Um Simulador para o Ensino de Sistemas Operacionais com a Tecnologia da Realidade Virtual (ESORV)

Vagner Scamati

Maio / 2017

Dissertação de Mestrado em Ciência da Computação

Um Simulador para o Ensino de Sistemas Operacionais com a Tecnologia da Realidade Virtual – (ESORV)

Esse documento corresponde à dissertação apresentação à Banca Examinadora no curso de Mestrado em Ciência da Computação da Faculdade Campo Limpo Paulista.

Membros da Banca:

_____

Prof. Dr. Marcelo Paiva Guimarães
(Orientador FACCAMP)

_____

Prof. Dr. Luís Mariano del Val Cura
(FACCAMP)

_____

Prof. Dr. Bianchi Serigue Meiguins
(UFPA)

Campo Limpo Paulista, 26 de maio de 2017

FICHA CATALOGRÁFICA

Dados Internacionais de Catalogação na Publicação (CIP)

Câmara Brasileira do Livro, São Paulo, Brasil.

Scamati, Vagner

Um simulador para o ensino de sistemas operacionais com a tecnologia da realidade virtual - ESORV / Vagner Scamati. Campo Limpo Paulista, SP: FACCAMP, 2017.

Orientador: Profº. Dr. Marcelo de Paiva Guimarães

Dissertação (Programa de Mestrado em Ciência da Computação) – Faculdade Campo Limpo Paulista – FACCAMP.

1. Sistemas operacionais. 2. Realidade Virtual. 3. Simulador de sistemas operacionais. 4. Escalonamento de processos. I. Guimarães, Marcelo de Paiva. II. Campo Limpo Paulista. III. Título.

CDD-005.2

Dedicatória

Dedico esta dissertação a minha esposa Renata Borges Scamati e meu filho Benicio Borges Scamati, minhas fontes de admiração e amor, que me faz acordar todos os dias renovado. Sem vocês esta jornada seria muito difícil. Esse projeto também é de vocês. Meu amor por vocês é inexplicável.

Em especial, aos meus pais Antônio Scamati e Vanda que sempre me apoiaram e nunca mediram esforços para o crescimento da família. São grandes incentivadores dos estudos e contribuíram muito na minha formação acadêmica e pessoal. Obrigado de coração mais uma vez. Minha eterna gratidão.

A minha sogra, Maria Lucia, que também foi uma grande incentivadora desta dissertação. Por meio das suas sabias palavras e espiritualidade sempre me orienta a evoluir praticando o bem e a caridade.

Agradecimentos

Inicialmente, quero agradecer a DEUS pelo dom da vida e por ter me ajudado a chegar até aqui, por ter me amparado com muita fé nos momentos difíceis desta caminhada. A nossa Senhora Aparecida de quem sou devoto, por sempre me proteger e mostrar o caminho correto para seguir em frente. Que seu manto sagrado continue abençoando nossas vidas.

Aos que sempre estiveram ao meu lado e são meus companheiros: minha família, Renata e Benicio; meus pais, Vanda e Antônio; minha sogra, Maria Lúcia; meus irmãos Vinicius e Cristiane; meus queridos sobrinhos Cauê, Lorena e Ana Clara.

Ao meu orientador prof. Dr. Marcelo de Paiva Guimaraes, pela confiança, pelos ensinamentos, pelas sabias orientações e principalmente pela sua mão amiga nas horas que precisei, não houve dia e nem horário que não pode colaborar. Sem a sua ajuda não teria sido possível a realização desta dissertação.

A todos os professore do programa de Mestrado em Ciência da Computação, em especial ao Prof. Dr. Shusaburo Motoyama, que sempre que era possível, em nossas conversas, me passava ensinamentos.

Aos funcionários da Faculdade Campo Limpo Paulista - FACCAMP, que sempre me atenderam muito bem e resolveram os problemas quando necessário com muito afinco. Em especial as secretárias Tatiane, Joice, Talita e Bruna e a responsável do financeiro, Leila Tolentino.

E por fim, a todos amigos que participaram desta caminhada no programa de mestrado, meus agradecimentos, em especial: Anderson Oliveira, Bruno Amaral, Bruno Ponsoni e Victor Souza. Também nos divertimos bastante.

"*Uma vida sem as suas dificuldades e desafios é como um diamante bruto sem ser lapidado não a forma e nem brilho. Todas as tempestades fazem do carvalho mais forte e resistente. Assim também somos nós enfrentando as dificuldades e desenvolvendo nossa capacidade de superar. Transformando nossas crenças e verdades em uma realidade mais profunda e sutil. O grande desafio do ser humano é reinventar – se todos os dias buscando na simplicidade das coisas a sua mensagem da vida. Que a nossa existência seja um legado de amor e paz. E que possamos ser lembrados pela coragem e esforço de termos edificados em nós os caracteres da bondade*".

Eduardo Lourenço

Resumo. *A realidade virtual tem demonstrado por meio de seus princípios - imersão, envolvimento e interação, que é capaz de influenciar positivamente e de modo direto a relação ensino-aprendizagem. Ensinar as disciplinas de computação que possuem conteúdos abstratos nas suas ementas tem se tornado um desafio cada vez mais presente nas salas de aula. É o caso da disciplina de Sistemas Operacionais. Embora o modelo tradicional seja útil e explorado em larga escala no ensino da computação, apresenta dificuldades para construir um conhecimento. Possibilitar ao aluno um ambiente que o habilite a aprender pela sua própria experiência e envolvimento faz com que haja um ganho de aprendizado. A realidade virtual permite que a simulação de um evento real seja demonstrada de forma tridimensional (3D). Para isso, esse trabalho desenvolveu um simulador virtual para o ensino da disciplina de Sistemas Operacionais, cujo foco é em escalonamento de processos. Nele, o aluno pode utilizar-se dos benefícios da tecnologia para imergir, interagir e navegar como se estivesse dentro do computador e podendo observar a placa mãe, as memórias e a comunicação abstrata entre os elementos computacionais por meio de seus processos. Além de conseguir configurar uma simulação e executá-la a qualquer momento em tempo real, o simulador pode ser executado em ambientes de computadores simples até as salas de multiprojeções sem necessidade de aquisição de equipamentos específicos. O simulador foi testado com alunos e os resultados obtidos evidenciam um ganho de aprendizado.*

Palavras-chave: *Informática na Educação; Realidade Virtual; Sistemas Operacionais; Simulador Virtual; Simulador de Sistemas Operacionais.*

Abstract. *Virtual reality has been demonstrating through its principles - immersion, involvement and interaction - that it can positively and directly influence the teaching/learning relationship. Abstract. Virtual reality has been demonstrating through its principles - immersion, involvement and interaction - that it can positively and directly influence the teaching/learning relationship. Teaching computing disciplines, which have abstract contents in their syllabus, has become an increasingly present challenge in the classrooms. It happens to the operating systems discipline. Although the traditional model is being useful and exploited on a large scale in computer education, it has difficulties in building knowledge. It provides the student an environment that contributes to their learning gain enabling them to learn from their own experience and involvement. Virtual reality allows the simulation of a real event to be demonstrated in a three-dimensional (3D way) way, a virtual way. In order to do that, this paper proposes the development of an immersive virtual simulator for teaching in the operating systems discipline. The student can use the benefits of technology to immerse, interact and navigate through this immersive virtual simulator, as if they were inside the computer being able to observe the motherboard, memories and abstract communication between the computational elements through their processes. Besides being able to configure a simulation and execute it at any time in real time, the simulator can be executed in simple computer environments or multi-projection rooms without the need of acquiring specific equipment. The simulator was tested with students and the results obtained show a learning gain.*

Key words*: Informatics in Education; Virtual reality; Operational systems; Virtual Simulator;*

Sumário

1. INTRODUÇÃO 1

1.1. Contexto e Motivação 3
1.2. Objetivos 4
1.3. Metodologia 5
1.4. Organização do Trabalho 6

2. CONCEITOS DE SISTEMAS OPERACIONAIS 8

2.1. Visão Geral 8
2.1.1. Monoprogramação e Multiprogramação 10
2.1.2. Interrupção 12
2.1.3. Memória 13
2.1.4. Dispositivos de Entrada e Saída 14
2.2. Processos 16
2.2.1. Conceitos básicos e principais características 16
2.2.2. Estados de um Processo 17
2.2.3. Criação, Finalização e Hierarquia de Processos 19
2.2.4. Tipos de Comportamentos de Processos 20
2.3. Bloco de Controle de Processos – BCP 21
2.3.1. Mudança de Contexto 22
2.4. Escalonamento 24
2.4.1. Conceitos e principais características 24
2.4.2. Algoritmos de Escalonamento 29
2.4.2.1. Escalonamento FCFS (First Come – First Served) 29
2.4.2.2. Escalonamento Menor Job Primeiro (SJF – Shortest Job First) 30
2.4.2.3. Escalonamento Round Robin (RR) 31
2.4.2.4. Escalonamento por Prioridades (PR) 33
2.4.2.5. Escalonamento de Múltiplas Filas (MLQ) 34
2.4.2.6. Escalonamento de Múltiplas Filas com Realimentação (MLFQ) 35

3. REALIDADE VIRTUAL 37

3.1. Definição da Realidade Virtual 37
3.2. Imersão, Interação e Envolvimento 38
3.3. Realidade Virtual Imersiva e Não Imersiva 39

3.4. Sistemas de Realidade Virtual 39
3.4.1. Ambiente Virtual 40
3.4.2. Ambiente Computacional 40
3.4.3. Hardware e Software 41
3.5. O Ensino-Aprendizado pela Realidade Virtual 41
3.5.1. Aplicabilidade do Ensino-Aprendizado pela Realidade Virtual 43
3.5.2. Trabalhos de Realidade Virtual na Área de Educação 43
3.5.3. Discussão 45
4. TRABALHOS CORRELATOS 47
4.1. Trabalhos 47
4.2. Discussão 54
5. SIMULADOR PARA O ENSINO DE SISTEMAS OPERACIONAIS COM REALIDADE VIRTUAL - (ESORV) 56
5.1. Características Principais e Arquitetura do Simulador 56
5.2. Simulação Virtual de Escalonamento de Processos 60
5.3. Principais tecnologias utilizadas na implementação 64
5.4. Considerações Finais 68
6. TESTES 69
6.1. Tipos de Testes e Metodologia 69
6.2. Cenário e Participantes 72
6.3. Resultados 75
6.3.1. Resultados dos testes da comparação do ESORV com o método tradicional de aula (instrutivista) 75
6.3.2. Resultado dos testes da comparação do ESORV com a ferramenta 2D - SOSIM 79
6.4. Considerações Finais 82
7. CONCLUSÕES 83
7.1. Contribuições da pesquisa 83
7.2. Trabalhos futuros 84
7.3. Considerações finais 84
REFERÊNCIAS 86

Apêndice I – Questionário de teste- Método Tradicional de aula (instrutivista) - elaborado pelo autor. 90

Apêndice II – Questionário de teste- Método ESORV (3D) - elaborado pelo autor.....92

Apêndice III – Questionário de teste- Método SOSIM (2D) - elaborado pelo autor. 95

Apêndice IV – Artigos aceitos para publicação 98

Glossário

2D - Duas dimensões

3D - Três dimensões

SBC - Sociedade Brasileira da Computação

BCP - Bloco de Controle de Processos

CPU - Unidade Central de Processamento

RAM - *Random Access Memory* (Memória de Acesso Aleatório)

ROM - *Read Only Memory* (Memória de Acesso à Leitura)

EPROM - *Erasable Programmable Read-Only Memory* (Memória Programável Apagável somente de Leitura)

EEPROM - *Electrically-Erasable Programmable Read-Only Memory* (Memória Programável Apagável por Eletricidade somente de Leitura)

I / O - *Input / Output* (E / S - Entrada e Saída)

USB - *Universal Serial Bus* (Barramento Universal Serial)

DMA - *Direct Memory Access* (Acesso à Memória Direto)

IEEE - Instituto de Engenheiros Eletricistas e Eletrônicos

ACM DL - *Association for Computing Machinery*

RR - *Round Robin* (algoritmo de escalonamento)

FCFS - *First Come - First Served* (Primeiro a Chegar, Primeiro a ser Atendido)

FIFO - *First In – First Out* (Primeiro a Entrar é o Primeiro a Sair)

PR – (Prioridade)

SJF - *Shortest Job First* (Processo Mais Curto)

MLQ - *Multiple Queues* (Múltiplas Filas)

MLFQ - *Multiple Queues with Feedback* (Múltiplas Filas com Realimentação)

HMD - *Head Mounted Display* (equipamento eletro-ótico compacto, montado no capacete)

TCL - *Tool Comand Language* (Linguagem de Comando de Ferramentas)

C - Linguagem C

MS-DOS - *Microsoft System Direct Operation* (Sistema Operacional Direto da Microsoft)

ESORV - Simulador de Ensino de Sistema Operacional com Realidade Virtual

SOSIM - Sistemas Operacionais Simulador

Lista de Tabelas

Tabela 1: Principais Objetivos do Escalonamento 28
Tabela 2: Descrição de duração e chegada dos processos para FCFS 30
Tabela 3: Descrição de duração e chegada dos processos para Round Robin 32
Tabela 4: Descrição de duração e chegada dos processos para Prioridades 33
Tabela 5: Relação de Artigos Selecionados 48
Tabela 6: Relação de Softwares Utilizados 67
Tabela 7: Principais Tópicos dos Testes 71
Tabela 8: Tipos de Questionários para os Testes 72
Tabela 9: Cenário de Aplicação dos Testes 73
Tabela 10: Resultado Estatístico com o Test t independente (Instrutivista x ESORV) . 77
Tabela 11: Resultado Estatístico com o Test t pareado do Grupo 1 78
Tabela 12: Resultado Estatístico com o Test t dependente do Grupo 2 78
Tabela 13: Resultado Estatístico com o Test t independente (SOSIM x ESORV) 81
Tabela 14: Resultado Estatístico com o Test t pareado do Grupo 1 81
Tabela 15: Resultado Estatístico com o Test t pareado do Grupo 2 82

Lista de Figuras

Figura 1: Fases da metodologia proposta 5

Figura 2 - Estrutura de um sistema de computação típico (fonte: adaptada de Maziero, 2013)..... 10

Figura 3 - Organização de um sistema de computador moderno 12

Figura 4 - Hierarquia de memórias (fonte: adaptada de Stallings, 2010)..... 14

Figura 5 - Estados de um processo e suas transições (fonte: adaptada de Silberschatz et al., 2015)..... 18

Figura 6 - Processos CPU-bound x I/O-bound (fonte: adaptada de Machado & Maia, 2013)..... 21

Figura 7 - Passo de uma troca de contexto (fonte: adaptada de Silberschatz et al., 2015) 23

Figura 8 - Representação do processo de escalonamento em diagrama de enfileiramento (fonte: adaptada de Silberschatz et al., 2015)..... 25

Figura 9 - Escalonamento FCFS..... 30

Figura 10 - Escalonamento SJF não preemptivo..... 30

Figura 11 - Escalonamento SJF preemptivo..... 31

Figura 12 – Escalonamento Round Robin..... 32

Figura 13 - Escalonamento por Prioridades 34

Figura 14 - Escalonamento com múltiplas filas (fonte: Silberschatz et al., 2015)..... 35

Figura 15 - Escalonamento com múltiplas filas com realimentação (fonte: adaptada de Silberschatz et al., 2015)..... 36

Figura 16 - Dispositivos de exploração de Realidade Virtual (fonte: Tori & Kirner, 2006)..... 40

Figura 17 - Simulação com campos geológicos por meio do VGE (fonte: Fengru et al., 2013)..... 43

Figura 18 - Laboratório Virtual de ensino de redes de computadores (fonte: Voss, 2013) 44

Figura 19 - Exibição de conteúdo utilizando slide (fonte: Banerjee et al., 2013) 45

Figura 20 - Demonstração de uma simulação prática (fonte: Suranauwarat, 2007)..... 49

Figura 21 - Interface do usuário para o Realtss (fonte: Diaz A et al., 2007)..... 52

Figura 22 - Interface de gerência do processador do SOSIM (fonte: Machado & Maia, 2013)..... 54

Figura 23 - ESORV – interface inicial representando os principais elementos internos de um computador..... 58

Figura 24 - ESORV - simulador para o ensino de escalonamento de processos com realidade virtual sendo executado em um mini-CAVE..... 59

Figura 25 - Arquitetura do simulador ESORV..... 60

Figura 26 - Ciclo de Simulação do ESORV..... 61

Figura 27 - Tipos de algoritmos que podem ser ecolhidos para executar uma simulação no ESORV..... 61

Figura 28 - O aluno cria o processo e configura os seus parâmetros no ESORV 63

Figura 29 - A rotina “despachante” do Sistema Operacional conduzindo o processo que estava no estado de “pronto” na memória RAM para ocupar a CPU..... 63

Figura 30 - O aluno visualiza o conteúdo do BCP do processo salvo no ESORV..... 64

Figura 31 - Interface da Unity 3D usada na construção do simulador ESORV 66

Figura 32 - Execução do Google CardBoard (olho esquerdo e direito) 67

Figura 33 (a) - Tipo de Teste Independente 71

Figura 33 (b) - Tipo de Teste Pareado..... 71

Figura 34 (a) - Cenário de execução dos testes 74

Figura 34 (b) - Cenário de execução dos testes..... 74

Figura 34 (c) - Cenário de execução dos testes 75

Figura 35 - Número de acertos e erros por grupo (Instrutivista x ESORV)..... 76

Figura 36 - Medidas características por grupo (Instrutivista x ESORV) 77

Figura 37 - Números de acertos e erros por grupo (SOSIM x ESORV) 79

Figura 38 - Medidas características por grupo (SOSIM x ESORV) 80

1. INTRODUÇÃO

A utilização de recursos computacionais empregados no processo de ensino-aprendizagem tem aumentado significativamente nos últimos anos impactando na metodologia de como o aluno pode absorver conhecimento e também alterando a forma de como o professor pode transmitir a informação.

É comum que os alunos enfrentem dificuldades de aprendizagem nas disciplinas dos cursos de Computação, como, por exemplo, nas disciplinas Sistemas Operacionais, Redes de Computadores, Estrutura de Dados e Arquitetura de Computadores. Um dos fatores que promovem essa dificuldade é o fato de elas envolverem conceitos abstratos, possuírem uma conotação de aprendizagem técnico e as vezes usar o treinamento prático, o que dificulta a relação ensino-aprendizagem.

O ensino das disciplinas dos cursos de Computação tem historicamente empregado uma abordagem instrutivista, que se baseia no desenvolvimento de um conjunto de atividades sequenciais com objetivos predefinidos. Esse modelo busca na relação ensino-aprendizagem expor a maior quantidade possível de informação ao aluno, sem focar no aspecto da formação e absorção do aprendizado pelo sujeito. Segundo Maziero (2002), isso não tem demonstrado ser suficiente para que a maioria dos alunos tenha uma compreensão clara e precisa sobre o que está sendo transmitido nas disciplinas.

Na teoria do Construtivismo, criada pelo biólogo suíço Jean Piaget (1932), cita em seu livro, que o aprendizado passa a ser um processo ativo, sendo a progressão da experiência o ponto principal para assimilar o conhecimento, evoluindo por meio de um processo interativo de construção, onde o foco principal é o sujeito. Essa teoria sustenta que o conhecimento não pode ser uma cópia, visto que é sempre uma relação entre objeto e sujeito e coloca menos ênfase nas sequências de aprendizagem, como na teoria instrutivista, e mais ênfase no projeto de aprendizagem, no qual o sujeito desempenha um papel ativo e essencial. O processo de construção do conhecimento passa a ser ativo na relação de aprendizagem, e ensinar é, sobretudo, dar suporte ao processo de aprendizado mais que transmitir o conhecimento simplesmente.

Sob essa ótica do Construtivismo, a utilização de tecnologias pode contribuir para melhorar e auxiliar na relação ensino-aprendizagem, sobretudo com o emprego da

realidade virtual, que por meio de seus ambientes de aprendizagem, de descoberta, da exploração, da simulação, podem contribuir com a evolução da relação ensino-aprendizagem.

Nas aplicações de realidade virtual, o aluno pode visualizar e interagir no ambiente de aprendizagem tridimensional (3D) aliado com estímulos aos outros sentidos humanos. Conforme o aluno interage com o ambiente, ocorrem mudanças na aplicação alvo. Assim o aluno tem a sensação de imersão, ou seja, de fazer parte do mundo simulado, e constrói seu conhecimento de forma participativa e ativa, convivendo com experiências que colaboram na relação ensino-aprendizagem. A realidade virtual pode permitir que os conceitos sejam aprendidos de forma concreta. Segundo Silva et al. (2005), essa característica está de acordo com os métodos construtivistas de ensino que estabelecem a necessidade de aprendizagem pela experiência do sujeito.

Segundo Tori & Kirner (2006), a realidade virtual pode ser definida como uma interface estendida ou avançada para propiciar a movimentação, visualização e interação do usuário em tempo real com a aplicação alvo sendo executada no computador em ambientes tridimensionais.

No âmbito do ensino dos cursos da Computação, empregar a tecnologia da realidade virtual pode trazer inúmeros benefícios, como, por exemplo, tornar o aprendizado mais interessante e eficiente, melhorar a motivação e a atenção a até reduzir custos quando a utilização do objeto e do ambiente real for mais dispendiosa que a simulação (Clark, 2006). Isso pode conduzir o aluno a prospectar um modelo de conhecimento, caminhando na direção adequada do aprendizado e, dessa maneira, permitir ir além do conhecimento transmitido somente.

Essa dissertação tem como motivação explorar o uso dos recursos da realidade virtual para simular o escalonamento de processos na disciplina Sistemas Operacionais. Para isso, desenvolveu-se um simulador tridimensional (3D), denominado Simulador para o ensino de Sistemas Operacionais com a tecnologia da realidade virtual (ESORV), que visa apoiar a relação ensino-aprendizagem. No simulador os alunos podem construir o aprendizado ao realizar simulações de um conteúdo de acordo com o seu ritmo. O aluno poderá usar desde celular até sistemas de multiprojeções para executar a simulação.

1.1. Contexto e Motivação

A disciplina de Sistemas Operacionais é integrante do núcleo de fundamentos dos cursos de Computação de acordo com a Sociedade Brasileira de Computação (SBC) e faz parte do currículo de referências para cursos de graduação de Computação e Informática no Brasil (SBC, 2003).

Segundo Maziero (2002), a disciplina de Sistemas Operacionais tem como proposta apresentar aos alunos os principais conceitos, fundamentos e técnicas usados na construção de Sistemas Operacionais, e também como esses conceitos podem ser usados e influenciam na construção de aplicações. Para isso, ela engloba na sua ementa módulos clássicos do assunto, como, por exemplo: Gerência de Processos, Escalonamento de Processador, Gerência de Memória e Comunicação entre Processos.

O Sistema Operacional é composto por módulos internos de softwares que são integrados e que possuem objetivos e funções distintas, contudo complementares e dependentes entre si. Esses módulos são utilizados para o gerenciamento dos recursos de hardware e software. Ao ensinar um módulo específico, se faz necessário demonstrar que vários conceitos e fundamentos são utilizados de forma dependente.

Uma das principais características no desenvolvimento da disciplina de Sistemas Operacionais é a dificuldade do estudante em abarcar os conhecimentos que lhe permitem uma visão em diversos níveis, do software (lógico) ao hardware (eletrônico), e entender, de maneira sistêmica e consistente, como suas interações se processam.

Demonstrar ao aluno como é realizada a interação existente em um sistema computacional que abrange o contexto de mais baixo nível de um computador, conhecido como hardware (registradores, memórias, chips), até o mais alto nível, no caso os softwares (aplicações), torna-se uma tarefa difícil tanto para o professor quanto para o aluno.

Um exemplo dessa dificuldade é fazer com que o aluno absorva adequadamente o conhecimento sobre Escalonamento de Processos. Para isso, é necessário explicar com recursos abstratos toda a interação entre o hardware e software, as características dos processos, os estados deles, os diversos algoritmos de escalonamento e outros pontos envolvidos.

Atualmente, os professores, de uma maneira geral, utilizam métodos tradicionais na aplicação de recursos didáticos de ensino, como, por exemplo: transparências de figuras, projeções, quadro acrílico, mapas de esquemas conceituais e principalmente laboratórios de informática com softwares restritos. Esses recursos, apesar de valiosos, não dispõem de mecanismos como o da tridimensionalidade e o de suporte à interação, que podem auxiliar a ilustrar os assuntos envolvidos na área de Sistemas Operacionais, como, por exemplo, a visão de dentro de um computador, a migração de um processo da memória principal para um processador, a alocação de um processador por um processo. Existem softwares que simulam Sistemas Operacionais, contudo a ilustração deles é limitada, por exemplo, não permitem a visão interna do hardware.

A partir deste contexto, torna-se relevante buscar alternativas e ambientes de aprendizado para a disciplina de Sistemas Operacionais, nos quais se possa suportar o envolvimento, a integração e a imersão do aluno no seu ambiente, contribuindo com a melhoria em seu conhecimento. Por isso, esse trabalho tem como uma das principais metas o desenvolvimento de um simulador baseado em realidade virtual para apoiar a disciplina de Sistemas Operacionais.

1.2. Objetivos

O objetivo dessa dissertação é responder a indagação: “Como contribuir e melhorar a relação de ensino aprendizado para a disciplina de Sistemas Operacionais ?”.

Para tanto, essa resposta leva ao ponto central desta dissertação: construir uma ferramenta de apoio ao ensino que:

- Seja apoiada em realidade virtual e, consequentemente, proporcione aos alunos a imersão, envolvimento e interação;
- Possibilite colaborar com o método construtivista de ensino, levando-se em consideração a própria experiência do aprendiz na construção do conhecimento;
- Facilite o entendimento do conteúdo da disciplina de Sistemas Operacionais.

Essa dissertação tem como propósito principal contribuir para que os alunos da área de Ciência da Computação entendam de forma fácil os conteúdos da disciplina de

Sistemas Operacionais. Além disso, visa explorar e validar os recursos de realidade virtual como a tecnologia que pode auxiliar o processo de ensino-aprendizagem.

1.3. Metodologia

A Figura 1 apresenta as fases que compõem a metodologia de pesquisa adotada nessa dissertação conforme detalhadas abaixo:

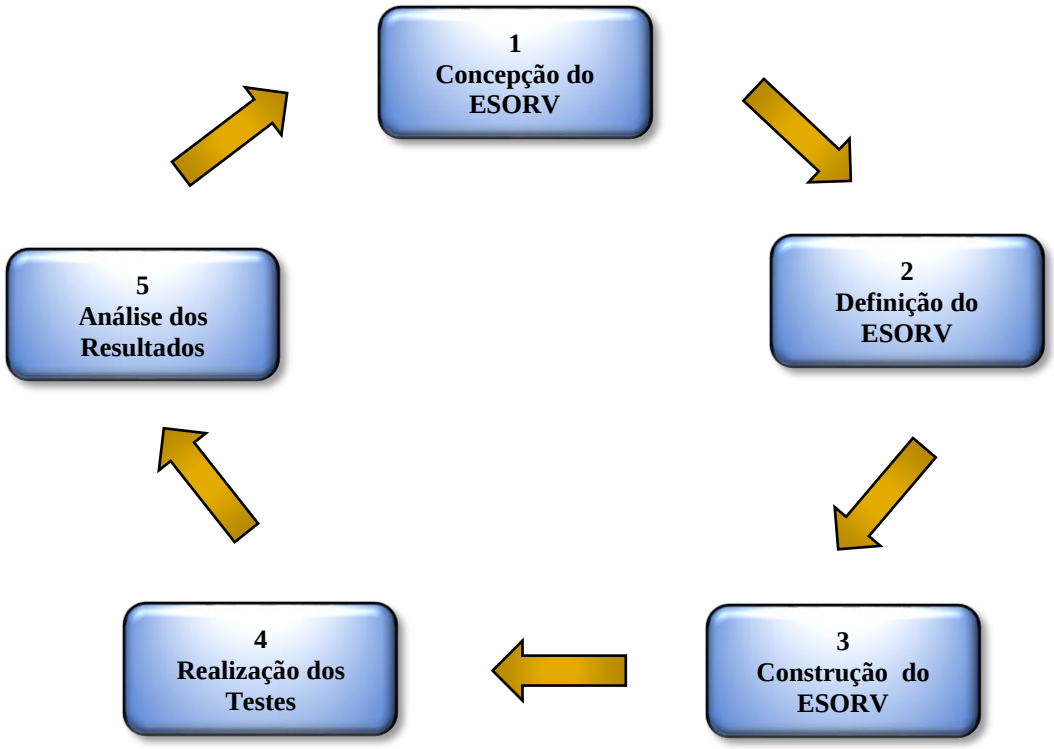


Figura 1: Fases da metodologia proposta

1. **Concepção do ESORV:** a necessidade de idealizar um projeto de ensino utilizando a realidade virtual foi concebida no cotidiano de aulas ministradas da disciplina de Sistemas Operacionais, observando e avaliando ao longo tempo características que poderiam contribuir na execução do projeto e na definição dos seus requisitos.
2. **Definição do ESORV:** nessa fase foi elaborado o protótipo levando-se em consideração o escopo dos requisitos levantados na fase anterior, a delimitação das áreas do conteúdo da disciplina de Sistemas Operacionais que seriam implementadas, a escolha de uma ferramenta de desenvolvimento que atenda

aos preceitos da realidade virtual, a usabilidade do sistema virtual, e os limites impostos para a definição do protótipo.

3. **Construção do ESORV:** essa fase implicou no desenvolvimento do simulador, utilizando-se de um motor de jogos. Construiu-se o ESORV de acordo com a concepção e definições do projeto. Esse simulador foi idealizado para avaliar a solução proposta.
4. **Realização dos Testes:** os testes foram implementados para derivar métricas que contribuam com a avaliação do simulador e principalmente para fornecer subsídios as futuras melhorias.
5. **Análise dos Resultados**: essa fase compila os resultados, avalia e faz comparação com outra ferramenta similar. Obtêm-se parâmetros para contextualizar o simulador.

