Speed Schedule - Jogo para auxílio no estudo das políticas de escalonamento em Sistemas Operacionais

Speed Schedule - Game to aid in learning the scheduling policies in Operating Systems

ROBERTO TENORIO FIGUEIREDO
Universidade Federal do Vale do São Francisco

VIVIANNI MARQUES LEITE DOS SANTOS
Universidade Federal do Vale do São Francisco

JORGE LUIS CAVALCANTI RAMOS
Universidade Federal do Vale do São Francisco

Resumo: Em cursos da área de exatas, disciplinas essencialmente teóricas muitas vezes se apresentam como uma dificuldade. Este artigo tem como objetivo geral apresentar um jogo que visa auxiliar a aprendizagem e fixação dos conceitos estudados na disciplina de Sistemas Operacionais, no que diz respeito às políticas de escalonamento de processos, o "Speed Schedule". Inicialmente, é mostrada a problemática dos alunos no aprendizado do conteúdo, as estratégias abordadas por alguns autores, e por fim o jogo é apresentado, sua descrição, funcionamento e como pode ser utilizado em sala de aula. Foram feitos testes preliminares com algumas turmas de Sistemas Operacionais do curso de Ciência da Computação de uma instituição de ensino superior e os resultados são apresentados. A conclusão é de que o jogo estimula o aprendizado e dá ao estudante uma visão real do funcionamento dos processos no processador dentro de um sistema operacional real.

Palavras Chave: Sistemas Operacionais. Escalonamento. Jogos Educativos

Abstract: In courses of the area of exact, essentially theoretical disciplines are always as a difficulty. This article has as general objective to present a game that aims to help the learning and fixation of the concepts studied in the discipline of Operating Systems, with regard to process scheduling policies, the "Speed Schedule". Initially the problem of students in content learning, the strategies addressed by some authors, and finally the game is presented, its description, how it works and how it can be used in the classroom. Preliminary tests were done with some classes of Operating Systems of the Computer Science course of a higher education institution and the results are presented. The conclusion is that the game stimulates learning and gives the student a real insight into how the processes work in the processor within an actual operating system.

Keywords: Operating Systems, Schedule, Educational Games.

FIGUEIREDO, Roberto Tenorio; SANTOS, Vivianni Marques Leite; RAMOS, Jorge Luis Cavalcanti. Speed Schedule - Jogo para auxílio no estudo das políticas de escalonamento em Sistemas Operacionais. *Informática na Educação*: teoria & prática, Porto Alegre, v. 23, n. 1, p. 45-61, jan./abr. 2020.

1 Introdução

Muitos dos conteúdos passados em um curso superior vão se perdendo com o tempo, principalmente aqueles ministrados de maneira essencialmente teórica. Uma grande parcela dos alunos tende a memorizar os conteúdos para realizar as avaliações e depois simplesmente não os veem mais e acaba simplesmente esquecendo. Para estudantes de tecnologia, entender como funciona um Sistema Operacional tanto no alto nível, como baixo nível, pode ser determinante no desenvolvimento de sistemas e de dispositivos automáticos (KIOKI, SANTIAGO e SOARES, 2008). Na criação de grandes aplicações, estar preparado para utilizar os recursos oferecidos pelo sistema pode ser a diferença na viabilidade desta aplicação, que podem simplificar o desenvolvimento e melhorar o desempenho de requisições e rotinas (TANENBAUM, 2010).

De maneira geral, a disciplina de Sistemas Operacionais é ministrada apenas repassando conceitos teóricos, sem que seja dada ao aluno uma experiência prática para consolidação da aprendizagem. O acesso a esta experiência prática torna-se praticamente impossível dado que os estágios oferecidos nesta área são extremamente escassos e projetos de pesquisa envolvendo o funcionamento interno dos sistemas operacionais não alcançam sua totalidade (MACHADO e MAIA, 2005), gerando dificuldade no aprendizado e na abstração do conteúdo, fato este que vem sendo verificado dentro e fora das salas de aula.

Diante desta problemática e sendo notório que jogos eletrônicos despertam o interesse de uma grande parcela de estudantes de computação, conjecturou-se que a criação de uma ferramenta atrativa e facilitadora, um jogo, feito de acordo com os métodos pedagógicos do curso, pode ajudar aos alunos a aprender melhor o conteúdo, fixando este na memória e despertando habilidades nos estudantes (MUÑOZ, 2009).

Assim, o desinteresse quanto ao conteúdo apresentado em sala de aula e as consequências para o aprendizado, percebe-se a necessidade de se criar algum meio para estimular os estudantes a participarem mais e a fixar os conteúdos.

Nesse sentido, o objetivo deste artigo é apresentar uma ferramenta de apoio à disciplina de Sistemas Operacionais, denominada "Speed Schedule", com foco nas políticas de escalonamento de processos, demonstrar sua utilização em sala de aula e mensurar ganhos no aprendizado.

O trabalho está dividido nos seguintes tópicos: Trabalhos Relacionados, que apresenta uma análise de outros trabalhos e pensamentos sobre o tema; Políticas de Escalonamento, que mostra os conceitos trabalhados no jogo; O Jogo, que descreve o funcionamento do jogo em si e suas contribuições para o aprendizado e as considerações finais, com as conclusões da pesquisa.

1.1 Metodologia do Trabalho

O presente artigo tem natureza aplicada (YIN, 2015), uma vez que objetiva gerar ferramenta de apoio para aplicação prática para solução de problemas relacionados ao processo

de aprendizagem. Também tem caráter terá caráter descritivo por apresentar o desenvolvimento e análise de uma ferramenta para melhoria do ensino, levando em consideração aspectos qualitativos, referentes aos efeitos causados durante sua implementação.

Quanto aos procedimentos técnicos, se caracteriza como um estudo de caso (YIN, 2015), pois é inclui validação com alunos de três turmas referentes à disciplina de Sistemas Operacionais, cujos resultados são específicos para as turmas em análise. Entretanto, deve-se ressaltar que os resultados poderão ser utilizados como referência para indicação de sua viabilidade em quaisquer outras turmas que se enquadrem na temática.

