaixo do painel de animação), (v) um painel para inserção de processos (à direita abaixo da legenda) e (vi) um conjunto de botões (parte inferior). Esta organização é a mesma para as políticas de gerência de processos não-

preemptivas. Para as políticas de gerência de processos preemptivas, há (somando aos outros componentes do painel de animação) (vii) o painel de processos em espera (à direita abaixo do painel de processos prontos). Neste painel, são armazenados os processos que passam para o estado “esperando” (Figura 7-8).

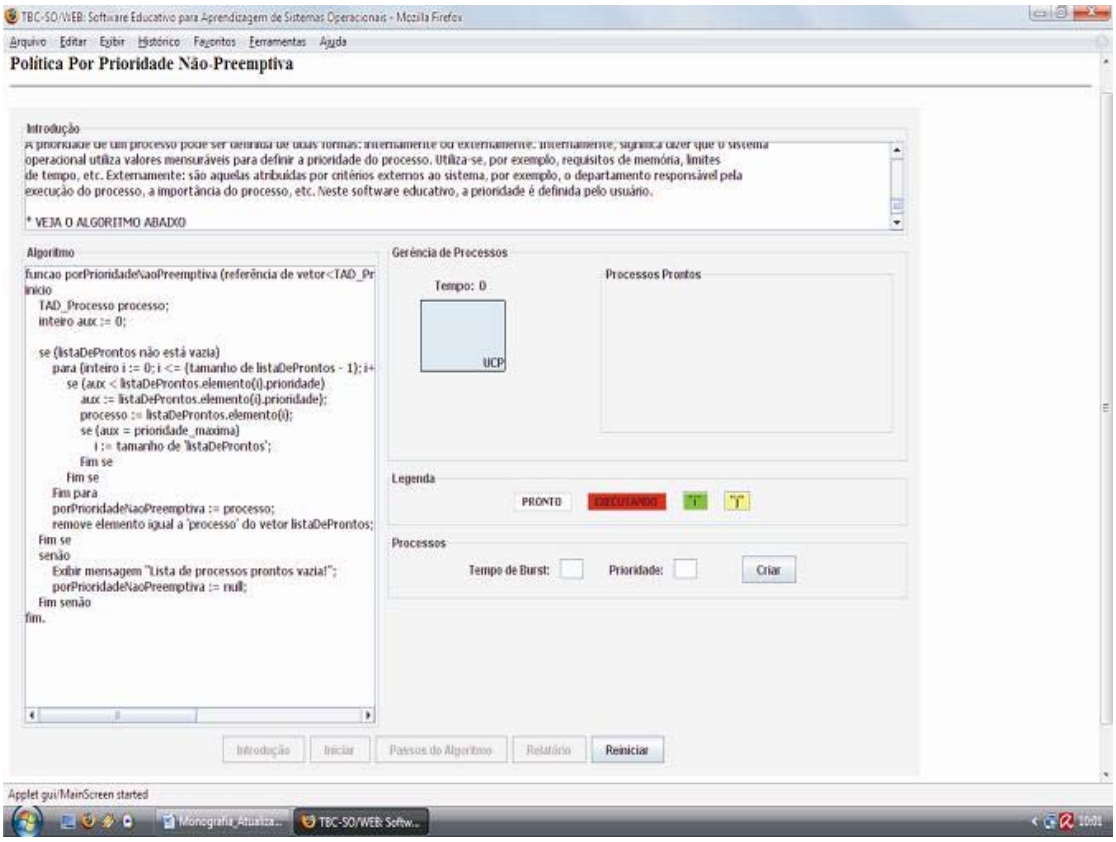

Figura 7-7 – Tela Inicial da Política de Gerência de Processos Por Prioridade Não-Preemptiva

Para efeito de passo a passo na execução do algoritmo, é usado um botão “Clique aqui para próximo passo” que aparece somente após o usuário clicar no botão “Passos do Algoritmo” (Figura 7-9). Durante a execução, são mostradas mensagens para o usuário contendo informações do uso do **TBC-**

SO/WEB. Para isso, foram usados métodos da classe "*JOptionPane*[11]". A Figura 7-10 mostra a mensagem inicial contendo primeiras informações para o uso do **TBC-SO/WEB**.

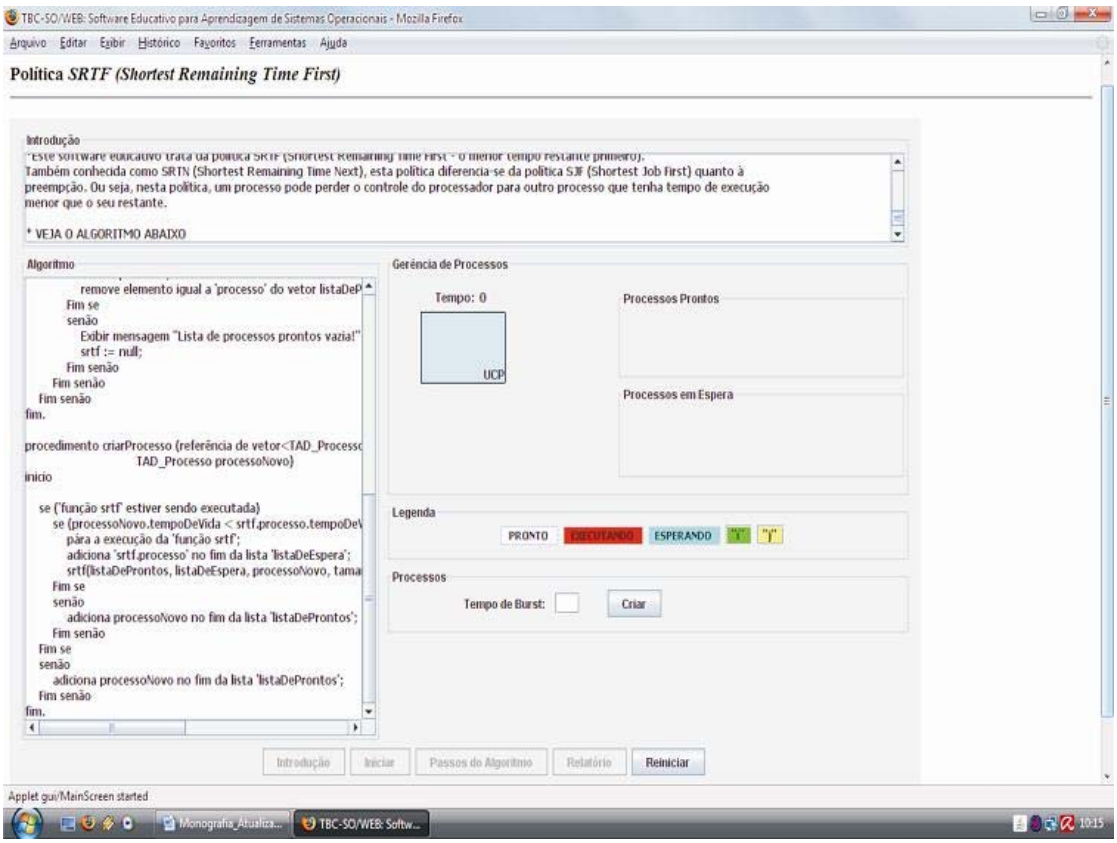

Figura 7-8 – Tela Inicial da Política de Gerência de Processos SRTF

Uma propriedade significativa do **TBC-SO/WEB** é ser auto-explicativo, basta deslizar o cursor do *mouse* sobre as partes da tela para visualizar breves mensagens sobre elas (Figura 7-11). O usuário é constantemente alertado com mensagens contendo dicas de uso do **TBC-SO/WEB** (Figura 7-12). Além disso,

[11]JOptionPane é uma classe do pacote javax.swing usada para criar sub-janelas de mensagem ao usuário (Sun Microsystems, 2009).

durante a execução do algoritmo, apenas os botões úteis ao usuário são apresentados. Este esquema evita falhas e deixa o usuário mais à vontade e sem preocupações quanto a quaisquer detalhes.