1.4. Organização do Trabalho

A estruturação dessa dissertação foi dividida da seguinte forma:

O Capítulo 2 apresenta o referencial teórico utilizado para contextualização dos conceitos da disciplina de Sistemas Operacionais. Inicialmente, os conceitos de entendimento dos projetos dos Sistemas Operacionais: conceitos organização e estrutura. Mas, sobretudo, os conceitos relacionados ao contexto em que está inserida a Gerência de Processos, Escalonamento e seus algoritmos e sobre o Bloco de Conceito de Processos (BCP).

O Capítulo 3 também apresenta o referencial teórico utilizado para a contextualização da realidade virtual, seus princípios, principais características e definições. Apresenta também uma pesquisa de trabalhos aplicados como os princípios da realidade virtual em diversas áreas de ensino educacional.

O Capítulo 4 reúne trabalhos correlatos pesquisados em bases científicas indexadas que relatam trabalhos sobre a construção de soluções para o ensino das disciplinas de Sistemas Operacionais, especificamente na área de escalonamento de processos e seus algoritmos.

O Capítulo 5 apresenta o simulador de Ensino de Sistemas Operacionais com Realidade Virtual (ESORV). Descreve as soluções utilizadas no projeto e demonstra suas principais funcionalidades.

O Capítulo 6 demonstra os testes realizados com o ESORV, comparado ao método tradicional de aula (instrutivista) e com outro projeto semelhante, porém com tecnologia 2D.

O Capítulo 7 descreve as conclusões finais e apresenta sugestões para os trabalhos futuros.

2. CONCEITOS DE SISTEMAS OPERACIONAIS

Este capítulo apresenta os principais conceitos e definições aplicados no ensino do conteúdo da disciplina de Sistemas Operacionais e que são tratados pelo simulador desenvolvido. No âmbito de um sistema computacional, segundo Tanembaum (2009), o Sistema Operacional é um conjunto de rotinas que gerencia os recursos de hardware e software, propiciando eficiência e segurança na utilização do computador. Obrigatoriamente, a disciplina de Sistemas Operacionais compõe a grade curricular dos cursos de Computação existentes atualmente no Brasil.

A seção 2.1 demonstra uma visão geral dos principais conteúdos comuns à maioria dos Sistemas Operacionais, como, por exemplo: monoprogramação e multiprogramação, interrupção, memória e dispositivos de entrada e saída. A seção 2.2 trata dos conteúdos que estão relacionados ao gerenciamento de processos: seus conceitos, tipos de comportamento, os estados e suas transições. A seção 2.3 descreve sobre o bloco de controle de processos (BCP) e a mudança de contextos entre os processos. A seção 2.4 demonstra os conceitos de escalonamento dos processos e seus algoritmos.

2.1. Visão Geral

Segundo Silberschatz et al. (2015), um sistema computacional é idealizado por três níveis fundamentais (hardware, software e usuários) que interagem entre si para proporcionar a utilização de um ambiente computacional.

- **Hardware**: são os elementos físicos dos computadores como processador, memória principal, memória secundária, teclados e mouse. Eles podem ser classificados em dispositivos de entrada, que se comunicam do ambiente do usuário ao ambiente do hardware; em dispositivos de saída, que se comunicam do ambiente do hardware ao ambiente do usuário e em dispositivos de entrada e saída simultaneamente, que são dispositivos que se comunicam em ambas as direções, do usuário ao hardware e vice-versa;
- **Software (aplicação) de propósito geral**: é o elemento lógico do sistema computacional, que é composto pelas instruções que implicam no funcionamento geral do sistema, por exemplo, o Sistema Operacional;

- **Usuários**: são as pessoas que utilizam os sistemas computacionais. Eles são variados desde profissionais da computação a usuários finais comuns.

Entre o hardware e o software existe uma lacuna, um distanciamento que dificulta a comunicação de ser efetuada. Os circuitos de hardware são complexos, acessados por meio de interfaces consideradas de baixo nível com instruções lógicas direcionadas às características das tecnologias utilizadas na sua construção, enquanto os softwares são geralmente em uma linguagem de programação de alto nível, direcionados ao entendimento do usuário.

Nessa lacuna, se faz necessária uma camada denominada genericamente de Sistemas Operacionais, que é necessária para o funcionamento adequado e seguro de um sistema computacional. É um tipo de software com tarefa específica.

O Sistema Operacional é considerado um software de estrutura complexa, que interage, comunica e gerencia o hardware (dispositivos) e o software (aplicações). Incorpora diversos aspectos complexos desde os de baixo nível (como drivers de dispositivos e gerência de memória física) até os de alto nível (como programas utilitários e a própria interface gráfica).

Segundo Tanembaum (2009), o Sistema Operacional pode ser considerado um dispositivo de software, cujo objetivo é fornecer um modelo de computador melhor aos programas do usuário, mais simples e lidar com o gerenciamento de todos os recursos. Já Silberschatz et al. (2015), conceitua que um Sistema Operacional é um programa que gerencia o hardware do computador e fornece para os programas aplicativos uma base para o seu funcionamento, além de atuar como intermediário entre o usuário e o hardware do computador.

O Sistema Operacional é composto por um conjunto de módulos com funções específicas e que se propõem a fornecer serviços e resolver a complexidade de operações do sistema. Dentre os módulos destacam-se: Gerenciamento de Processos, Gerenciamento de Processador, Gerenciamento de Memória, Gerenciamento de Entrada e Saída e Gerenciamento de Sistemas de Arquivos (Maziero, 2013). Cada módulo possui entradas, saídas e funções bem definidas. Alguns Sistemas Operacionais são projetados para dar eficiência e conveniência de acordo com sua

finalidade, tornando um sistema de computação utilizável.

A Figura 2 demonstra a arquitetura geral de um sistema de computação. Nela, pode-se observar os elementos de hardware (discos, memórias, portas USB e redes); o Sistema Operacional, que tem a função de gerenciar o hardware e o software, e alguns programas aplicativos próximos aos usuários. E, além disso, mostra a integração executada pelo próprio Sistema Operacional entre os softwares de propósito geral e o hardware.

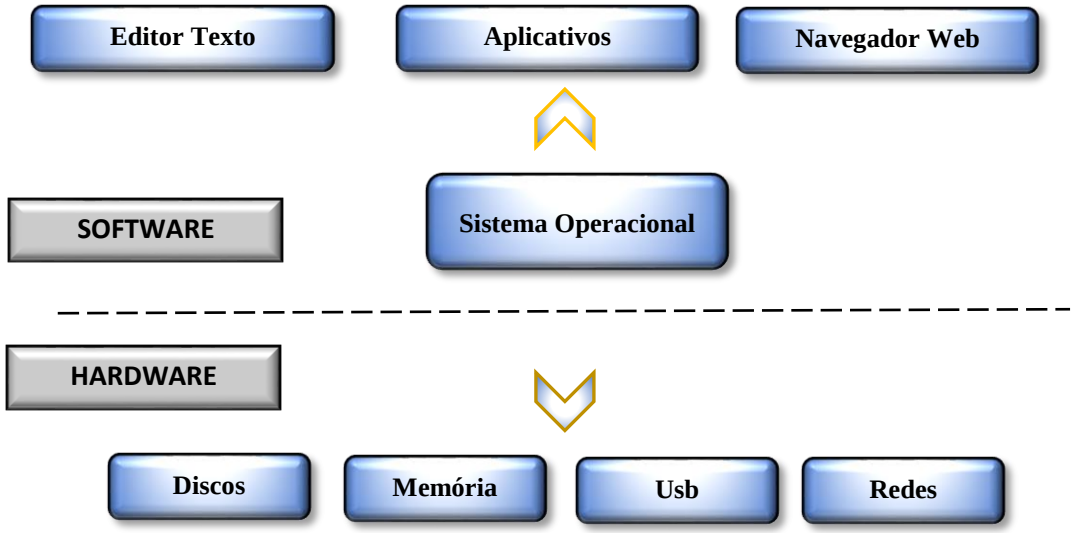


Figura 2 - Estrutura de um sistema de computação típico (fonte: adaptada de Maziero, 2013)

Os programas são executados em um ambiente interno fornecido pelo Sistema Operacional, que pode variar de acordo com suas características, como quantidade de núcleos e registradores dos processadores, modelo de comunicação entre processos e algoritmos de escalonamento. Contudo, há muitos conceitos importantes, um deles é a capacidade da multiprogramação.

2.1.1. Monoprogramação e Multiprogramação

Antes do advento de sistemas multiprogramados, os computadores possuíam Sistemas Operacionais que utilizavam o conceito da monoprogramação. Nesse tipo de sistema, os recursos computacionais, por exemplo, processador, memórias e dispositivos de entrada e saída eram subutilizados, limitando o desempenho desse tipo de arquitetura devido ao sistema comportar a execução de um programa por vez.

Nos sistemas monoprogramáveis, segundo Machado & Maia (2013), somente um programa pode estar em execução por vez, permanecendo o processador dedicado exclusivamente a essa tarefa. O tempo de espera por uma tarefa pode ser relativamente longo, uma vez que as operações com dispositivos de entrada e saída são muito lentas se comparadas com a velocidade do processador. Um aspecto importante é a subutilização da memória principal, quando um programa que não ocupe totalmente a memória ocasiona a existência de áreas livres sem utilização. Por isso, a memória é subutilizada. Uma outra característica é que os programas são executados de maneira sequencial, um espera pela execução do outro para iniciar. Esses problemas são solucionados em sistemas multiprogramáveis.

A multiprogramação está relacionada com a capacidade de várias atividades simultâneas serem executadas ao mesmo tempo. O ambiente interno do Sistema Operacional deve permitir que essa função seja executada de forma eficiente. Para isso, necessita prover uma interface conveniente a seus usuários fornecendo uma alocação de recursos controlada e de forma ordenada de processadores, memórias e dispositivos de entrada e saída entre vários programas que competem por eles (Silberschatz, et al., 2015).

Nos sistemas multiprogramáveis, vários programas podem estar residentes em memória, concorrendo pela ocupação do processador. Sendo assim, quando um programa solicita uma operação de E/S outros programas poderão utilizar o processador, permanecendo a CPU menos ociosa. Então o recurso da memória principal é utilizado de maneira mais eficiente, pois existe vários programas residentes se revezando na utilização do processador, tornando o gerenciamento de recursos como a CPU e a memória mais produtivo e adequado às necessidades da computação.

O compartilhamento de recursos é realizado de acordo com sua forma de gerenciamento. Segundo Tanembaum (2009), por meio de duas maneiras: no tempo, quando diferentes programas ou usuários aguardam sua vez de usar o recurso de forma sequencial; e, no espaço, em vez de usuários ou programas esperarem sua vez, cada um ocupa uma parte do recurso.

O mecanismo que é considerado fundamental para o uso da multiprogramação é conhecido como interrupção.

2.1.2. Interrupção

A maioria dos Sistemas Operacionais modernos e de uso geral é composta por uma ou mais CPUs e vários dispositivos conectados por meio dos barramentos (um conjunto de linhas de comunicação que permitem a interligação entre dispositivos, como a CPU e a memória) que proporcionam acesso à memória compartilhada.

A Figura 3 ilustra a organização de um sistema computacional moderno em que é possível visualizar como é realizada a comunicação entre as partes. Observa-se, por exemplo, como as placas controladoras (disco, usb, controlador gráfico) e a CPU se comunicam com a memória compartilhada por meio de barramentos do sistema.

Para o atendimento de novas solicitações a CPU, elas são enviadas por meio de um modelo de organização computacional. São chamados de eventos e podem atuar de forma síncrona ou assíncrona. Um evento pode ser advindo do usuário ou do próprio sistema. O evento é indicado por uma interrupção (hardware ou de software).

Portanto, a interrupção indica à CPU que naquele momento uma determinada tarefa necessita de atendimento, de uma atenção especial. Segundo Silberschatz et al. (2015), quando uma CPU é interrompida, ela para o que está fazendo e transfere imediatamente a execução para uma localização onde se encontra a rotina de serviço de interrupção que entrará em operação imediatamente.

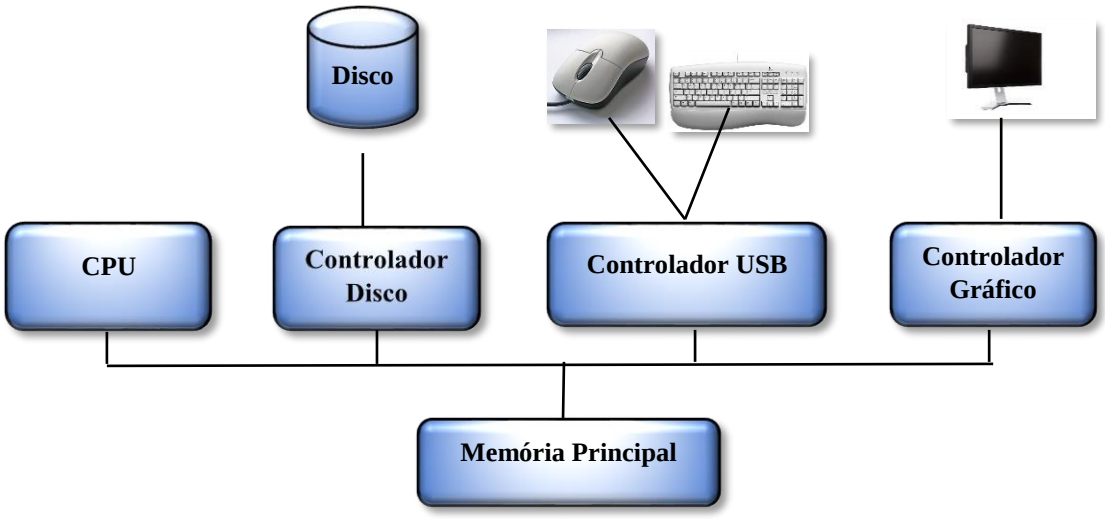


Figura 3 - Organização de um sistema de computador moderno

O sistema de interrupções facilita a execução das rotinas do próprio Sistema Operacional e a dos programas dos usuários por meio da sincronização. Em uma arquitetura de computadores, as interrupções são mecanismos utilizados com frequência.

Entre outros cenários possíveis no simulador desenvolvido nesse trabalho é possível observar o que acontece quando uma interrupção é disparada; por exemplo, quando ocorre uma interrupção de disco, o sistema toma a decisão de parar de executar o processo corrente e retomar o processo do disco que foi bloqueado para aguardar essa interrupção. Finalizada a leitura de disco, o processo que aguarda por isso é desbloqueado e torna-se disponível para ser executado novamente.

2.1.3. Memória

Na organização dos computadores outro tópico importante é a estrutura de armazenamento utilizada nos sistemas computacionais. Segundo Stallings (2010), todos os tipos de memória de um computador fornecem uma estrutura de armazenamentos que possibilita guardar e recuperar instruções quando necessário. Existe uma ampla variedade de sistemas de armazenamento em um sistema de computação que pode ser organizada em uma hierarquia. A Figura 4 demonstra a hierarquia de memórias de acordo com o custo de fabricação, capacidade de armazenamento e velocidade de acesso.

Os níveis mais altos são os mais caros para construir com velocidade de acesso mais rápidas e com pouca capacidade de armazenamento, que é o caso dos registradores. O nível abaixo dos registradores na hierarquia e que possui as mesmas características é a memória cache, que é mais rápida que a memória principal e visa armazenar os dados que estão sendo utilizados mais frequentemente ou que provavelmente serão acessados em breve (Stallings, 2010).

Quando se desce na escala da hierarquia, geralmente o custo de fabricação diminui, a velocidade de acesso fica mais lenta e o armazenamento aumenta. O próximo nível nessa hierarquia é a memória principal (também chamada de memória RAM - *random accesss memory*), que é volátil porque depende da energia elétrica para o seu funcionamento e seu conteúdo pode ser alterado. Porém a velocidade de acesso é menor que a memória cache. E no último nível, destacam-se as memórias secundárias (*hard disks* internos ou externos). Além dessas, outros tipos de memória são utilizados:

memórias ROM (*read only memory*), que o conteúdo não pode ser alterado, como, por exemplo: *EPROM*, *EEPROM* e *FLASH*.

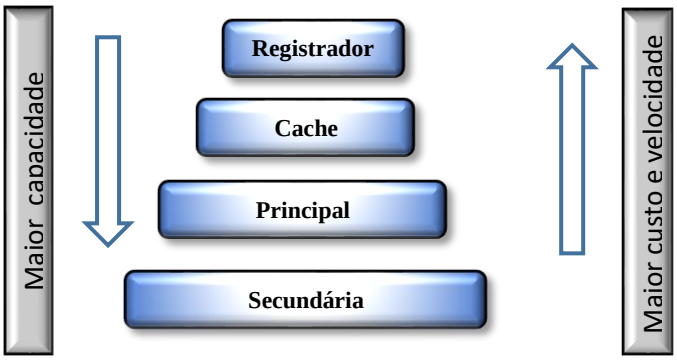


Figura 4 - Hierarquia de memórias (fonte: adaptada de Stallings, 2010)

O conceito de execução dos programas na memória principal por meio de seus processos pode ser visto no simulador ESORV. Nele, os processos são enfileirados na memória principal, que é chamada de fila de prontos, em que aguardam sua vez para ocupar seu tempo de execução na CPU. É possível observar no simulador que um processo pode ocupar memórias diferentes, como a memória principal e os registradores que se encontram na CPU.

2.1.4. Dispositivos de Entrada e Saída

O gerenciamento da estrutura de dispositivos de entrada e saída é uma das principais e mais complexas funções na organização dos computadores. Cuida da confiabilidade e do desempenho do sistema como um todo, uma vez que gerencia as formas de comunicação entre os próprios dispositivos com a CPU e a memória. Nesse cenário, se faz necessária a integração das camadas:

- **Hardware:** o próprio dispositivo de entrada e saída e seu controlador (que são dependentes do dispositivo);
- **Software**: possui as camadas *device driver* (software do dispositivo que depende do hardware) e o subsistema de entrada e saída (isola a complexidade de operações específicas para cada tipo de dispositivo e independe do dispositivo).

Segundo Machado & Maia (2013), a troca de informações e dados entre dispositivos de entrada e saída e a memória ocorre por meio de protocolos, em que três técnicas básicas são utilizadas:

- **Entrada e Saída Programada (*Polling*):** o processador consulta de forma sequencial e repetida todos os periféricos com o objetivo de encontrá-los e saber se estão prontos para enviar ou receber dados. A CPU possui controle total da comunicação, determinando os instantes de tempo que ocorrerão transferências. Para que todas as solicitações geradas pelos periféricos sejam atendidas, o processador realiza consultas com a frequência do dispositivo que possui a maior taxa de latência. Esse método é fácil de implementar, mas implica uma grande dedicação do processador, pois a execução do programa não pode prosseguir enquanto a operação de entrada e saída não terminar, mesmo sendo o processador mais rápido do que os periféricos;
- **Orientada a Interrupção:** quando ocorrer uma interrupção gerada pelo periférico, o processador responde em vez de ficar consultando de forma sequencial e repetida todos os periféricos com o objetivo de encontrá-los e saber se estão prontos para enviar ou receber dados. Para tanto, o processador pode continuar a execução de um programa sem que haja um procedimento de parada de forma desnecessária para verificar se um periférico precisa ou não ser atendido;
- **Acesso Direto à Memória (DMA):** o uso de técnicas como *Polling* ou de Interrupção elevam muito a carga de trabalho do processador. Para resolver esse problema foi desenvolvido um mecanismo que produz o acesso direto. Ele permite transferir dados diretamente da memória ou para a memória sem que efetivamente o processador participe. O mecanismo de interrupção é usado somente para notificar a CPU sobre ocorrência de erros ou informar que a transferência terminou.

Os dispositivos de entrada e saída são retratados no ESORV quando existe a possibilidade de um processo do tipo entrada e saída ocupar a CPU. Quando isso acontece no simulador, o processo em execução é interrompido e seu estado transita de “execução”

para “aguardando”, e ele passa a ocupar a fila de processos chamada de “fila de entrada e saída”, liberando a CPU para o outro processo.

A seção 2.1 explicou, como um todo, os fundamentos básicos e gerais do funcionamento de um Sistema Operacional, como, por exemplo: monoprogramação/ multiprogramação, interrupção, memória e dispositivos de entrada e saída. A próxima seção apresenta a fundamentação sobre processos que foi utilizada para a construção do ESORV.

2.2. Processos

Os sistemas computacionais atuais podem executar várias aplicações ao mesmo tempo e disputar a concorrência de recursos computacionais, como, por exemplo a CPU e a memória. Para isso, um programa ao ser executado deve estar obrigatoriamente associado a um processo. Esta seção apresenta os principais conceitos, definições e mecanismos empregados na gerência de processos.

2.2.1. Conceitos básicos e principais características

O conceito central em qualquer Sistema Operacional é o de processo. Segundo Tanenbaum (2009), um processo é uma abstração de um programa em execução. Silberschatz et al. (2015) descreve que o processo é mais que o código do programa. Ele também inclui uma atividade corrente que está interagindo com o processo naquele momento, representado, por exemplo, pelo contexto de hardware: o valor do contador do programa e o conteúdo dos registradores do processador. Já Machado & Maia (2013) concluem que um processo pode estar integrado ao ambiente no qual o programa é executado. Esse ambiente, além das informações sobre a execução do processo, pode possuir também a quantidade de recursos do sistema que cada programa pode utilizar.

Para o melhor entendimento se faz necessário e conveniente saber diferenciar um programa de sua execução. Para isso é usado o conceito de processo. Um programa é uma sequência de instruções escritas pelo usuário por meio de uma linguagem de programação que tem como objetivo resolver um problema específico. Ele é representado por um conceito estático, não ativo, não altera o seu próprio estado e não executa interações com outras entidades.

Já o processo é a própria execução das instruções de um determinado programa. Para ser um processo, é necessário que o programa ou alguma tarefa seja executado. Assim, de forma ativa e dinâmica, o processo aloca recursos de hardware e software de um ambiente computacional para sua execução. Interage com outras entidades: usuário, periféricos e/ou outros processos. Por isso, o processo é um programa em estado de execução mais as variáveis de hardware e software alocadas.

Um mesmo programa pode ter vários processos no mesmo intervalo de tempo. Um mesmo usuário, por exemplo, pode iniciar diversas abas em um mesmo navegador de internet, sendo que cada aba é um processo independente. A mesma cópia do código da aba do navegador é utilizada por todos os processos, porém existe uma área de variáveis privativa para cada processo.

Nessa ótica dos sistemas multiprogramáveis, a CPU chaveia, muda de programa para programa, executando cada um deles por frações de milissegundos. Enquanto a cada instante a CPU executa somente um programa, em um intervalo de tempo ela pode trabalhar diversos programas, dando aos usuários a ilusão de paralelismo (Tanenbaum, 2009).

Os Sistemas Operacionais evoluíram historicamente para dar suporte ao modelo conceitual de processos sequenciais em único processador. Porém, os processos podem ser executados em mais de um processador, conceito conhecido como paralelismo. Segundo Tanembaum (2009), o paralelismo denominado real só acontece em computadores com multiprocessadores ou núcleos, nos quais existe a possibilidade de processos diferentes serem executados em processadores ou núcleos distintos no mesmo intervalo de tempo. O paralelismo denominado de lógico acontece em computadores com um único processador, no qual os processos são executados de maneira sequencial.

2.2.2. Estados de um Processo

Quando um programa é invocado, o processo vai cumprindo fases diferentes durante seu processamento, moldando-se de acordo com cada situação determinada em função de eventos originados por ele próprio (eventos voluntários) ou pelo próprio Sistema Operacional (eventos involuntários). Cada vez que acontece uma nova fase o processo muda seu estado. Os estados de um processo, segundo Tanembaum (2009),

são:

- **Execução:** instruções do processo estão sendo executadas na CPU;
- **Espera:** o processo está aguardando que algum evento ocorra;
- **Pronto:** o processo está esperando ser atribuído à CPU.

A Figura 5 ilustra os estados de um processo e suas possíveis transições.

Os estados de um processo vão se alternado no curso de seu processamento em cada fase

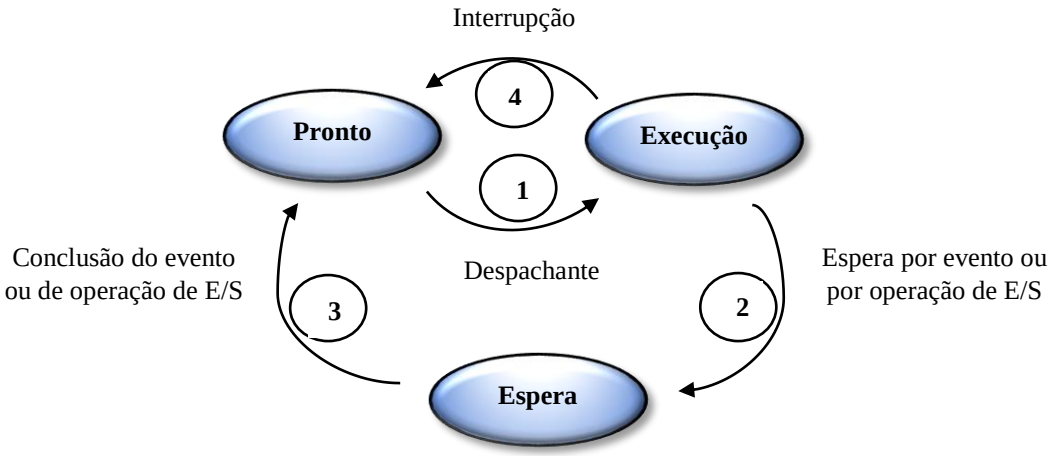


Figura 5 - Estados de um processo e suas transições (fonte: adaptada de Silberschatz et al., 2015)

Existem quatro mudanças de estado de um processo:

1. **Pronto para Execução:** os processos são inseridos na fila de prontos da memória principal após sua criação. Eles serão alocados na CPU de acordo com uma ordem atribuída por um algoritmo. Quando ocupam o processador, o processo sai da fila do estado de pronto e entra em estado de execução na CPU;
2. **Execução para Espera:** o processo passará do estado de execução para o estado de espera devido a eventos gerados pelo próprio processo. Por exemplo, uma operação de entrada e saída ou por solicitação de eventos externos;
3. **Espera para Pronto:** o processo no estado de espera passará para o estado de pronto quando o recurso que o processo está solicitando é concedido, por exemplo, operação de entrada e saída ou um recurso aguardado. É obrigatório

o processo transitar entre o estado de espera para o estado de pronto antes de ser selecionado novamente para o estado de execução;

4. **Execução para Pronto:** um processo em execução passará para o estado de pronto quando um evento for gerado pelo sistema, como, por exemplo, sinalizando o término de *quantum* (fatia de tempo) que o processo possui ou que outro processo prioritário irá ocupar a CPU.

Esse modelo de processos facilita saber o que está ocorrendo dentro do sistema. No ESORV, é possível observar quando os processos transitam em seus estados. Por exemplo, quando um programa é invocado pelo usuário e se torna um processo em estado de pronto ou quando o processo passa do estado de pronto para execução na CPU, ou ainda quando passa do estado de pronto para espera.

2.2.3. Criação, Finalização e Hierarquia de Processos

Diversos mecanismos ou eventos que ocorrem em um Sistema Operacional ou por alguma interação do usuário que poder ser de forma voluntária ou não, podem criar e finalizar processos. Há quatro eventos principais que fazem com que processos sejam criados (Tanembaum, 2009):

1. **Início do Sistema**: quando se carrega o Sistema Operacional, em geral, criam-se diversos processos. Alguns deles são considerados processos de primeiro plano (*foreground*), ou seja, processos que interagem com usuários e realizam tarefas para eles. Outros processos são conhecidos como segundo plano (*background*), e apresentam alguma função específica, por exemplo, pode ser responsável a aceitar mensagens eletrônicas sendo recebidas, ficando inativo na maior parte do tempo, mas tornando-se ativo quando uma mensagem chega;
2. **Execução de uma chamada de sistema de criação de processo por um processo em execução**: muitas vezes, um processo em execução fará chamada de sistema para criar um ou mais novos processos com o intuito de colaborar em seu trabalho. Quando a tarefa a ser executada puder facilmente ser dividida em vários outros processos relacionados, criar novos processos passa a ser mais eficiente, ágil e colaborativo;

3. **Uma requisição do usuário para criar um novo processo**: nos sistemas interativos, os usuários podem inicializar uma aplicação a qualquer momento. A criação inicia um novo processo e executa no seu contexto a aplicação selecionada;
4. **Início de uma tarefa em lote**: aplica-se somente a sistemas em lote localizados em computadores de porte maior. Nesses sistemas, usuários podem submeter tarefas em lote para o sistema. Quando o Sistema Operacional julgar que tem recursos para executar outra tarefa, ele criará um novo processo.

Em todos os casos, um novo processo filho é criado por um processo existente (processo pai). Esse processo pai pode ser um processo de usuário que está executando no momento e executará uma chamada de sistema para criar um novo processo filho e assim determinar qual a aplicação a ser executar nele.

Os términos dos processos, após sua criação, serão finalizados em consequência de algumas condições (Tanembaum, 2009):

1. **Saída Normal (voluntária)**: termina porque seu trabalho foi executado;
2. **Saída por erro (voluntária)**: quando um erro fatal é descoberto por um processo;
3. **Erro Fatal (involuntário)**: o processo causa um erro, que muitas vezes pode ser de um programa, por exemplo, a execução de uma instrução ilegal ou quando ocorre uma divisão por zero;
4. **Cancelamento por outro processo (involuntário):** quando uma chamada de sistema é executada pelo processo informando ao Sistema Operacional para cancelar algum outro processo.

Quando um processo cria outro processo filho, em alguns sistemas, eles ficam associados. O próprio processo filho pode gerar mais processos sucessivamente.