Para desenvolvimento do jogo, inicialmente foi escolhido o tópico das políticas de escalonamento de processos. Em seguida, foi realizada pesquisa na literatura sobre o que está sendo desenvolvido em relação aos jogos estimuladores de aprendizado, com foco na disciplina de Sistemas Operacionais. Esta pesquisa foi realizada em periódicos, revistas especializadas e anais de eventos. Tendo como base essas referências e também livros utilizados na disciplina em foco, foi idealizada uma forma de transformar o conteúdo em um jogo, que não fosse um mero jogo de perguntas e respostas.

A partir da ideia central do que seria o jogo e considerando um incentivo relacionado a velocidade com que o aluno precisa acertar as respostas do escalonador, foi nomeado o jogo.

Adicionalmente, foi desenvolvido o jogo com base nos seguintes critérios: agregar conhecimento à agilidade no pensamento; criar um sistema lúdico e competitivo; dimensionar o jogo para pudesse ser finalizado em uma aula de até uma hora.

Definidos as etapas anteriores, seguiu-se o planejamento do jogo, com elaboração do esboço e toda sua descrição. Por fim, a implementação e os testes feitos em três turmas de estudantes da disciplina.

2 Trabalhos Relacionados

Uma gama grande de trabalhos apresenta a criação e uso de jogos e aplicativos simuladores no auxílio do aprendizado de matérias específicas em cursos de tecnologia. Um exemplo disso pode ser lido no artigo "*Desenvolvimento de Jogos Educacionais para o Ensino de Engenharia de Software"* (LIMA, PACHECO, *et al.*, 2011), que nos mostra informações sobre o desenvolvimento de um jogo para a disciplina de Engenharia de Software e faz uma análise do jogo "UbiRE".

Outro artigo nesta mesma área é o "*Tira-Teima, um software educacional para o ensino de programação em C"* (SENA, 2011), onde o estudante tem a chance de visualizar, a cada linha de código, informações gráficas do que ocorre com o algoritmo em execução. Ainda relacionado ao tema, temos o artigo: "*Construção de simuladores gráficos para Teoria da Computação: uma proposta para auxiliar a aprendizagem da construção lógica das Máquinas de Turing*" (OLIVEIRA e MARCO, 2007), com o objetivo de facilitar a compreensão do alto grau de abstração presente nas máquinas de Turing e por fim, é interessante falar do artigo "*WarGrafos – Jogo para Auxílio na Aprendizagem da Disciplina de Teoria dos Grafos"* (FIGUEIREDO e

FIGUEIREDO, 2011), que apresenta um jogo de War, modelado ao estudo de diversos conceitos da disciplina de Teoria dos Grafos.

Falando especificamente de Sistemas Operacionais, tem-se o artigo "*SOsim, um framework construtivista no aprendizado da disciplina de Sistemas Operacionais*" (MACHADO e MAIA, 2005), onde o discente pode fazer experimentações, através de simulações gráficas visuais, de vários conceitos da disciplina, inclusive do escalonador de processos.

O artigo *"OS Simulator: Um Simulador de Sistema de Arquivos para Apoiar o Ensino/Aprendizagem de Sistemas Operacionais"* (GADELHA, AZEVEDO, *et al.*, 2010) nos mostra um simulador para auxílio na aprendizagem especificamente de temas relacionados ao sistema de arquivos. Nesta mesma ideia, o artigo "*Simulador para a Prática de Sistemas Operacionais*" (CARVALHO, BALTHAZAR, *et al.*, 2012) apresenta o desenvolvimento do S^2O, um simulador das principais funcionalidades de um sistema operacional que permitirá uma interação com o usuário/aluno de forma a facilitar a assimilação dos conceitos ensinados na disciplina.

Com um tema mais próximo deste trabalho, temos o artigo "*TBC-SO/WEB: Software educativo para aprendizagem de políticas de escalonamento de processos e de alocação de memória em sistemas operacionais*" (REIS e COSTA, 2009) que nos mostra um simulador que trata especificamente das Políticas de Gerência de Memória (First-Fit, Next-Fit, Best-Fit e Worst-Fit) e das Políticas de Escalonamento de Processos: FIFO, SJF, HRRN, Por Prioridade Preemptiva e Não-Preemptiva, SRTF e Round-Robin. O artigo "*Sistema Operacional Integrado Simulado: Módulo de Entrada e Saída*" (CRUZ, SILVA e GONÇALVES, 2007) apresenta o SOIS, um simulador abrangente dentro da disciplina, mas com foco principal nos conceitos de Entrada e Saída.

O artigo "*Sistema Operacional Integrado Simulado: Módulo de Entrada e Saída*" (KIOKI, SANTIAGO e SOARES, 2008) propõe a utilização de um simulador de sistema operacional genérico em sala de aula, ainda em desenvolvimento. Já o artigo "*Simulador de Rotinas do Sistema Operacional para Auxílio às Aulas Teóricas*" (RIBEIRO, BERNARDES e LOBO, 2014) apresenta um simulador que trata especificamente das rotinas de gerenciamento de memória real e virtual e do escalonamento de processos em arquiteturas com multiprocessadores ou com processadores multicore de forma gráfica e interativa. E por fim, mas sem esgotar a totalidade de artigos, o "*Simulador para o aprendizado de sistemas operacionais*" (TONINI e LUNARDI, 2006) que apresenta o desenvolvimento de uma aplicação, idealizada como um simulador para o aprendizado da disciplina de sistemas operacionais.

Estas pesquisas nos mostram ferramentas com o mesmo objetivo do Speed Schedule, todavia, na grande maioria dos trabalhos encontrados, quando se fala em aplicação para auxílio na aprendizagem da disciplina de Sistemas Operacionais, muito se fala em "simulador" e pouco se fala especificamente em jogos. Neste caso um simulador tenta mostrar ao aluno como as coisas acontecem dentro de um sistema, mas um jogo propõe uma situação competitiva, estimulando os estudantes a vencer através do seu conhecimento.

Este artigo soma em apresentar uma ferramenta jogável, com bons resultados em sala, que pode ser trabalhado em uma única aula, e bem diferente do que foi apresentado em outros

artigos, porém, todos eles mostram à necessidade de se alterar as técnicas tradicionais de ensino e fazer da sala de aula um local, de certa forma, prazeroso e instigante aos estudantes.