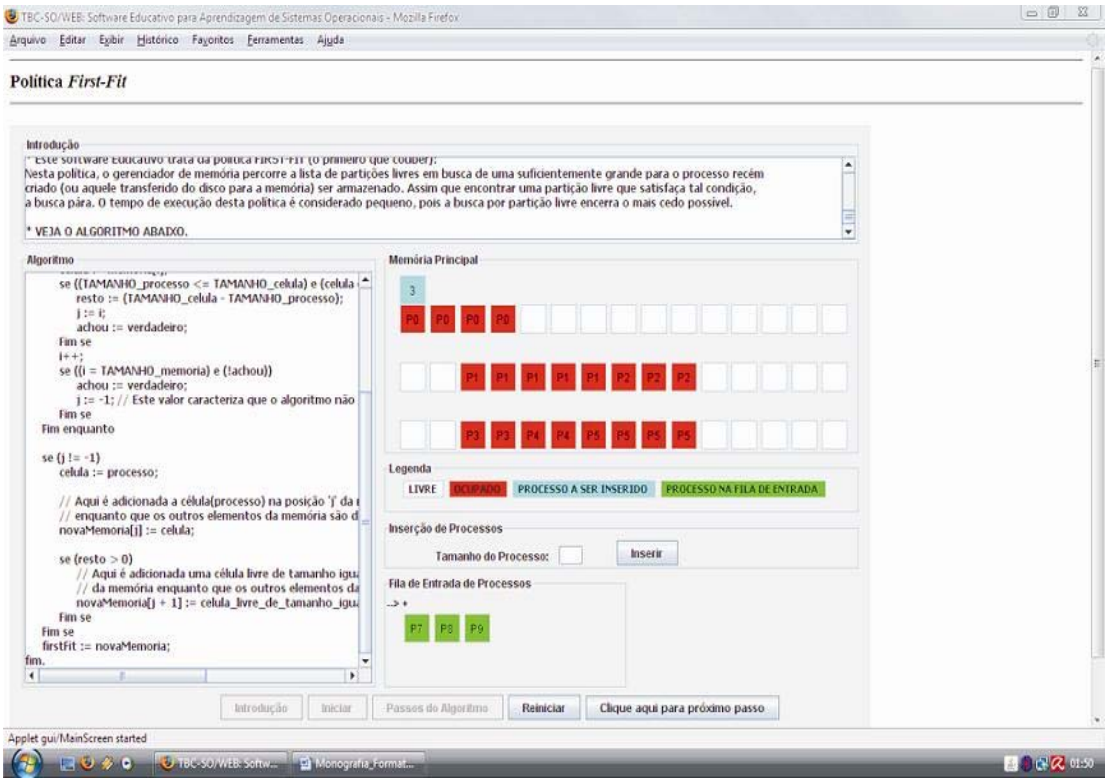

Figura 7-9 – Tela de Execução da Política de Gerência de Memória *First-Fit*

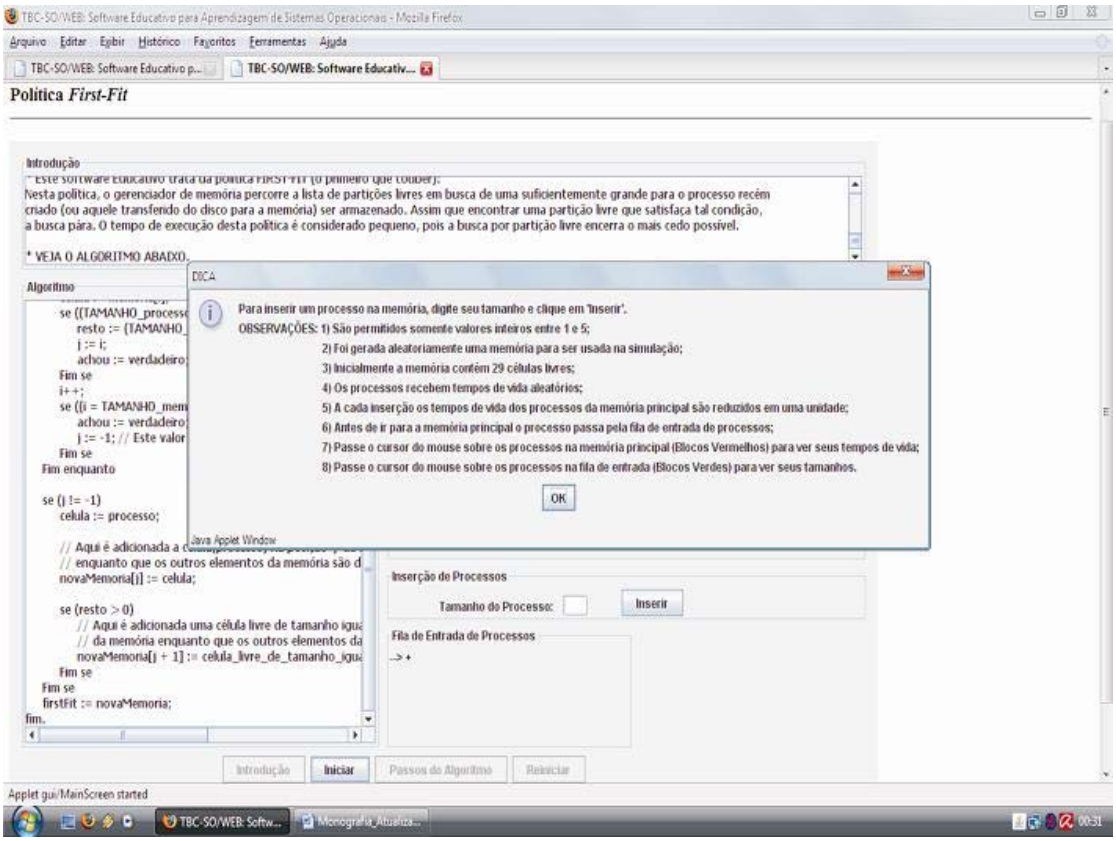

Figura 7-10 – Tela da Política de Gerência de Memória *First-Fit*

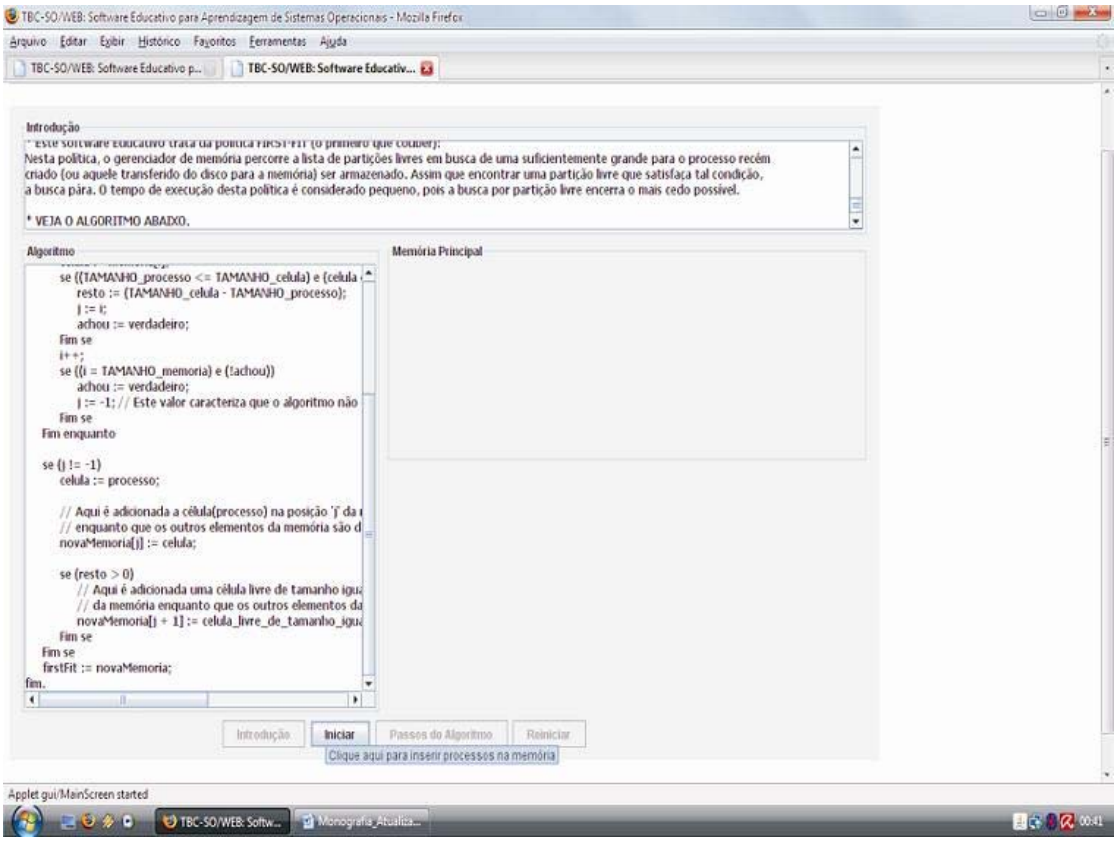

Figura 7-11 – Tela com Breve Mensagem Explicativa Associada ao Botão Iniciar

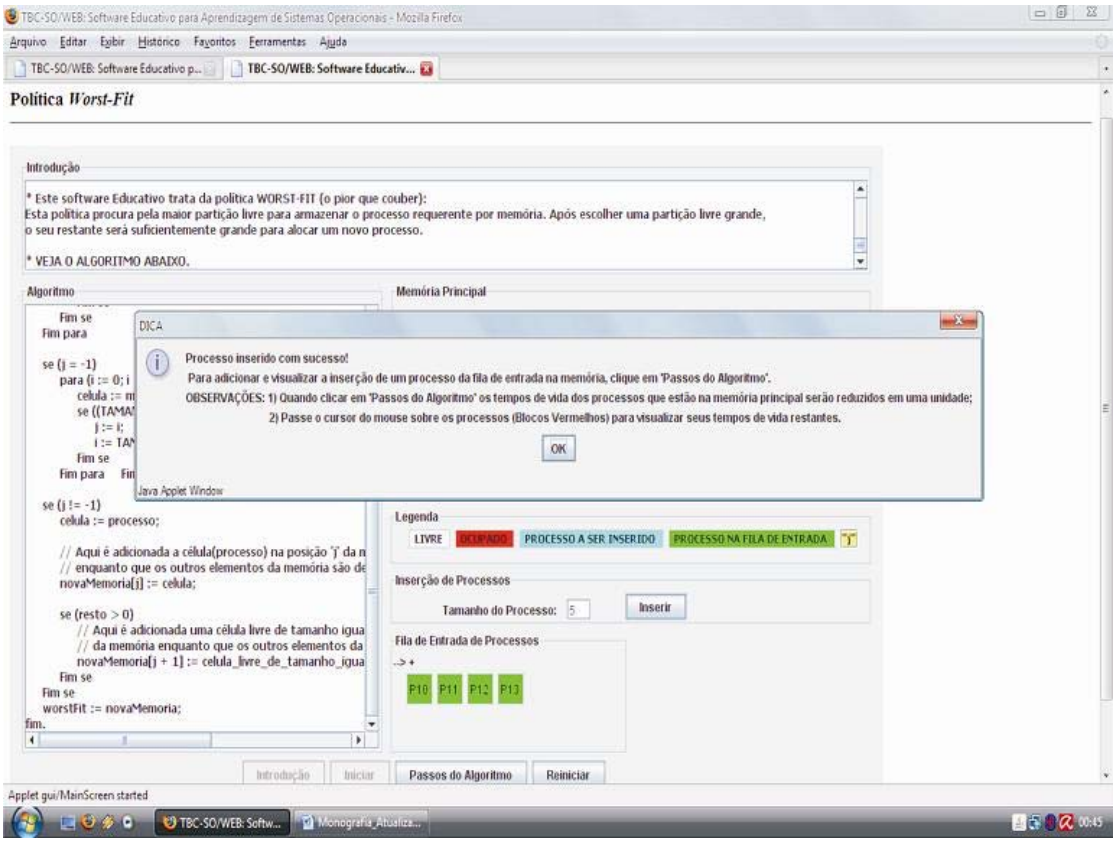

Figura 7-12 – Tela com Mensagem de Sucesso após Inserir um Processo

Assim, o **TBC-SO/WEB** é uma excelente ferramenta para o ensino de gerência de processos e de gerência de memória em sistemas operacionais, pois são estimulados no estudante os processos de compreensão e de autocorreção dos algoritmos tratados por ele. Quando melhor ensinado um assunto, o rendimento e o desempenho dos alunos melhoram, proporcionando melhores currículos e melhores profissionais para o mercado.