2.2.4. Tipos de Comportamentos de Processos

Segundo Machado & Maia (2013) os processos podem ser classificados de acordo com a sua forma de execução no uso processador e nos dispositivos de entra e saída, como:

- **CPU-*bound* (ligado à CPU):** quando fica a maior parte do tempo no

estado de execução, utilizando intensivamente o processador.

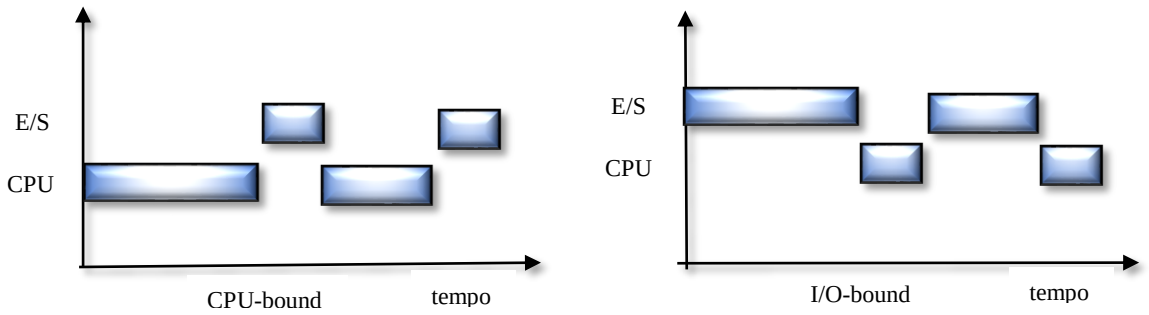
- **I/O-*bound* (ligado à E/S):** quando fica a maior parte do tempo no estado de espera, executando operações de E/S. Dependem muito mais dos dispositivos de entrada e saída que do processador.

A Figura 6 (a) demonstra a ocupação dos processos do tipo CPU *bound*, em que é possível observar o tempo de ocupação do processo na CPU, que é utilizado mais intensamente do que nos dispositivos de E/S. A Figura 6 (b) ilustra o tempo de ocupação dos dispositivos de E/S que é utilizado mais do que a CPU em um determinado intervalo de tempo.

Os conceitos de processo relatados nesta seção foram empregados na construção do ESORV. O usuário escolhe por exemplo, qual tipo de comportamento que o processo terá na simulação, CPU/*bound* ou I/O-*bound*. Além disso, ele permite observar a criação e finalização dos processos dentro do ambiente do simulador.

2.3. Bloco de Controle de Processos – BCP

Na Seção 2.2.1 foi visto que o processo possui um estado interno bem definido, que representa sua situação atual. Esse estado se modifica conforme evolui a execução do processo. O contexto de um processo é o seu estado em um determinado instante, que contém por exemplo, a posição de código que está sendo executada, os valores de suas variáveis e os arquivos utilizados.



a) - Processos CPU-*bound* **b) - Processos I/O-*bound***

Figura 6 - Processos CPU-*bound* x I/O-*bound* (fonte: adaptada de Machado & Maia, 2013)

Segundo Maziero (2013), cada processo no sistema possui um descritor associado, ou seja, uma estrutura de dados que o representa no núcleo. Nessa estrutura são armazenadas as informações relativas ao seu contexto e os demais dados necessários à

sua gerência. E é essa estrutura de dados que geralmente é chamada de BCP (Bloco de Controle de Processos).

Cada processo no Sistema Operacional é representado por um BCP, que tipicamente contém as seguintes informações principais (Silberschatz et al., 2015):

- **Estado do processo:** o estado pode ser pronto, execução e espera (bloqueado);
- **Registradores da CPU:** dependendo da arquitetura do computador, os registradores podem variar. Eles incluem acumuladores, registradores índice, ponteiros de pilha e registradores de uso geral, além de qualquer informação de código de condição;
- **Informações de Escalonamento da CPU:** essas informações incluem a prioridade de um processo, ponteiros de filas de escalonamento ou qualquer outro parâmetro de escalonamento;
- **Informações de Gerenciamento de Memória:** trata dos valores dos registradores base e limite das tabelas de paginação ou segmentação.

2.3.1. Mudança de Contexto

Para que haja uma troca de contexto, ou seja, o processador possa interromper a execução de um processo e retornar a ele mais tarde sem que corrompa seu estado interno, é necessário definir operações para salvar e restaurar o contexto do processo. Essas operações ocorrem com frequência nos Sistemas Operacionais. O ato de salvar os valores do contexto atual em seu BCP e possivelmente restaurar o contexto de outro processo, previamente salvo em outro BCP, é denominado mudança de contexto.

Segundo Silberschatz et al. (2015), o intervalo de mudança do contexto é considerado puro *overhead*[1], uma vez que a implementação da troca de contexto é uma operação delicada, que envolve manipulação de registradores específicos de cada processador, sendo por essa razão geralmente codificada em linguagem de máquina.

Os passos da mudança de contexto são: a interrupção do processo, o salvamento de seu contexto, o escalonamento e a reativação do processo escolhido. Desses passos, o

[1] O termo *overhead* na computação trata-se de qualquer processamento ou armazenamento em excesso, uma sobrecarga no sistema.

que é considerado mais demorado é a execução do escalonamento, que pode ser ativado, por exemplo, quando há necessidade de reordenar a fila de processos prontos que se encontra na memória principal.

A Figura 7 demonstra um diagrama com os passos envolvidos em uma troca de contexto. Inicialmente é possível ver que o Processo *P0* inicia a execução, então uma interrupção é gerada pelo sistema, que pretende colocar o Processo *P1* em estado de execução. Isso obriga o Sistema Operacional a iniciar os passos para a mudança de contexto, salvando no BCP o estado do Processo *P0*. Após a entrada em execução do Processo *P1*, o sistema gera novamente uma interrupção, que pretende trazer de volta a execução do Processo *P0*. Novamente, o Sistema Operacional inicia os passos para a mudança de contexto só que nesse momento, salvando no BCP o estado do Processo *P1* e recuperando o contexto salvo do Processo *P0* para que ele entre em execução.

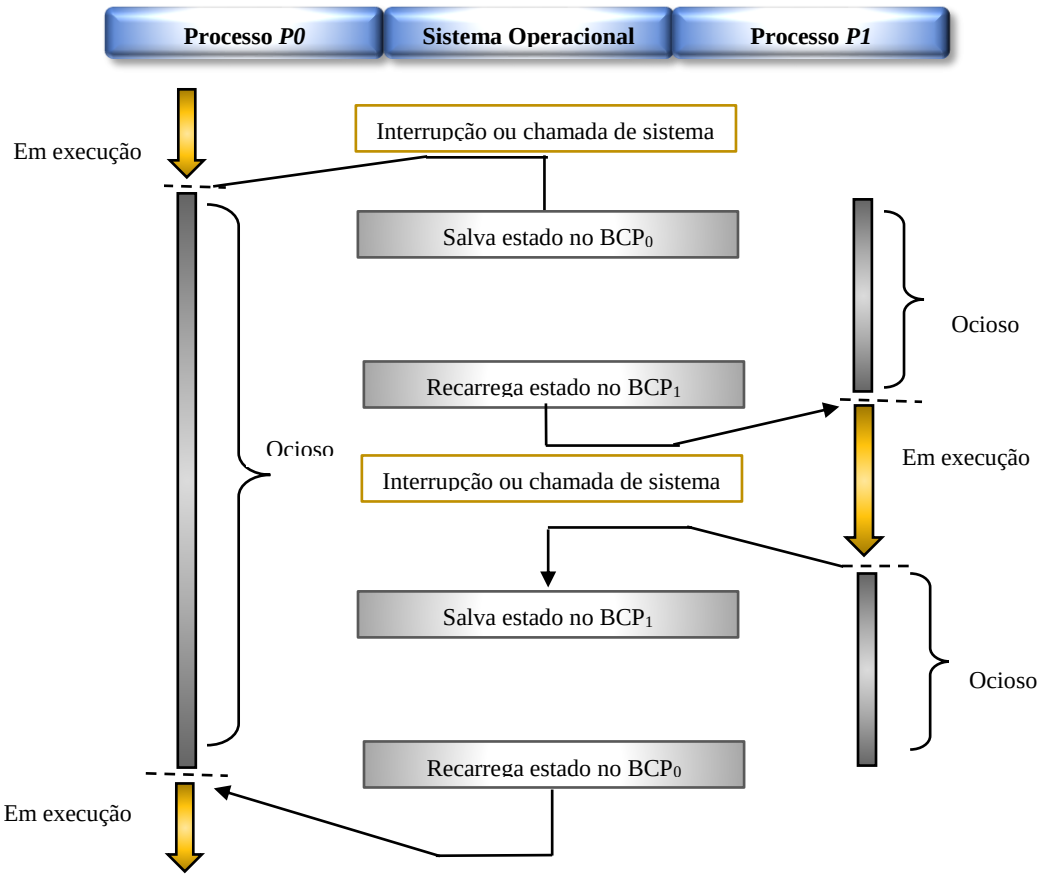


Figura 7 - Passo de uma troca de contexto (fonte: adaptada de Silberschatz et al., 2015)

Um dos pontos de destaque no ESORV é possibilitar que o aluno visualize a funcionalidade e informações do BCP durante a mudança de contexto podendo ver de fato como as interações são feitas e observar o salvamento das informações no BCP.

2.4. Escalonamento

Um dos componentes mais importantes da gerência de processos é o escalonador. (Silberchatz et al., 2015) explica que nos sistemas modernos com a capacidade da multiprogramação, a alternância de tempo entre os processos por meio do compartilhamento de tempo é essencial. Essa alternância é executada com tanta frequência que possibilita que os usuários possam interagir com todos os programas que estão executando (pseudoparalelismo). Os conceitos sobre o escalonamento e os algoritmos utilizados para escalonar são mostrados na próxima seção.

2.4.1. Conceitos e principais características

Um processo disponível é selecionado pelo escalonador de processos do Sistema Operacional a partir de um conjunto de vários processos disponíveis na fila de prontos para ser levado até a CPU para a execução. A ordem de processos a serem executadas é determinada pelo escalonador, sendo que o algoritmo de escalonamento determina o comportamento que o Sistema Operacional terá. Tanembaum (2009) indica que o escalonador também deve se preocupar em fazer o uso eficiente da CPU, uma vez que realizar a mudança de contexto dos processos é um procedimento muito custoso computacionalmente, conforme visto na seção 2.3.1.

Os recursos que necessitam ser escalonados em um sistema computacional de forma geral são: dispositivos de entrada e saída, memória e CPU. O Sistema Operacional necessita de diversas rotinas para a implementação do escalonamento, que podem operar no sistema em conjunto. As principais rotinas relacionadas ao escalonamento são (Machado & Maia, 2013):

- **Escalonador (*scheduler*):** tem como função principal implementar os critérios da política de escalonamento. Principalmente nos sistemas multiprogramáveis, o escalonador se torna fundamental, uma vez que, todo o compartilhamento do processador é dependente dessa rotina;

- **Despachante *(dispatcher)*:** responsável pela mudança de contexto do processo após o escalonador determinar qual processo deve ocupar a CPU. Quanto mais rápido o despachante realizar a mudança de contexto do processo, melhor será o desempenho do sistema, uma vez que esse tempo utilizado é ocioso para o sistema como um todo.

O escalonador poderá ser aplicado a uma ou mais filas de processos prontos para utilizar-se do procedimento de escalonamento. Silberschatz et al. (2015) denomina essa situação de diagrama de enfileiramento. Os processos são representados por meio de filas. Duas filas estarão presentes: a fila de prontos e um conjunto de filas de dispositivos (entrada e saída).

A Figura 8 representa o diagrama de enfileiramento onde são ilustradas as filas de pronto e de entrada e saída. A caixa retangular representa uma fila e os círculos representam os recursos que servem às filas.

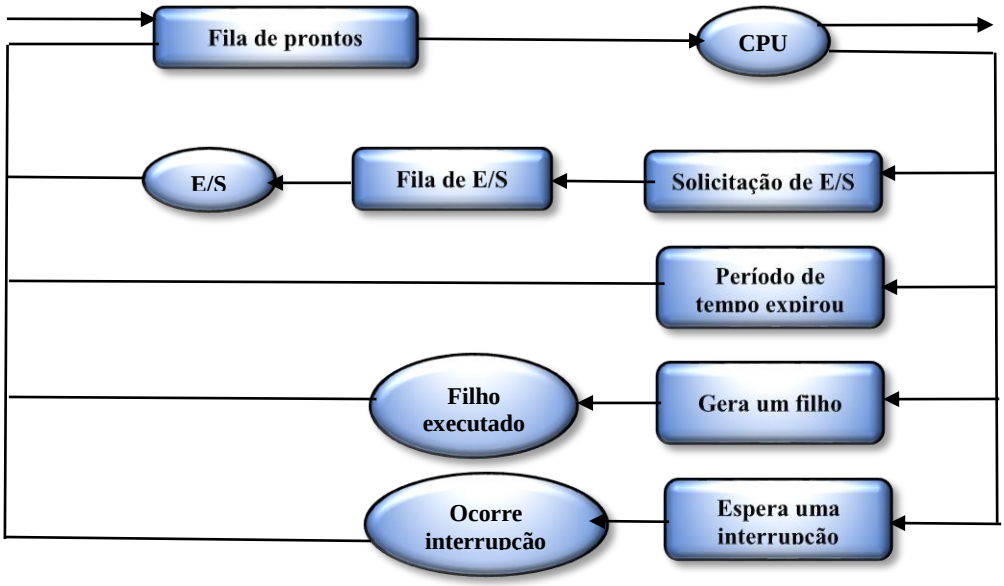


Figura 8 - Representação do processo de escalonamento em diagrama de enfileiramento (fonte: adaptada de Silberschatz et al., 2015)

Essas filas são manipuladas e sempre se alteram conforme o comportamento dos processos. Inicialmente, um novo processo é inserido na fila de prontos. Ele aguarda até ser selecionado para execução pela rotina do escalonador ou ser despachado pela rotina do despachante. Uma vez que o processo é alocado na CPU e entra em execução, um entre vários eventos pode ocorrer Silberschatz et al. (2015):

- O processo pode emitir uma solicitação de E/S e então ser inserido em uma fila E/S;
- O processo pode criar um novo subprocesso e esperá-lo terminar;
- O processo pode ser removido e ser devolvido à fila de prontos.

Nos dois primeiros casos, o processo passa do estado de espera para o estado de pronto e então é devolvido à fila de prontos. Um processo continua esse ciclo até terminar, momento em que é removido de todas as filas e tem seu BCP e seus recursos são desalocados.

Durante sua existência, um processo pode passar por várias filas de escalonamento. De alguma forma, o Sistema Operacional deve selecionar alguns deles. Segundo Silberschatz et al. (2015), existem os seguintes tipos de escalonadores:

- **Escalonamento de longo prazo:** gerencia a criação de processos. Seleciona os processos que estão na memória secundária e que serão levados para a memória principal e os carrega para execução posterior. É executado com menos frequência. Geralmente é o escalonamento que controla o grau de multiprogramação (a quantidade de processos na memória);
- **Escalonamento de médio prazo**: é responsável pelo *swapping* (troca). Ele remove processo da memória e coloca no disco, ou seja, o escalonador de médio prazo é responsável por reduzir o grau de multiprogramação;
- **Escalonamento de curto prazo:** seleciona entre os processos que estão prontos para a execução e aloca a CPU para um deles. É uma rotina escrita em linguagem de máquina que se encontra na memória principal executada pelo despachante. Esse escalonamento ocorre constantemente, com alta frequência, uma vez que o chaveamento do processador entre os processos ativos e a troca de contexto é alto.

Antes de se definir e conhecer os principais algoritmos usados na execução de um escalonador é importante conhecer mais alguns assuntos que contribuem para formar uma visão sobre o escalonamento: a definição de preempção e não preempção, a natureza dos ambientes de execução dos processos, as características desejáveis de um escalonamento e o escalonamento com multiprocessadores.

A definição de processos preemptivos e não preemptivos está relacionada com a possibilidade de o Sistema Operacional se utilizar de uma chamada do sistema por meio de uma interrupção para a troca de processos. Segundo Machado & Maia (2013), as classificações são:

- **Preemptivo:** é quando um processo pode perder a CPU. O processo em execução é interrompido e é passado para o estado de pronto caso ocorra um processo mais prioritário (que estava suspenso aguardando um evento). Com uso da preempção é possível priorizar a execução;
- **Não preemptivo:** nestes sistemas, nenhum evento externo pode promover a perda do uso do processador quando um processo está em execução. O processo sairá do estado de execução somente quando terminar seu processamento ou gerar instruções do próprio código da aplicação que permitam uma mudança para o estado de espera.

A natureza dos ambientes de execução dos processos está relacionada com a forma que um sistema pode ser executado em tipos de Sistemas Operacionais diferentes. Sendo assim, cada escalonador deve ser otimizado conforme o ambiente que será executado. Três ambientes merecem distinção de acordo com Tanembaum (2009):

- **Lote:** não existe um tempo de resposta rígido do sistema para o usuário. Possui pouca interação com o usuário e geralmente é executado de tempos em tempos, como procedimentos de cálculo mensal de uma empresa;
- **Interativo:** a preempção visa evitar que um processo monopolize a CPU, e com isso negue serviço aos outros processos. Possui uma interação do sistema com o usuário com respostas mais rápidas em relação ao ambiente por lote de tempo do sistema para o usuário. Estes ambientes incluem a maior parte das aplicações dos usuários comuns que utilizam editores de texto, navegadores, internet e e-mail;
- **Tempo real:** os processos destes escalonadores devem respeitar o limite de tempo de finalização estabelecido. Então, eles exigem previsibilidade em seu tempo de resposta aos eventos, uma vez que, geralmente, estão associados aos controles de sistemas críticos, como: processos industriais, automobilísticos e usinas nucleares.

As características desejáveis de um escalonador estão relacionadas aos objetivos para projetar um algoritmo de escalonamento. É necessário se ter uma concepção do que um bom algoritmo deve fazer. Alguns objetivos, segundo Tanembaum (2009), dependem da natureza do ambiente (lote, interativo, tempo real). Os principais objetivos estão relacionados na Tabela 1.

Tabela 1: Principais Objetivos do Escalonamento

Todos os Sistemas
Justiça – dar a cada processo uma porção justa da CPU
Aplicação da política – verificar se a política estabelecida é cumprida
Equilíbrio – manter ocupadas todas as partes do sistema
Sistemas em Lote
Vazão (*throughput*) – maximizar o número de tarefas por hora
Tempo de retorno (*turnaround)* – minimizar o tempo entre a submissão e o término
Utilização de CPU – manter a CPU ocupada o tempo todo
Sistemas Interativos
Tempo de Resposta – responder rapidamente às requisições
Proporcionalidade – satisfazer às expectativas dos usuários
Sistema de Tempo Real
Cumprimento dos prazos – evitar a perda de dados
Previsibilidade – evitar a degradação da qualidade em sistemas multimídia

Outro ponto importante em relação ao escalonamento é o referente ao multiprocessamento, que é a situação na qual o sistema possui mais de uma CPU e, então, os processos podem ser executados em paralelo**.** O ideal para essa situação seria executar um balanceamento de carga, no entanto o escalonamento passa a ter um nível de dificuldade maior quanto a sua funcionalidade.

Segundo Silberschatz et al. (2015), as abordagens para o escalonamento com multiprocessadores podem ser:

- **Multiprocessamento Assimétrico:** todas as decisões de escalonamento

(processamento, dispositivos de entrada e saída e outras atividades) são manipuladas por um único processador chamado de mestre. Os outros processadores executam apenas códigos de usuário. Esse modelo reduz a necessidade de compartilhamento de dados porque só um processador acessa a estrutura de dados do sistema;

- **Multiprocessamento Simétrico (SMP):** cada processador faz seu próprio escalonamento. Todos os processos podem ficar na mesma fila de prontos ou cada processador pode ter sua própria fila de processos prontos. Um exemplo típico de utilização desse modelo de escalonamento é encontrado nos processadores *multicore* (diversos núcleos de processadores no mesmo circuito físico).

2.4.2. Algoritmos de Escalonamento

O escalonamento da CPU lida com a decisão da escolha de qual dos processos da fila de prontos da memória principal será alocado a CPU. Para tanto, há necessidade de empregar algoritmos que tenham essa finalidade nos Sistemas Operacionais. Há diversos algoritmos de escalonamento. Os principais são descritos nessa seção e foram utilizados na implementação do ESORV.

2.4.2.1. Escalonamento FCFS (First Come – First Served)

Este tipo de algoritmo consiste em alocar os processos da memória principal para a CPU de forma sequencial em uma única fila. A CPU executa os processos de acordo com a fila de prontos e conforme os processos chegam à sequência. É um algoritmo não preemptivo (Tanembaum, 2009).

O FCFS é um algoritmo simples de escalonar, e também é chamado de FIFO (*first in – first out*). Ele pode ocasionar um tempo de espera alto, pois irá depender do tempo de execução do processo na CPU e assim, processos pequenos podem esperar por muito tempo atrás de processos longos. Para o entendimento do algoritmo FCFS, considere os seguintes processos na fila de prontos P_1, P_2, P_3 com suas durações previstas de processamento e chegadas ao sistema descritas na Tabela 2.

Tabela 2: Descrição de duração e chegada dos processos para FCFS

Processos	Duração	Chegada
P_1	23	0
P_2	4	2
P_3	3	1

O diagrama da Figura 9 mostra o escalonamento do processador usando o algoritmo FCFS. Os quadros coloridos representam o uso do processador (observe que em cada instante apenas um processo ocupa o processador). Assim, o processo P_3 será iniciado somente no tempo 27.

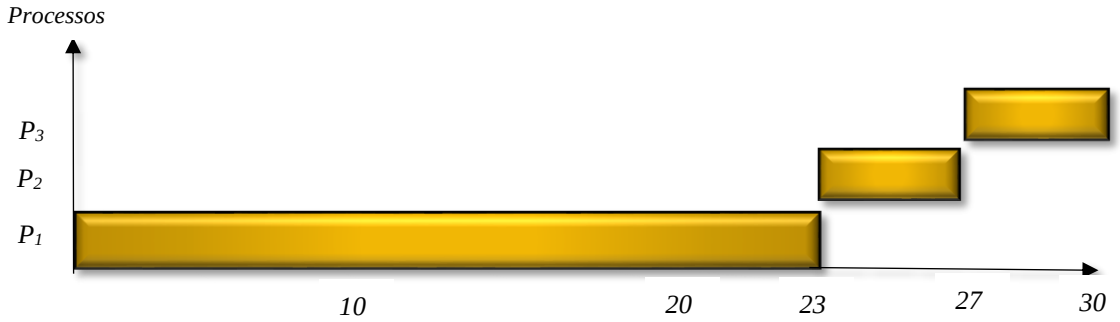


Figura 9 - Escalonamento *FCFS*

2.4.2.2. Escalonamento Menor Job Primeiro (SJF – *Shortest Job First*)

O algoritmo *SJF* leva em consideração o processo que tiver o menor tempo de processador ainda por executar. Obtêm-se as durações de tempo de execução dos processos anteriores. Assim, é selecionado o processo em estado de pronto que necessitar de menos tempo de CPU para terminar sua execução (Machado & Maia, 2013).

Ele é um algoritmo não preemptivo. Aplicando-se ao SJF os processos da Tabela 2, obtém-se o escalonamento apresentado na Figura 10. Nela, é possível observar o escalonamento do processador com o algoritmo *SJF*. Os quadros coloridos demonstram o uso do processador pelos processos de menor duração. Assim o processo P_3 será iniciado somente no tempo 0.

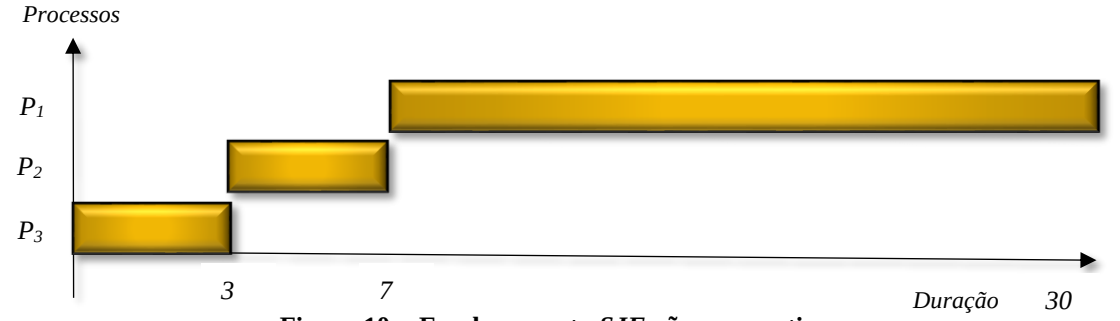


Figura 10 - Escalonamento *SJF* não preemptivo

Existe a possibilidade desse algoritmo utilizar-se da preempção. O algoritmo é denominado por alguns autores como menor tempo restante primeiro - SRTF (*Short Remaining Time First*). Utilizando-o, o escalonador compara a duração prevista de cada novo processo que ingressará no sistema com o tempo restante de processamento dos demais processos, até mesmo aquele que está sendo executado no momento (Maziero, 2013). A Figura 11 ilustra a versão do SJF preemptivo aplicado aos valores da Tabela 2. Os quadros coloridos demonstram o uso do processador pelos processos e representa a troca na CPU toda vez que surgem processos mais curtos. Assim o processo P_3 será iniciado somente no tempo 0, irá para o estado de pronto no momento 2 e retornará para a execução no momento 5.

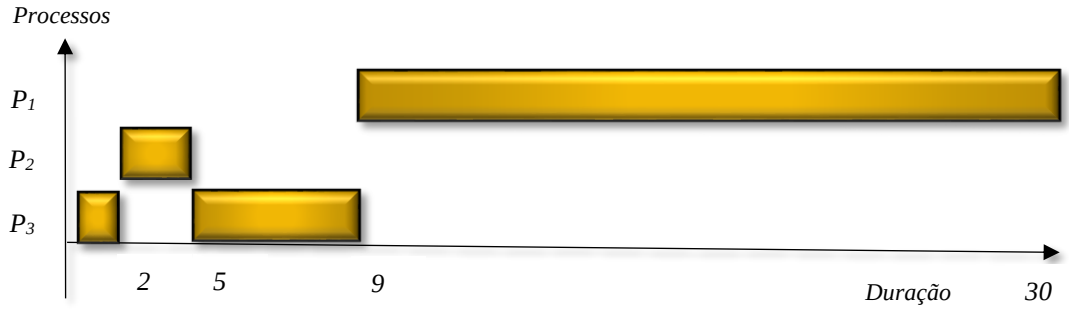


Figura 11 - Escalonamento *SJF* preemptivo

Um problema associado ao escalonamento SJF é a possibilidade de ocorrer a inanição (*starvation*) dos processos mais longos, uma vez que o algoritmo SJF não preemptivo é considerado um exemplo de escolha prioritária. Caso o fluxo de processos curtos chegando ao sistema seja elevado, as tarefas com duração de tempos maiores nunca serão escolhidas para receber o processador e provavelmente não irão ocupar a CPU. Esse problema pode ser resolvido com a técnica conhecida como *aging* (envelhecimento). Esse mecanismo incrementa gradualmente a prioridade dos processos que estão à espera na fila de prontos por muito tempo. Se esse aumento fizer com que o processo tenha maior prioridade sobre os outros, ele deverá ganhar a CPU, resolvendo o problema da inanição.

2.4.2.3. Escalonamento Round Robin (RR)

Escalonamento Round Robin é o algoritmo de tipo preemptivo, conhecido também como escalonamento circular (*Round Robin Scheduling*), e é parecido com o FCFS. Porém, possui o conceito de preempção para permitir alternância entre os processos quando alocam a CPU. Isso é possível atribuindo uma pequena unidade de tempo chamada de

quantum para cada processo. Após cumprir o seu término do *quantum* na CPU, o processo é removido da CPU e colocado no final da fila de prontos, assim permitindo que outro processo a ocupe (Silberschatz, et al., 2015).

O *quantum* atribuído a cada processo afeta diretamente o desempenho do sistema. Se o quantum é grande tende a se comportar de forma semelhante ao FCFS, pois o processo ocupará um grande tempo de processador. Se o quantum é pequeno, então ocorrerá muito *overhead* devido às muitas mudanças de contexto. Portanto, o *quantum* deve ser pequeno o suficiente para garantir o tempo compartilhado e deve ser grande o bastante para compensar as trocas de contexto.

Geralmente o tempo médio de espera na política *RR* é longo. Considerando os processos definidos na Tabela 3 e um *quantum* de 2 milissegundos, a Figura 12 demonstra a sequência de escalonamento obtida desconsiderando a mudança de contexto. Os quadros coloridos demonstram o uso do processador pelos processos com o *RR*.

O principal benefício do algoritmo *RR* é permitir que um processo não monopolize a CPU, ocupando-a por tempo indeterminado. Sendo assim, o *quantum* do processo alocado na CPU é definido pelo sistema e distribuído igualmente entre os processos. É um algoritmo considerado justo com a alocação dos processos na CPU, é ideal para os sistemas de tempo compartilhado.