2.1 Aspectos pedagógicos

Antes de trabalhar um jogo como um elemento lúdico, deve-se deixar de lado a ideia do jogo pelo jogo e fazer desta ferramenta, mais um instrumento de trabalho (RIEDER, ZANELATTO e BRANCHER, 2004).

Os jogos educacionais possuem objetivos pedagógicos bem específicos no desenvolvimento dos estudantes (RIEDER, ZANELATTO e BRANCHER, 2004). Esses tipos de jogos, além do entretenimento, têm como finalidade principal trazer novos conhecimentos ou auxiliar a fixação de conceitos anteriormente estudados. Eles, por definição, devem explorar o máximo possível, o processo de ensino e aprendizagem. Educadores e profissionais pedagógicos concordam que jogos educativos, quando bem aplicados, trazem diversos benefícios para os alunos e professores no dia-a-dia de sala da aula (RODRIGUES, 2006).

Algumas pesquisas apresentam, de maneira mais efetiva, a importância dos jogos na aprendizagem e como fazê-los. Neste sentido, o trabalho "*Jogos Eletrônicos na Sala de Aula"* (RODRIGUES, 2006), apresenta as possibilidades de ensino através do lúdico e do digital, mostrando as técnicas de aprendizagem com o uso de jogos eletrônicos. Já o artigo "*Por que ter Jogos Eletrônicos em Sala de Aula*" (MUÑOZ, 2009) defende a utilização de jogos eletrônicos no dia-a-dia do aprendizado e seus benefícios no processo de ensino. Enquanto que o artigo "*Análise de métodos de avaliação de jogos educacionais*" (TSUDA, SANCHES, *et al.*, 2014) apresenta um estudo realizado com o objetivo de analisar diferentes métodos de avaliação que possam ser utilizados para apoiar o desenvolvimento de jogos educacionais que efetivamente ajudam no aprendizado.

3 Políticas de Escalonamento de Processos

Neste tópico, serão apresentadas as políticas de escalonamento apresentadas pelos autores Tanenbaum (2010) e Deitel (et al., 2008). Todas foram contempladas no Speed Schedule.

- **FIFO:** (Primeiro a chegar, primeiro a ser servido – FCFS) organiza a fila por ordem cronológica de requisição a CPU, ou seja, o primeiro que solicitou será o primeiro a ser atendido. Possui uma estratégia Não-Preemptiva. Dentre as vantagens estão à simplicidade
  - **Prós:** É simples entendimento e de fácil implementação. Boa para processos I/O Bound;
  - **Contras:** Baixo desempenho e fácil degradação (processos curtos podem demorar muito tempo na fila);
- **Round Robin (RR):** também chamado de alternância circular (ou chaveamento circular), assim como o FIFO, organiza a fila por ordem cronológica de requisição a CPU, porém, no RR, é atribuído a cada processo que entra na CPU um quantum.

Quando o quantum acaba, esse processo é retirado e outro é escolhido para entrar na CPU. É o chamado FIFO Preemptivo ou FIFO de estratégia preemptiva.

- **Prós:** É indicado para processos curtos e interativos, além de ser simples e de fácil implementação;
- **Contras:** Gasta tempo fazendo o chaveamento de processos. Não é recomendado para processos de tamanho equivalente, pois processos pequenos terão seu tempo de execução aumentado devido a várias interrupções do fim do quantum, além de não ser fácil estabelecer o valor do quantum. Se o quantum for pequeno demais, o tempo do chaveamento será percebido pelo usuário (o desempenho do sistema é comprometido devido à quantidade de chaveamentos realizados), se o referido tempo for grande demais, o processo monopoliza os recursos do sistema, pois pode gerar um aumento no tempo de resposta dos processos com requisições interativas. Um quantum em torno de 20ms a 50ms é bastante razoável.

- **SPN – STCF (*Shortest Process Next - Shortest Time to Completion First*):** Também conhecido como Processo Mais Curto Primeiro (em inglês *Shortest Process First* – SPF) ou Menor Job Primeiro (em inglês, *Shortest Job First* – SJF). Organiza a fila por ordem do tamanho do tempo de uso da CPU, ou seja, para cada processo, é estimado o tempo de toda sua execução e os menores processos ficam na frente da fila. Essa estimativa é realizada pelo escalonador com base em diversos parâmetros, como tamanho do processo, histórico de execução, quantidade de espaço ocupado em memória real e virtual, entre outros. Possui uma estratégia não preemptiva;
  - **Prós:** Menor tempo de resposta dos processos curtos, obtendo assim uma fila de escalonamento mais curta;
  - **Contras:** É impossível estimar com precisão o tempo de todos os processos, essas estimativas podem gastar tempo de CPU e ter uma margem de erro elevada em sistemas interativos;
- **SPN – SRTCF (*Shortest Process Next – Shortest Remaining Time to Completion First*):** Também conhecida como Menor Tempo de Execução Restante (em inglês *Shortest-Remaining Time*). Assim como a política anterior, também organiza a fila por ordem do tamanho do tempo de uso da CPU, porém, usa o quantum, ou seja, é uma estratégia preemptiva;
  - **Prós:** Menor tempo de resposta entre os processos. Quem gasta mais tempo de CPU, virá mais vezes para frente da fila;
  - **Contras:** É impossível estimar com precisão o tempo de todos os processos, essas estimativas podem gastar tempo de CPU e ter uma margem de erro elevada em sistemas interativos;
- **HRRN (*Highest Response Ratio Next*):** Também conhecida como Próxima Taxa de Resposta Mais Alta. Este algoritmo corrige algumas das deficientes do SPN-STCF, principalmente o favoritismo excessivo dos processos cursos. Nele, é definido um fator (prioridade) para cada processo. Esse fator é calculado com a fórmula:

  (Tamanho do Tempo + Tempo de Espera na Fila) / Tamanho do Tempo.