7.5. Temas Abordados e Utilização

O **TBC-SO/WEB** aborda temas envolvidos aos subtópicos escalonamento de processos e alocação de memória presentes, respectivamente, nos tópicos gerência de processos e gerência de memória de sistemas operacionais. No **TBC-SO/WEB**, foram implementados os algoritmos de alocação de memória: i) *First-Fit*; ii) *Next-Fit*; iii) *Best-Fit* e iv) *Worst-Fit* e os algoritmos de escalonamento de processos: i) FIFO; ii) SJF; iii) HRRN; iv) Por Prioridade Não-Preemptivo; v) SRTF; vi) Por Prioridade Preemptivo e vii) *Round-Robin*.

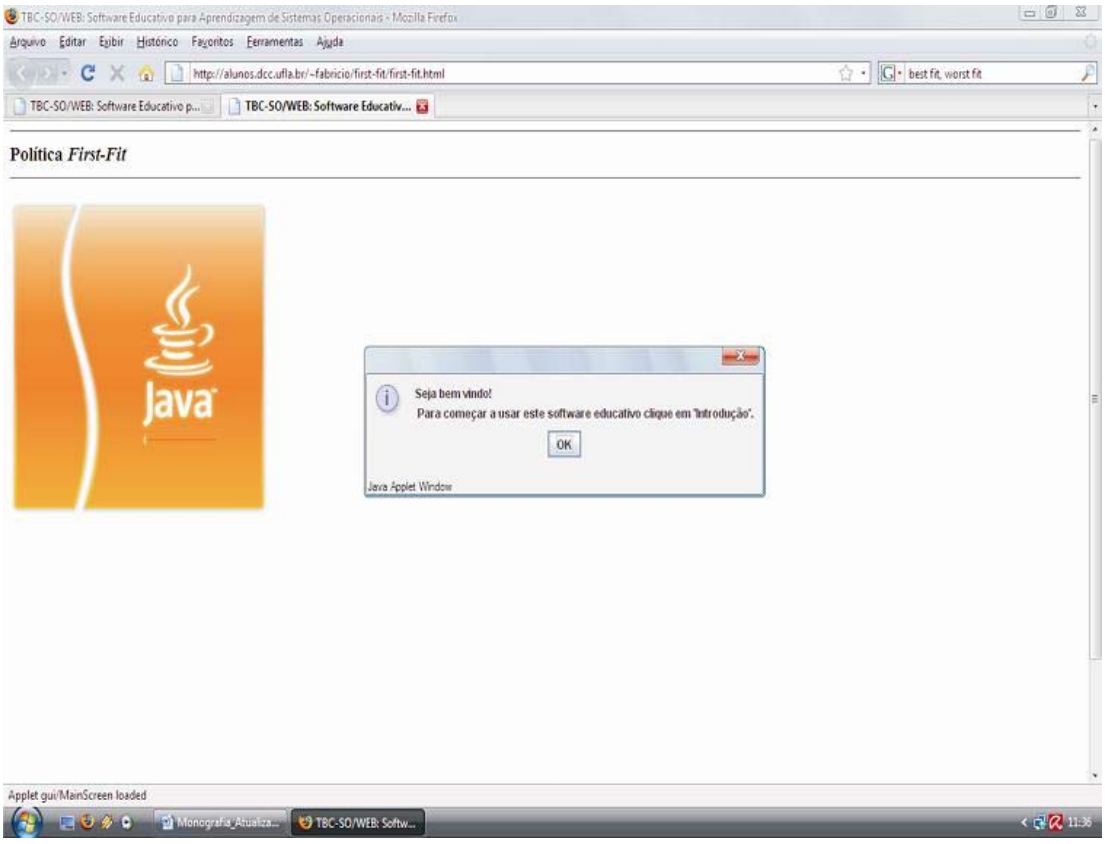

Figura 7-13 – Tela com Mensagem de Abertura

7.5.1. Gerência de Memória

Neste tópico, a forma de uso é semelhante para todos seus programas. Assim, analisando o funcionamento tem-se:

- Ao selecionar um dos itens *"First-Fit"*, *"Next-Fit"*, *"Best-Fit"* ou *"Worst-fit"* (ver o site do **TBC-SO/WEB** – Figura 7-4), uma nova janela ou aba é aberta no navegador contendo o ambiente gráfico correspondente. Em seguida, é exibida uma mensagem de boas vindas com instrução para próximo passo (Figura 7-13);

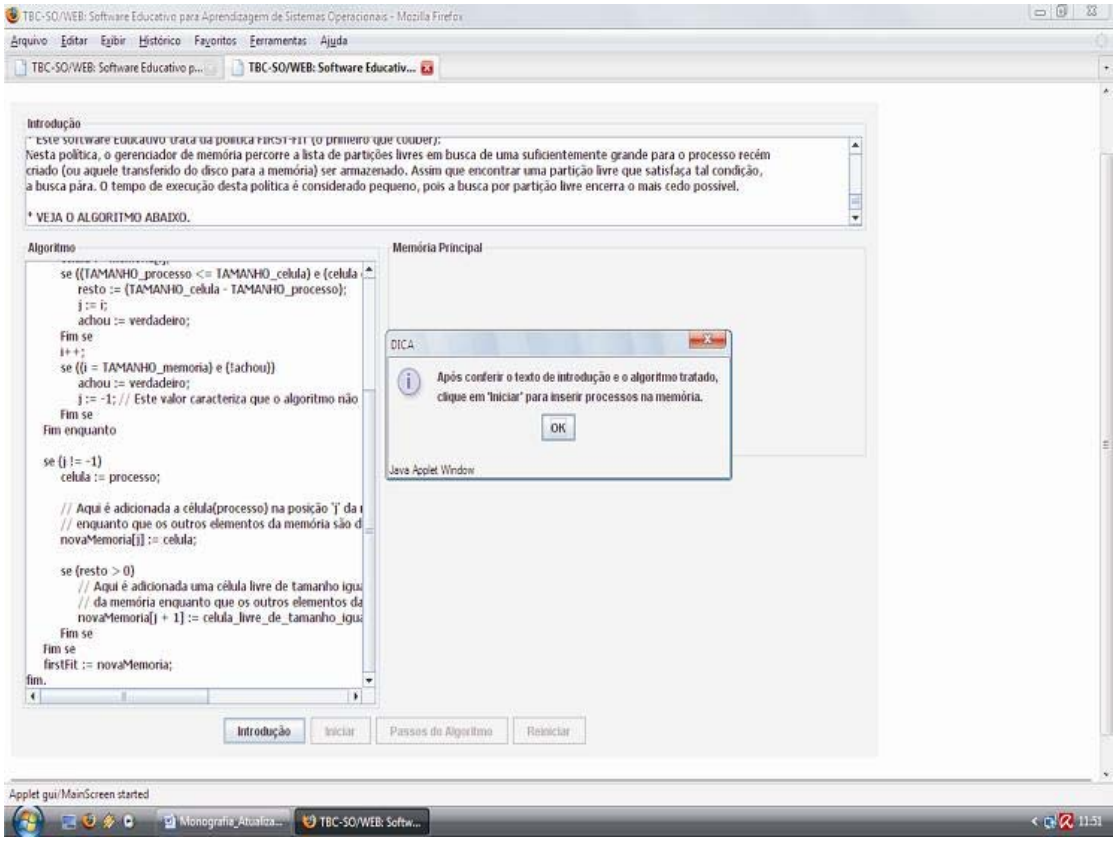

Figura 7-14 – Tela com Mensagem após Clicar no Botão Introdução

- Por apenas os botões necessários estarem habilitados, o usuário deve clicar no botão “Introdução” para que uma mensagem de instruções de uso apareça. Junto a esta mensagem, uma breve introdução (parte superior do ambiente) da política tratada e o seu algoritmo em *Portugol* (à esquerda) são exibidos no ambiente (Figura 7-14);
- Ao clicar no botão “OK”, o botão “Iniciar” é habilitado para que o usuário possa prosseguir com o uso do **TBC-SO/WEB**. Ao clicar nesse botão, é exibida uma mensagem contendo informações e dicas de uso do **TBC-SO/WEB** (Figura 7-15). Em seguida, é exibida a tela inicial completa do **TBC-SO/WEB**, onde o usuário pode inserir processos na fila de entrada (Figura 7-6);