Tabela 3: Descrição de duração e chegada dos processos para Round Robin

Processos	Duração	Chegada
P_1	23	0
P_2	4	2
P_3	3	1

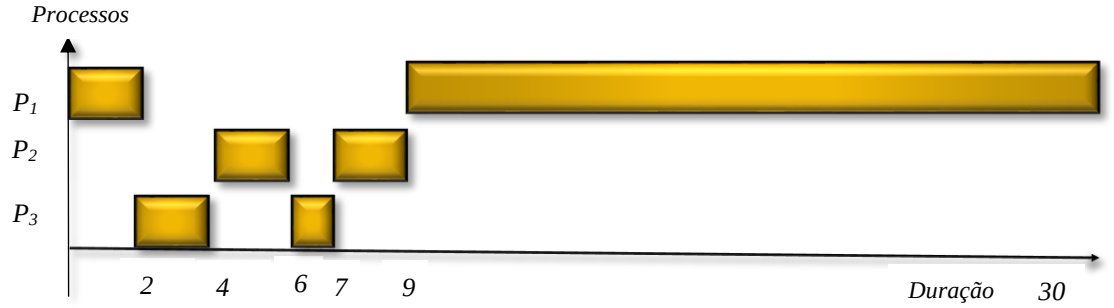


Figura 12 – Escalonamento *Round Robin*

2.4.2.4. Escalonamento por Prioridades (PR)

Escalonamento por Prioridades é um algoritmo de tipo preemptivo no qual um valor predeterminado é associado a cada um dos processos. Esse valor é a prioridade de execução. O processo que possui maior prioridade no estado de pronto é sempre o escolhido para a execução, e os processos com valores iguais são escalonados seguindo o critério FCFS (Silberschatz, et al., 2015).

As maneiras de determinar a prioridade de execução podem ser (Maziero, 2013):

- **Interna:** usa alguma medida para atribuir a prioridade, por exemplo, tempo de execução do processo e limites de memória;
- **Externa:** usa informações oriundas do usuário em geral, definido por algum critério externo ao Sistema Operacional, por exemplo, tipo de processo e custo do processo.

Nesse tipo de algoritmo só ocorrerá a perda do processador no caso de uma mudança voluntária do usuário ou do sistema, passando o estado do processo em execução para espera ou quando um processo de maior prioridade estiver no estado de pronto, forçando a interrupção do corrente processo e entregando a CPU ao outro processo com maior prioridade. Nesse caso o Sistema Operacional salva o BCP e coloca-o em estado de pronto isso ocorre por meio de uma interrupção de *clock.*

Para ilustrar o funcionamento do escalonamento por prioridades, considere os processos, suas durações e as suas prioridades descritas na Tabela 4:

Tabela 4: Descrição de duração e chegada dos processos para Prioridades

Processos	Duração	Prioridade
P_1	23	2
P_2	4	1
P_3	3	3

O diagrama da Figura 13, mostra o escalonamento do processador usando o algoritmo por prioridades. Os quadros coloridos demonstram o uso do processador pelos processos por meio da prioridade. Assim, respeitando a prioridade o P_3 é o último a ser escalonado.

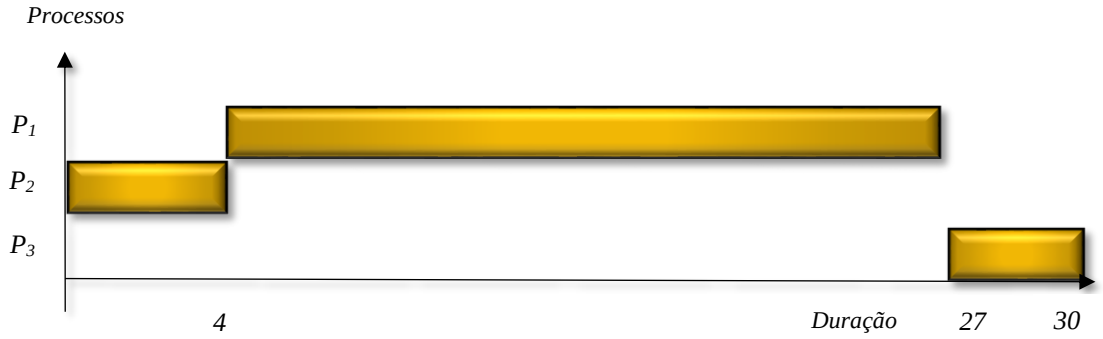


Figura 13 - Escalonamento por Prioridades

Um dos principais problemas no escalonamento por prioridades é conhecido como inanição (*starvation*). Aqueles processos com prioridade baixa podem não ser escolhidos para o escalonamento, permanecendo indefinidamente na fila de prontos (Silberschatz et al., 2015). Os processos de alta prioridade tendem a monopolizar a CPU. Esse problema pode ser resolvido, utilizando-se da técnica de envelhecimento (*aging*.)

2.4.2.5. Escalonamento de Múltiplas Filas (MLQ)

Neste tipo de escalonamento o ponto principal é dividir os processos no estado de pronto em diversas filas distintas. Para cada fila é atribuída um valor que é uma prioridade específica em função de características próprias, como, por exemplo a importância que os processos da fila têm para a aplicação e o tipo de processamento de cada processo que esta na fila.

Segundo Silberchatz et al. (2015), a prioridade associada a cada fila, é a que estabelece uma relação de prioridade entre elas e o sistema. A partir disso, somente escala processos de uma determinada fila quando todas as outras de maior prioridade estiverem vazias. Um comportamento importante para este tipo de algoritmo é que o processo nunca muda de fila.

A Figura 14 mostra os processos organizados em filas separadas com prioridades fixas por fila. Cada fila é separada por uma determinada característica que se encontra em ambiente de processos específicos. É possível observar na figura 14 que as características que determinam a separação das filas são: processos de sistemas; processos de interação; processos de edição; processos em lotes e processos de estudantes. O escalonamento

ocorre pela prioridade e somente mudará de fila quando todos os processos desta fila forem escalonados.

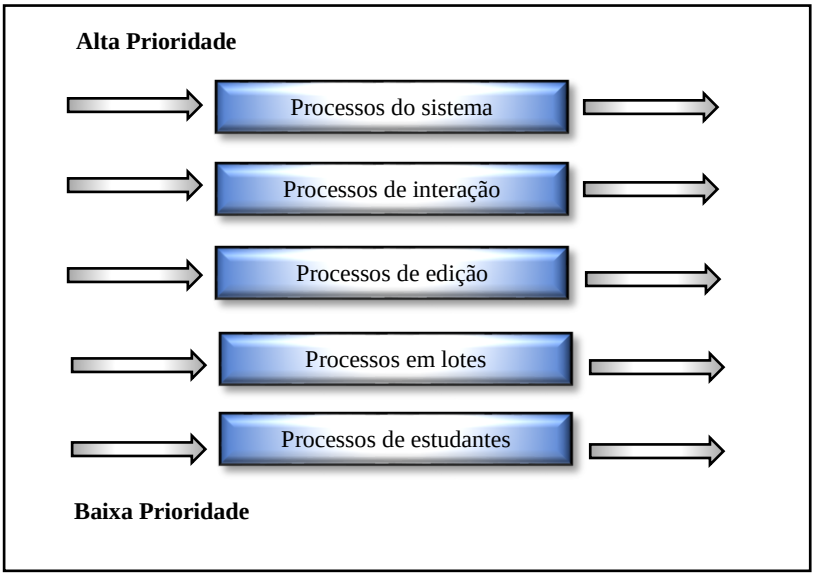


Figura 14 - Escalonamento com múltiplas filas (fonte: Silberschatz et al., 2015)

2.4.2.6. Escalonamento de Múltiplas Filas com Realimentação (MLFQ)

Este tipo de algoritmo é bem semelhante ao escalonamento de múltiplas filas, porém os processos podem trocar de fila. Esse é o ponto principal desse tipo de escalonamento: a troca de filas pelos processos. O algoritmo permite separar processos com características semelhantes, identificando dinamicamente o comportamento de cada processo e direcionando-o para uma das filas (Tanembaum, 2009).

Os processos não são previamente associados às filas de pronto, e sim direcionados pelo escalonador para as filas existentes conforme o seu comportamento. Assim, os processos são selecionados de acordo com afinidade do comportamento de cada um para compor a fila.

Seguindo o exemplo de Silberschatz et al. (2015), a Figura 15 destaca as filas diferentes em que o processo poderá transitar. As três filas com características de execução diferentes: F0 (quantum de tempo 8 milissegundos); F1 (quantum de 16 milissegundos); e F3 (escalonamento FCFS). Supondo que uma nova tarefa entra na fila F0, que é atendida com base no escalonamento *Round Robin* quando ganha a CPU, o processo recebe 8 milissegundos. Senão terminar nesse tempo, o processo é movido para

a fila F1. Em F1 o processo é atendido novamente em *Round Robin* e receberá 16 milissegundos adicionais. Se ainda não estiver completa a fila, o processo é movido para a fila F2.

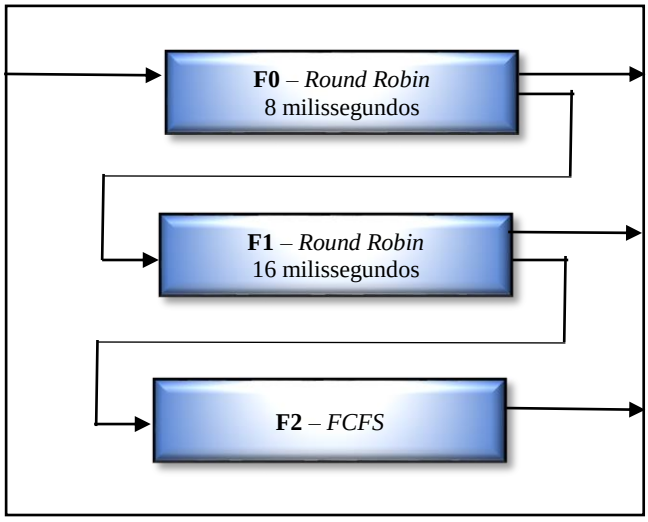


Figura 15 - Escalonamento com múltiplas filas com realimentação (fonte: adaptada de Silberschatz et al., 2015)

O escalonamento é um dos assuntos principais tratados neste trabalho, assim foi diretriz para a construção do ESORV. O simulador utiliza os algoritmos de escalonamento, que é escolhido pelo usuário no início do uso dele. Além disso, o aluno também configura outros parâmetros, como, por exemplo, o tempo de execução de CPU. O aluno/professor pode configurar os parâmetros de simulação em qualquer momento.

3. REALIDADE VIRTUAL

As definições sobre realidade virtual, de forma geral, procuram fazer referência a uma imersiva, interativa e envolvente experiência que se baseia em imagens gráficas tridimensionais, passando a gerar ambientes interativos e em tempo real (Tori & Kirner, 2006). Este capítulo apresentará os assuntos e temas alinhados com a realidade virtual contemplando a fundamentação teórica utilizada nesse trabalho.

A seção 3.1 descreve sobre as definições e conceitos de realidade virtual. A seção 3.2 apresenta as definições sobre imersão, interação e envolvimento. A seção 3.3 apresenta a realidade virtual imersiva e não imersiva. A seção 3.4 mostra os componentes de um sistema de realidade virtual: ambiente virtual e computacional. A seção 3.5 demonstra a utilização de exemplos de projetos que utilizam a realidade virtual em diversas áreas da educação.

3.1. Definição da Realidade Virtual

O termo realidade virtual foi inventado no final da década de 1980 por Jaron Lanier, que criou um simulador que foi compartilhado com vários usuários simultaneamente. A partir deste fato, o termo realidade virtual foi designado para diferenciar simulações feitas por uma máquina de simulações, as quais envolvem vários usuários em um mesmo ambiente. Segundo Pimentel (1995), um ponto importante na realidade virtual é o convencimento do usuário de que ele está em outra realidade, provocado pelo seu envolvimento no uso da tecnologia.

Segundo Tori & Kirner (2006), a realidade virtual tem como características a visualização, a movimentação e a interação do usuário com o ambiente tridimensional que está utilizando em tempo real. Assim, as aplicações de realidade virtual são consideradas interfaces avançadas ou estendidas dos usuários para acessar os ambientes simulados.

Um ponto importante e comum na realidade virtual é a possibilidade de o usuário interagir com o ambiente virtual em tempo real, podendo estimular e ser estimulado a realizar uma ação por meio da interação com as cenas reais. Segundo Bowman et al. (2005), isso está relacionado com a reação do usuário quando o computador e o sistema

têm a capacidade de detectar tal reação. Essa interação possibilita promover alterações na aplicação. Assim, o usuário poderá participar mais efetivamente, explorando e construindo um conhecimento com a aplicação.

Podem existir diversas formas possíveis de interação em ambiente virtuais por meio de seus dispositivos de entrada, que são usados para representar ações do usuário direto para o sistema, permitindo a manipulação de objetos. Segundo Tori & Kirner (2006), a interação mais simples é a navegação, decorrente da movimentação do usuário no seu espaço. Não há mudanças no ambiente virtual, o que fica caracterizada é a imersão na aplicação, como se o usuário estivesse participando do contexto real por meio das interfaces.

Outro quesito a considerar em sistemas de realidade virtual é a interface, pois pode envolver um nível de interação avançado por meio do controle tridimensional que ela pode possuir. O usuário se sente inserido no contexto virtual da aplicação, conseguindo manipular, visualizar e explorar os dados da aplicação em tempo de execução, usando seus sentidos, particularmente os movimentos tridimensionais do corpo.

Segundo Kirner & Siscouto (2008), os dispositivos que promovem esse tipo de interação podem ser classificados em Convencionais - são os periféricos comuns de sistemas computacionais de propósito geral, como, por exemplo, mouse, monitor e teclado. Não convencionais - são periféricos que não são utilizados comumente na maioria das aplicações, como, por exemplo, capacetes de visualização e luvas. Esses dispositivos devem ser implementados sob interfaces intuitivas, que permitem levar o usuário à cena da aplicação, de maneira imersiva e dando a sensação de realismo.

3.2. Imersão, Interação e Envolvimento

Pode-se afirmar que a realidade virtual pode ser caracterizada ainda pela integração de três ideias básicas: imersão, interação e envolvimento. A concepção da imersão está alinhada com a ideia de mostrar que o usuário, quando participa de um ambiente imersivo de realidade virtual, tem a sensação de estar dentro do ambiente, explorando e participando, e não sendo somente um observador. De acordo com Cruz-Neira (1992), isso pode se dar por meio da utilização dos dispositivos não convencionais, como, por exemplo, Cavernas Digitais, que são sistemas imersivos baseados nas

projeções em salas, nas paredes e tetos. Uma mini Caverna Digital (Dias et al., 2012) foi utilizada para simulações no ESORV.

A interação está associada à capacidade de o computador detectar entradas do usuário e modificar em tempo real o mundo virtual e as ações sobre ele. Em geral, o usuário realiza a interação para explorar e observar as cenas mudarem. Para parecer mais realista o ambiente virtual inclui objetos simulados e existe também a inserção de sons nos ambientes e sons associados a objetos específicos (Araújo, 1996).

A perspectiva de envolvimento, por sua vez, está associada ao grau de estimulação para o comprometimento de uma pessoa com determinada atividade, podendo ser ativo (participar de um jogo, visualizar um ambiente virtual) ou passivo (ler um livro, participar de uma cirurgia virtual)

3.3. Realidade Virtual Imersiva e Não Imersiva

Segundo Kirner (2011), o usuário percebe o mundo virtual por meio da interface que é construída em uma aplicação e pode ser explorada por uma tela de um monitor, ilustrado na Figura 16(a), ou de projeção, inserindo o usuário no mundo virtual com a utilização de dispositivos não convencionais, por exemplo o capacete (HMD), na Figura 16(b), ou ainda salas com multiprojeções (Cavernas Digitais). Esse trabalho utilizou a realidade virtual imersiva. O usuário tem a sensação do realismo como se estivesse dentro, imerso no ambiente virtual, podendo interagir com objetos como se fossem reais, manipulando-os.

Na realidade virtual não imersiva o usuário é transportado parcialmente para o domínio da aplicação, preservando seu senso de presença no mundo real. O usuário manipula o ambiente virtual, porém não tem a mesma sensação de fazer parte da simulação, de estar dentro do ambiente.

3.4. Sistemas de Realidade Virtual

Segundo Vince (2004), a análise de um sistema de realidade virtual deve levar em consideração alguns quesitos considerados importantes para sua definição, construção e funcionamento, que são: o ambiente virtual; o ambiente computacional; a tecnologia de realidade virtual; e as formas de interação.

a) Realidade Virtual explorada por uma tela de monitor **b) Realidade Virtual explorada por um capacete HMD**

Figura 16 - Dispositivos de exploração de Realidade Virtual (fonte: Tori & Kirner, 2006)

3.4.1. Ambiente Virtual

Em geral, um ambiente virtual pode conter referências do mundo real com vários atributos associados, como formas diferentes, simulando cenas, objetos, figuras e formatos, restrições físicas e atributos acústicos (Tori & Kirner, 2006). Dá-se a esses atributos, cores, texturas, iluminação, posicionamento e demais necessidades. Contudo, há situações que o ambiente virtual não precisa ter referências ao mundo real, constituindo de um modelo abstrato.

Os elementos utilizados para construir os ambientes virtuais utilizam técnicas para dar a sensação de realismo. Existem casos em que o grau de realismo ocorre por meio da utilização de modelo da simulação do comportamento de um determinado objeto por meio de uso de animações. O modelo utilizado na composição do ESORV usa do recurso de animação de objetos.

3.4.2. Ambiente Computacional

O ambiente computacional envolve aspectos relacionados ao hardware e o sistema como um todo, por exemplo: processador, sistemas operacionais e mecanismos de entrada e saída. Vale lembrar que a plataforma de utilização de modelos de computadores pode ser heterogênea. Variando de dispositivos móveis até sistemas multiprocessados, o que pode influir diretamente no ambiente computacional (Tori & Kirner, 2006).

Como em qualquer tipo de sistema de realidade virtual, os serviços de tempo real são fundamentais para produzir tempo de respostas mínimos. O ponto central do ambiente

computacional passa a ser a capacidade de suportar e coordenar a interação de entrada e saída dos estímulos dos usuários em tempo real com uma perda aceitável (Tori & Kirner, 2006). Por exemplo, um sistema com *estereoscopia*[2] necessita de no mínimo 60 quadros por segundo, caso contrário, efeitos na imagem não serão perceptíveis pelos usuários.

O importante será que, de qualquer forma, todas as tarefas deverão funcionar integradas e com o tempo de resposta suficiente para garantir a imersão, a interação e o envolvimento do usuário.

3.4.3. Hardware e Software

O ambiente de hardware de interação de realidade virtual é caracterizado por uma vasta variedade de dispositivos de entrada, como, por exemplo: rastreadores, mouses 3D, teclado, luvas, sensores, sintetizadores e reconhecedores de voz. Além disso, geralmente, o ambiente utiliza para processamento placas gráficas específicas. Os computadores utilizados dependem do propósito da aplicação. Geralmente, as aplicações são imersivas, então a solução utilizada pode variar desde um dispositivo móvel até um supercomputador.

Já os softwares voltados para sistemas de realidade virtual tratam desde a gerência da simulação até a construção dos objetos tridimensionais. Esses softwares usam geralmente padrões específicos, como: *VRML*[3] e *X3D*[4]. Outros são bibliotecas gráficas, como *OpenGL*[5], *OpenSG*[6], *Ogre*[7] e *Unity Unreal*[8], e outros ainda, que são ferramentais de modelagem, como o *Blender* [9]e *Autodesk Maya*[10].

3.5. O Ensino-Aprendizado pela Realidade Virtual

A realidade virtual baseia-se no uso de interfaces computacionais interativas com o

---

2 estereoscopia é uma técnica usada para se obter informações do espaço tridimensional, através da análise de duas imagens obtidas em pontos diferentes.
[3] *VRML* (*Virtual Reality Modeling Language*) é uma linguagem que serve para descrever objetos e mundos 3D interativos por meio da web.
[4] http://www.web3d.org/x3d/what-x3d
[5] https://www.opengl.org/
[6] http://www.opensg.org/
[7] http://www.ogre3d.org/
[8] https://unity3d.com/
[9] https://www.blender.org/
[10] http://www.autodesk.com.br/

objetivo de criar no usuário uma sensação de realidade. Assim, o usuário pode se envolver de forma exploratória para que ganhe conhecimento por meio do próprio ambiente de estudo, e de forma interativa para que aprenda sobre um determinado assunto a partir de sua imersão e participação. Desse modo, a aprendizagem realizada por meio da vivência pessoal do participante pode estimular o aluno a aprender e, consequentemente, constituir uma experiência pessoal, que poderá facilitar a aprendizagem (Guimarães & Martins, 2012).

Segundo Clark (2006), alguns benefícios podem ser observados com o uso de realidade virtual na educação, que são:

- Tornar o aprendizado mais interessante e divertido com o objetivo de melhorar a motivação e a atenção;
- Reduzir custos quando a utilização do objeto e do ambiente real for mais dispendiosa que a simulação;
- Possibilitar que se façam coisas que são impossíveis de serem feitas no mundo real;
- Integrar habilidades e conhecimento;
- Aumentar a retenção por meio do realismo;
- Melhorar a transferência de aprendizagem para o mundo real;
- Acessar o conteúdo de aprendizagem em qualquer lugar e em qualquer tempo.

Nos últimos anos ocorreu um aumento do uso das aplicações de realidade virtual na educação. Isso se deve, principalmente ao surgimento de novas ferramentas de software. Contudo, segundo Guimaraes & Martins (2012), o uso efetivo de ambiente de realidade virtual no contexto educacional é ainda limitado a apenas algumas instituições. Isso ocorre devido a motivos como: a necessidade de equipamentos, de professores treinados, de aplicações de realidade virtual para determinados conteúdos e de pessoal de apoio. A incorporação dos ambientes de realidade virtual no meio educacional somente acontecerá quando essas barreiras forem superadas. Além disso, é fundamental que estejam atrelados a um propósito educacional bem definido. Somente assim essa tecnologia poderá agregar

valores ao processo de ensino aprendizagem.

3.5.1. Aplicabilidade do Ensino-Aprendizado pela Realidade Virtual

Existem na literatura exemplos de uso de ambientes de realidade virtual em diversas áreas do conhecimento e em variados níveis de instrução, como, por exemplo, em simulação de processos químicos e físicos, e na aprendizagem de pilotagem de aviões. Segundo Dede (2009), esses ambientes apresentam indícios de contribuição na melhoria e aprimoramento do processo ensino-aprendizagem.

A seção seguinte apresenta o resultado de uma revisão da literatura sobre ambientes virtuais imersivos voltados ao ensino (Scamati et al., 2015).

3.5.2. Trabalhos de Realidade Virtual na Área de Educação

Em 2013, Fengru et al. (2013) apresentaram um estudo sobre a construção e visualização de ambientes geológicos complexos baseados no VGE (ambiente virtual geográfico). Esse ambiente simulado permite que os alunos interajam com o ambiente virtual geológico (Figura 17). Essa solução foi construída usando a ferramenta *OpenSimulator*[11]. Esse estudo apontou que a prática de visitas a campos geológicos no ambiente simulado pode ser eficiente e eficaz para o aprendizado do aluno.

Figura 17 - Simulação com campos geológicos por meio do VGE (fonte: Fengru et al., 2013)

Já no estudo de Wagner et al. (2013), os autores apresentam um ambiente virtual 3D voltado a cursos a distância e que foi combinado com outras tecnologias, como, por

[11] http://www.opensimulator.org

exemplo, WEB 2.0 para prover uma simulação na internet. A solução apresentada teve como objetivo diminuir a distância geográfica entre os professores, tutores e alunos, tornando-se então uma ferramenta de apoio a esta modalidade de curso. Esse ambiente virtual 3D foi desenvolvido com o *OpenSimulator*.

Voss (2013) apresentou uma proposta de criação de um laboratório virtual imersivo para o ensino de redes de computadores por meio de um cenário de laboratório base denominado ASTERIX. Esse ambiente proporciona aos alunos o conhecimento técnico em redes de computadores e até mesmo a capacidade de aplicar esse conhecimento em mundo real, bem como a noção de buscar a melhoria por meio da aprendizagem e do trabalho colaborativo. Foi utilizado o *OpenSimlator* no processo de simulação, focando nas características da realidade virtual. A aplicação permite que o aluno efetue configurações e testes sobre uma rede virtual de computadores por meio de uma interface gráfica do usuário (GUI). A Figura 18 apresenta o ambiente virtual do ASTERIX.

Figura 18 - Laboratório Virtual de ensino de redes de computadores (fonte: Voss, 2013)

Os resultados do projeto apresentado por Callaghan et al. (2015) exploram como os mundos virtuais e técnicas de jogos de vídeo podem ser usadas para criar ambientes altamente imersivos e envolventes para o ensino de material relacionado à engenharia. Esse processo descreve como os educadores podem aprender a usar efetivamente as características mais atraentes de jogos de computador, por exemplo: a participação ativa, intrínseca e o retorno imediato. Para os cursos de educação à distância relacionados com a utilização da engenharia é um único método realista de proporcionar o aprendizado

prático, permitindo que alunos utilizem remotamente os laboratórios para práticas sem restrições de tempo ou considerações geográficas.

Esse projeto explorou como todos esses componentes individuais podem, com sucesso, ser integrados para criar um ambiente virtual. Além disso, esse trabalho demonstra como o ambiente existente é reforçado por meio da adição de uma camada de base de jogo, permitindo que equipes de estudantes trabalhem em conjunto de forma colaborativa e competitiva.

O trabalho de Banerjee et al. (2013) apresenta um estudo de caso que demonstra um mundo virtual 3D online compartilhado como um auxílio educacional para redes de ensino e comunicação nas universidades indianas, promovendo benefícios na tarefa de aprendizagem. O estudo utilizou a ferramenta *OpenSimulator*. Os autores desse projeto forneceram dois estudos de caso sobre o mesmo conteúdo educacional. O primeiro teste foi um ambiente 2D baseado na web e o segundo, um ambiente 3D. Eles concluíram que um ambiente virtual de aprendizagem tem resultados afirmativos e positivos com relação ao envolvimento dos alunos, podendo, inclusive, melhorar as notas dos alunos.

As informações apresentadas dentro do mundo virtual ajudam o aluno a explorar o tema a fundo, podendo usar esse conhecimento no cenário do mundo real. A educação imersiva ajuda os usuários a entenderem assuntos que tenham dificuldade em aprender no ambiente de sala de aula. A Figura 19 demonstra a solução.

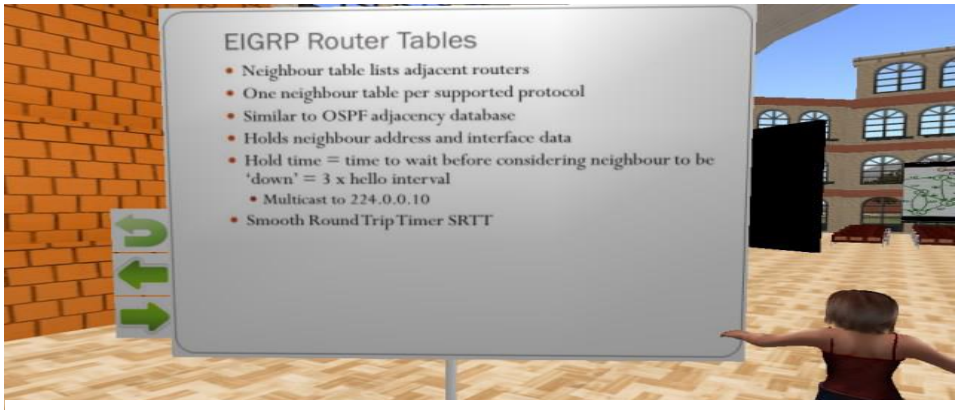


Figura 19 - Exibição de conteúdo utilizando slide (fonte: Banerjee et al., 2013)

3.5.3. Discussão

Os avanços tecnológicos experimentados nos últimos tempos em diversos aspectos

acabaram modificando áreas da sociedade. A área educacional é uma das que têm sofrido diversas mudanças, entre elas o uso da tecnologia para apoiar o processo de ensino-aprendizagem. Nesse contexto, o uso de ambientes virtuais é uma das soluções que se tem destacado devido a sua capacidade de favorecer a aprendizagem com facilidades, como, por exemplo, a possibilidade de executar o experimento diversas vezes que seriam difíceis no mundo real algumas vezes.

A adoção de ambientes virtuais, segundo Banerjee et al. (2013), permite que os professores recriem um mundo real ou que criem mundos completamente novos, provendo experiências que podem ajudar estudantes no entendimento de conceitos ou na execução de tarefas específicas do aprendizado.

Nesse tipo de abordagem não existe separação entre ação, reflexão e questionamentos, visto que o aluno aprende a teoria por meio da apresentação de conceitos e definições, vivenciando a prática por meio das simulações próprias desse tipo de ambiente e realizando os questionamentos utilizando os recursos de comunicação das ferramentas. Por essa razão, a aprendizagem virtual e imersiva, conforme relata Wagner et al. (2013), é indicada para melhorar a relação ensino e aprendizagem, sobretudo no que diz respeito às atividades práticas, que permitem ao aluno aplicar o conhecimento adquirido.

O uso de ambientes de realidade virtual na educação tende a promover um novo paradigma educacional, no qual os alunos não apenas recebem informações, mas, também, interagem uns com os outros por meio da manipulação de conteúdo em múltiplas formas, modificando o conhecimento que deve ser examinado, explorado e estudado. Uma característica fundamental do ambiente virtual é a possibilidade da interação, fornecendo a capacidade de ações instantâneas e reações em tempo real.

O ESORV foi desenvolvido para ser uma aplicação educacional para o ensino como as citadas anteriormente e que ilustram as potencialidades de se aplicar os ambientes virtuais imersivos ao ensino com o objetivo de melhorar e contribuir com um modelo de educação construtivista por meio da relação ensino-aprendizagem.

4. TRABALHOS CORRELATOS

Este capítulo apresenta o estado da arte em temas relacionados à dissertação. A pesquisa abrangeu publicações nas bases de dados IEEE, ACM e o Banco de Teses da Capes (http://servicos.capes.gov.br/capesdw/). Os tipos de trabalhos considerados foram: teses, dissertações, monografias, artigos de revisão e artigos publicados em conferências ou periódicos. Os seguintes descritores foram utilizados para a busca dos artigos: "Operating Systems" and "simulator" and "scheduling". As buscas nos sites foram realizadas por meio de opções avançadas de cada motor de pesquisa, fazendo uso de filtros de busca em um intervalo de publicação entre 2005 e 2015.