Como o tempo de serviço está no denominador, os processos curtos ainda terão privilégios, mas como o tempo de espera aparece no numerador, os processos longos também terão sua chance e não serão sempre preteridos, evitando que o escalonador adie esses processos indefinidamente. Possui uma estratégia não preemptiva.

- **Prós:** Equilibra o tempo de resposta entre os processos curtos e longos e impede que o escalonador adie processos indefinidamente;
- **Contras:** É impossível estimar com precisão o tempo de todos os processos, essas estimativas podem gastar tempo de CPU e ter uma margem de erro muito grande;

- **Com Prioridade:** Dentro de um sistema, principalmente os multiusuários, processos podem ter diferentes importâncias devido a fatores externos como ordem hierárquica da empresa. Para esses casos, o escalonamento por prioridade é muito indicado. Nele, é atribuído a cada processo um número chamado de Prioridade (ver o tópico: Características dos Processos Escalonados) e quem estiver pronto e tiver recebido a maior prioridade, entra na CPU. Para processos com a mesma prioridade e sub-prioridade é usada a Política *Round Robin* (*RR*). O escalonador pode reduzir a prioridade do processo a cada pulso do relógio ou pode ser atribuído um quantum máximo ao processo e este fica na CPU até o quantum acabar. Possui uma estratégia preemptiva;
  - **Prós:** Considera fatores externos, importantes para definição da ordem de execução pelos processos de usuário;
  - **Contras:** Processos de prioridade baixa podem ficar esperando indefinidamente. Para reduzir esse fator contra, em prioridades dinâmicas, o escalonador poderá reduzir a prioridade de processos com alta prioridade para dar lugar a processos de baixa prioridade na CPU. Essa redução de prioridade deve ser calculada a cada interrupção do sistema.
- **Múltiplas Filas:** Também conhecido como Filas Multiníveis de Retorno (em inglês, *Multilevel FeedBack Queues*). Essa estratégia foi pensada para:
  - Favorecer processos pequenos;
  - Favorecer processos limitados por E/S para atingir uma boa utilização dos dispositivos de E/S;
  - Determinar a natureza de um processo tão rápido quanto possível e escalonar o processo de acordo.

  Nesta política, cada processo é associado a uma fila, de acordo com a sua prioridade, ou seja, ao invés de uma única fila para todos os processos, temos várias filas menores de acordo com os níveis de prioridade. O processo de uma fila só é executado quando todas as filas mais altas estão vazias. Cada fila é feita baseada nas características dos processos (CPU Bound, I/O Bound, Batch, etc.) e poderá ter seu próprio escalonador. Quando a prioridade for dinâmica, os processos podem mudar de fila, caso mudem seu comportamento inicial (um processo I/O Bound pode passar a ser CPU Bound, por exemplo), aumentando assim a desempenho do sistema. Essa mudança de fila não acontece no caso da prioridade ser estática. Possui uma estratégia preemptiva.

- **Prós:** Considera fatores externos, importantes para definição da ordem de execução pelos processos de usuário. Favorece processos pequenos e limitados por E/S, além de agilizar a determinação da característica de um processo escalonado;
- **Contras:** Processos em filas de baixa prioridade podem ficar esperando indefinidamente;

- **Escalonamento Garantido:** Organiza a fila por ordem do tamanho do tempo de uso da CPU, assim como o SPN-SRTCF, porém o quantum que é atribuído aos processos é igual para todos, ou seja, os processos ficaram o mesmo tempo na CPU. Para o cálculo do quantum de cada processo é dividido o tempo total calculado para os processos pela quantidade de processos, o resultado dessa divisão será o quantum de cada um dos processos. Sempre que um novo processo chega à fila de escalonamento ou um processo conclui sua execução, esse cálculo é refeito e os quantuns dos processos são atualizados. Possui uma estratégia preemptiva.
  - **Prós:** Facilita e agiliza o cálculo do quantum. Permite que o sistema saiba exatamente quando um processo irá entrar na CPU.
  - **Contras:** É difícil estimar com precisão o tempo de todos os processos o que dificulta a sua implementação;
- **Escalonamento Por Fração Justa (*Fair Share Scheduling* – FSS):** Organiza a fila por ordem de prioridade, e faz o cálculo do quantum da mesma forma que o Escalonamento Garantido. Porém, o resultado da divisão é atribuído, não a um processo, mais a um grupo de processos. Esses grupos são definidos devido a sua afinidade na execução, mas principalmente a identificação do usuário que o criou. Por exemplo, imagine uma rede de computadores onde os usuários dos terminais fazem várias requisições ao servidor, através de vários processos. O S.O do servidor criará um grupo para conter cada requisição de um mesmo usuário e atenderá esses grupos em um tempo que será igual a todos. Dentro de um grupo o atendimento será por ordem de prioridade. Possui uma estratégia preemptiva.
  - **Prós:** Facilita e agiliza o cálculo do quantum. Permite que o sistema saiba exatamente quando um processo irá entrar na CPU. Usuários de mesma prioridade terão tempos iguais de uso do sistema;
  - **Contras:** Processos em filas de baixa prioridade podem ficar esperando indefinidamente.
- **Escalonamento Por Loteria.** Neste algoritmo, bilhetes de loteria são atribuídos aos processos, cujos prêmios são os vários recursos do sistema, como tempo de CPU. O escalonador escolhe aleatoriamente um bilhete de loteria e o processo que tiver o bilhete, receberá o recurso. Quando aplicado somente ao escalonamento da CPU, o sistema pode fazer 50 sorteios por segundo e cada vencedor ganhará o quantum (que pode ser de 20ms) como prêmio. É possível entregar mais bilhetes a processos mais importantes (que teriam mais prioridade). Caso queiram, os processos podem trocar

bilhetes entre si. Quando um processo é escolhido, perde alguns bilhetes que são redistribuídas com os demais. Possui uma estratégia preemptiva.

- **Prós:** É um algoritmo altamente responsivo e de fácil implementação;
- **Contras:** Processos de baixa prioridade podem ser escolhidos e um processo que acabou de sair pode ser escolhido novamente para entrar;

4 O Jogo

O jogo Speed Schedule é baseado no cálculo do tempo de retorno médio. Com base nos tempos totais de cada processo e na ordem com que os processos chegam à CPU, dependendo da política de escalonamento indicada, o estudante terá que ordenar os processos e calcular o tempo de retorno médio no menor tempo possível. Neste jogo, quem responder certo uma maior quantidade de vezes e em menos tempo será o campeão.