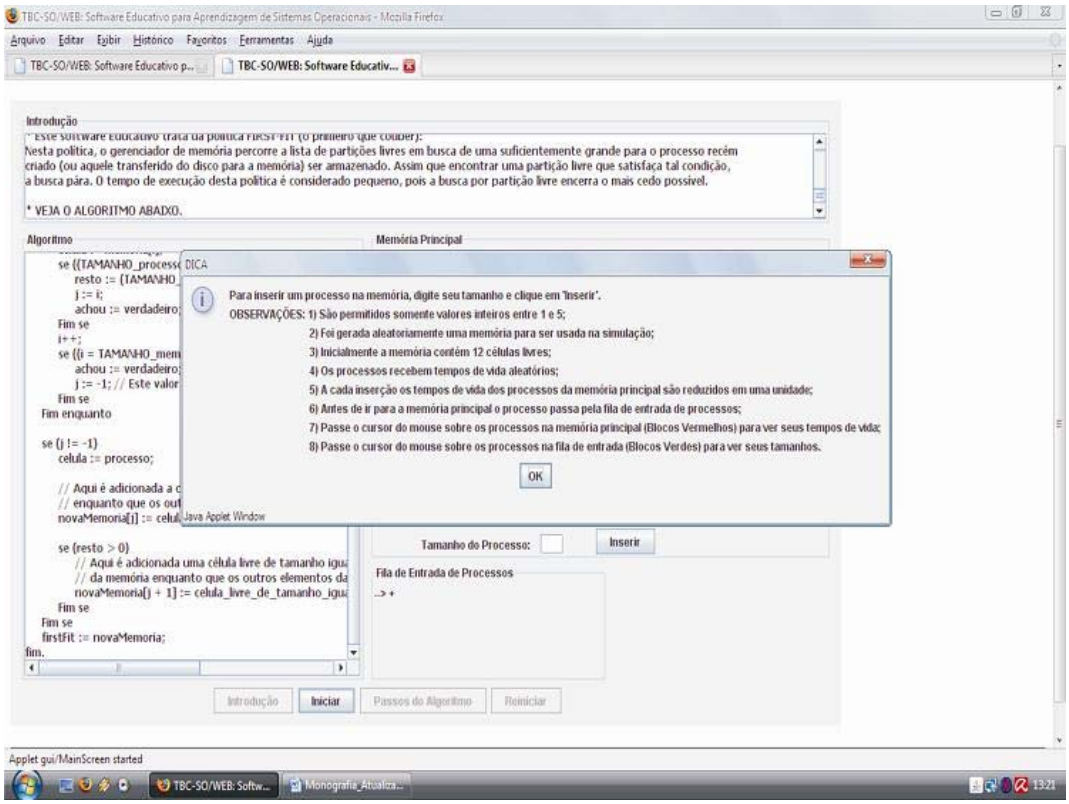

Figura 7-15 – Tela com Mensagem após Clicar no Botão Iniciar

- Para inserir um processo, o usuário deve fornecer o tamanho do processo e clicar no botão “Inserir”. Após clicar neste botão, é exibida uma mensagem informativa sobre a ação. Caso esta ação seja realizada com sucesso, o processo inserido é apresentado na fila de entrada e o botão “Passos do Algoritmo” é habilitado para que o usuário possa acompanhar passo a passo a alocação de espaço na memória para um processo (Figura 7-16);

- Ao clicar no botão “Passos do Algoritmo”, uma nova janela é aberta contendo um botão “OK” para prosseguir passo a passo o algoritmo (Figura 7-9);

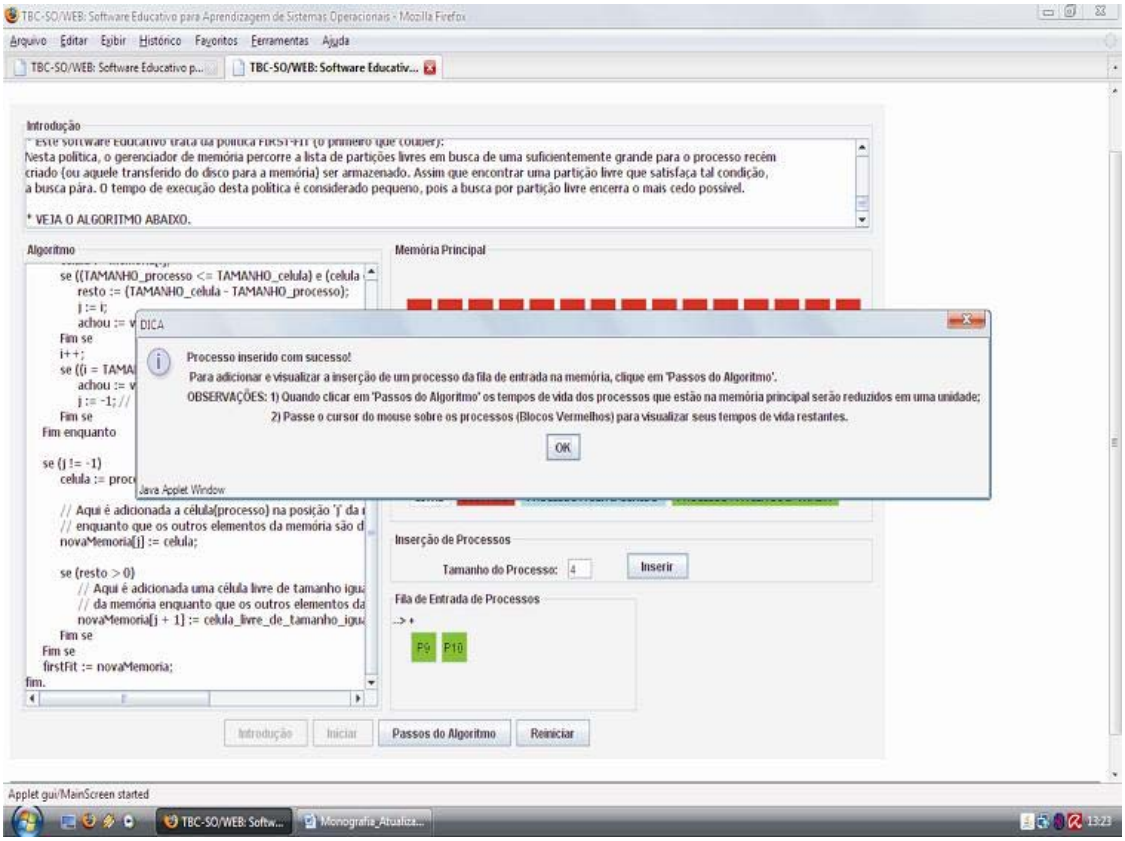

Figura 7-16 – Tela com Mensagem após Inserir um Processo

- O usuário tem opção de inserir processos ou reiniciar o estado do **TBC-SO/WEB** quando quiser.

7.5.2. Gerência de Processos

Neste tópico, as políticas de gerência de processos estão divididas em duas categorias, porém a forma de uso do **TBC-SO/WEB** é semelhante para todos seus programas e é igual a forma de uso dos programas do tópico "Gerência de Memória". Entretanto, para programas deste tópico, há um botão adicional (botão "Relatório") (Figura 7-17). Este botão é habilitado após escalonar pelo menos um processo da fila de processos e, quando clicado, é exibida uma mensagem contendo informações acerca dos processos inseridos (Figura 7-18). Estas informações são: i) tempo de criação; ii) tempo de *burst*; iii) prioridade (quando aplicável); iv) tempo de espera; v) tempo médio de espera; vi) tempo de retorno; e vii) tempo médio de retorno.

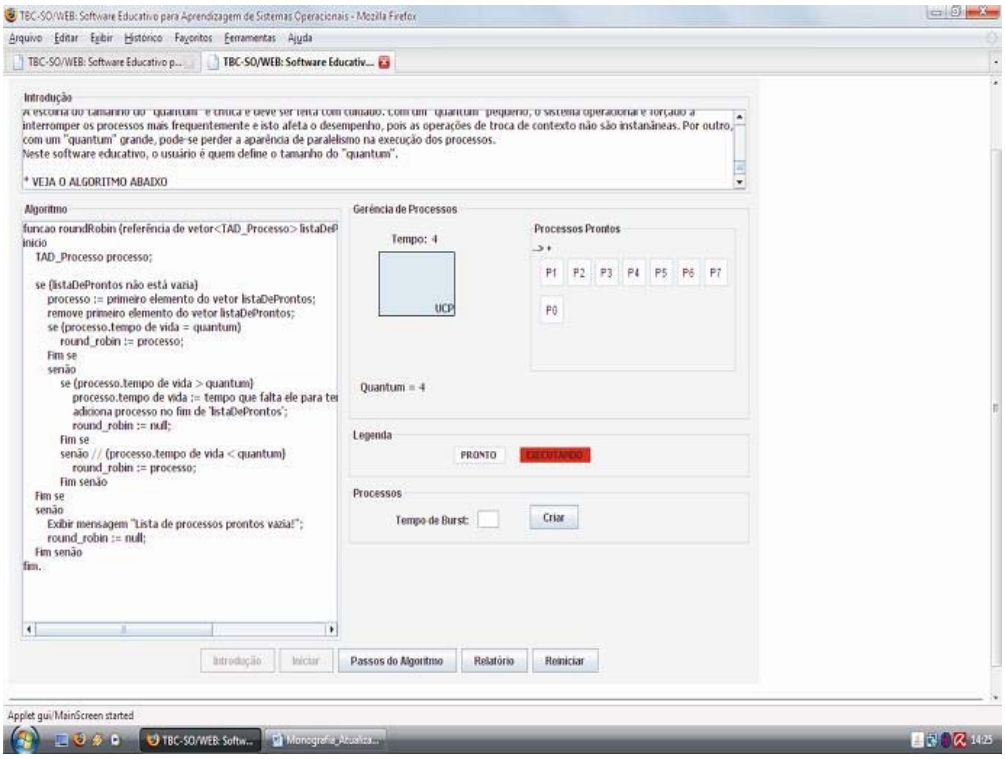

Figura 7-17 – Tela de Execução da Política *Round-Robin*

Durante a execução de um processo, uma barra de progresso (abaixo da figura do processador) é exibida ao usuário. Ela altera seu valor quando o usuário clica no botão “Clique aqui para próximo passo” que aparece após o usuário clicar no botão “Passos do Algoritmo” (Figura 7-19). Para a política de gerência de processos *Round-Robin*, quando o usuário clica no botão “Passos do Algoritmo” pela primeira vez, é aberta uma nova janela para que ele insira o valor do *quantum* a ser usado pela política (Figura 7-20).