Após executadas as buscas com a combinação dos termos, os resumos dos artigos retornados pelos buscadores foram lidos e analisados, segundo os seguintes critérios de inclusão: ter sido publicado no período entre 2005 e 2015; estar em língua inglesa ou portuguesa, e obedecer aos descritores. Os critérios de exclusão considerados foram artigo que não apresentasse a versão completa para leitura e artigos que não relacionavam o conceito solicitado com o objetivo do tema da revisão. Os artigos resultantes estão listados na Tabela 5.

A principal busca da pesquisa foi sobre trabalhos que visam otimizar o escalonamento de processos de Sistemas Operacionais. Foram encontrados diversos simuladores de algoritmos de escalonamento que podem ser divididos em simulações por meio de programas de linha de comando ou em programas gráficos. A próxima seção, a 4.1, demonstra quais os trabalhos correlatos que foram encontrados na literatura e quais são suas principais características e funcionalidades.

4.1. Trabalhos

O trabalho de Suranauwarat (2007) apresenta um simulador de algoritmo de escalonamento de CPU. Ele usa animação gráfica para transmitir os conceitos de vários algoritmos de escalonamento para uma única CPU.

Esse simulador permite que o usuário configure os seus parâmetros após a execução. Ele descreve graficamente o que cada processo fez em um determinado momento.

Tabela 5: Relação de Artigos Selecionados

Artigos	Objetivos	Algoritmos
A CPU Scheduling Algorithm Simulator. 37th ASEE/IEEE Frontiers in Education Conference, October 2007	Criar simulador que usa animação gráfica para transmitir os conceitos de vários algoritmos de escalonamento para uma única CPU.	FCFS RR SJF SRTF MLFQ
Design and Implementation of a Process Scheduler Simulator and an Improved Process Scheduling Algorithm for Multimedia. International Conference on Advanced Computing and Communications, 2006.	Criar um simulador para concentra-se em avaliar a adequação de vários algoritmos de escalonamento de processos para um sistema operacional Multimídia. Análise de resultados para diferentes algoritmos relacionadas ao desempenho.	FCFS RR SJF EDF MLFS
Development and Evaluation of Adobe Flash based CPU Scheduling Simulator Executable on Major Multiple Web Browsers. Intelligent Networking and Collaborative Systems (INCOS), International Conference on, Sept 2015.	Criação de um simulador baseado em Adobe Flash para escalonamento dos principais algoritmos de CPU e executável em diferentes navegadores WEB.	FCFS RR SJF
Realtss: a real-time scheduling simulator Electrical and Electronics Engineering 2007. ICEEE 2007. 4th International Conference on.	Realtss é uma proposta de ferramenta de ensino de um simulador de escalonamento em tempo real, que é adequado para simular algoritmos de escalonamento sem a necessidade de implementá-los em um Sistema Operacional de Tempo Real (RTOS) e de código aberto. Comparou diferentes políticas de escalonamento entre eles.	Prioridade EDF DM PCP
Simulation of CPU scheduling Algoritms 2010 IEEE – Institute Electrical and Electronics Engineering	Envolveu o desenvolvimento de um simulador para escalonamento da CPU por diferentes algoritmos. Mediu o desempenho de diferentes algoritmos de escalonamento.	FCFS RR SJF Prioridade
A Constructivist Framework for Operating Systems Education: a Pedagogic Proposal Using the SOsim. 2005 ACM - ITiCSE '05 Proceedings of the 10th science education. 2005. Pages 218-222.	O SOSIM foi desenvolvido com uma proposta de ensino-aprendizagem no modelo construtivista. O simulador é 2D e tem como objetivo apoiar o ensino da disciplina de Sistemas Operacionais como foco em escalonamento de processos e gerenciamento de memória.	Prioridade RR

Usando essa representação do simulador, o usuário pode entender o que está acontecendo por dentro do sistema, incluindo a percepção de qual é o processo candidato para atribuição da CPU em um tempo diferente. Além disso, permite ao

usuário testar e aumentar seu entendimento dos conceitos estudados. Ele também pode tomar as suas próprias decisões de configuração do simulador por meio da interface gráfica, por exemplo, colocando ou retirando mais processos na fila de prontos para escalonar o processador. O simulador foi desenvolvido na linguagem *Java*[12] 5.0.

Esse simulador demonstra todos os processos mantidos em uma fila de prontos e todos os processos em espera que são colocados em uma fila de entrada e saída. O usuário utiliza um algoritmo de escalonamento para decidir qual processo a partir da fila de pronto irá ser executado, e quando e durante quanto tempo ocupará o processador. O simulador utiliza os algoritmos FCFS, RR, SJF, SRTF e MLFQ.

O simulador tem dois modos de funcionamento: de simulação e de prática. No modo de simulação o usuário pode assistir ao passo a passo de como um algoritmo funciona.

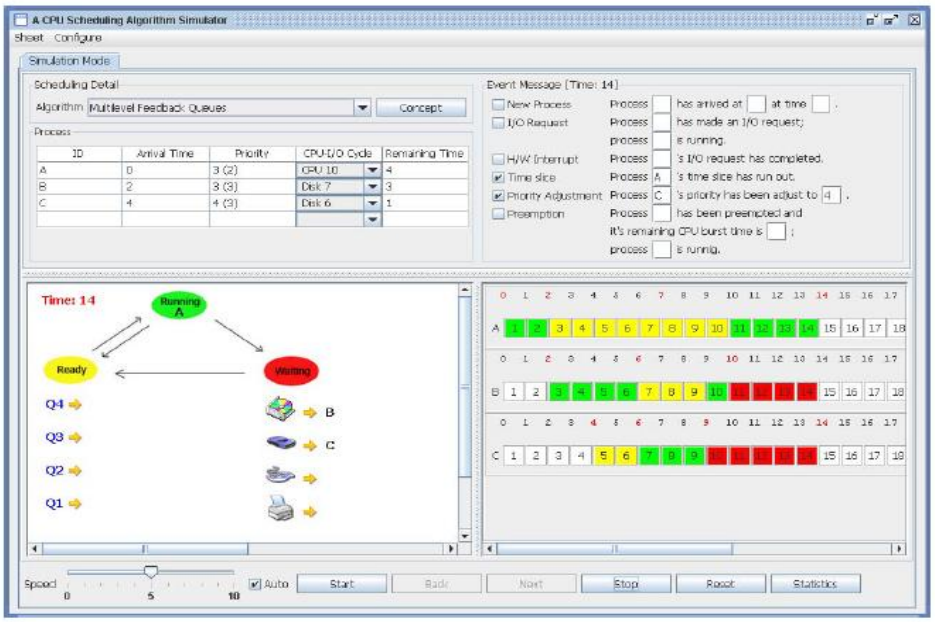


Figura 20 - Demonstração de uma simulação prática (fonte: Suranauwarat, 2007)

No modo de prática, representado na Figura 20, o usuário pode tomar as suas próprias decisões de simulação, criando um cenário para a execução do simulador, ou seja, decidir qual algoritmo, escolher o processo e optar por quanto tempo cada processo é executado no processador.

[12] *Java* é uma linguagem de programação orientada a objetos que atualmente faz parte do núcleo da Plataforma Java da Oracle®.

O trabalho correlato de Saraswat & Gupta (2007) mostrou um simulador de Sistema Operacional multimídia. O simulador concentra-se em avaliar à adequação de vários algoritmos de escalonamento de processos para um Sistema Operacional Multimídia. O simulador foi concebido de tal forma que vários algoritmos de escalonamento podem ser conectados ao código existente facilmente, com pouco conhecimento sobre as várias estruturas envolvidas. Cinco algoritmos de escalonamento padrão foram utilizados: FCFS, RR, MLFQ, SJF e EDF.

O algoritmo EDF (Earliest Deadline First) é utilizado em ambientes de Sistemas Operacionais em Tempo Real (RTOS). Ele tem como ideia principal a atribuição dinâmica de prioridades de acordo com o *deadline*[13] de cada processo (Shaw, 2003). O simulador foi escrito em linguagem *C*[14] e é compatível com sistemas rodando *Linux*[15]. A ideia do simulador está focada em medir o desempenho por meio de avaliações de amostras de tráfego em tarefas mistas distintas (áudio e vídeo). Para isso, permite medir parâmetros como prazos não cumpridos, interrupções de contextos totais, tempo médio de espera e tempo médio de retorno.

Já o trabalho apresentado por Imai & Takeichi (2015) descreve um simulador baseado em algoritmo de escalonamento de CPU com o *Adobe Flash*[16]. Ele foi criado com o objetivo de ser executado em ambientes da Web para obter uma avaliação quantitativa e qualitativa sobre escalonamento de processos de Sistemas Operacionais. O simulador suporta três principais algoritmos de escalonamento de CPU: FCFS, o RR, e o SJF. Ele pode ser executado sobre os principais navegadores da Web, como, por exemplo, o *Microsoft Explorer*[17], o *Mozilla FireFox*[18] e o *Google Chrome*[19].

[13] *Deadline* é o tempo máximo para realização de uma tarefa.
[14] *C* é uma linguagem de programação compilada de propósito geral, estruturada, imperativa, procedural.
[15] *Linux*® é um sistema operacional, programa responsável pelo funcionamento do computador,
[16] *Adobe Flash Player*® é um *plugin* multiplataforma, baseado em navegador, que permite a exibição de conteúdos e vídeos.
[17] *Internet Explorer* é uma série de navegadores web gráficos desenvolvidos pela Microsoft®.
[18] *Mozilla FireFox* é um navegador livre e multi-plataforma desenvolvido pela Mozilla Foundation®.
[19] *Google Chrome* é um navegador de internet, de código aberto, desenvolvido pela companhia Google®.

Na interface do simulador, escrito em *Action Script* [20] para *Adobe Flash,* é possível visualizar as quatro grandes etapas do cenário para o comportamento típico dele: definir a velocidade de entrada para o início da simulação; selecionar o tipo de algoritmo de simulação; fazer a simulação com botões (para trás e para a frente); e, por último, apresentar os resultados da simulação de acordo com a configuração desse cenário e compara-los as métricas utilizando outros algoritmos.

Nesse simulador também foi possível que os alunos que o utilizaram na disciplina de Sistemas Operacionais respondessem questões qualitativas sobre a eficácia do uso do simulador. Por exemplo, na questão: É eficaz para aprender algoritmo de escalonamento de CPU com este simulador visual?; 90% dos entrevistados responderam que sim.

Diaz et al. (2007) apresentou o *Realts,* que é um simulador voltado para algoritmos de escalonamento em tempo real sem a necessidade de implementá-los em um Sistema Operacional. No *Realtss* as políticas de escalonamento de Sistemas Operacionais em tempo real atual ou novas podem ser implementadas e seu desempenho avaliado. O simulador pode ser utilizado como ferramenta de ensino, no qual os alunos podem aprender a programação sem a necessidade de conhecer programação de sistemas em tempo real.

Ele foi escrito em *TCL*[21] e é distribuído como software de código aberto. Seu design modular permite a integração de algoritmos de escalonamento. Novos algoritmos de escalonamento podem ser adicionados como módulos escritos em *TCL*, *C* ou *C*[22] ++. Ele pode ser executado em vários Sistemas Operacionais como o Linux ou Windows.

A utilização do simulador envolve três etapas: definição de parâmetros, simulação e análise de resultados. Na primeira fase, o usuário define os parâmetros das tarefas. A Figura 21 mostra a interface do *Realtss*, no qual é possível realizar configurações de parâmetros: o período, o tempo de execução do pior caso, a fase, e o prazo.

---

[20] *ActionScript* é a linguagem de criação de scripts do Flash, para controlar objetos em filmes do Flash a fim de criar elementos para a navegação e interatividade, possibilitando a criação de filmes e aplicativos da Web com grande interatividade.

[21] *TCL* é uma linguagem de programação para scripts que pode ser utilizada em diversas aplicações, e ser executada em diferentes plataformas, como, por exemplo, Linux e outros sistemas operacionais.

[22] *C++* é uma linguagem de programação compilada multiparadigma (seu suporte inclui linguagem imperativa, orientada a objetos e genérica) e de uso geral.

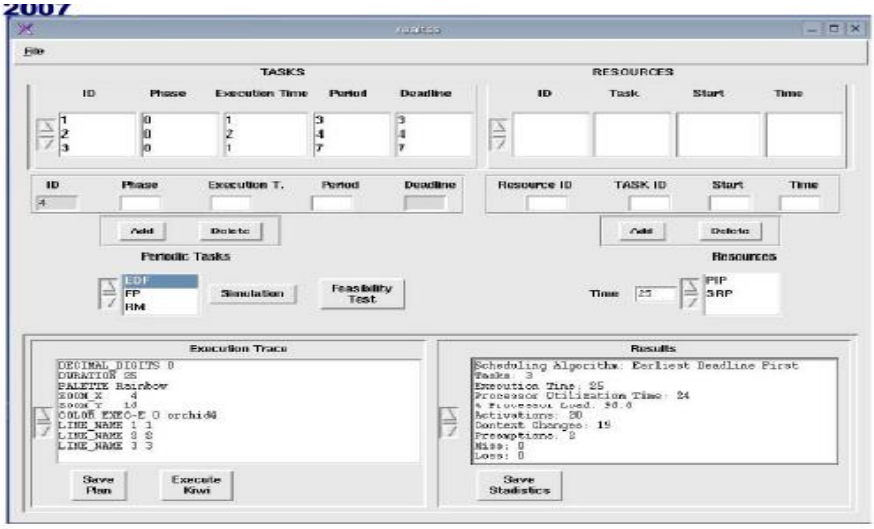


Figura 21 - Interface do usuário para o *Realtss* (fonte: Diaz A et al., 2007)

O trabalho de Umar & Javed (2010) mostrou o desenvolvimento de um simulador para escalonamento de CPU chamado de *SCSA*. Foi desenvolvida uma ferramenta que faz uma simulação em tempo real e gera dados úteis a serem utilizados para a avaliação. Os resultados da avaliação são úteis para a concepção e desenvolvimento de Sistemas Operacionais modernos. Esse simulador pode ser usado para medir o desempenho de diferentes algoritmos de escalonamento, por exemplo: FCFS, RR, SJF e Prioridade. Também é usado para a relação de aprendizado para os alunos.

O sistema foi projetado para rodar em Sistemas Operacionais *MS-DOS*[23] ou *Windows*[24]. Ele fornece ao usuário uma oportunidade para simular escalonamento de processos em um ambiente de multiprogramação e multiusuário. Consegue ler dados de simulação a partir de um arquivo de dados definido pelo usuário a partir de linhas de comando e de acordo com os dados, organiza a fila de prontos na memória.

O sistema simula a criação, a execução e o encerramento de processos simultâneos, e simula também a utilização compartilhada do processador, da unidade de disco, da impressora e da unidade de fita magnética. Mantém filas do sistema na memória

23 O sistema operacional *MS-DOS*® é responsável pela comunicação entre o usuário e o computador. Foi desenvolvido pela Microsoft® há muito tempo. É um sistema monoprogramado.

24 O *Windows*® é um sistema operacional com interface gráfica que foi desenvolvido pela Microsoft®. Possui ambiente multiprogramado.

como listas vinculadas isoladamente. Essas filas incluem: fila de prontos, fila de disco, fila de impressão e fila de unidade de fita magnética.

O SOSIM (sistema operacional simulador) desenvolvido por Machado & Maia (2005), é um simulador criado para o apoio do ensino-aprendizado da disciplina de Sistemas Operacionais em especial o conteúdo de escalonamento de processos e gerenciamento de memória. Ele foi criado por meio do modelo de ensino-aprendizagem conhecido como Construtivista. O SOSIM permite que o professor apresente os conceitos e mecanismos de um sistema operacional multiprogramável e/ou multitarefa de forma simples e animada. A partir das opções de configuração é possível selecionar diferentes políticas e alterar o funcionamento do simulador. Por exemplo, o usuário pode escolher qual o algoritmo deseja simular e no próximo passo criar os processos que podem ser do tipo CPU-*bound* ou I/O-*bound*. Após a configuração da para observar na interface do SOSIM os processos escolhidos alinhados na fila de prontos da memória principal. Em seguida, a animação 2D do simulador mostra os processos ocupando a CPU e retornando a fila de prontos na memória quando necessário ou sendo conduzidos até a fila de dispositivos de entrada/saída. E assim para todos os processos selecionados.

Para cada ciclo de simulação é possível refazer o cenário. Por meio da interface do simulador pode-se alterar as variáveis de configurações dos processos em tempo real, incluindo ou retirando processos. Desta forma, o aluno tem a oportunidade de visualizar os conceitos teóricos apresentados em aula de forma simples e animada e que pode colaborar no ensino de escalonamento de processos da disciplina de Sistemas Operacionais.

Por meio do SOSIM pode-se implementar a simulação sobre o conceito de escalonamento de processos (criar processos e visualiza-los em seu BCP, visualizar seus estados); visualizar estruturas internas do sistema (BCP e TEP - Tabela de Entrada de Páginas); e ainda visualizar os conceitos relacionados a gerência de memória (memória virtual por paginação e política de busca e substituição de página). A Figura 22 mostra a interface do SOSIM para o escalonamento de processos. Observa-se as características principais do processo: o estado que ele se encontra, o tempo de espera de E/S, a fatia de tempo, a frequência do *clock* e a prioridade da fila de pronto.

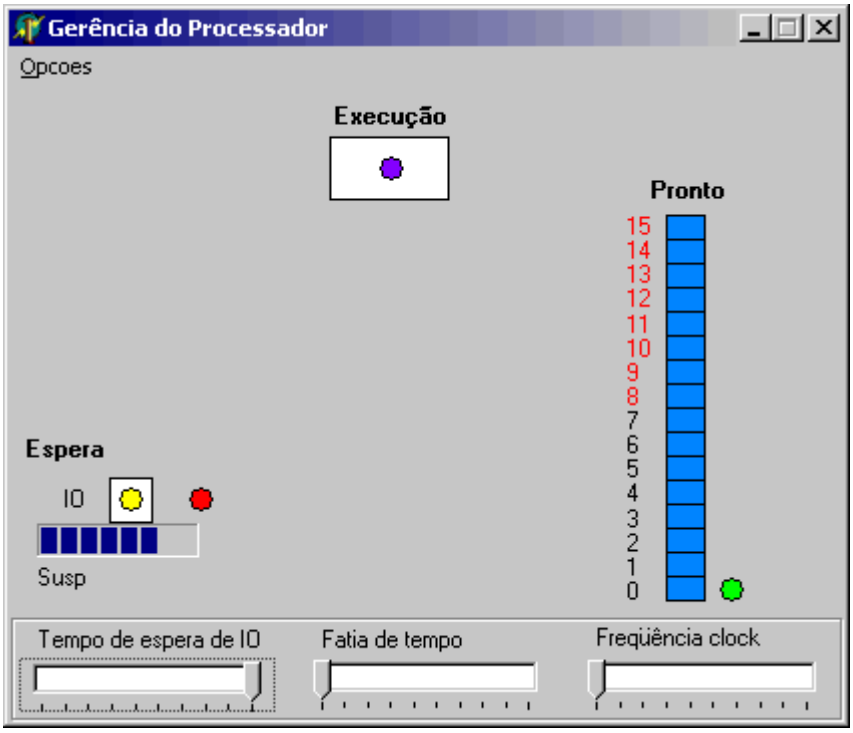


Figura 22 - Interface de gerência do processador do SOSIM (fonte: Machado & Maia, 2013)

O SOSIM é um projeto utilizado em larga escala no ensino de escalonamento de Sistemas Operacionais, possui comunidades na internet que colaboram com sua evolução e integração. Ele foi escolhido para ser comparado ao ESORV nos testes aplicados nessa dissertação.

4.2. Discussão

Os trabalhos correlatos demonstram que simuladores foram construídos com base na utilização de diversos algoritmos de escalonamento para ocupar uma única CPU. Eles são executados em diferentes cenários de ambiente, como desktop e internet. Podem ser configurados por parâmetros pelos usuários, customizando uma configuração para realizar a simulação. Nela, os usuários conseguem ditar o comportamento desejado da execução da aplicação. É possível, por exemplo, escolher qual algoritmo deverá compor a simulação, seu tempo de execução, seu estado e observar os resultados.

Foram desenvolvidos para Sistemas Operacionais distintos, Linux e Windows, utilizando-se de diversos ambientes de programação diferentes. Os propósitos são os mais variados possíveis. O trabalho de Imai &Takeichi (2015) tem como foco medir o

comportamento de desempenho, por exemplo, o tempo de resposta da execução da aplicação escrita em *Adobe Flash* em navegadores de Web diferentes, aplicados com algoritmos de escalonamento escolhido pelo usuário. Já o trabalho de Suranauwarat (2007) apresenta um simulador de algoritmo de escalonamento de CPU, que utiliza animação gráfica com o foco de melhorar o entendimento na transmissão dos conceitos envolvidos no escalonamento. Por meio de etapas os alunos observam o funcionamento dos algoritmos de escalonamento e suas características, como, por exemplo, a prioridade de execução dos processos. O trabalho de Saraswat, & Gupta (2007) apresentou um simulador de Sistema Operacional multimídia, que é focado na medição de desempenho nos tipos de mídia contínua (vídeo e áudio) com tarefas diferentes. Esse tipo de situação exige que o escalonador execute com restrições de tempo estritas para o processamento.

Embora alguns deles tivessem a preocupação de melhorar o entendimento dos conteúdos aplicados e a funcionalidade dos algoritmos, os simuladores são baseados em programas de linha de comando - Saraswat & Gupta (2007) ou uma interface em duas dimensões (2D) - Machado & Maia (2005), que é uma característica semelhante na maioria dos simuladores. Esse tipo de interface permite ao aluno visualizar a CPU e a memória principal compartilhada de uma forma dinâmica e animada, mas não é possível que o aluno tenha imersão no ambiente. No levantamento dos trabalhos correlatos não é observado que o aluno possa criar experiências imersivas com o simulador, no qual ele tem a condição de estimular o comportamento da simulação e observar o resultado obtido, e novamente estimular o sistema, características que são os propósitos da realidade virtual e que foram utilizadas na construção do ESORV por meio de uma interface virtual – 3D.

5. SIMULADOR PARA O ENSINO DE SISTEMAS OPERACIONAIS COM REALIDADE VIRTUAL - (ESORV)

Este capítulo apresenta o simulador desenvolvido nesse trabalho, o ESORV (Simulador para o Ensino de Sistemas Operacionais com Realidade Virtual). Ele foi desenvolvido levando-se em consideração os princípios dos escalonadores de processos e os principais propósitos da realidade virtual, proporcionando aos alunos uma ferramenta para construir uma experiência inovadora na relação ensino-aprendizagem. É possível com o simulador que o aluno se envolva com o conteúdo por meio da imersão, da interação e do envolvimento, sentindo-se como se estivesse dentro do computador. Para isso, o aluno utiliza um ambiente imersivo e tridimensional (3D) para realizar as simulações de escalonamento de processos.

A seção 5.1 demonstra as principais características e a arquitetura de componentes do simulador. A seção 5.2 mostra como é realizada a simulação virtual de escalonamento de processos. Por fim, a seção 5.3 apresenta as principais tecnologias utilizadas na construção do simulador.

5.1. Características Principais e Arquitetura do Simulador

Uma das barreiras encontradas na adoção de simuladores de realidade virtual na educação é o custo. Contudo, os avanços tecnológicos têm contribuído para que as instituições de ensino criem e tenham o acesso a laboratórios específicos de realidade virtual a custos menores. Dispositivos como *Google CardBoard*[25] e softwares específicos já estão disponíveis no mercado por meio de software livre de qualidade.

Utilizando o ESORV os alunos conseguem realizar a imersão em uma placa mãe do computador, no qual eles tem condições de visualizar, manipular, e explorar o processo de escalonamento de um processo na CPU em tempo real. Essa atividade de simulação pode fortalecer e consolidar a compreensão dos alunos em relação ao escalonamento de processos. Para tanto, uma estratégia pedagógica bem delineada deve ser inserida com o intuito de assegurar que os alunos compreendam o conteúdo.

[25] https://vr.google.com/cardboard/

As principais atividades de simulação que podem ser executadas e observadas no ESORV são:

- **Criação de Processos**: o aluno pode criar os processos no ambiente virtual de acordo com as configurações necessárias. Escolher se é um processo do tipo *CPU-bound* ou *I/O-bound* e determinar o tempo de execução dos processos;
- **Escolher o algoritmo**: é possível escolher qual tipo de algoritmo será aplicado no ambiente virtual (PR, SJF, RR);
- **Alteração em tempo real**: o aluno pode a qualquer instante alterar o algoritmo escolhido e recarregar as configurações novamente;
- **Acompanhar as mudanças de estado do processo**: por meio das animações se torna possível visualizar e ouvir as mudanças de estado do processo,
- **Visualizar as filas de pronto e de entrada/saída dos processos**: o layout do simulador proporciona visualizar os processos que estão enfileirados na fila de pronto e na fila de dispositivos de entrada/saída;
- **Observar a inserção e recuperação das informações da tabela de BCP de cada processo**: quando ocorre a interrupção de um processo ou retorno dele a CPU é possível observar de fato como é realizado o processo de inserção/recuperação das informações do processo na BCP;
- **Entender o funcionamento dos algoritmos de escalonamento**: o ambiente virtual permite que se veja e ouça como é feito a aplicação de cada algoritmo escolhido para ser executado;
- **Observar a integração da memória e processador com os processos**: em uma das animações do ambiente virtual é possível observar como o processo é conduzido entre a memória e o processador;
- **Entender as transições dos estados de um processo**: o aluno pode entender em quais situações possíveis um processo pode mudar seu estado,
- **Entender quando ocorre a preempção**: o aluno consegue visualizar a retirada do processo de uma CPU, ainda que não terminou seu *quantum*, e a entrada do outro processo na CPU.

A Figura 23 mostra a interface inicial de utilização do ESORV, antes do aprendiz iniciar o processo de simulação. É possível observar em uma perspectiva central da câmera os principais elementos que fazem parte do cenário que corresponde a visão interna de uma placa mãe do computador, que são:

- **Elemento número 1:** conhecida como “fila de prontos” dos processos que estão localizados na memória principal (RAM);
- **Elemento número 2:** localização da memória principal (RAM), local de armazenamento das principais rotinas do Sistema Operacional e dos processos que irão ocupar a CPU;
- **Elemento número 3:** localização da CPU;
- **Elemento número 4:** tabela de armazenamento das variáveis de cada processo, a BCP;
- **Elemento número 5:** conhecida como “fila de E/S” dos processos que estão em estado de “espera” nos dispositivos de entrada e saída;
- **Elemento número 6:** dispositivos de entrada e saída, por exemplo, uma unidade de armazenamento em massa, o hard disk;
- **Elemento número 7:** os barramentos da placa mãe de um computador.

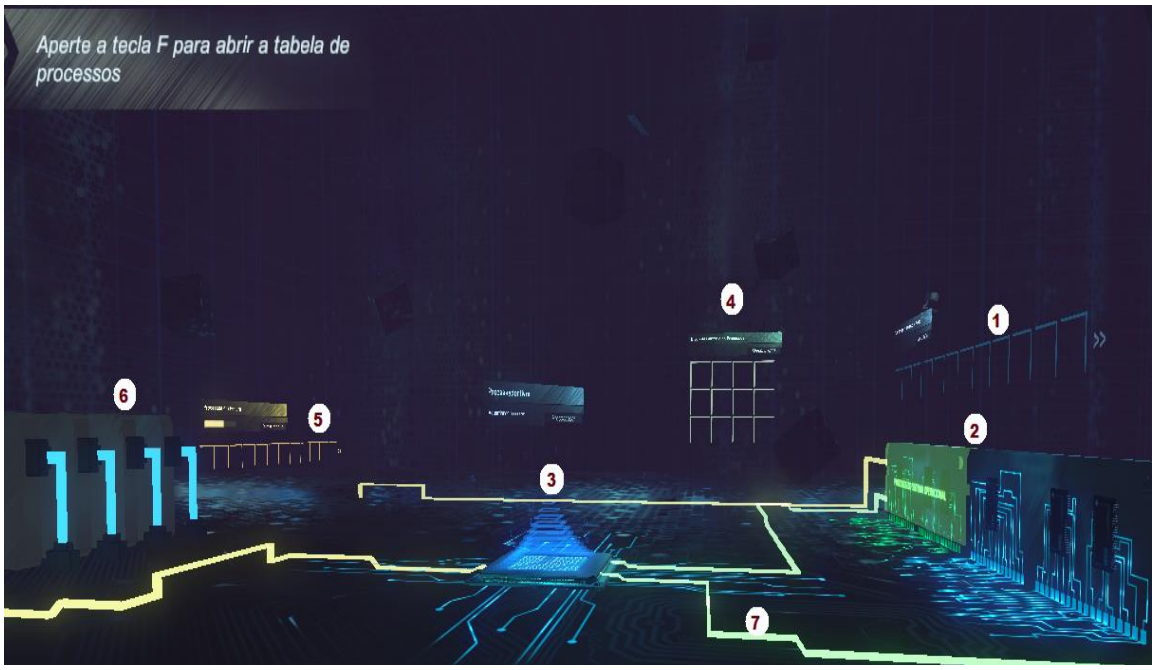


Figura 23 - ESORV – interface inicial representando os principais elementos internos de um computador

A Figura 24 mostra o ESORV sendo executado. Nesse exemplo, o simulador é executado em um mini-CAVE (Dias et al., 2012), que é um ambiente composto de três paredes laterais com *estereoscopia passiva*[26], com suporte a vários dispositivos de interação, por exemplo: o *Microsoft Kinect*[27], o *Wii Remote*[28], e o teclado e/ou mouse. A interação cumpre um dos requisitos básicos de aplicações de realidade virtual - a simulação reagir às ações do usuário de forma adequada (isto é, a perspectiva reflete a posição e orientação do usuário). O ambiente de processamento consiste de um cluster de computadores de seis nós com placas gráficas tridimensionais (3-D) e seis projetores de alta definição.