Diferente de outros projetos para auxílio da disciplina de Sistemas Operacionais, onde o estudante cria um ambiente e a aplicação simplesmente mostra o que acontece dentro do kernel naquele ambiente, o Speed Schedule coloca o aluno na posição do sistema, de modo que o aluno passa a ser o escalonador. Assim, o jogo apresenta ao aluno um conjunto de processos e suas informações pertinentes e o jogador, na posição de sistema, ordenará os processos de acordo com a política escolhida e calculará o tempo médio de retorno naquela situação específica.

4.1 Criação do Jogo

Inicialmente, como qualquer outro jogo, a primeira atividade a ser realizada é a idealização do jogo em si e sua descrição no papel, criando um *Game Designer Document*. Esta tarefa é feita pelo profissional conhecido como Projetista do Jogo (Game Designer).

Para a criação do Speed Schedule, depois de realizadas as pesquisas no referencial teórico sobre jogos educacionais na área de tecnologia, principalmente para a disciplina de Sistemas Operacionais, foi realizada uma pesquisa informal junto aos discentes sobre quais tópicos são mais complicados em aprender. Com um estudo mais aprofundado do tópico escolhido para desenvolvimento, foi feita a ideia da dinâmica do jogo.

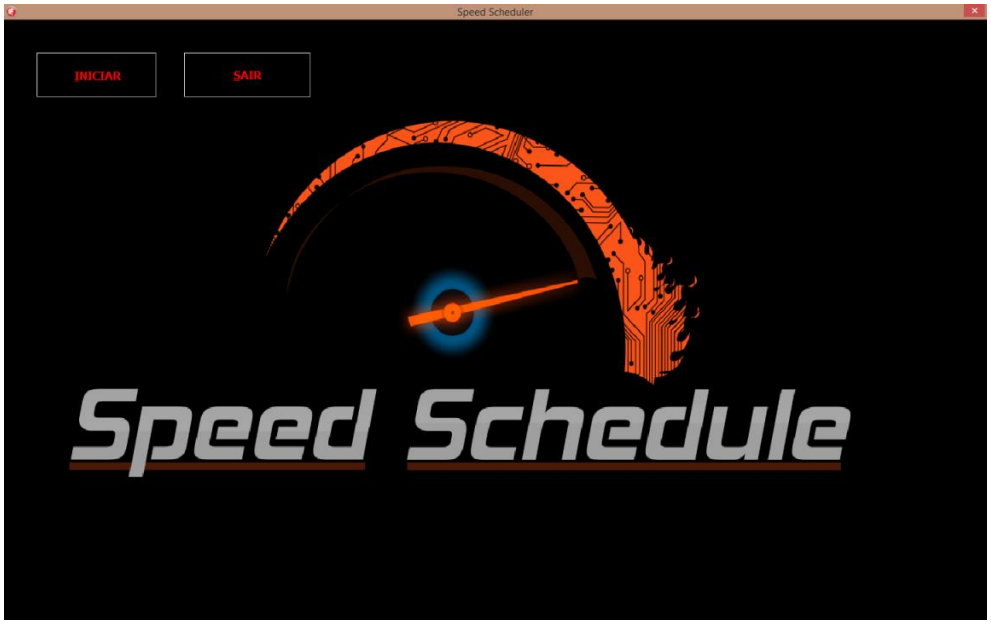
Após da definição do jogo estar pronta, foram analisadas algumas *engines* de programação, entre elas o *Game Maker* e o *Flash*, porém, a linguagem de programação escolhida foi o Delphi RAD Studio XE5, por ser de fácil desenvolvimento, devido à larga experiência do *Game Designer* na linguagem, que é compatível com todas as versões do Windows, sistema operacional mais utilizado pelos alunos no campus e em suas residências.

4.2 Telas do jogo

No início do jogo, os alunos são divididos em equipes. Cada equipe escolhe um nome, que a identificará no jogo e a ordem em que cada grupo irá jogar, pois as equipes não jogam simultaneamente, e sim em rodadas alternadas.

Ao iniciar, o jogo abre a tela inicial, mostrada na Figura 1. O clicar no botão "INICIAR", o jogo leva à tela de inserção dos grupos, apresentada na Figura . Nesta tela, é possível informar a quantidade de equipes e, na ordem da jogatina, o nome com que cada uma será identificada.

Figura 1: Tela inicial do Jogo Speed Schedule



Fonte: O autor

Figura 2: Tela de formação das equipes

Fonte: o autor

Depois que a quantidade e os nomes forem inseridos, clica-se em "Iniciar Jogo" e o jogo começa de fato, abrindo a tela do jogo mostrada na

Figura .

Figura 3: Tela principal, onde o jogo acontece.

Fonte: O autor

A tela do jogo é composta por quatro botões superiores, o primeiro "FECHAR JOGO" é para ser clicado quando o jogo terminar. O segundo, "INICIAR JOGO", reinicia uma partida com as mesmas equipes cadastradas. O terceiro, "PARCIAIS", mostra os resultados parciais (quantidade de acertos e pontos) de cada equipe até o momento em que o botão foi clicado e o quarto, "INICIAR RODADA - JÁ", inicia a rodada de uma equipe.

Abaixo dos botões superiores, temos cinco ou seis colunas (a quantidade de colunas é aleatória entre cinco e seis), com as informações referentes aos processos, que estão em ordem de chegada (tempo entre uma chegada e outra é insignificante). Na primeira linha, são listados os processos a partir do processo P1. Na segunda linha, o tempo de retorno, na terceira linha, a prioridade dos processos, conforme conceitos de Tanembaum (2010). Na

quarta linha temos o turnaround relativo dos processos e na última linha a quantidade de bilhetes que cada processo recebeu. Para manter um valor constante para a política de Loteria, foi combinado que o processo que tiver mais bilhetes será o primeiro a ser escolhido para entrar na CPU.