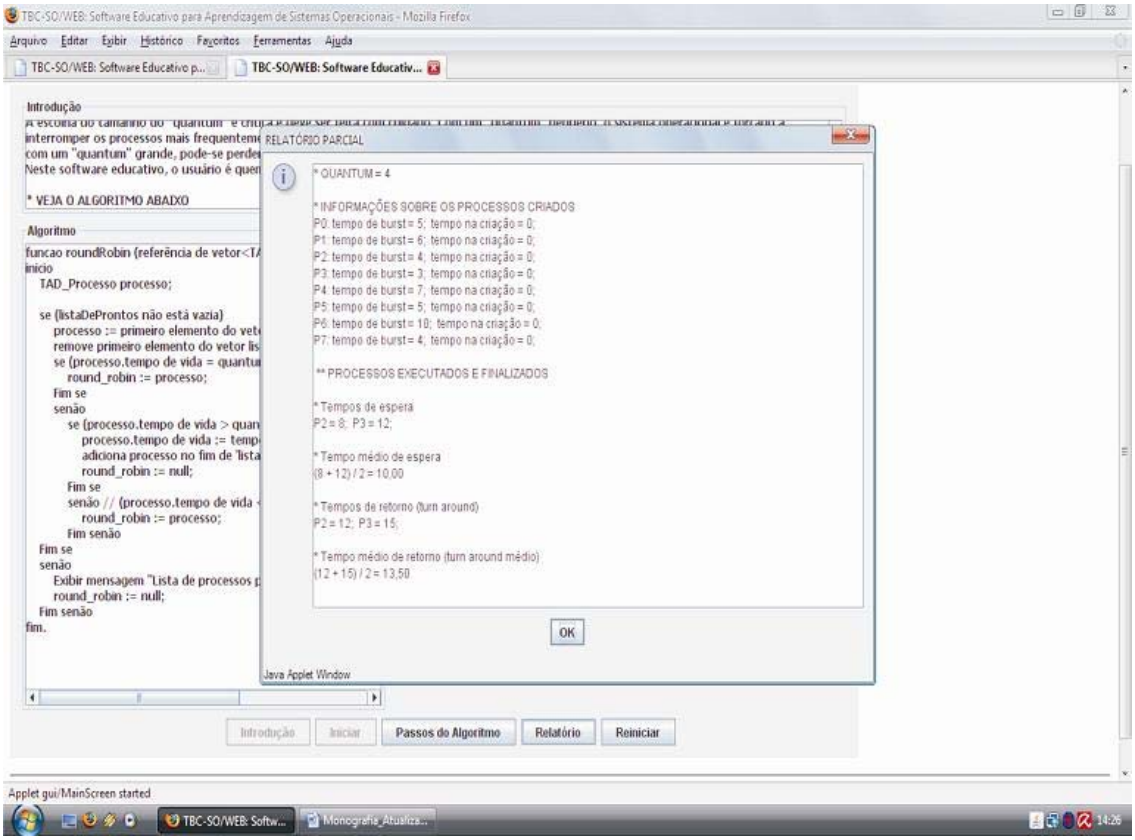

Figura 7-18 – Tela de Execução da Política de Gerência de Processos Round-Robin com Janela de Relatório

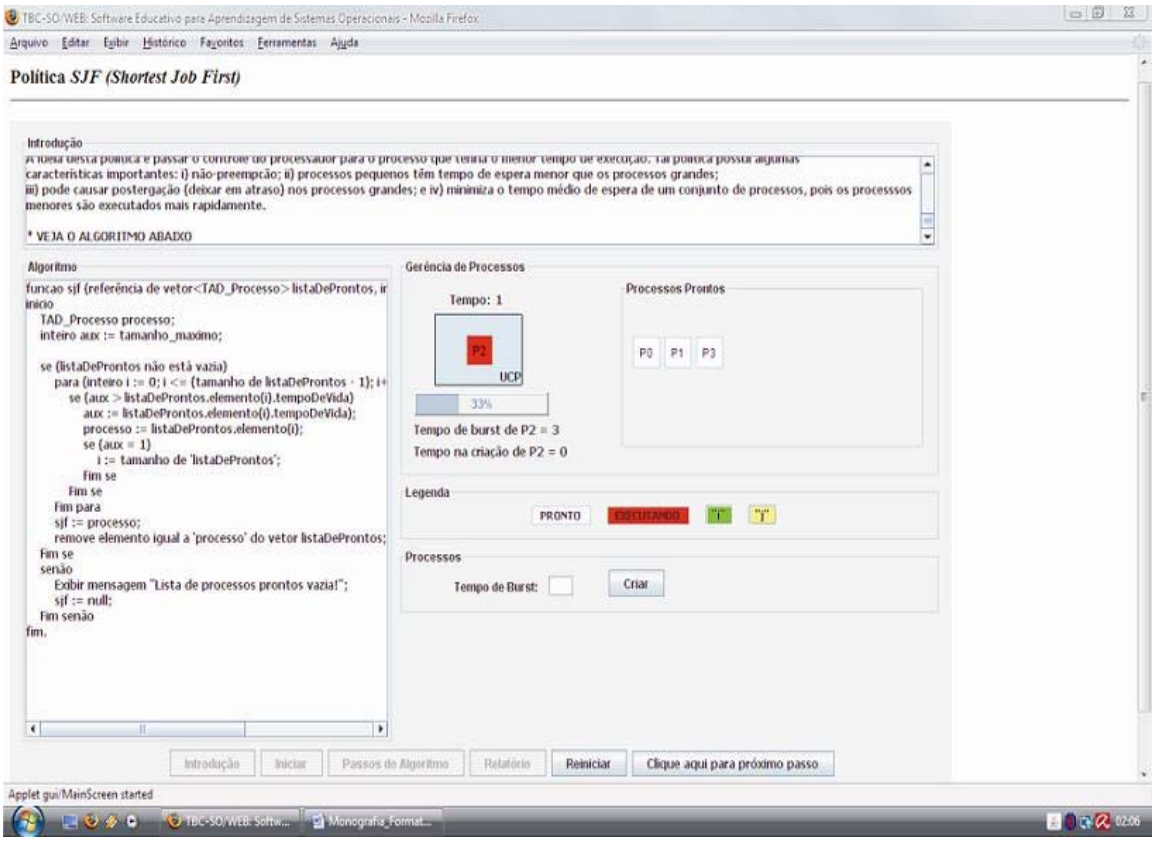

Figura 7-19 – Tela da Política SJF Executando um Processo

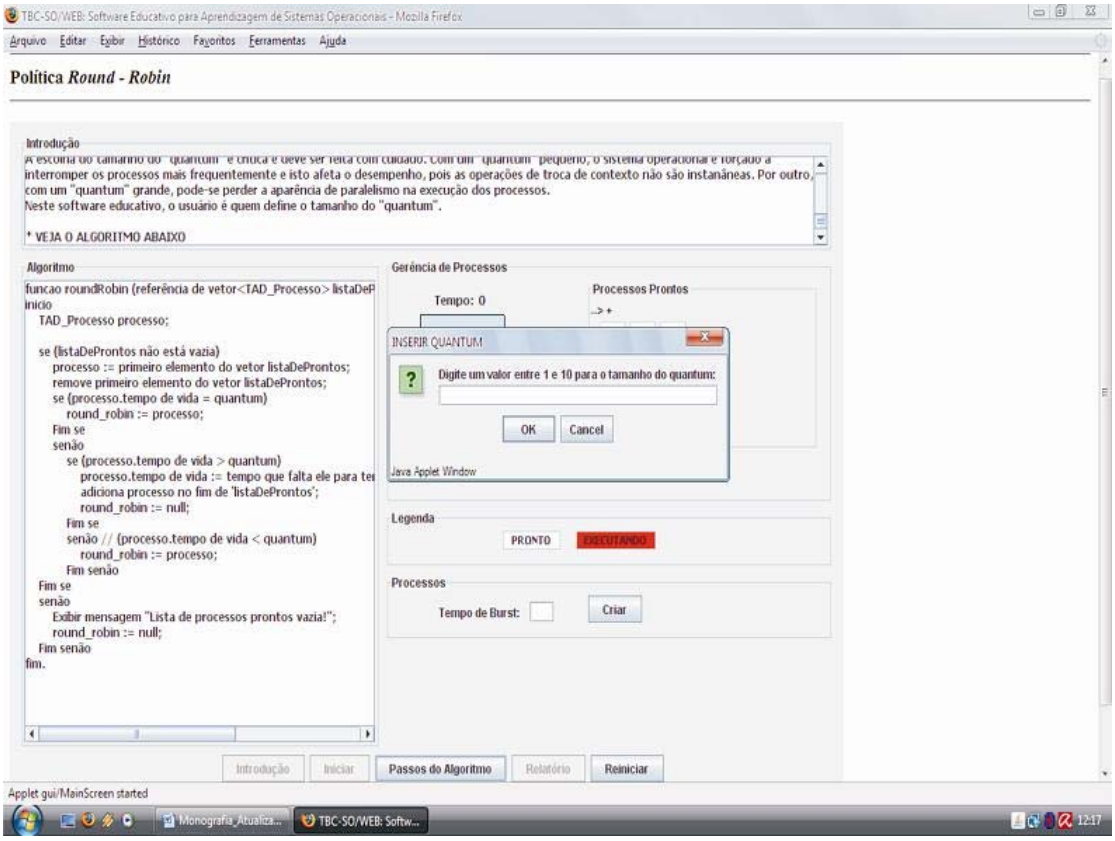

Figura 7-20 – Tela de Execução da Política *Round-Robin* Exibindo Janela "Inserir Quantum"