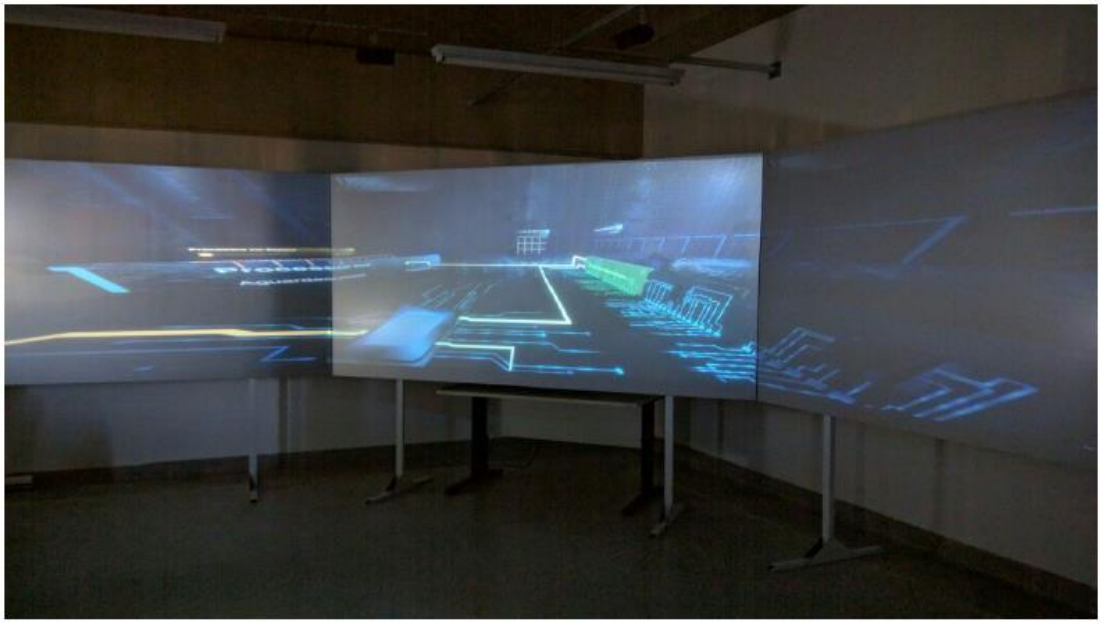


Figura 24 - ESORV - simulador para o ensino de escalonamento de processos com realidade virtual sendo executado em um mini-CAVE

Nessa versão da simulação que executa no mini-CAVE (multiprojeção) foi implementada usando o *Unity Cluster Package* (Neto et al., (2015)). Com ele, tornou-se possível executar o simulador com suporte passivo para estereoscópio, correção de perspectiva de acordo com o ponto de vista do usuário e o acesso a servidores específicos

[26] *estereoscopia passiva* é a possibilidade de ser mostrada duas imagens simultaneamente, por exemplo, olho esquerdo e direito.
[27] *Microsoft Kinect* é um sensor de movimentos desenvolvido para os consoles de jogos Xbox 360 e Xbox One. O Kinect criou uma nova tecnologia capaz de permitir aos jogadores interagir com os jogos eletrônicos sem a necessidade de ter em mãos um controle.
[28] *Wii Remote Wii* é um console de videogame doméstico produzido pela Nintendo®.

para fornecer recursos de dispositivos independentes. A Figura 25 ilustra a arquitetura dessa ferramenta que foram separadas em atividades de simulação e visualização:

- **Escalonamento de Processos**: a aplicação de realidade virtual que simula algoritmos de escalonamento de processo;
- **Interação**: permite que professor e alunos manipulem a simulação utilizando dispositivos de entrada, como *Kinec*t, *Wii Remote*, teclado e mouse;
- **Navegação**: permite que professor e alunos percorram os componentes da placa mãe (ou seja, memória, CPU e I/O dos dispositivos);
- **Visualização**: permite que professores e alunos observem os componentes da placa mãe usando seu computador pessoal, HMD, ou um dispositivo multiprojeção.

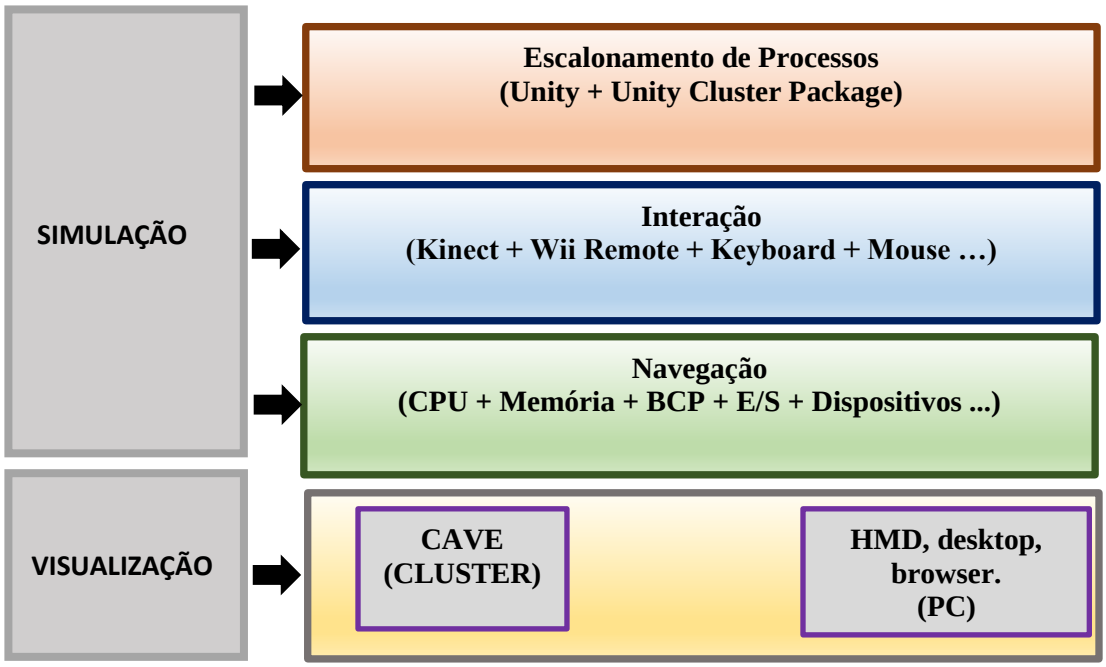


Figura 25 - Arquitetura do simulador ESORV

5.2. Simulação Virtual de Escalonamento de Processos

O funcionamento adequado do simulador depende do ciclo ilustrado na Figura 26. Inicialmente, são as etapas sequenciais que devem ser cumpridas para realizar a simulação. O algoritmo de escalonamento é escolhido (PR, RR, SJF), conforme a Figura 27. Depois disso, o usuário poderá escolher quais os processos participarão da simulação

e isso inclui a definição dos seus parâmetros, como, por exemplo, o algoritmo (PR, SJF, RR); o tempo de execução; e o tipo de processo (*CPU-bound*, *I/O-bound*), ilustrado na Figura 28.

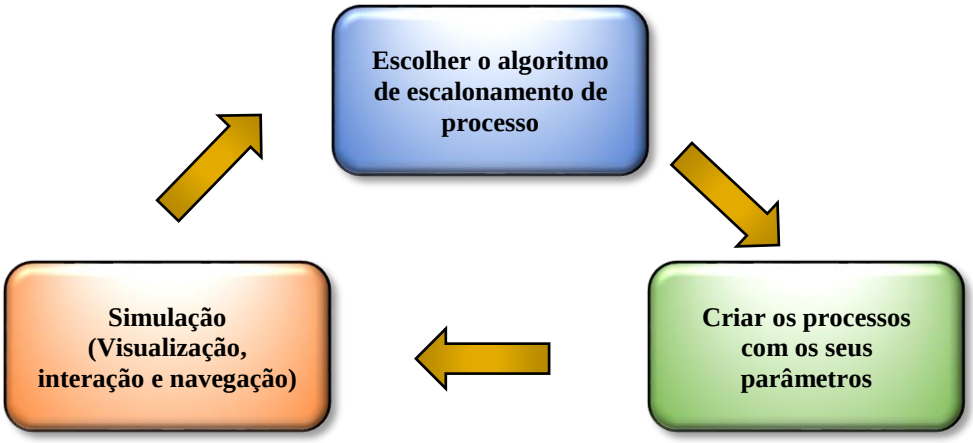


Figura 26 - Ciclo de Simulação do ESORV

Finalmente, a simulação começa. Cada processo é executado por uma quantidade de tempo gerado e, a partir deste momento, o usuário é livre para navegar e visualizar a placa mãe do computador e interagir com os processos.

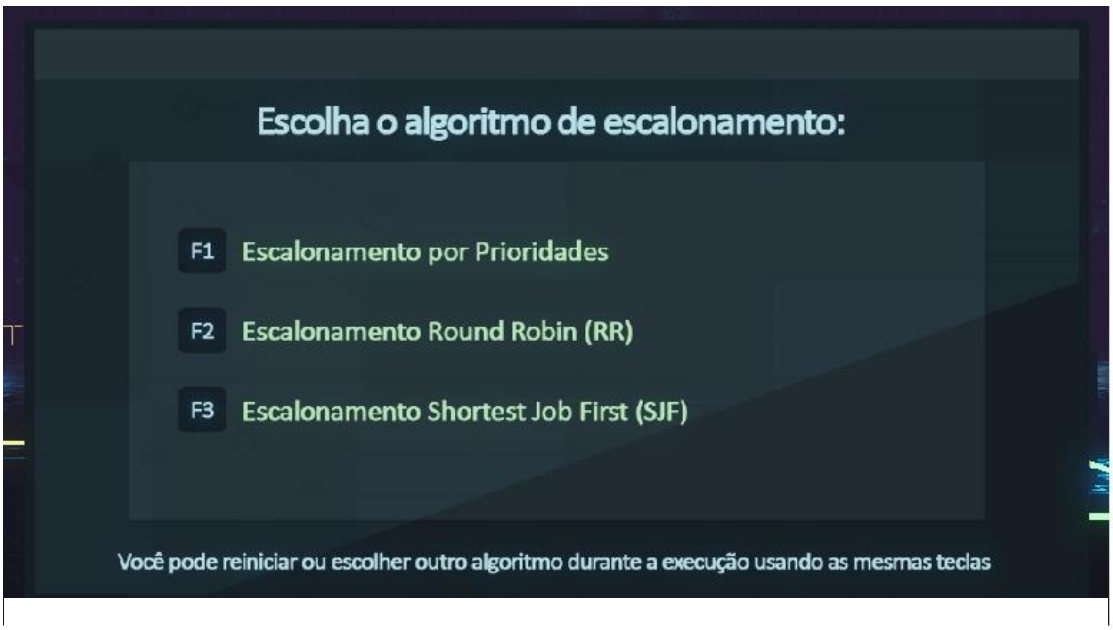


Figura 27 - Tipos de algoritmos que podem ser ecolhidos para executar uma simulação no ESORV

Durante a navegação, o usuário pode visualizar mensagens e ou ouvir descrições da atividade em curso, e de transições do processo do algoritmo (por exemplo, "Um novo processo ocupou a CPU" ou "O BCP do processo atual em execução foi salvo").

Animações gráficas são usadas para ilustrar a etapa ou o passo de transações de estado dos processos e do movimento de processos de um componente para outro (por exemplo, a partir da CPU para o disco rígido por meio do sistema de barramentos do computador). É possível, por exemplo, assistir a uma mudança de processo da CPU para um dispositivo de entrada e saída e observar o estado do processo sendo alterado do estado de execução para aguardando.

Torna-se fácil de entender o que está acontecendo por dentro do algoritmo de escalonamento e por que, em determinado momento, um processo qualquer vai executar ou não na CPU. Além disso, uma guia de ajuda está disponível durante a simulação para fornecer informações sobre os controles e os algoritmos de escalonamento. Sobre o aspecto da navegação, o simulador implementa as características necessárias de uma aplicação interativa, como a detecção de colisão, que é necessária para ambiente virtual.

Quando o aluno está criando os processos, ele pode determinar os parâmetros de cada processo de acordo com o algoritmo escolhido (Figura 28). Exemplos desses parâmetros é incluir uma prioridade no caso de um algoritmo de escalonamento por prioridade, ou o tempo de execução de um processo do algoritmo SJF ou RR. Em todos os casos é possível determinar se um processo está vinculado à CPU/*bound* ou I-O/*bound*. O professor pode configurar os parâmetros do algoritmo e de processo para simular situações especiais. Por exemplo, se um processo tem baixa prioridade dentro de um conjunto de processos sofrerá uma atividade denominada *starvation* (ele nunca é atribuído a CPU). Uma possível solução para a inanição, nesse caso, é o professor ou o aluno optar por escolha de um algoritmo de escalonamento com uma fila de prioridade que aumenta a prioridade dos processos que têm esperado na fila de pronto por um longo período de tempo.

O simulador foi projetado para ser capaz de adicionar novos algoritmos de escalonamento, necessitando apenas de uma nova rotina para reordenar a fila de pronto de acordo com os critérios. Os algoritmos suportados atualmente são **FCFS, RR e PR.**

O ESORV utiliza o modelo clássico de três estados para fazer a simulação (pronto, em execução e em espera). Nele é possível visualizar todo o processo de transições (pronto para execução, execução para pronto, execução para espera e espera para pronto).

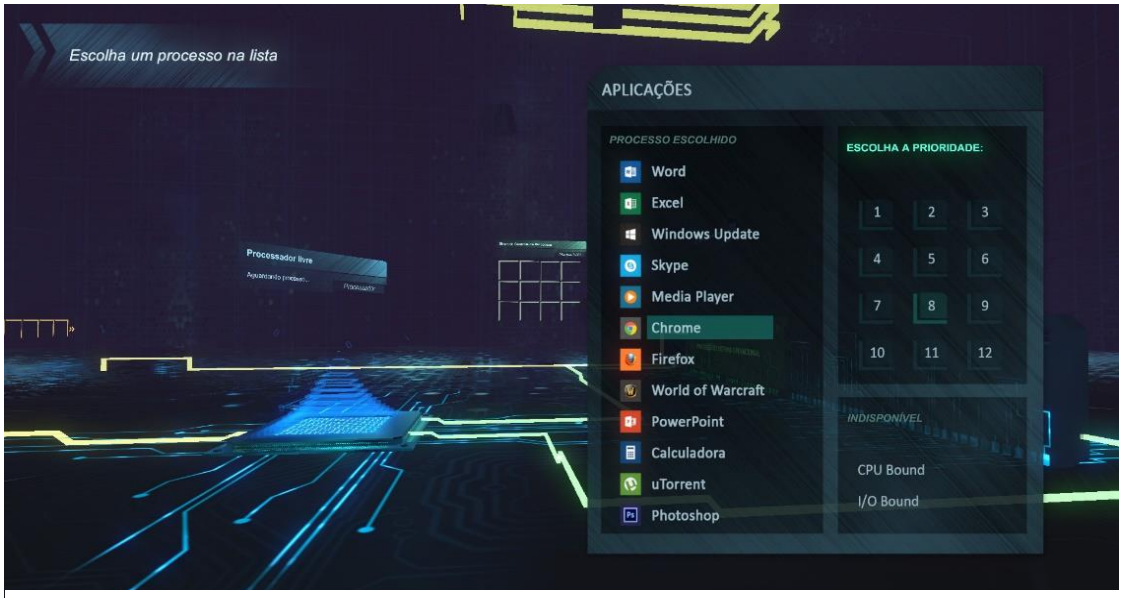


Figura 28 - O aluno cria o processo e configura os seus parâmetros no ESORV

A Figura 29 mostra a transição do processo no estado de pronto que se encontra na memória RAM para o estado de execução na CPU. É possível observar em uma perspectiva da câmera a condução do processo por meio do barramento sendo realizado pela rotina “despachante” do Sistema Operacional.

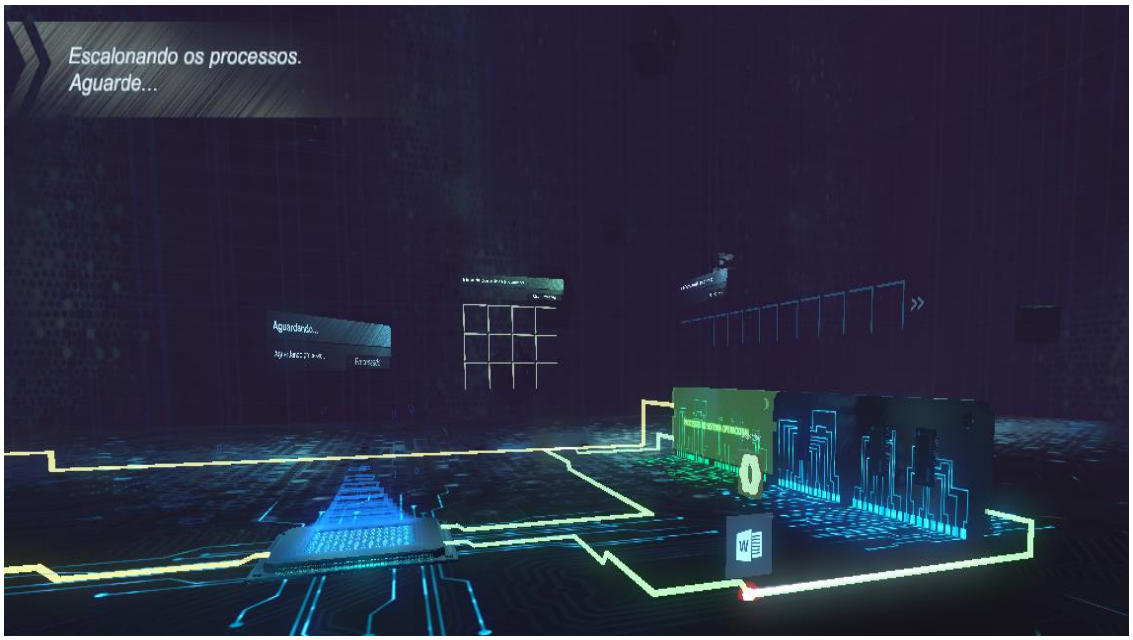


Figura 29 - A rotina “despachante” do Sistema Operacional conduzindo o processo que estava no estado de “pronto” na memória RAM para ocupar a CPU

A Figura 30 mostra o conteúdo de uma BCP de um processo sendo salvo no momento da mudança de contexto entre um processo e outro na CPU. Nela é possível observar os processos que estão com o seu BCP ativo. Nesse exemplo são: o Word, o Media Player e a Calculadora. Ao aproximar-se de cada processo é permitido visualizar as suas informações internas, como, por exemplo, o número identificador do processo, o nome, o estado, o tempo restante de execução, o registrador de CPU, e o limite de memória.

Figura 30 - O aluno visualiza o conteúdo do BCP do processo salvo no ESORV

5.3. Principais tecnologias utilizadas na implementação

O ESORV foi desenvolvido utilizando o motor de jogos chamado *Unity Unreal*[29], que é usado para criar jogos para plataformas de computadores pessoais, dispositivos móveis, aplicações Web e consoles. Seu motor de tempo de execução é integrado com um conjunto de ferramentas que facilitam os fluxos de trabalho rápidos e permite conteúdo interativo tridimensional (3-D). É um dos motores de jogos mais populares.

O motor de jogo é o programa de computador utilizado na confecção de jogos digitais. Segundo Anderson et al. (2013), motores de jogo fornecem a infraestrutura para a criação de jogos, não sendo somente um conjunto de componentes reutilizáveis, mas uma camada que conecta tais componentes. É nesse motor de jogo que o desenvolvimento desse simulador foi feito, unindo arquivos de áudio, imagens e modelos tridimensionais

[29] https://unity3d.com/pt

(3D) para criar os diversos cenários e ambientes do jogo. O motor de jogos possui diversas bibliotecas de *scripts* já embutidas, que facilitam o desenvolvimento de um ambiente virtual. Por exemplo, s*cripts* para cuidar da renderização dos gráficos (motor gráfico) e da física básica envolvida no jogo (motor de física).

No *Unity* é possível montar sua arte e seus recursos em cenas e ambientes, adicionar física, editar e testar simultaneamente seu jogo e, quando finalizado, publicar em suas plataformas escolhidas, como, por exemplo, *Web*[30], *iOS*[31], *Android*[32], *Wii Remote*, *PS4*[33] e *Xbox 360*[34].

A Figura 31 demonstra a interface da *Unity* 3D usada na construção do ESORV. De forma geral, ela é basicamente dividida em quatro seções: 1-cena, 2-inspetor, 3-hierarquia e 4-arquivos:

- **A seção 1:** mostra o ambiente da cena com todos os seus elementos. Quando um objeto 3D é colocado na cena, ele pode ser manipulado. Uma cena tridimensional (3D) possui três eixos, que são os eixos X, Y e Z. São as três direções que definem um ambiente tridimensional (3D);
- **A seção 2:** mostra a hierarquia, que possui a lista de objetos presentes na cena. O ícone selecionado nessa lista é o que está no centro da cena. Selecionar um objeto na cena ou na lista de hierarquia tem o mesmo resultado. Em cenas complexas é muito mais fácil encontrar os objetos por meio da hierarquia do que tentando encontrar manualmente no meio da cena;
- **A seção 3:** mostra a pasta com os arquivos. Ao se criar um novo projeto, a *Unity* vai definir por padrão uma pasta raiz chamada *Assets*. Dentro dessa pasta deverão ser salvos todos os arquivos a serem usados no projeto, como, por exemplo, os modelos 3D, os scripts, os efeitos sonoros e as texturas. É possível criar várias pastas para organizar melhor os arquivos dentro dessa pasta raiz;

[30] *Web* é o nome pelo qual a rede mundial de computadores, a internet, se tornou conhecida.
[31] *iOS* é um sistema operacional para dispositivos móveis da Apple Inc®
[32] *Androi*d é um sistema operacional do Google® para dispositivos móveis baseado no Linux.
[33] *PS4* é um console de jogos da Sony®.
[34] *Xbox 360* é um console de jogos da Microsoft®.

- **A seção 4:** chamada de *Inspector*, que indica todos os detalhes e configurações sobre o objeto selecionado na cena. Observe como o nome do objeto no *Inspector* é o mesmo selecionado na hierarquia, que por fim é o mesmo selecionado no centro da cena. Ele mostra vários detalhes, como a posição do objeto na cena, se esse objeto possui colisores e scripts anexados. Também mostra o seu material e as configurações das texturas (*shader*), que definem como esse objeto será afetado visualmente pela iluminação da cena.

Figura 31 - Interface da Unity 3D usada na construção do simulador ESORV

Há também uma versão para dispositivos baseada no *Google CardBoard* e uma solução HMD fornecida por meio da utilização de smartphones para a geração de imagem. A interação é capturada usando um controle Bluetooth[35] integrado ao ambiente *Unity* 3D.

A Figura 32 mostra a execução do *Google CardBoard*. O mesmo conteúdo pode ser visualizado nesse dispositivo, embora com recursos limitados de interação, imersão e envolvimento.

[35] *Bluetooth* é a tecnologia de comunicação sem fio que permite transmissão de dados e arquivos de maneira rápida e segura por meio de diversos tipos de dispositivos móveis.

Além do ambiente descrito para o desenvolvimento do simulador ESORV até o momento, também foram utilizadas outras tecnologias de softwares conforme demonstra a Tabela 6.

Tabela 6: Relação de Softwares Utilizados

Relação de Software	
Autodesk Maya	Programa de modelagem 3D. Todos os modelos 3D do protótipo serão criados nesse programa. Isso inclui o cenário, os ícones, as memórias, o processador, e os outros modelos que aparecem visualmente na cena.
Adobe Photoshop	Programa de edição de imagem. Será usado para criar as texturas dos modelos 3D e texturas da interface.
Adobe Soundbooth	Programa de edição de áudio que será utilizado no protótipo para efeitos sonoros.
Monodevelop	Editor de scripts utilizado para dar manutenção nos scripts que foram criados no Unity 5 por meio de JavaScript.
Unity 5 Unreal	É um motor de jogo que será utilizado para criar as cenas de simulação do protótipo semelhante às de um jogo. Onde se insere os modelos 3D, as texturas, a iluminação e os efeitos sonoros, por meio dos scripts. Os scripts são criados dentro da Unity.

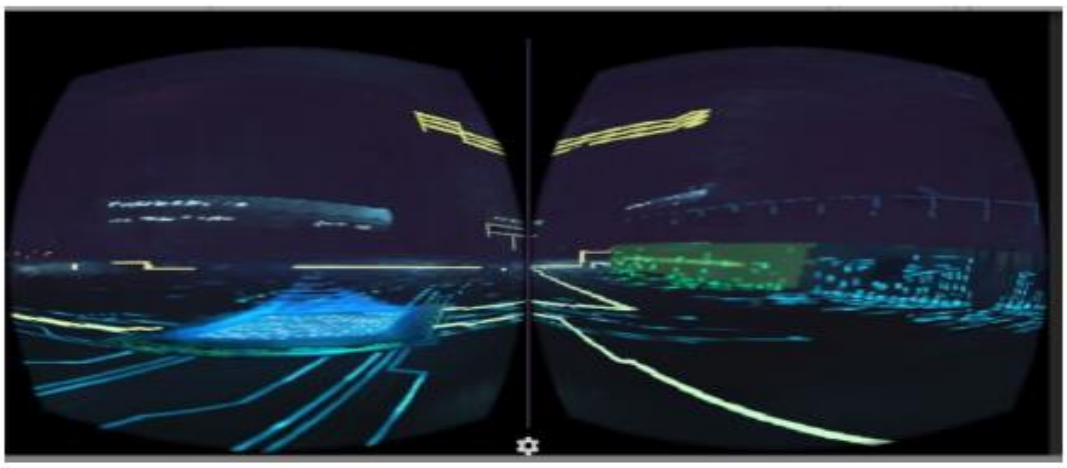


Figura 32 - Execução do *Google CardBoard* (olho esquerdo e direito)

5.4. Considerações Finais

Este capitulo apresentou o ESORV, que é um simulador que apoia o ensino de Sistemas Operacionais por meio da simulação virtual de escalonamento de processos. É possível que o aluno ou aprendiz faça um ciclo de simulação em tempo real a qualquer momento que seja necessário e que esteja de acordo com o seu ritmo de aprendizagem.

Um dos pontos fortes do simulador é permitir que o aprendiz navegue na aplicação simulando o ambiente interno de um computador, como, por exemplo, observar a sequência de integração entre os processos do sistema operacional e a memória e a ocupação da CPU, ou analisar a funcionalidade dos algoritmos de escalonamento de processos. O aprendiz também pode na simulação entender como são salvas as informações em sua tabela de BCP. Todo o processo de execução do simulador pode ser realizado de acordo com o ritmo de aprendizado aplicado pelo aprendiz.

Dessa forma o ESORV foi concebido e construído. Foram realizados testes de funcionamento em um ambiente imersivo virtual de multiprojeção, o mini-CAVE da UFSCAR (Universidade Federal de São Carlos) e com o óculos virtual de papelão da Google, o *google cardboard*. Também é possível executar simulação semi imersiva por meio dos computadores pessoais. Os testes de aplicabilidade do ESORV e a comparação com outro método de simulação foram realizados no IFPR - Instituto Federal do Paraná - Campus Palmas (PR).

6. TESTES

Este capítulo demonstra os testes aplicados ao simulador 3D desenvolvido nesse trabalho, o ESORV, e apresenta seus resultados. A seção 6.1 discute os tipos de testes que foram aplicados e as suas características (metodologia, conteúdo e questionários). A seção 6.2 apresenta o cenário da realização dos testes (local e recursos utilizados), o perfil dos participantes incluindo alunos e professores. A seção 6.3 apresenta os resultados obtidos. A seção 6.4 finaliza o capítulo, demonstrando uma breve discussão sobre os resultados.

6.1. Tipos de Testes e Metodologia

Nesta seção, primeiramente são apresentadas as comparações dos testes que foram utilizadas, os principais pontos do conteúdo que foram inseridos, os questionários e a metodologia.

Os testes aplicados nesta dissertação embasaram-se na comparação dos resultados das médias entre acertos e erros dos grupos dos indivíduos. Para isso, utilizam o conceito estatístico chamado de *teste t de Student* criado em 1908 por William Sealy Gosset. É o método mais utilizado para se avaliar as diferenças entre as médias de dois grupos. Os testes possuem amostras de resultados de dois conjuntos de resultados de médias diferentes e são chamados de *teste t* independente quando os grupos de alunos são diferentes e realizam somente um teste e *teste t* pareado quando os grupos de alunos são diferentes, porém os mesmos indivíduos realizam dois tipos de testes, um teste chamado de Pré Teste e após os testes com o ESOR e o SOSIM.

Embora o ponto principal dos testes não é coletar informações qualitativas, alguns questionamentos não obrigatórios foram inseridos aos testes com o objetivo de ter informações com o enfoque da usabilidade e funcionalidade do ESORV. Alguns entrevistados descreveram as percepções positivas e negativas sobre a utilização do ESORV. Por exemplo, no questionamento: "*Qual a vantagem do uso da realidade virtual para o processo de ensino-aprendizagem*?". A resposta do indivíduo 7: "O aprendizado se torna mais fácil, no caso do experimento, você consegue visualizar todo o processo realizado como se estivesse dentro do computador", já o indivíduo 16 respondeu: "Aumenta consideravelmente a vontade de aprender e deixa mais legal o ensino". Na pergunta: *"Qual é a desvantagem do uso da realidade virtual para o processo de ensino-*

aprendizagem?". A resposta do indivíduo 6 foi: "Se perde nos processos" e o indivíduo 11 relatou: "A falta de equipamento adequado". Assim pode-se observar apontamentos gerais de alguns indivíduos que poderão contribuir para melhorias futuras do ESORV.

A Figura 33(a) e 33(b) apresentam os tipos de testes que foram empregados. É possível observar que foram utilizados os *Testes t de Student independente e pareado* para dois modelos de grupos de testes.