Todas as informações do processo são geradas de maneira aleatória, dentro de um intervalo estabelecido, ao clicar no botão "INICIAR RODADA - JÁ".

Ao lado das colunas, temos o botão "EXPLICAÇÃO DA POLÍTICA" que pode ser clicado quando a resposta for computada e o jogo apresenta uma explicação teórica sobre aquela política apresentada. Também possui o botão "RESPOSTAS", onde o jogo, também após a resposta ser computada, apresenta as respostas corretas na aplicação de cada uma das políticas de escalonamento para a situação apresentada no jogo. Estes dois botões não tão uteis durante a jogatina, mas são de grande importância aos alunos, quando utilizam a ferramenta para estudos fora da sala de aula.

Abaixo, temos a política aleatoriamente escolhida, em vermelho, no canto direito, a equipe que deve jogar naquele momento, o campo para colocar a resposta e o botão "OK", que deve ser clicado quando a resposta for digitada no campo Resposta.

No rodapé da tela, temos a rodada atual, que pode ser no máximo 10, mas depende do tempo da aula, podendo se ajustar a aulas de uma hora até aulas de quatro horas. Uma informação de como as respostas devem ser escritas e a pontuação recebida caso a equipe erre a resposta. Mais adiante, veremos que quanto mais pontos, pior no jogo.

4.3 Dinâmica do Jogo

O professor conduz a partida. Após inserir os dados das equipes, o condutor dá início às rodadas. A primeira equipe elege um representante na rodada que irá escrever a resposta. O professor clica em "INICIAR RODADA – JÁ".

Os estudantes devem analisar a situação que se apresenta e calcular, naquela situação, qual é o tempo de retorno médio. A resposta deve ser escrita no campo abaixo do nome de sua equipe e em seguida clicar em "ok". Enquanto o botão "ok" não é clicado, a cada segundo, a equipe ganha um ponto. Após o clique do "ok", o sistema analisa a resposta digitada e apresenta a informação do acerto ou erro. Caso o aluno tenha acertado, é incrementado o número de vitórias da equipe, caso contrário, o sistema informa qual é a resposta correta e a equipe ganha 500 pontos extras. Em seguida, um representante da segunda equipe é escolhido e repete a operação, ou seja, o condutor clica em "INICIAR PARTIDA – JÁ" e o novo aluno deve escrever a resposta no campo e clicar em "ok". Todos os alunos da equipe ajudam no cálculo, mas apenas o eleito digita a resposta. Quando todas as equipes tiverem jogado uma vez (colocado uma resposta), fecha-se um ciclo, de um total de dez ciclos. Um membro de uma equipe só pode jogar novamente, se todos os membros já tiverem jogado. Todos os membros da equipe devem jogar a mesma quantidade de vezes, ou, no máximo, uma vez a mais que os demais, caso a quantidade de membros não seja divisível por dez ou não seja múltipla da quantidade de rodadas adequadas ao tempo da aula.

Ao final das rodadas, uma lista com a pontuação geral aparece. Vence quem obteve menos pontos. O nome da equipe vencedora aparece destacado na lista.

Para tornar o jogo mais acessível, é utilizado um teclado numérico móvel, que permita alcançar os estudantes, sem que os mesmos precisem se locomover à mesa do professor. Com isso, os estudantes podem digitar as respostas com mais comodidade.

4.4 Validação do jogo

Depois dos testes de bugs, foi feita a validação do jogo em sala. Esta validação se deu em três turmas de Sistemas Operacionais do curso de Ciência da Computação de uma instituição de ensino superior, e jogado três vezes em cada uma das turmas. O jogo foi aplicado na aula posterior após a explanação do conteúdo teórico pelo professor, o qual foi feito em aulas expositivas, usando projetor multimídia para exibição de slides.

No início da aula, os estudantes foram divididos em equipes com três componentes cada e jogaram por aproximadamente uma hora e trinta minutos. Durante a jogatina, foi observado à reação dos participantes, conforme sugere o modelo de avaliação de jogos educacionais, com base no nível 1, do modelo de avaliação de treinamentos de Kirkpatrick, em que a qualidade do jogo é indicada pela reação dos estudantes durante sua execução (SAVI, BORGATTO e VON WANGENHEIN, 2011).

Em relação às reações dos estudantes, foi observado que, inicialmente eles ficavam dispersos enquanto sua rodada não chegava e muito apreensivos durante sua vez de jogar, mas com o tempo, eles passaram a tentar resolver os desafios das outras equipes como forma de treinamento enquanto esperavam sua vez e também conseguiram reduzir o tempo de resolução dos desafios e com menos consultas as referências permitidas. (livro texto da disciplina).

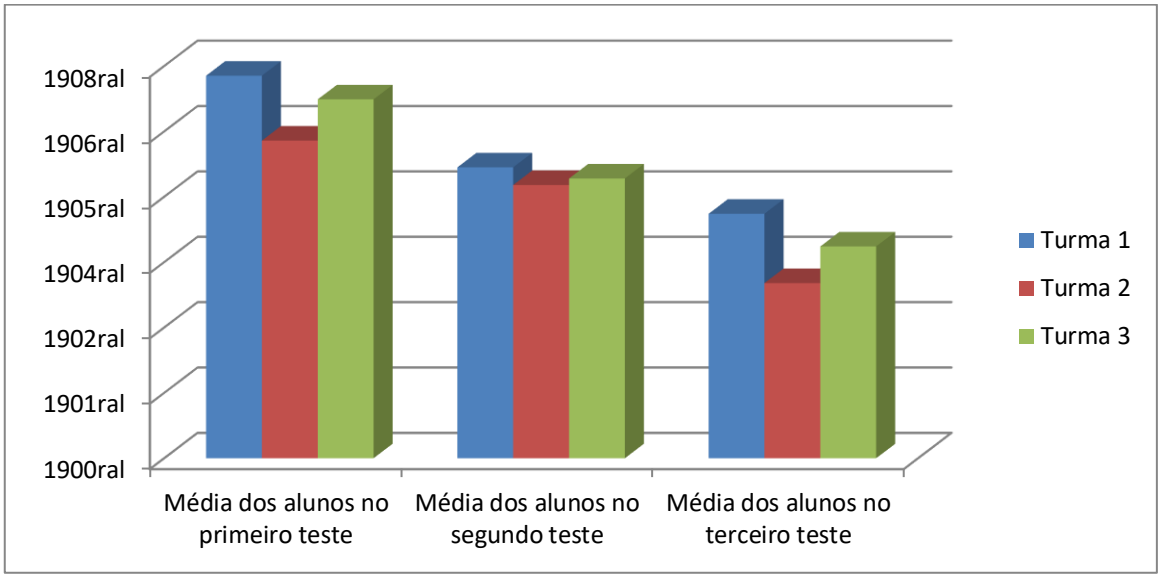
Outros fatores que influenciam na qualidade de um jogo educativo são a capacidade de motivar seu público-alvo a estudar, o ganho em aprendizagem e uma boa experiência de jogo (SAVI, BORGATTO e VON WANGENHEIN, 2011).