7.6. Análise Comparativa dos Ambientes Educacionais

Esta seção apresenta uma análise comparativa dos ambientes educacionais citados no Capítulo 6 com o **TBC-SO/WEB**. São consideradas as seguintes características: i) facilidade de uso; ii) visibilidade do *status* de execução; iii) concordância entre os resultados obtidos e a teoria; iv) facilidade de aprendizado do conteúdo abordado; v) uso de linguagem de fácil entendimento para o usuário; vi) apresentação de relatório com dados sobre a execução; vii)

visualização gráfica dos passos dos algoritmos tratados; e viii) apresentação de texto teórico explicativo.

Com esta análise, pode-se notar que o critério facilidade de uso para o **TBC-SO/WEB** é o melhor, pois ele apresenta constantemente janelas contendo dicas de uso e breves legendas explicativas sobre suas partes e funções e direciona o usuário no seu uso com habilitação e desabilitação de botões. Enquanto que os outros ambientes educacionais, com exceção do SOSim, não são simples de usar, pois, de acordo com os testes realizados para a análise, eles não apresentam instruções claras de uso ou disposição de partes com nomes sugestivos. O wxEscalProc é parcialmente fácil de usar, pois, mesmo apresentando botões com nomes sugestivos, ele apresenta erros de execução, caso o usuário não siga os passos corretos de seu uso.

Com relação à visibilidade do *status* de execução, assim como acontece no SOSim, o **TBC-SO/WEB** demonstra visualmente o *status*. Nele, o usuário pode visualizar graficamente o *status* de execução, verificando a posição de blocos coloridos ilustrativos e o seu relógio lógico. No SOSim, o usuário possui o mesmo recurso de visualização gráfica; para isso, ele deve ler os dados de execução apresentados na tela. Os outros ambientes educacionais não apresentam boa visualização do *status* de execução, pois, após inserir dados de entrada, o usuário só terá o resultado por meio de um relatório gerado no final da execução.

Todos os ambientes educacionais possuem relatórios com dados sobre a execução, contudo é preciso ressaltar que o **TBC-SO/WEB** e o SOSim, possuem à disposição do usuário, relatórios parciais de execução. Com estes relatórios parciais, o usuário pode visualizar dados de entrada e execução das políticas abordadas por eles e continuar executando o software, sem finalizar

suas inserções de dados. Além disso, todos os ambientes educacionais apresentaram concordância entre os resultados obtidos e a teoria, salvo o wxEscalProc que pode apresentar erros mediante o não seguimento correto dos passos.

Quanto à facilidade de aprendizado das políticas implementadas pelos ambientes educacionais, o **TBC-SO/WEB** mostra ter a melhor, visto que o seu foco principal ser voltado ao ensino destas políticas e ao uso de recursos gráficos animados para ilustrar os passos envolvidos nelas. O MOSS apresenta as mesmas políticas, porém ele não mostra claramente os passos dos algoritmos envolvidos. Com ele, o usuário insere valores e aguarda o resultado que será armazenado em arquivo. O wxEscalProc mostra ter o mesmo foco do **TBC-SO/WEB**, mas ele não explora bem recursos gráficos animados, tornando sua capacidade de ensino pobre. Entretanto, vale ressaltar que todos eles, com exceção do MOSS, apresentam linguagem fácil de entender. O MOSS não apresenta esta característica, pois a forma de usá-lo faz com que sua linguagem não seja de fácil entendimento.

Dentre os ambientes educacionais analisados, um recurso exclusivo do **TBC-SO/WEB** é a presença de textos teóricos explicativos. Com estes textos, o usuário consegue contextualizar a aplicação das políticas que o **TBC-SO/WEB** aborda. Outra importante característica exclusiva do **TBC-SO/WEB** é a apresentação dos algoritmos (em Portugol) de cada política.

A tabela 7-1 apresenta uma comparação dos ambientes educacionais abordados neste trabalho. As colunas são referentes aos seguintes aspectos:

Tabela 7-1 – Tabela Comparativa dos Softwares Citados e o TBC-SO/WEB

Ambientes Educacionais	1	2	3	4	5	6	7	8
SOSim	Sim	Alta	Sim	Regular	Sim	Sim	Sim	Não
MOSS	Não	Baixa	Sim	Regular	Não	Sim	Não	Não
wxEscalProc	Parcialmente	Regular	Sim	Regular	Sim	Sim	Sim	Não
TBC-SO/WEB	**Sim**	**Alta**	**Sim**	**Alta**	**Sim**	**Sim**	**Sim**	**Sim**

1. Facilidade de uso;
2. Visibilidade do *status* de execução do software;
3. Concordância entre os resultados do software e a teoria;
4. Facilidade de aprendizado das políticas implementadas pelo software;
5. Utilização de linguagem do usuário;
6. Apresentação de relatório com dados sobre a execução das políticas implementadas;
7. Visualização gráfica dos passos do algoritmo tratado;
8. Apresentação de texto teórico explicativo.

7.7. Avaliação em Sala de Aula

Assimilar o conceito e o desenvolvimento de algoritmos é alvo de dificuldade encontrada por alunos de cursos da área de Computação e Informática, visto que disciplinas que apresentam diversos algoritmos.

Com relação ao uso do **TBC-SO/WEB**, o *link* de acesso ao *site* foi divulgado pela lista de *e-mails* dos estudantes de computação e pelo ambiente virtual de ensino (Moodle[12]) da disciplina Sistemas Operacionais, oferecida pelo Departamento de Ciência da Computação da Universidade Federal de Lavras

[12] Moodle é uma aplicação Web que educadores podem usar para criar *sites* de aprendizado *on-line* (Moodle, 2009).

(DCC/UFLA), durante o 1º semestre letivo de 2009. Para isso, o orientado entrou em contato com o professor desta disciplina que o adicionou na sala virtual correspondente para a divulgação do **TBC-SO/WEB**. A lista de *e-mail* dos alunos dos cursos de Ciência da Computação e Sistemas de Informação da mesma universidade foi outro veículo de divulgação e aquisição de respostas para o questionário de avaliação (Anexo A).

No próprio *site* do software, o usuário é convidado a preencher um questionário de avaliação sobre o **TBC-SO/WEB**. Até o momento, as respostas obtidas mostram que o **TBC-SO/WEB** atende às expectativas dos alunos, pois, com as estas respostas, pôde-se perceber que eles o consideram útil, bem organizado (fácil de entender, usar e navegar) e com interface de boa usabilidade (a maneira de usá-lo é intuitiva e suas dicas de como usá-lo são claras). Além disso, os alunos consideraram que houve melhor entendimento do assunto tratado pelo **TBC-SO/WEB**, facilitando a aprendizagem destas políticas, e possibilitou esclarecimento de dúvidas fora da sala de aula, bem como a quantidade de políticas implementadas foi satisfatória. Além dos atuais alunos dos dois cursos de graduação citados, ex-alunos desses cursos (atuais profissionais da área de Computação e Informática) participaram respondendo o questionário.

Neste questionário de avaliação, o usuário teve espaço livre destinado a comentários não contemplados nas questões. Neste espaço, muitos usuários elogiaram o **TBC-SO/WEB** e sugeriram o desenvolvimento de novos ambientes educativos similares para abordar outras áreas da computação. Assim, esta sugestão está descrita como trabalhos futuros. Por meio do *feedback* dos usuários, também pode-se perceber que eles obtiveram êxito na utilização do **TBC-SO/WEB**. Isso resulta em maior expectativa de aumento do aprendizado dos alunos de cursos da área de Computação e Informática.

Com isso, pode-se verificar que a organização do **TBC-SO/WEB** é didática, sendo útil para disciplinas que possuem em suas ementas os tópicos relacionados ao conteúdo abordado por ele. Além disso, é preciso ressaltar que é uma experiência desafiadora aos acadêmicos da área de computação, tornar o ensino de Computação e Informática mais dinâmico, para aumentar e aprimorar a formação de novo pessoal para a área de tecnologia da informação.

Dessa forma, o **TBC-SO/WEB** pode continuar a ser usado nas aulas da disciplina Sistemas Operacionais do DCC/UFLA, incluindo exercícios e análise comparativa dos resultados das simulações realizadas em sala de aula.

7.8. Considerações Finais

Apesar das vantagens que o **TBC-SO/WEB** pode oferecer, é preciso ressaltar que algumas políticas relacionadas com alocação de memória e escalonamento de processos em sistemas operacionais não foram abordadas, como escalonamento garantido, múltiplas filas e outros. Isso é deixado aos docentes para sugerirem aos alunos que pesquisem mais políticas envolvidas, quais são usadas nos sistemas operacionais atuais, etc. para aprofundar conhecimentos e melhorar a discussão durante as aulas.