- **Grupo 1 – Testes independentes**
  - **1.1. Comparação do SORV com o método tradicional de ensino (instrutivista)**: faz análise da comparação entre o modelo tradicional de ensino em salas de aulas, empregado em larga escala no ensino de escalonamento de Sistemas Operacionais, com a proposta da dissertação, o simulador em perspectiva 3D, o ESORV.
  - **1.2. Comparação do SORV com outro simulador de ensino (2D) - SOSIM**: faz a análise de comparação entre uma ferramenta de simulação similar ao propósito desse projeto, que usa animações em perspectiva 2D, e é conhecida como SOSIM. Essa proposta é comparada ao dessa dissertação, o simulador em perspectiva 3D, o ESORV.
- **Grupo 2 – Teste dependentes ou pareados**
  - **2.1. Comparação do Pré Teste com o método tradicional de ensino (instrutivista)**: faz a análise de comparação entre a média do grupo de respostas dos testes executados no início dos procedimentos (Pré Teste) com a média da base de respostas após a aplicação do método tradicional de aulas.
  - **2.2. Comparação do Pré Teste com o simulador ESORV**: faz análise de comparação entre a média do grupo de respostas dos testes executados no início dos procedimentos (Pré Teste) com a média da base de respostas após a aplicação do simulador ESORV.

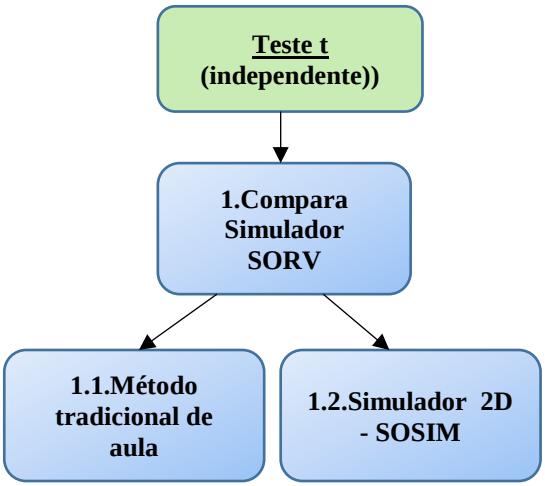


Figura 33 (a) - Tipo de Teste Independente

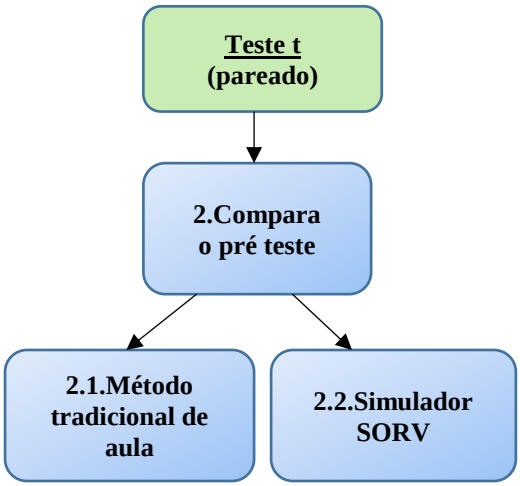


Figura 33 (b) - Tipo de Teste Pareado

Os principais pontos que foram questionados nos testes são relacionados diretamente ao escalonamento de processos da disciplina de Sistemas Operacionais. A Tabela 7 destaca os conteúdos inseridos na aplicação dos testes.

Tabela 7: Principais Tópicos dos Testes

Tópico do Conteúdo	Avaliação do Aprendizado
Processo	Avaliar o conceito de processos no sistema operacional e seus estados. Observar os contextos envolvidos: software, hardware e endereçamento.
Transição de estados de um processo	Analisar em quais situações os processos podem transitar de um estado para outro.
Escalonamento da fila de prontos	Observar o conceito de escalonamento e observar a formação da fila de prontos.
Função do despachante do escalonador do sistema operacional	Aprender qual a função exercida pelo despachante do sistema operacional no contexto de escalonar.
Bloco de Controle de Processos - BCP	Avaliar qual a utilidade do BCP no contexto de escalonamento de processos
Mudança de Contexto	Observar como é realizado o procedimento da mudança de contexto entre hardware e software dos processos.
Escalonamento dos processos para a CPU	Medir qual o conceito de escalonamento para o sistema operacional e observar o procedimento de ocupação da CPU.
Tipos de Processos	Analisar e distinguir os conceitos de processos CPU *bound* e I/O *bound*

A técnica empregada para a coleta e investigação das informações dos entrevistados foi por meio de questionário. A Tabela 8 demonstra as principias características de cada questionário.

Tabela 8: Tipos de Questionários para os Testes

Questionário	Objetivo	Anexo
Pré Teste	Construir uma base inicial de respostas dos testes para ser comparada com outras bases.	(Anexo I)
Método Tradicional	Construir uma base de respostas utilizando o método de ensino tradicional de aula para ser comparada com outras bases.	(Anexo II)
Método SOSIM	Construir uma base de respostas utilizando o método de ensino com uma ferramenta com perspectiva 2D para ser comparada com outras bases.	(Anexo III)
Método SORV	Construir uma base de respostas utilizando o método de ensino com uma ferramenta 3D, virtual, construída nessa dissertação para ser comparada com outras bases.	(Anexo IV)

E por último, para poder fazer as comparações entre os testes, a metodologia utilizada para a aplicação obedeceu aos seguintes critérios:

- Todos os entrevistados foram divididos aleatoriamente em duas turmas;
- Uma das turmas seguiu para a sala para responder ao questionário após o uso de um dos métodos, quais sejam, SOSIM ou Tradicional;
- A outra turma seguiu para uma sala diferente para responder ao questionário após o uso da solução construída nessa dissertação, o ESORV. Para isso usou o questionário chamado método ESORV.

Assim, pode-se construir as bases de respostas dos entrevistados para compilar os resultados e fazer as comparações para evidenciar os resultados finais.

6.2. Cenário e Participantes

Os testes foram aplicados às turmas de alunos do primeiro, segundo, quarto e sexto semestres do curso de Bacharelado em Sistemas de Informação, Artes Visuais e

Administração de Empresas do Instituto Federal do Paraná (IFPR) campus de Palmas (PR). Os alunos ainda não tiveram contato com a disciplina de Sistemas Operacionais, que é aplicada no curso de graduação A Tabela 9 demonstra o perfil do cenário em que os testes foram realizados.

Tabela 9: Cenário de Aplicação dos Testes

<table>
<tr><th colspan="2">L o c a l i d a d e</th></tr>
<tr><td colspan="2">Local: IFPR – Instituto Federal do Paraná – Campus Palmas (PR)
Av. Bento Munhoz da Rocha Neto, PRT 280, Palmas - PR, CEP: 85555-000
Telefone: (46) 3263-8900</td></tr>
<tr><th colspan="2">P ú b l i c o A l v o</th></tr>
<tr><td colspan="2">Curso: Bacharelado em Sistemas de Informação/Artes Visuais/Adm. Empresas
Semestres: 1,2, 4 e 6 Quantidade alunos: 110</td></tr>
<tr><th colspan="2">P a r t i c i p a n t e s</th></tr>
<tr><td>Prof.: Tarlis Portella
Formação: Mestre Engenharia Produção
Tempo de Docência: 5 anos
Disciplina: Sistemas Operacionais
Possui experiência no módulo de ensino da ferramenta: Sim (há 4 anos)</td><td>Prof: Vagner Scamati
Formação: Especialista em Desenvolvimento de Software para WEB
Tempo de Docência: 15 anos
Disciplina: Sistemas Operacionais
Possui experiência no módulo de ensino da ferramenta: Sim (há 8 anos)</td></tr>
<tr><th colspan="2">R e c u r s o s</th></tr>
<tr><td>Hardware:
Laboratórios de Computação do IFPR campus Palmas.
40 micros DELL - INTEL CORE 2 QUAD CPU 6600 2.4 Ghz - 2GB RAM HD 80 GB, Monitor 15 polegadas LCD.</td><td>Software:
- Simulador 2D de ensino de Sistemas Operacionais SOSIM.
- Simulador 3D de ensino de Sistemas Operacionais SORV.</td></tr>
</table>

A Figura 34 visualiza as fotos das etapas de realização dos testes; as fotos (a), (b) e (c) apresentam os alunos que participaram dos testes em seu ambiente de aplicação de cada sessão nos laboratórios de computação do IFPR. Cada laboratório está sendo

monitorado por um professor responsável pela aplicação dos testes e recolhimento dos questionários.

Figura 34 (a) - Cenário de execução dos testes

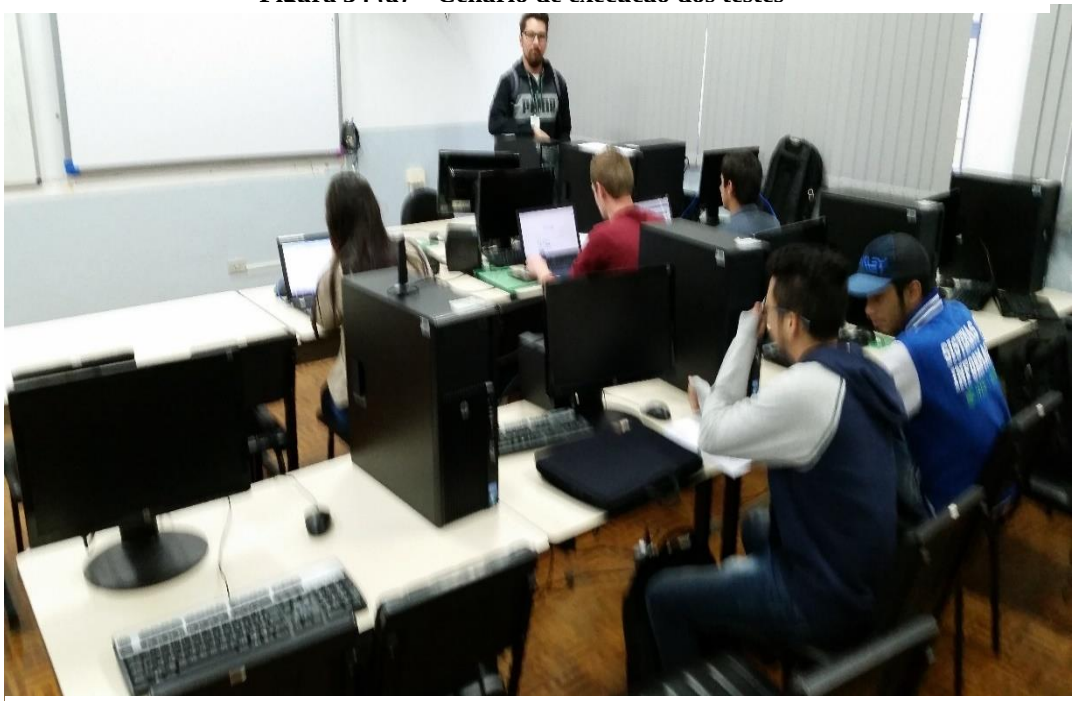


Figura 34 (b) - Cenário de execução dos testes

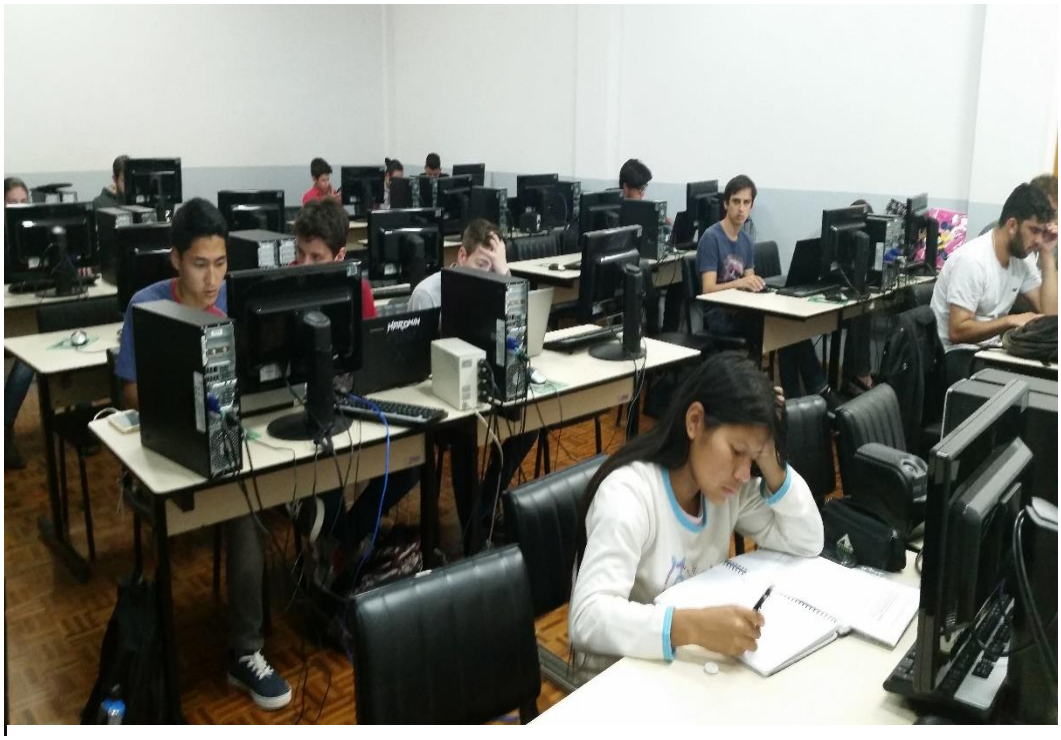


Figura 34 (c) - Cenário de execução dos testes

6.3. Resultados

Essa seção apresenta os resultados obtidos nos testes e suas interpretações. Os resultados estão relacionados à comparação do ESORV com o método tradicional de aula (instrutivista) e com uma ferramenta 2D, o SOSIM.

6.3.1. Resultados dos testes da comparação do ESORV com o método tradicional de aula (instrutivista)

O total de entrevistados nessa comparação foi uma amostra de 44 entrevistados, que foram divididos aleatoriamente em dois grupos. O Grupo 1, com 22 entrevistados, foi composto por aqueles que usaram o método tradicional de ensino (instrutivista) e responderam ao questionário chamado de método tradicional. O Grupo 2, com os outros 22 entrevistados, foi composto por aqueles que usaram o ESORV e responderam ao questionário chamado de método ESORV. Cada grupo teve aproximadamente 15 minutos de instruções e aula.

Inicialmente realizou-se a comparação entre o Grupo 1 (método tradicional) e o Grupo 2 (ESORV). A Figura 35 demonstra as quantidades de erros e acertos para cada grupo. É possível visualizar que o Grupo 1 obteve uma frequência de acertos menor que o Grupo 2, porém a quantidade de erros foi superior aos do Grupo 2. Isso significa que o ESORV obteve um desempenho de acertos maior e com um número de erro menor quando comparado com o método tradicional de aula (instrutivista).

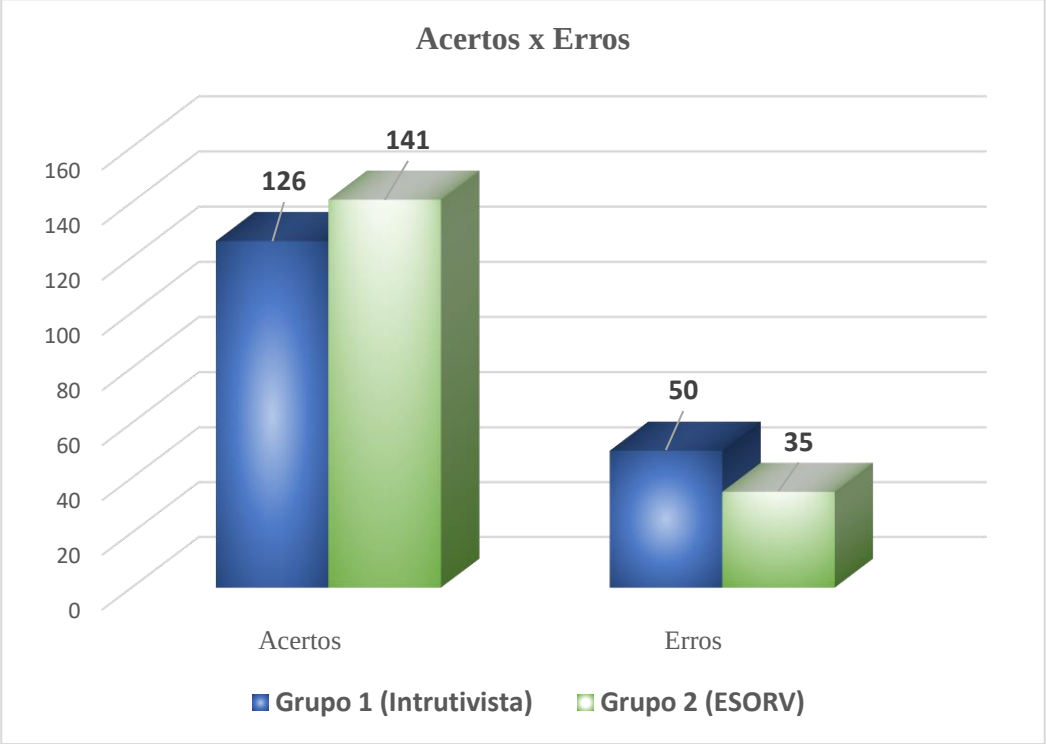


Figura 35 - Número de acertos e erros por grupo (Instrutivista x ESORV)

A Figura 36 apresenta as medidas características de cada grupo. Pode-se perceber que para o Grupo 2 a média de acertos dos indivíduos teve um acréscimo se comparada à média do Grupo 1. As medidas de dispersão revelam que no Grupo 2 a variabilidade dos resultados foi um pouco menor, isto é, o número de acertos dos participantes desse grupo se concentra mais em torno da média do que no Grupo 1. O desvio padrão foi maior no Grupo 1 do que no Grupo 2.

A partir dos resultados, principalmente os das médias de cada Grupo, pode ser feita uma comparação. É possível, então, calcular uma relação que comprove a efetividade do uso do ESORV como ferramenta de apoio ao ensino de escalonamento de Sistemas Operacionais.

Para comparar as médias do Grupo 1 com as do Grupo 2 foi utilizado o *Test t* independente. As hipóteses do *Test t* são:

H0: µGrupo1 = µGrupo2 (os grupos apresentam médias iguais, hipótese nula)

H1: µGrupo1 ≠ µGrupo2 (os grupos apresentam médias de acertos diferentes, hipótese alternativa)

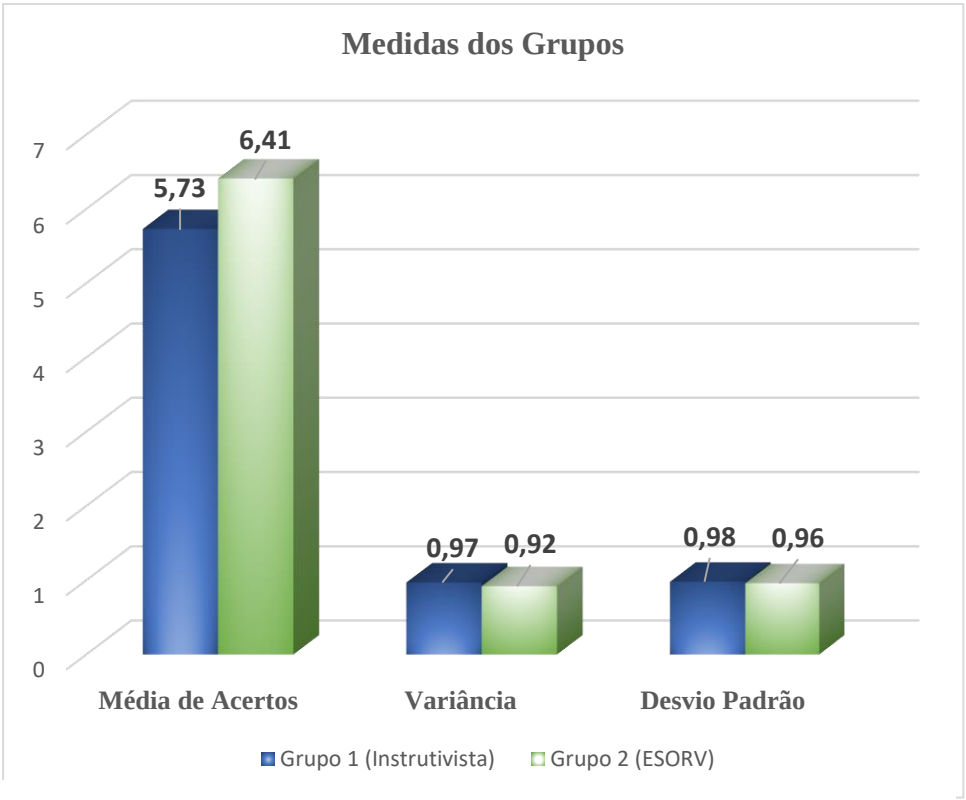


Figura 36 - Medidas características por grupo (Instrutivista x ESORV)

A tabela 10 demonstra o resultado do cálculo do valor p com nível de significância de 5% ($\alpha = 5\%$) e $n_1 + n_2 - 2$ graus de liberdade.

Tabela 10: Resultado Estatístico com o Test t independente (Instrutivista x ESORV)

Teste t independente	
t	**-2, 3265**
Graus de liberdade	**42**
Valor-p	***0,0249***

Como o resultado do valor p é menor que 0,05, temos evidências para rejeitar a hipótese nula, isto é, o resultado indica que há uma diferença significativa entre as médias

das duas amostras. Assim, conclui-se que o aumento nos acertos para o Grupo 2 é significativo, ou seja, a utilização do ESORV apresenta um ganho de aprendizado quando comparado ao método tradicional de ensino (instrutivista).

Para comparar o número de acertos de cada grupo com os acertos no pré-teste, utilizou-se o ***Teste t de Student pareado***, pois nesse caso as amostras são dependentes. A tabela 11 apresenta a comparação da média do Grupo 1 com a média da base de respostas do questionário chamado de Pré teste para esses mesmos entrevistados, o cálculo leva em consideração uma taxa de erro de 5% (**α** = 5%) e n – 1 graus de liberdade.

Tabela 11: Resultado Estatístico com o Test t pareado do Grupo 1

Teste t pareado do Grupo 1	
Médias	
Grupo 1: 5,73	**Pré-teste: 4, 136**
t	**6, 3154**
Graus de liberdade	**21**
Valor-p	**0,00000291**
Diferença estimada entre as médias	***1, 5909***

A tabela 12 apresenta a comparação da média do Grupo 1 com a média da base de respostas do questionário Pré teste para esses mesmos entrevistados, o cálculo leva em consideração uma taxa de erro de 5 (**α** = 5%) e n – 1 graus de liberdade.

Tabela 12: Resultado Estatístico com o Test t dependente do Grupo 2

Teste t pareado do Grupo 2	
Médias	
Grupo 2: 6,41	**Pré-teste: 3,773**
t	**10,8872**
Graus de liberdade	**21**
Valor-p	**0,000000000428**
Diferença estimada entre as médias	***2,6363***

Os resultados evidenciam que ambos os grupos obtiveram médias de acertos maiores do que a média obtida no Pré-teste, sendo que para o Grupo 2 a diferença estimada é superior à do Grupo 1, ou seja, a utilização de ambas as soluções: o ESORV e do método tradicional de aula; apresentam uma evidencia de ganho de aprendizado quando comparados a base de respostas do Pré teste (sem a aplicação de nenhum método).

6.3.2. Resultado dos testes da comparação do ESORV com a ferramenta 2D - SOSIM

Nessa comparação, o total foi uma amostra de 66 entrevistados, que foram divididos aleatoriamente em dois grupos. O Grupo 1, com 33 entrevistados, foi composto por aqueles que usaram a ferramenta de simulação 2D para o ensino. Eles responderam ao questionário chamado de método SOSIM. O Grupo 2, com os outros 33 entrevistados, foi composto por aqueles que usaram o ESORV e responderam ao questionário chamado de método ESORV.

Inicialmente realizou-se a comparação entre o Grupo 1 (SOSIM) e o Grupo 2 (ESORV) com o objetivo de avaliar as duas propostas. A Figura 37 demonstra as quantidades de erros e acertos para cada grupo. É possível visualizar que o Grupo 1 obteve uma frequência de acertos menor que a do Grupo 2; assim, a quantidade de erros foi superior à do Grupo 2. Isso significa que o ESORV obteve um desempenho de acertos maior e com um número de erro menor quando comparado com o SOSIM.

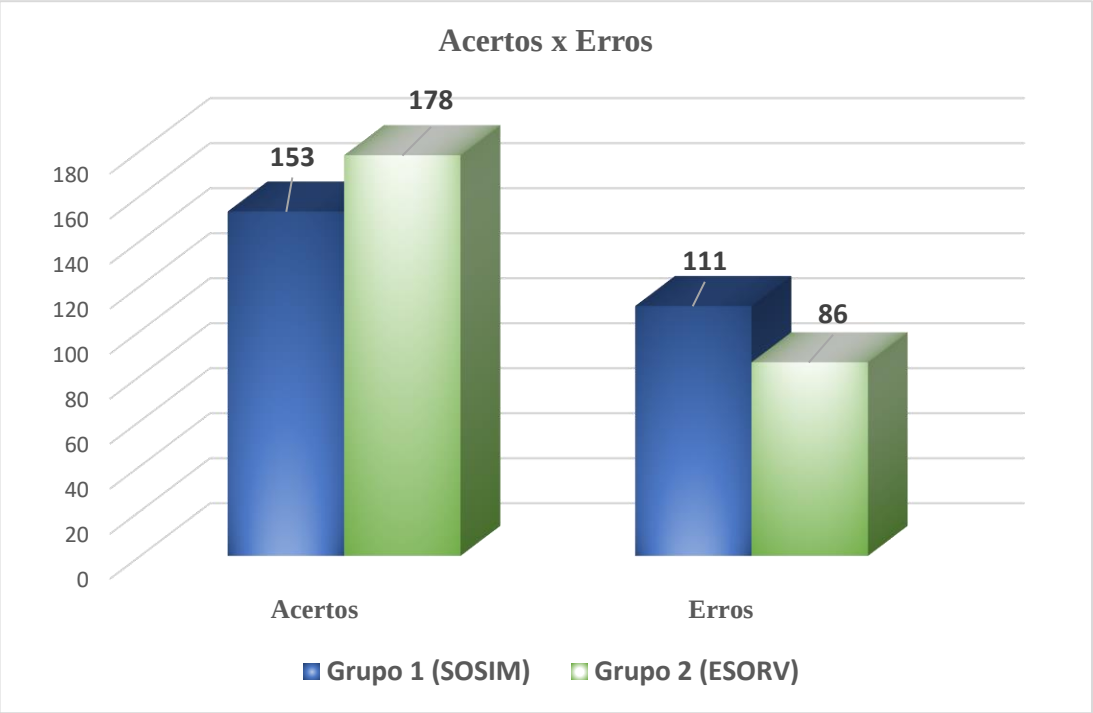


Figura 37 - Números de acertos e erros por grupo (SOSIM x ESORV)

A Figura 38 apresenta as medidas características de cada grupo. Pode-se perceber que para o Grupo 2 (ESORV) a média de acertos dos indivíduos teve um acréscimo se comparada com a do Grupo 1 (SOSIM). As medidas de dispersão revelam que no Grupo

2 a variabilidade dos resultados foi um pouco menor, isto é, o número de acertos dos participantes desse grupo se concentra mais em torno da média do que no Grupo 1.

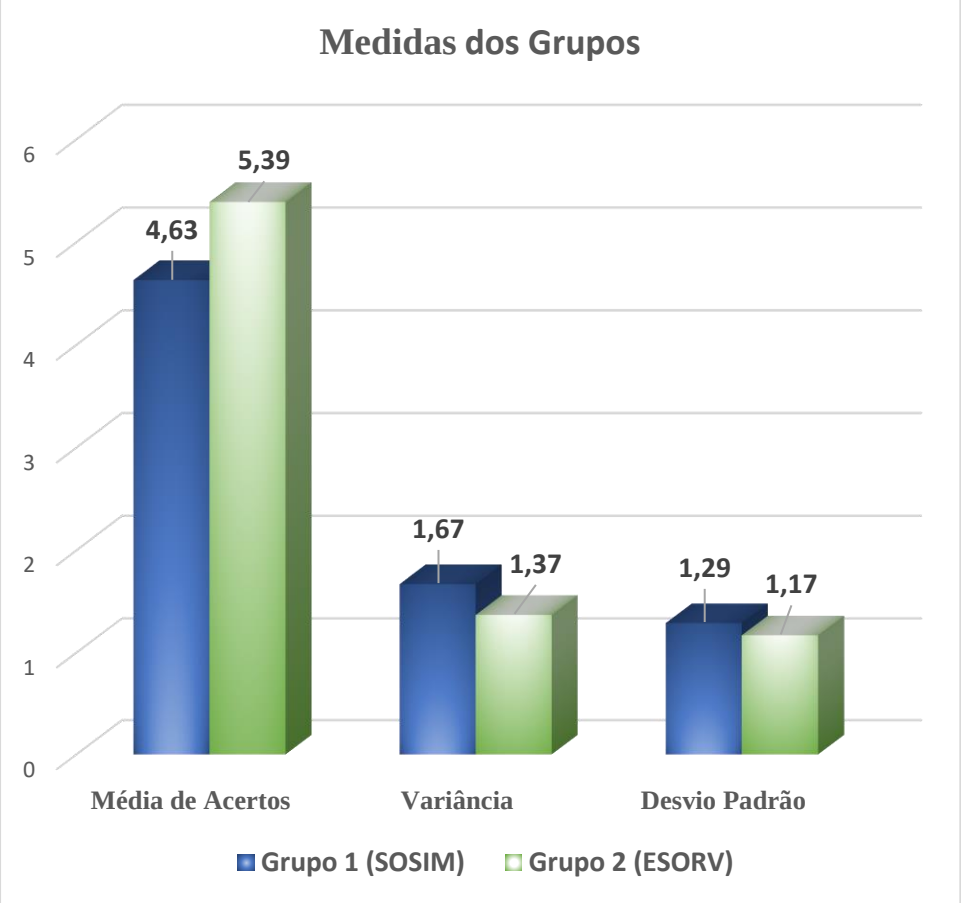


Figura 38 - Medidas características por grupo (SOSIM x ESORV)

A partir dos resultados, principalmente o das médias de cada Grupo, pode ser feita uma comparação. É possível então calcular uma relação que comprove a efetividade do uso do ESORV como ferramenta de apoio ao ensino de escalonamento de Sistemas Operacionais comparada a outras soluções.