O clima de competição entre as equipes e o vislumbre de diminuir os recordes anteriores, foram elementos importantes na motivação do jogo por parte dos estudantes. Tal motivação levou os alunos a memorizarem os conteúdos para conseguir tempos ainda menores, aumentando os acertos e o interesse pelo jogo.

O ganho na aprendizagem das políticas foi verificado através do aumento da taxa de acertos entre os dias em que o jogo foi aplicado em sala e na avaliação escrita da matéria, visto que a taxa de acerto das questões referentes ao tema, dos estudantes que participaram dos jogos, foi de 86%, enquanto dos alunos que faltaram à jogatina, ficou na faixa dos 72%. Recordando que quanto mais pontos, mais erros foram cometidos ou mais demorado foi para impostar a resposta, o ganho na aprendizagem pode ser visto no

Gráfico 1, que mostra a quantidade média de pontos obtidos pelas equipes.

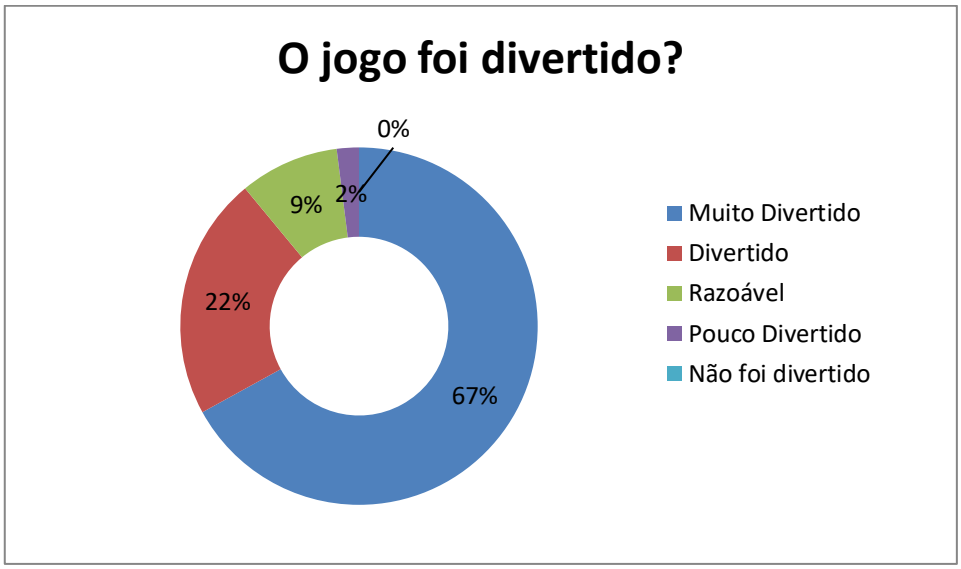
Gráfico 1: Média de pontos das equipes, em cada uma das tentativas.



Fonte: O autor

A experiência dos alunos ao jogar deve gerar imersão e uma satisfação em interagir com o jogo e seus desafios, que são compatíveis com o que foi ministrado em sala. O orientador da matéria fez um comparativo da exposição dos conteúdos em semestres anteriores e neste, em que o jogo foi utilizado em sala e foi percebido um grau de interesse elevado em participar da atividade lúdica e muitos solicitaram que outros conteúdos da disciplina fossem incluídos no jogo. Com isso, foi feita uma pergunta aos participantes sobre o quão divertido foi o jogo e, conforme mostra o Gráfico 2, foi concluído que os estudantes gostaram da iniciativa e se divertiram na jogatina.

Gráfico 2: Gráfico sobre pergunta feita aos alunos.



Fonte: O autor

5 Considerações Finais

O jogo *Speed Schedule* mostrou-se promissor como uma ferramenta para auxiliar os estudantes no estudo das políticas de escalonamento de processos, assunto constante na disciplina de Sistemas Operacionais. Conforme observado junto aos estudantes, o jogo facilitou a compreensão sobre os conteúdos propostos de maneira mais ágil e atrativa.

Conforme as observações feitas em sala, o *Speed Schedule* pode estimular o aprendizado e a concentração dos estudantes em compreender como uma política de escalonamento escolhe qual é o próximo processo a entrar na CPU de maneira lúdica.

Com este artigo foi possível destacar a importância do uso de jogos em sala de aula, com objetivos claros de melhorar o aprendizado dos estudantes e despertar o interesse para as disciplinas, principalmente aquelas em que se tem verificado um maior desinteresse e dificuldades no aprendizado.

Após a análise dos resultados obtidos junto aos estudantes, percebeu-se que o desenvolvimento de jogos não pode se limitar a um tópico da disciplina de Sistemas Operacionais, portanto, para pesquisas futuras, sugere-se desenvolver jogos para outros tópicos da disciplina, como por exemplo, as estratégias de substituição de páginas da memória RAM, os algoritmos de escalonamento do braço de leitura de um disco e a tradução de endereços virtuais em reais. Recomenda-se ainda, priorizar as disciplinas com maior índice de reprovação.

Outro ponto importante para pesquisas futuras é a análise da interface de interação com o jogador, para tornar o jogo mais atrativo e mais fácil de usar. Além disso, estudos futuros devem focar na análise de acessibilidade, para tornar o jogo iniludível aos portadores de deficiência.