O **TBC-SO/WEB** propicia ao docente a opção de ministrar aulas mais dinâmicas e atrativas, diferentes daquelas que usam somente o quadro negro como ferramenta de ensino, proporcionando maior tempo para resolução de exercícios e esclarecimento de dúvidas.

A implementação de mais políticas e a inclusão de outros aspectos (por exemplo, considerar tempo nas operações de troca de contexto) relativos às gerências tratadas pelo **TBC-SO/WEB** é proposta como trabalhos futuros.

8. CONSIDERAÇÕES FINAIS

Neste capítulo são apresentadas as considerações finais do presente trabalho. A seção 8.1 apresenta as principais conclusões. Na seção 8.2 são citadas contribuições proporcionadas pelo desenvolvimento deste. Finalizando, a seção 8.3 discorre sobre algumas propostas de trabalhos futuros.

8.1. Conclusões

A partir das informações obtidas, pode-se perceber que o uso de ambientes educativos como ferramentas de ensino é uma idéia interessante, visto que proporciona novas experiências para professores e alunos. Além disso, eles terão oportunidade de avançar mais rapidamente no campo do conhecimento, por causa da agilidade fornecida por esses ambientes.

Após a pesquisa, constatou-se que a qualidade de ensino é melhorada com o uso de ambientes educativos, pois, além de serem facilitadores da aquisição de conhecimento, eles estimulam o raciocínio e processos abstratos, frequentemente encontrados em disciplinas de cursos da área de Computação e Informática que abordam algoritmos, podendo ser mais bem visualizados com uso de processos gráficos animados. Com isso, pode-se concluir também que futuros educadores, considerando aqueles que utilizaram deste recurso, terão interesse em usar o mesmo formato didático nas disciplinas que estiverem ministrando.

Pôde-se perceber que existem vários recursos tecnológicos que possibilitam o desenvolvimento de ambientes educativos, com destaque a plataforma Java, que dispõe de recursos gráficos, portabilidade e forte suporte para técnicas de desenvolvimento de aplicações. Concluiu-se também que as

políticas implementadas pelo **TBC-SO/WEB** apresentam características coerentes com as descritas pelos principais autores referenciados.

Dessa forma, pode-se constatar que a educação precisa ser reavaliada de forma a incorporar novas tecnologias a seu favor a fim de despertar o interesse e a consciência crítica das pessoas.

8.2. Contribuições

O presente trabalho deixa como contribuição um software educativo, **TBC-SO/WEB**, que reuni políticas de gerência de memória e de gerência de processos, mais especificamente, de alocação de memória e de escalonamento de processos. Além disso, deixa um documento científico que revisa a literatura agrupando teorias sobre uso da informática na educação, história e conceitos de sistemas operacionais, políticas de alocação de memória e escalonamento de processos em sistemas operacionais, análise de alguns softwares de mesmo tema e documentação do **TBC-SO/WEB**.

Com relação à divulgação do **TBC-SO/WEB**, em 2008 ele foi apresentado no congresso de iniciação científica da UFLA (XXI CIUFLA) e aos alunos da disciplina Sistemas Operacionais ofertada no segundo semestre letivo pelo Departamento de Ciência da Computação da mesma universidade. O **TBC-SO/WEB** foi apresentado a estes alunos usando o fórum de discussão virtual da disciplina. Foi pedido que os alunos que respondessem um questionário de avaliação do **TBC-SO/WEB**, no intuito de melhorá-lo. Assim, espera-se que o **TBC-SO/WEB** seja utilizado nas aulas da disciplina Sistemas Operacionais a partir do segundo semestre de 2009. Além disso, contribui-se com a comunidade de software, pois seu código fonte está hospedado em servidor disponível na Internet (os *sites* disponibilizados contêm os *links*).

8.3. Trabalhos Futuros

Uma das sugestões de desenvolvimento de trabalhos futuros, como desdobramento deste, é o desenvolvimento de estudo e de implementação de mais políticas de gerência de memória e de gerência de processos, além do desenvolvimento de outras políticas relacionadas a sistemas operacionais, como algoritmos de substituição de página e acesso a disco. Com relação ao desenvolvido, considerar tempo em operações de entrada e saída nas políticas de gerência de processos, na política *Round-Robin*, incluir opção no software para o gerenciador de processos escolher o tamanho do *quantum* e mostrar como o cálculo é feito, melhorar a parte visual tornando-a ainda mais atrativa e traduzir para a língua inglesa.

Além disso, analisar o currículo de cursos da área de computação e continuar a linha de desenvolvimento de ambientes educativos do orientador, porém com relação a outros temas, como redes de computadores, inteligência artificial, banco de dados, etc.

A partir dessa discussão, percebe-se que a área de ensino da computação é ampla e de muitos trabalhos a serem feitos, não só por se tratar de uma área amplamente dinâmica e tecnológica, mas no intuito de atrair mais alunos para fortalecer a geração de novos recursos humanos e promovendo o desenvolvimento do país.

REFERÊNCIAS BIBLIOGRÁFICAS

Anderson, D. A. Tutorial Series 9: Operating Systems. Los Alamitos, California: IEEE Computer Society Press, 1981.

Arruda, F. R. Escalonamento Round-Robin. Disponível em http://www.ime.usp.br/~kon/MAC5755/trabalhos/software/FlavioArruda/node2.html. Consultado em 13/06/2008.

Brookshear, J. Glenn. Ciência da Computação: Uma Visão Abrangente. Traduzido por C. M. Lee. 5ª Edição. Porto Alegre: Bookman, 2002.

Buzin, P. F. W. K. A Epistemologia da Ciência da Computação: Desafio do Ensino dessa Ciência. Revista da Educação, Ciência e Cultura, v. 6, n° 2. Centro Universitário La Salle. Canoas, RS, Brasil.

Campos, E. A. V.; Ascencio, A. F. G. Fundamentos de Programação de Computadores. São Paulo: Prentice Hall, 2002.

Cano, C. A. Os Recursos da Informática e os Contextos de Ensino e Aprendizagem. In: Sacho, Juana M. Para Uma Tecnologia Educacional. Porto Alegre: ArtMed, 1998.

Carvalho, D. S.; Balthazar, G. R.; Dias, C. R.; Araújo, M. A. P.; Monteiro, P. H. R. S2O: Uma Ferramenta de Apoio ao Aprendizado de Sistemas Operacionais. In: XXVI Congresso da SBC – XIV Workshop sobre Educação em Computação (XIV WEI). Campo Grande, MS, 2006.

Comer, D. E. Computer Network and Internets. Upper Saddle River: Pearson Prentice Hall, 1997.

Coscarelli, C. V. O Uso da Informática e os Contextos de Ensino e Aprendizagem. Disponível em http://bbs.metalink.com.br/~lcoscarelli/PrespedMM.pdf. Consultado em 25/04/2008.

Cosnard, M.; Trystram, D. Parallel Algorithms and Architectures. London: International Thomson Computer Press, 1995.

Cysneiros, P. G. Professores e Máquinas: Uma Concepção de Informática na Educação. Disponível em

http://edutec.net/Textos/Alia/PROINFO/prf_txtie08.htm. Consultado em 25/04/2008.

Davis, W. S. Sistemas Operacionais: Uma Visão Sistemática. Traduzido por D. C. Alencar. 3ª Edição. Rio de Janeiro: Campus, 1990.

Deitel, H. M.; Deitel, P. J. Java TM: Como Programar. 6ª Edição. São Paulo: Pearson Prentice Hall, 2005.

Deitel, H. M.; Deitel, P. J.; Choffnes, D. R. Sistemas Operacionais. Traduzido por A. S. Marques. 3ª Edição. São Paulo: Pearson Prentice Hall, 2005.

Flynn, I. M.; Mchoes, A. M. Introdução aos Sistemas Operacionais. 1ª Edição. São Paulo: Thomson, 2002.

Garcia, I. C.; Rezende, P. J.; Calheiros, F. C. Astral: Um Ambiente para Ensino de Estrutura de Dados Através de Animações de Algoritmos. Disponível em http://www.ic.unicamp.br/~rezende/garcia.htm. Consultado em 24/09/2008.

Gasparini, A. F. L.; Barrella, F. E. TCP/IP. 3ª Edição. São Paulo: Érica, 1996.

Guedes, J. R.; Guedes, C. L. Hipermídia Auxílio ao Ensino de Sistemas Operacionais. In: II Congresso Sul Catarinense de Computação (SulComp2006). Criciúma, SC, 2006.

Hayes, J. P. Computer Architecture and Organization. 2nd Edition. New York: McGraw Hill, 1988.

Holcombe, J.; Holcombe, C. Dominando os Sistemas Operacionais: Teoria e Prática. 1ª Edição. Rio de Janeiro: Alta Books, 2003.

Iizuca, K. Ligação Micro-Mainframe. São Paulo: Atlas, 1987.

Jorge, L. Sistemas Operativos: Escalonador da CPU. Disponível em http://www.ipb.pt/~ljorge/so0203/aula22.pdf. Consultado em 13/06/2008.

Kay, J.; Lauder, P. A Fair Share Scheduler. Communications of the ACM. 31(1):44-55, 1988.

Kurose, J. F.; Ross, K. W. Redes de Computadores e a Internet: Uma Abordagem Top-down. Traduzido por A. S. Marques. 3ª Edição. São Paulo: Pearson Prentice Hall, 2005.

Laine, J. M. Aula 13: Gerência de Memória. Disponível em http://regulus.pcs.usp.br/~jean/so/AULA%2013%20-%20Ger%EAncia%20de%20Mem%F3ria.pdf. Consultado em: 11/05/2008.