Para comparar as médias do Grupo 1 com as do Grupo 2, foi utilizado o *Test t* independente. As hipóteses do *Test t* são:

H0: μGrupo1 = μGrupo2 (os grupos apresentam médias iguais, hipótese nula)

H1: μGrupo1 ≠ μGrupo2 (os grupos apresentam médias de acertos diferentes, hipótese alternativa)

A tabela 13 demonstra o resultado do cálculo do valor p com nível de significância de 5% ($\alpha = 5\%$) e $n_1 + n_2 - 2$ graus de liberdade.

Tabela 13: Resultado Estatístico com o Test t independente (SOSIM x ESORV)

Teste t independente	
t	**-2,493**
Graus de liberdade	**64**
Valor-p	***0,01529***

Como o valor p é menor que 0,05, temos evidências para rejeitar a hipótese nula, isto é, o resultado indica que há uma diferença significativa entre as médias das duas amostras. Portanto, conclui-se que o aumento nos acertos para o Grupo 2 é significativo, ou seja, a utilização do ESORV apresenta um ganho de aprendizado quando comparado à ferramenta de ensino SOSIM.

Para comparar o número de acertos de cada grupo com os acertos no pré-teste, utilizamos o ***Teste t de Student pareado***, pois nesse caso as amostras são dependentes. A tabela 14 apresenta a comparação da média do Grupo 1 com a média da base de respostas do questionário chamado de Pré teste para esses mesmos entrevistados, o cálculo leva em consideração uma taxa de erro de 5% (**α** = 5%) e n – 1 graus de liberdade.

Tabela 14: Resultado Estatístico com o Test t pareado do Grupo 1

Teste t pareado do Grupo 1	
Médias	
Grupo 1: 4,636	**Pré-teste: 3, 515**
t	**3,9869**
Graus de liberdade	**32**
Valor-p	**0,0003633**
Diferença estimada entre as médias	***1, 12***

A tabela 15 apresenta a comparação da média do Grupo 1 com a média da base de respostas do questionário Pré teste para esses mesmos entrevistados, o cálculo leva em consideração uma taxa de erro de 5 (**α** = 5%) e n – 1 graus de liberdade.

Tabela 15: Resultado Estatístico com o Test t pareado do Grupo 2

Teste t pareado do Grupo 2	
Médias	
Grupo 2: 5,394	**Pré-teste: 4,121**
t	**5,8172**
Graus de liberdade	**32**
Valor-p	**0,000001843**
Diferença estimada entre as médias	***1,27***

Os resultados evidenciam que ambos os grupos obtiveram médias de acertos maiores do que a média obtida no Pré teste, sendo que para o Grupo 2 a diferença estimada é superior à do Grupo 1, ou seja, a utilização de ambas as soluções: o ESORV e do SOSIM apresentam uma evidencia de ganho de aprendizado quando comparados a base de respostas do Pré teste (sem a aplicação de nenhum método).

6.4. Considerações Finais

Os testes realizados nessa dissertação tiveram o objetivo de avaliar se houve evidência de ganho de aprendizado utilizando o ESORV na relação ensino aprendizado. Foram utilizados *Teste t de Student independente e pareado* em dois tipos de testes de comparações: um com o método tradicional de aula (instrutivista) e o outro com um simulador 2D, o SOSIM.

Embora os dois tipos de testes tenham demonstrado diferenças em algumas métricas envolvidas, os resultados se apresentaram similares e demonstram que principalmente as médias dos grupos possuem diferença significativa nas comparações. Isso evidencia e dá um norte que ao utilizar o ESORV como apoio ao ensino de escalonamento de processos em Sistemas Operacionais é possível que haja um ganho de aprendizado.

7. CONCLUSÕES

Atualmente, a demanda por novas soluções tecnológicas que apoiem a relação ensino-aprendizagem é crescente. O ESORV pode ser considerado um recurso com grande potencial a ser explorado. Este capítulo apresenta o fechamento da dissertação com suas conclusões. A seção 7.1 demonstra as principais contribuições da pesquisa. A seção 7.2 apresenta os trabalhos futuros. A seção 7.3 faz as considerações finais dessa dissertação.

7.1. Contribuições da pesquisa

O simulador ESORV foi o resultado da pesquisa realizada nessa dissertação. Ele foi desenvolvido para que o aluno possa ter um ganho de aprendizado no ensino de escalonamento de processos na disciplina de Sistemas Operacionais.

O ESORV, diferentemente de outros simuladores, é inovador. Fundamentado na tecnologia de realidade virtual, fazendo o uso dos propósitos dessa tecnologia, por exemplo, navegar em um ambiente de aprendizagem 3D, sendo possível envolver-se na construção do seu próprio conhecimento de maneira exploratória e interativa. Com essa abordagem tecnológica, é possível simular situações de ensino-aprendizagem nas quais o aluno aprendiz possa melhorar seu entendimento de conceitos abstratos na área do ensino da computação. O uso da tecnologia da realidade virtual na construção deste protótipo tornou-se um diferencial em relação a outros simuladores. Além de contribuir para o ensino, apresenta uma potencialidade a ser explorada com o avanço da tecnologia, por exemplo, a simples aquisição do *google cardboard*, pode colaborar com a popularização do ESORV entre acadêmicos e usar toda a potencialidade existente da realidade virtual aplicada no simulador a favor do aprendiz.

Essa dissertação incluiu um estudo possível de observar e analisar o comportamento que a realidade virtual possa demonstrar como ferramenta de ensino-aprendizagem. As análises realizadas a partir dos resultados dessa pesquisa evidenciam que ao utilizar o ESORV, pode-se contribuir positivamente no processo de aprendizagem. Isso responde a principal questão que norteia essa pesquisa, conforme apresentada no início dessa dissertação: “Como contribuir e melhorar a relação de ensino-aprendizado para a disciplina de Sistemas Operacionais? ”.

As principais contribuições destacadas com a utilização do ESORV foram na área de computação: oferecendo uma solução tecnológica para minimizar o problema da absorção do aprendizado dos conceitos abstratos encontrados no ensino de Sistemas Operacionais; e na área de educação, na qual houve uma melhoria no ganho de aprendizagem.

A partir dos resultados obtidos também foi possível visualizar e identificar desafios futuros, como sugestões de melhorias que poderão ser implementadas.

7.2. Trabalhos futuros

Como trabalho futuro, são propostas as seguintes orientações:

- Realizar testes com grupos maiores e mais heterogêneos no que diz respeito à relação ensino-aprendizagem dos alunos;
- Testes em ambientes diversos de projeção (por exemplo, Cavernas Digitais e capacetes de visualização);
- Desenvolver novos módulos sobre a disciplina de Sistemas Operacionais, por exemplo: Gerência de Memória;
- Implementar uma solução do ESORV para a arquitetura de processadores de vários núcleos (multiprocessado);
- Ampliar a forma de interação do ESORV com o aluno.

7.3. Considerações finais

A realidade virtual vem sendo explorada atualmente em projetos importantes de algumas áreas, que vão desde a cirurgia virtual e treinamentos com simuladores até o apoio na relação ensino-aprendizagem em áreas diferentes: geografia, química e outras. Contudo, tem sido pouco utilizada no apoio ao ensino da Computação, especificamente na disciplina de Sistemas Operacionais.

Essa dissertação explorou no contexto computacional, os benefícios que a realidade virtual pode entregar para um projeto de simulação de ensino-aprendizagem. Nessa perspectiva, todas as evidências e possibilidades de ganho final de aprendizagem apresentada nos resultados obtidos estão apoiadas e concentradas na utilização da tecnologia de realidade virtual. É possível dizer que todo o esforço computacional

aplicado ao projeto, que vai desde a estratégia de desenvolvimento do simulador até a utilização de qual tipo de tecnologia seria aplicado ao ESORV teve uma relação direta de impacto nos resultados de ganho de aprendizagem.

Essa pesquisa apresenta resultados preliminares que podem evidenciar o êxito do uso do ESORV. Eles sugerem uma possível melhora no desempenho dos grupos de alunos que fizeram uso do ESORV por meio da realidade virtual.

O simulador tem grande potencial para contribuir e melhorar o ensino de escalonamento de processos da disciplina de Sistemas Operacionais, transformando a aprendizagem em uma situação de exploração, descoberta, observação e construção de conhecimento ao criar para os alunos uma nova forma de compreender o conteúdo. Essa abordagem pode impactar os alunos nos processos cognitivos, auxiliando a sua compreensão, fornecendo experiência prática de conceitos abstratos e indo ao encontro dos conceitos da teoria do Construtivismo.

Por fim, um aspecto de cunho pessoal: esse trabalho me proporcionou conhecer novos assuntos e informações, contribuindo para o meu crescimento profissional e pessoal, e assim, permitindo que explorasse mais o tema principal discutido nessa dissertação com o objetivo de expandir e melhorar constantemente a solução desenvolvida.

REFERÊNCIAS

Anderson, E. F., Mcloughlin, L., Watson, J., Holmes, S., Jones, P., Pallett H., Smith, B. (2013), "Choosing the infrastructure for entertainment and serious computer games - a whiteroom benchmark for game engine selection". in*:* Games and Virtual Worlds for Serious Applications (VS-GAMES), 2013 5th International Conference on. [S.l.: s.n.], p 1–8.

Anido, R. (2000), "Uma proposta de plano pedagógico para a matéria Sistemas Operacionais", in: Anais do II Curso de Qualidade, Workshop sobre Educação em Computação, XX Congresso da Sociedade Brasileira de Computação, Curitiba, PR, pp 125–148.

Araújo, R. B. "Especificação e análise de um sistema distribuído de realidade virtual", São Paulo, Junho, 144 Pp., Tese (Doutorado), Departamento de Engenharia de Computação e Sistemas Digitais, Escola Politécnica da Universidade de São Paulo, 1996.

Banerjee, I., (2013), "Introducing immersive technologies for learning: A study evaluating the readiness for using virtual worlds for learning and teaching in an Indian University", Contemporary Computing (IC3), Sixth International Conference on.

Bowman, D., (2005), "3D users Interfaces: Theory and Practice". Boston, MA: Addison-Wesley".

Callaghan, M. J., (2013), 'Using Game-Based Learning in Virtual Worlds to Teach Electronic and Electrical Engineering", Industrial Informatics, IEEE Transaction on, Vol. 9, Issue 1.

Carvalho, D. S., Balthazar, G. R., Dias, C. R., Araújo, M. A. P. and Monteiro, P. H. R. (2006), "S²O: Uma Ferramenta de Apoio ao Aprendizado de Sistemas Operacionais", Anais do XIV Workshop sobre Educação em Computação, Campo Grande/MS.

Clark, D. (2006) "Motivation in e-learning". Disponível em http://leolearning.com/our-expertise/learning-strategy/. Acesso em 01/10/2016.

Cruz-Neira, C., (1992), "The CAVE audio visual experience automatic virtual environment", Communication of the ACM, 35(6):64-72.

Dede, C. (2009), “Immersive Interfaces for Engagement and Learning”, Science Magazine, V. 323, p. 66-69.

Dias, D. R. C., Brega, J. R. F., Trevelin, L. C., Neto, M. P., Gnecco, B. B. and Guimarães, M. de P. (2012), “Design and evaluation of an advanced virtual reality system for visualization of dentistry structures”, in: Virtual Systems and Multimedia (VSMM), 18th International Conference on. [S.l.: s.n.], p. 429–435.

Diaz, A., Batista, R. and Castro, O. (2007), “Realtss: a real-time scheduling simulator”, Electrical and Electronics Engineering 2007. ICEEE 2007. 4th International Conference on, pp. 165-168.

Fengru Huang, (2013), “Three-dimensional construction and visualization of complex geologic environments for virtual field practice and virtual Education”, Geoinformatics (GEOINFORMATICS), 21st International Conference.

Google, “Google cardboard – google vr,”. Disponível em https://vr.google.com/cardboard/index.html. Acesso em: 01/10/2016.

Guimarães, P. M., and Martins F. V. (2012) “Desafios para o uso de Realidade Virtual e Aumentada de maneira efetiva no ensino”, Workshop de Desafios da Computação Aplicada à Educação.

Imai, Y. and Takeichi K., (2015), “Development and evaluation of adobe flash based cpu scheduling simulator executable on major multiple web browsers,” in: Intelligent Networking and Collaborative Systems (INCOS), International Conference on, pp. 149–155.

Kirner, C. and Pinho, M.S. (1996), “Introdução a Realidade Virtual”, Mini curso, JAI/SBC, Recife, PE.

Kirner, C., Siscoutto, R.A., (2008), “Fundamentos de Realidade Virtual e Aumentada”, in: EDS: Uma Abordagem Tecnológica. 2008. pp. 1-20.

Kirner, C., and Kirner, T.G., (2011), “Evolução e tendências da Realidade Virtual e da Realidade Aumentada”, in: M. W. de S. Ribeiro, E. R. Zorzal, EDS., Realidade Virtual e Aumentada: Aplicações e Tendências, XIII Symposium on Virtual and Augmented Reality. Uberlândia: Sociedade Brasileira de Computação - SBC, 2011. pp. 8-23.

Machado, B., F. and Maia, P. L. (2013), “Arquitetura de Sistemas Operacionais”, 5ª. Edição, Ed. LCT, Rio de Janeiro, RJ.

Maziero, C. A. (2002). “Reflexões sobre o ensino prático de Sistemas Operacionais", Anais do X Workshop sobre Educação em Computação (WEI2002), XXII Congresso da SBC.

Maziero, C. A. (2013). “Sistemas Operacionais: Conceitos e Mecanismos”. Online: http://wiki.inf.ufpr.br/maziero/lib/exe/fetch.php?media=so:so-livro.pdf. Acesso em 01 de julho de 2016.

Piaget, J. (1932). “The Moral Judgement of the Child”, 1965, New York: Free Press.

Pimentel, K. and Teixeira, K. (1995), “Virtual reality - through the new looking glass”, 2ª. Edição, New York, McGraw-Hill.

Popolin Neto, M., Agostinho, I. A., Dias, D. R. C., Rodello, I. A., Brega, J. R. F. and Rebelo, I. B. (2004), “ Proposta de uma Ferramenta de Verificação dos Procedimentos de Interação em Sistemas de Realidade Virtual”. Florianópolis, Abril 2004, Tese (Doutorado), Engenharia de Produção da Universidade Federal de Santa Catarina.

Saraswat, P., H. and Gupta, P. "Design and implementation of a process scheduler simulator and an improved process scheduling algorithm for multimedia operating systems", International Conference on Advanced Computing and Communications, pp. 513-517.

Shaw, Alan C. (2003), “Sistemas e Software De Tempo Real”, Bookman Companhia 2ª Edição.

Silberschatz, A., Gagne E, G. and Galvin, P. B. (2015), “Fundamentos de Sistemas Operacionais”, Ed. LCT, (9ª. edição), Rio de Janeiro, RJ.

Silva, L. F., Cardoso, A., Mendes, E. B. and Lamounier, E. (2005). “Associando Ferramentas Cognitivas e Realidade Virtual não Imersiva para o ensino de Física”, in: IADS – Conferência Ibero Americana.

Suranauwarat, S. (2007), “A CPU Scheduling Algorithm Simulator”, October 10-13, Milwaukee, WI 37th ASEE/IEEE Frontiers in Education Conference.

Scamati, V., Dias, D. R. C., Guimarães, M. de P. and Martins, F. V. (2015), "Utilização de ambientes virtuais para o ensino", XV Symposium on Virtual and Augmented Reality. Presidente Prudente, pp-165-169.

Sociedade Brasileira de Computação (SBC) (2003). "Currículo de referências para cursos de graduação de Computação e Informática". Disponível em: http://www.sbc.org.br/documentos-da-sbc/send/131-curriculos-de-referencia/764-curriculo-de-referencia-is-versao-2003. Acesso em: 12 setembro 2016.

Stallings, Willian (2010), "Arquitetura e Organização de Computadores", 8.ed., Pearson Pratice Hall, 2010, São Paulo.

Tanembaum, A. S. (2004a) (2009b). "Sistemas Operacionais Modernos", Ed. Pearson, São Paulo.

Tori, R. and Kirner, C. (2006), "Fundamentos de Realidade Virtual", in: VIII Simpósio de Realidade Virtual, Livro do Pré-Simpósio: Fundamentos e Tecnologia de Realidade Virtual e Aumentada, ISBN 857669068-3.

Umar, S. and Javed, M., Y. (2010), "Simulation of CPU Scheduling Alogrithm", IEEE – Institute Electrical and Electronics Engineering.

Vince, J. (1995), "Virtual Reality Systems". Reading, Massachusetts, Addison-Wesley. Visualization of dentistry structures, in: Virtual Systems and Multimedia (VSMM), 2012 18th International Conference on. [S.l.: s.n.], 2012. p. 429–435.

Voss, G. B. (2013), "Context-Aware Virtual Laboratory for Teaching Computer Networks: A Proposal in the 3D OpenSim Environment", Virtual and Augmented Reality (SVR), XV Symposium on M. Young, The Technical Writer's Handbook, University Science, Mill Valley, CA.

Wagner, R. (2013), "Using 3D virtual learning environments in new perspective of education", Information Technology Based Higher Education and Training (ITHET), International Conference on.

Apêndice I – Questionário de teste- Método Tradicional de aula (instrutivista) - elaborado pelo autor.

T E S T E S - A U L A (instrutivista)

Testes

Para responder as próximas questões, você deverá ler os itens das instruções e executar os procedimentos.

Instruções:

- A duração dos testes será a mesma da aula;
- 5 minutos para explicar como funciona os testes;
- 35 minutos realizando uma aula no modelo tradicional (instrutivista), sobre o conteúdo dos testes;
- 10 minutos para responder as questões.

Procedimentos da Aula:

- Explicar sobre conceito de processos, seus estados, comportamentos e transições.
- Explicar sobre escalonamento de processos, os tipos de escalonador e os principais algoritmos.
- Demonstrar o que é uma mudança de contexto e o que significa o BCP (Bloco de Controle de Processos).
- Observar a criação da fila de prontos, e a alternância entre o processador a fila de prontos e a fila de dispositivos de entrada e saída e como o processo se comporta nas suas transições.

Questões

Questões
1.Sobre processos: ☐ É um programa ativo ☐ É um software ☐ Não possui estados

2.Quando o processo sai da CPU e vai para a fila de dispositivos de entrada e saída, ele está transitando entre quais estados?

☐ Pronto-Execução ☐ Execução-Espera ☐ Espera-Pronto

3.Quando a fila de prontos é organizada através do *quantum* do processo, essa organização refere-se a qual tipo de algoritmo de escalonamento?

☐ Fifo ☐ Round Robin ☐ Prioridades

4.Quando um processo tem o uso intensivo de dispositivos de entrada e saída, ele é considerado?

☐ CPU *bound* ☐ I/O *bound* ☐ Memória *Bound*

5.O "despachante" do escalonador do sistema operacional, pode ser utilizado em qual situação?

☐ Quando o processo é conduzido da fila de prontos da memória principal para a CPU

☐ Quando o processo é conduzido da fila de prontos da memória principal para a fila de processos que estão em estado de espera

☐ Quando o processo é conduzido da CPU para a tabela de Bloco de Controle de Processos (BCP)

6.Sobre BCP (bloco de controle de processos):

☐ É uma tabela onde são armazenados os dados dos processos que foram interrompidos do uso da CPU. E esses dados serão recuperados quando necessário

☐ Não é uma tabela, e sim uma área que serve para o uso dos algoritmos de escalonamento

☐ É um local que só é usado para armazenar variáveis de programas

7.O tempo que um processo utiliza quando ocupa a CPU através de um algoritmo de escalonamento, é denominado?

☐ Quantum ☐ Overhead ☐ Skew

8. Quando um processo vai da memória principal para a CPU, ele está transitando entre quais estados?

☐ Execução-Espera ☐ Pronto-Execução ☐ Execução-Pronto

Apêndice II – Questionário de teste- Método ESORV (3D) - elaborado pelo autor.

T E S T E S - S O R V

Testes

Para responder as próximas questões, você deverá ler os itens das instruções e executar os procedimentos.

Instruções:

- A duração dos testes será a mesma da aula;
- Para executar os testes você deverá conferir se o simulador ESORV está instalado na área de trabalho do seu computador;
- Ler atentamente as instruções iniciais do ESORV para fazer a simulação. A tecla "H" pode ajuda-lo caso necessite;
- 10 minutos para explicar como funciona a simulação;
- 35 minutos realizando os procedimentos de interação e envolvimento;
- 10 minutos para responder as questões.

Procedimentos:

- Criar uma simulação com o tipo de algoritmo de escalonamento "Prioridade" – tecla (F1). Selecionar um processo (tecla "F") com a prioridade 1 e tipo de processo CPU *bound;*
- Quando esse processo criado estiver em execução na CPU, criar outro processo (tecla "F"), com a prioridade 2 e tipo de processo I/O *bound;*
- Observar a criação da fila de prontos, e a alternância entre o processador a fila de prontos e a fila de dispositivos de entrada e saída, como ocorre o salvamento dos dados no BCP e como o processo se comporta nas suas transições;
- Criar uma simulação com o tipo de algoritmo de escalonamento "Round Robin" - (tecla F2). Selecionar um processo (tecla "F") com o quantum de 10, tempo de execução de 20 segundos e tipo de processo CPU *bound;*

- Quando esse processo criado estiver em execução na CPU, criar outro processo (tecla “F”), com tempo de execução de 20 segundos e tipo de processos I/O *bound;*
- Observar a criação da fila de prontos, e a alternância entre o processador a fila de prontos e a fila de dispositivos de entrada e saída, como ocorre o salvamento dos dados no BCP e como o processo se comporta nas suas transições;
- Criar uma simulação com o tipo de algoritmo de escalonamento “SJF” (tecla F3). Selecionar um processo (tecla “F”) com o tempo de execução de 20 segundos e tipo de processo CPU *bound;*
- Quando esse processo criado estiver em execução na CPU, criar outro processo (tecla “F”), com tempo de execução de 20 segundos e tipo de processos I/O *bound;*
- Observar a criação da fila de prontos, e a alternância entre o processador a fila de prontos e a fila de dispositivos de entrada e saída e como o processo se comporta nas suas transições;

Questões

Questões
1.Sobre processos: ☐ É um programa ativo ☐ É um software ☐ Não possui estados
2.Quando o processo sai da CPU e vai para a fila de dispositivos de entrada e saída, ele está transitando entre quais estados? ☐ Pronto-Execução ☐ Execução-Espera ☐ Espera-Pronto
3.Quando a fila de prontos é organizada através do *quantum* do processo, essa organização refere-se a qual tipo de algoritmo de escalonamento? ☐ Fifo ☐ Round Robin ☐ Prioridades
4.Quando um processo tem o uso intensivo de dispositivos de entrada e saída, ele é considerado? ☐ CPU *bound* ☐ I/O *bound* ☐ Memória *Bound*
5.O “despachante” do escalonador do sistema operacional, pode ser utilizado em qual situação? ☐ Quando o processo é conduzido da fila de prontos da memória principal para a CPU

Quando o processo é conduzido da fila de prontos da memória principal para a fila de processos que estão em estado de espera

☐ Quando o processo é conduzido da CPU para a tabela de Bloco de Controle de Processos (BCP)

6.Sobre BCP (bloco de controle de processos):

☐ É uma tabela onde são armazenados os dados dos processos que foram interrompidos do uso da CPU. E esses dados serão recuperados quando necessário

☐ Não é uma tabela, e sim uma área que serve para o uso dos algoritmos de escalonamento

☐ É um local que só é usado para armazenar variáveis de programas

7.O tempo que um processo utiliza quando ocupa a CPU através de um algoritmo de escalonamento, é denominado?

☐ Quantum ☐ Overhead ☐ Skew

8. Quando um processo vai da memória principal para a CPU, ele está transitando entre quais estados?

☐ Execução-Espera ☐ Pronto-Execução ☐ Execução-Pronto

9.Assinale na escala abaixo, o quanto que você acha que o ESORV contribuiu para o aprendizado de escalonamento de sistemas operacionais.

☐ Muito Pouco ☐ Pouco ☐ Indiferente ☐ Muito ☐ Muitíssimo

10.Qual a vantagem do uso da Realidade Virtual para o processo de ensino-aprendizagem?

11. Qual a desvantagem do uso da Realidade Virtual para o processo de ensino-aprendizagem?

Apêndice III – Questionário de teste- Método SOSIM (2D) - elaborado pelo autor.

T E S T E S - S O S I M

Testes

Para responder as próximas questões, você deverá ler os itens das instruções e executar os procedimentos.

Instruções:

- A duração dos testes será a mesma da aula;
- Para executar os testes você deverá conferir se o simulador ESORV está instalado na área de trabalho do seu computador;
- Ler atentamente as instruções iniciais do ESORV para fazer a simulação. A tecla "H" pode ajuda-lo caso necessite;
- 10 minutos para explicar como funciona a simulação;
- 35 minutos realizando os procedimentos de interação e envolvimento;
- 10 minutos para responder as questões.

Procedimentos:

- Criar uma simulação com o tipo de algoritmo de escalonamento "Prioridade" – tecla (F1). Selecionar um processo (tecla "F") com a prioridade 1 e tipo de processo CPU *bound;*
- Quando esse processo criado estiver em execução na CPU, criar outro processo (tecla "F"), com a prioridade 2 e tipo de processo I/O *bound;*
- Observar a criação da fila de prontos, e a alternância entre o processador a fila de prontos e a fila de dispositivos de entrada e saída, como ocorre o salvamento dos dados no BCP e como o processo se comporta nas suas transições;
- Criar uma simulação com o tipo de algoritmo de escalonamento "Round Robin" - (tecla F2). Selecionar um processo (tecla "F") com o quantum de 10, tempo de execução de 20 segundos e tipo de processo CPU *bound;*

- Quando esse processo criado estiver em execução na CPU, criar outro processo (tecla “F”), com tempo de execução de 20 segundos e tipo de processos I/O *bound;*
- Observar a criação da fila de prontos, e a alternância entre o processador a fila de prontos e a fila de dispositivos de entrada e saída, como ocorre o salvamento dos dados no BCP e como o processo se comporta nas suas transições;
- Criar uma simulação com o tipo de algoritmo de escalonamento “SJF” (tecla F3). Selecionar um processo (tecla “F”) com o tempo de execução de 20 segundos e tipo de processo CPU *bound;*
- Quando esse processo criado estiver em execução na CPU, criar outro processo (tecla “F”), com tempo de execução de 20 segundos e tipo de processos I/O *bound;*
- Observar a criação da fila de prontos, e a alternância entre o processador a fila de prontos e a fila de dispositivos de entrada e saída e como o processo se comporta nas suas transições;

Questões

Questões
1.Sobre processos: ☐ É um programa ativo ☐ É um software ☐ Não possui estados
2.Quando o processo sai da CPU e vai para a fila de dispositivos de entrada e saída, ele está transitando entre quais estados? ☐ Pronto-Execução ☐ Execução-Espera ☐ Espera-Pronto
3.Quando a fila de prontos é organizada através do *quantum* do processo, essa organização refere-se a qual tipo de algoritmo de escalonamento? ☐ Fifo ☐ Round Robin ☐ Prioridades
4.Quando um processo tem o uso intensivo de dispositivos de entrada e saída, ele é considerado? ☐ CPU *bound* ☐ I/O *bound* ☐ Memória *Bound*
5.O “despachante” do escalonador do sistema operacional, pode ser utilizado em qual situação? ☐ Quando o processo é conduzido da fila de prontos da memória principal para a CPU

Quando o processo é conduzido da fila de prontos da memória principal para a fila de processos que estão em estado de espera

☐ Quando o processo é conduzido da CPU para a tabela de Bloco de Controle de Processos (BCP)

6.Sobre BCP (bloco de controle de processos):

☐ É uma tabela onde são armazenados os dados dos processos que foram interrompidos do uso da CPU. E esses dados serão recuperados quando necessário

☐ Não é uma tabela, e sim uma área que serve para o uso dos algoritmos de escalonamento

☐ É um local que só é usado para armazenar variáveis de programas

7.O tempo que um processo utiliza quando ocupa a CPU através de um algoritmo de escalonamento, é denominado?

☐ Quantum ☐ Overhead ☐ Skew

8. Quando um processo vai da memória principal para a CPU, ele está transitando entre quais estados?

☐ Execução-Espera ☐ Pronto-Execução ☐ Execução-Pronto

9.Assinale na escala abaixo, o quanto que você acha que o ESORV contribuiu para o aprendizado de escalonamento de sistemas operacionais.

☐ Muito Pouco ☐ Pouco ☐ Indiferente ☐ Muito ☐ Muitíssimo

10.Qual a vantagem do uso da Realidade Virtual para o processo de ensino-aprendizagem?

11. Qual a desvantagem do uso da Realidade Virtual para o processo de ensino-aprendizagem?

Apêndice IV – Artigos aceitos para publicação

Esse trabalho resultou em dois artigos cientificos aceitos e publicados:

- *"Utilização de ambientes virtuais imersivos para o ensino",* publicado nos anais do XI Workshop de Computação da FACCAMP - WCF2015, Campo Limpo Paulista, São Paulo, Brasil;
- *"A Process-Scheduling Simulator Based on Virtual Reality Technology"*, publicado nos anais do "13th ACS/IEEE International Conference on Computer Systems and applications - AICCSA 2016", Agadir, Marrocos. Estrato B1.