Finalmente, após a publicação do trabalho e seu registro, o código fonte será liberado, em acordo com o titular do registro, para utilização de outros pesquisadores que queiram contribuir com seu desenvolvimento.

Referências

CARVALHO, Denise Schimitz; BALTHAZAR, Glauber da Rocha; DIAS, Carlos Rodrigo ; ARAÚJO, Marco Antônio Pereira; MONTEIRO, Paulo Henrique Rezende. Simulador para a Prática de Sistemas Operacionais. **In. Revista Eletrônica da FMG**, Juiz de Fora, 26 out. 2012.

CRUZ, E. H. M.; SILVA, V. P.; GONÇALVES, R. A. L. Sistema Operacional Integrado Simulado: Módulo de Entrada e Saída. **In. Proceedings of XIV ERI-PR 2007**, p. 89-98, 2007.

DEITEL, H. M.; DEITEL, ; CHOFFNES, D. R. **Sistemas Operacionais 3ª Edição**. São Paulo: Pearson, 2008.

FIGUEIREDO, R. T.; FIGUEIREDO, C. B. C. WarGrafos Jogo para Auxilio na Aprendizagem da Disciplina de Teoria dos Grafos. **In. Proceedings of SBGames 2011**, 2011.

GADELHA, R. N. S. et al. OS Simulator: Um Simulador de Sistema de Arquivos para Apoiar o Ensino/Aprendizagem de Sistemas Operacionais. **In. Proceedings of SBIE 2010**, 2010.

KIOKI, E. Y.; SANTIAGO, P. P.; SOARES, A. C. Um Simulador Didático como Ferramenta de Apoio ao Ensino da Disciplina de Sistemas Operacionais. **INICIA**, Santa Rita do Sapucaí (MG), p. 41-48, 2008.

LIMA, Thaiana; PACHECO, Beatriz; SANTOS, Rodrigo; WERNER, Cláudia; LIMOEIRO, Fernando. Desenvolvimento de Jogos Educacionais para o Ensino de Engenharia de Software. **In. Proceedings of SBGames 2011**, 2011.

MACHADO, F. B.; MAIA, L. P. Um framework construtivista no aprendizado de Sistemas Operacionais – uma proposta pedagógica com o uso do simulador SOsim. **In. Proceedings of 10th Annual Conference on Innovation and Technology in Computer Science Education**, 2005.

MUÑOZ, Y. J. Por que ter Jogos Eletrônicos em Sala de Aula? **In: VII Enpec – Encontro Nacional de Pesquisa em Educação e Ciências. Anais do Evento.**, Florianópolis: UFCS, 2009.

OLIVEIRA, G. P.; MARCO, A. S. Construção de Simuladores Gráficos para Teoria da Computação: uma proposta para o ensino do conceito de Máquinas de Turing. **In: IV SEGeT – Simpósio de Excelência em Gestão e Tecnologia. Anais do evento.**, Resende: AEDB, 2007.

REIS, F. P.; COSTA, H. A. X. **TBC-SO/WEB:** Software Educativo para Aprendizagem de Gerência de Processos e Gerência de Memória em Sistemas Operacionais. Universidade Federal de Lavras: Monografia, 2009.

RIBEIRO, T. P.; BERNARDES, R. L.; LOBO, E. A. Simulador de Rotinas do Sistema Operacional para Auxílio às Aulas Teóricas. **In. Proceedings of Simpósio Brasileiro de Sistemas de Informação 2014**, p. 737-748, 2014.

RIEDER, R.; ZANELATTO, E. M.; BRANCHER, D. J. Observação e Análise da Aplicação de Jogos Educacionais Bidimensionais em um Ambiente Aberto. **In: Taller Internacional de Software Educativo. Actas do IX Taller Internacional de Software Educativo.**, Santiago, Chile, 2004.

RODRIGUES, G. D. Jogos Eletrônicos na sala de aula: possibilidades de ensino através do lúdico e do digital. **In: Intercom – Sociedade Brasileira de Estudos Interdisciplinares da Comunicação. Anais do XXIX Congresso Brasileiro de Ciências da Comunicação.**, Brasília, 2006.

SAVI, R.; BORGATTO, A. F.; VON WANGENHEIN, C. G. Um Modelo de Avaliação de Jogos Educacionais na Engenharia de Software. **In**. **Proceedings of 25th Brazilian Symposium on Software Engineering (SBES)**, São Paulo, 2011.

SENA, V. F. **Um Software Educacional para o ensino de programação em C.** UnB. Brasília. 2011.

TANENBAUM, A. S. **Sistemas Operacionais Modernos 3ª Edição**. São Paulo: Pearson, 2010.

TONINI, G. A.; LUNARDI, S. Simulador para o Aprendizado de sistemas Operacionais. **In. Proceedings of SIRC 2006**, 2006.

TSUDA, Marcos; SANCHES, Vinícius Moro; FERREIRA, Thalles Gonçalves; OTSUKA, Joice Lee; BEDER, Delano Medeiros. Análise de métodos de avaliação de jogos educacionais. **In. Proceedings of SBGames 2014**, p. 158-166, 2014.

YIN, Robert K. **Estudo de Caso: Planejamento e métodos**. Bookman editora, 2015.

Recebido em maio de 2019.
Aprovado para publicação em fevereiro de 2020.

Roberto Tenorio Figueiredo
Estudante do Programa de Pós-Graduação em Agroecologia e Desenvolvimento Territorial - Universidade Federal do Vale do São Francisco – UNIVASF, Brasil, vivianni.santos@gmail.com

Dra. Vivianni Marques Leite dos Santos
Programa de Pós-Graduação em Agroecologia e Desenvolvimento Territorial - Universidade Federal do Vale do São Francisco – UNIVASF, Brasil, vivianni.santos@gmail.com

Dr. Jorge Luis Cavalcanti Ramos
Programa de Pós-Graduação em Agroecologia e Desenvolvimento Territorial - Universidade Federal do Vale do São Francisco – UNIVASF, Brasil, jorge.cavalcanti@univasf.edu.br