Laureano, M. A. P. Sistemas Operacionais. Disponível em http://www.ppgia.pucpr.br/~laureano/puc_2007/asu/sistemas_de_arquivos.ppt. Consultado em 17/04/2008.

Lee, V.; Schneider, H.; Schell, R. Aplicações Móveis: Arquitetura, Projeto e Desenvolvimento. Traduzido por A. Bentes e D. Rüdiger. São Paulo: Makron Books, 2005.

Lima, C. A. M. Gerenciamento de Memória. Disponível em http://www.dca.fee.unicamp.br/~moraes/aulas/Aula11.pdf. Consultado em 14/05/2008.

Lima, I. R.; Toledo, M. C. P.; Costa, H. A. X. Um Software Educacional para o Ensino de Geometria Analítica e Álgebra Linear via Web. XXIX Congresso Nacional de Matemática Aplicada e Computacional. 2006.

Machado, F. B.; Maia, L. P. Arquitetura de Sistemas Operacionais. 4ª Edição. Rio de Janeiro: LTC, 2007.

Magalhães, M. F.; Cardozo, E.; Faina, L. F. Introdução aos Sistemas Operacionais. Disponível em http://www.facom.ufu.br/~faina/BCC_Crs/INF09-2S2007/DwLd_SO1/so-1992.ps.gz. Consultado em 18/04/2008.

Medeiros, R. Gerência de Memória. Disponível em http://www.ucb.br/prg/professores/raissad/disciplinas/2006_2/SO/material/memoria.html. Consultado em 11/05/2008.

Menezes, M. O. Sistemas Operacionais – Sistemas de Arquivos. Disponível em http://www2.dem.inpe.br/ijar/SistArquivos.pdf. Consultado em 17/04/2008.

Mercado, L. P. L. Novas Tecnologias na Educação: Reflexões Sobre a Prática. Maceió: EDUFAL, 2002.

Moodle, Open-Souce Community-Based Tools for Learning. Disponível em: http://moodle.org. Consultado em 20/05/2009.

Neitzel, L. C. A Rede Digital na Rede Educacional: Um Reencantamento. Disponível em http://www.geocities.com/neitzeluiz/reencan.html. Consultado em 24/04/2008.

Oliveira, R. Introdução aos Sistemas Operacionais. Disponível em http://www.getec.cefetmt.br/~ruy/pos-graduacao/SO/SO_introduction.pdf. Consultado em 05/04/2008. Ontko, Ray; Reeder, Alexander. MOSS – Modern Operating System Simulators. Disponível em http://www.ontko.com/moss/. Consultado em 22/06/2008.

Oliveira, R. S.; Carissimi, A. S.; Toscani, S. S. Sistemas Operacionais. 2ª Edição. Porto Alegre: Sagra Luzzatto, 2001.

Open Source Initiative. The Open Source Definition. Disponível em http://www.opensource.org/. Consultado em 31/03/2008.

Portal MEC. Disponível em http://portal.mec.gov.br/index.php?option=com_content&view=article&id=289&Itemid=86. Consultado em 03/05/2009.

Rocha, A. R.; Schneider, A.; Alves, J. C.; Silva, R., M. A. wxEscalProc – Um Simulador de Políticas de Escalonamento Multiplataforma. Disponível em http://www.ic.unicamp.br/~rocha/grad/src/wxEscalProc.tar.gz. Consultado em 27/09/2008.

Rocha, A. R.; Schneider, A.; Alves, J. C.; Silva, R., M. A. WxProc – Um Simulador de Políticas de Escalonamento Multiplataforma. INFOCOMP – Jornal of Computer Science. Vol. 3, N. 1: p.43-47, 2004.

Sancho, J. M. Para uma Tecnologia Educacional. Porto Alegre: ArtMed, 1998.

Sandholtz, J. H.; Ringstaff, C.; Dwyer, D. C. Ensinando com Tecnologia: Criando Salas de Aula Centradas nos Alunos. Porto Alegre: ArtMed, 1997.

Santos, R. P. dos; Costa, H. A. X. Desenvolvimento de Aplicativos Gráficos para o Ensino de Estruturas de Dados e Algoritmos em Grafos para Web. Relatório Final apresentado à Universidade Federal de Lavras, como parte das exigências do PBIICT/FAPEMIG, referente ao período de março/2005 a fevereiro/2006. Lavras, 2006.

Santos, R. P.; Costa, H. A. X. TBC-AED (Treinamento Baseado em Computador para Algoritmos e Estruturas de Dados) e TBC-GRAFOS (Treinamento Baseado em Computador para Algoritmos em Grafos). XIX Simpósio Brasileiro de Informática na Educação, 2008. Mostra de Software. v. 1. p. 1-1.

Santos, R. P.; Costa, H. A. X. TBC-AED e TBC-AED/WEB: Um Desafio no Ensino de Algoritmos, Estruturas de Dados e Programação. IV Workshop em Educação em Computação e Informática do Estado de Minas Gerais. 2005a.

Santos, R. P.; Costa, H. A. X. TBC-AED: Um Software Gráfico para Apresentação de Algoritmos e Estruturas de Dados aos Iniciantes em Computação e Informática. I Congresso de Computação do Sul do Mato Grosso. v. 1. p. 215-234. 2005b.

Santos, R. P.; Costa, H. A. X. TBC-GRAFOS/WEB – Treinamento Baseado em Computador para Algoritmos em Grafos Via Web. International Conference on Engineering and Computer Education. 2007. v. 1. p. 825-829.

SBC. Currículo de Referência para Cursos de Ciência da Computação, Engenharia da Computação e Sistemas de Informação. Disponível em http://www.sbc.org.br/index.php?language=1&subject=28&content=downloads&id=82. Consultado em 23/05/2009.

Scama, R. Gerência do Processador. Disponível em http://www.professor.rodrigoscama.com.br/textos/SO_gerencia_processador.pdf. Consultado em 12/06/2008.

Shay, W. A. Sistemas Operacionais. Traduzido por M. M. Fechio. São Paulo: Makron Books, 1996.

Silberchatz, A.; Galvin, P. B.; Gagme, G. Fundamento de Sistemas Operacionais. Tradução da 6ª Edição. Rio de Janeiro: LTC, 2004a.

Silberchatz, A.braham; Galvin, P. B.; Gagme, G. Sistemas Operacionais com Java. Traduzido por D. Vieira. Tradução da 6ª Edição. Rio de Janeiro: Elsevier, 2004b.

Silva, F. J. S. Sistemas Operacionais, Escalonamento de Processos. Disponível em www.deinf.ufma.br/~fssilva/graduacao/so/aulas/escalonamento.pdf. Consultado em 20/12/2008

Stallings, W. Operating Systems: Internals and Design Principles. 5th Edition. Upper Saddle River: Pearson Prentice Hall, 2005.

Starke, M. R. Controle Dinâmido de Recursos em Sistemas Operacionais. Dissertação de mestrado, Pontifícia Universidade Católica do Paraná, 2005.

Sun Microsystems. How to Make Dialogs (The JavaTM Tutorials <Creating a GUI with JFC/Swing> Using Swing Components). Disponível em http://java.sun.com/docs/books/tutorial/uiswing/components/dialog.html. Consultado em 24/04/2009.

Tanenbaum, A. S. Distributed Operating Systems. Upper Saddle River: Pearson Prentice Hall, 1995a.

Tanenbaum, A. S. Sistemas Operacionais Modernos. Traduzido por R. A. L. Gonçalves; L. A. Consularo. 2ª Edição. São Paulo: Pearson Prentice Hall, 2003b.

Tanenbaum, A. S.; Woodhull, A. S. Sistemas Operacionais: Projeto e Implementação. Porto Alegre: Bookman, 1997.

UML. UML® Resource Page. Disponível em http://www.uml.org/. Consultado em 24/05/2009.

Valente, J. A. Diferentes Usos do Computador na Educação. Disponível em http://www.nied.unicamp.br/publicacoes/separatas/Sep1.pdf. Consultado em 23/04/2008.

Willrich, R. Sistemas Operacionais. Disponível em http://www.inf.ufsc.br/%7Ewillrich/Ensino/INE5602/restrito/ii-cap5.PDF. Consultado em 05/04/2008.

Zagari, E. N. F. Gerência de Memória. Disponível em http://www.las.ic.unicamp.br/edmar/PUC/2006/SO/SO-Aula5.pdf. Consultado em: 14/05/2008.

Zambalde, A. L.; Alves, R. M. Interface Homem-Máquina e Ergonomia. Lavras: UFLA/FAEPE, 2003.

ANEXO A – Questionário de Avaliação do Software

Qual teu nome?

Que curso você faz?

Que período?

1) Você considera o TBC-SO/WEB útil?
( ) Sim ( ) Não. Por que?

2) Você acha que o site está bem organizado (fácil de entender e navegar)?
( ) Sim ( ) Não. Por que?

3) Você acha a interface do TBC-SO/WEB (applets) amigável?
( ) Sim ( ) Não. Por que?

4) Você aprendeu / esclareceu uma dúvida / entendeu melhor o funcionamento de alguma política abordada pelo TBC-SO/WEB?
( ) Sim ( ) Não. Por que?

5) Existe alguma(s) política(s) que você gostaria que o TBC-SO/WEB tivesse?
( ) Sim. Quais? ( ) Não

6) Você acha que o TBC-SO/WEB facilita a aprendizagem das políticas abordadas por ele?
( ) Sim ( ) Não. Por que?

7) Espaço livre para sugerir, tirar dúvida, elogiar, criticas, ...