UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

JOGOS SÉRIOS EM MUNDOS VIRTUAIS: UMA ABORDAGEM PARA O ENSINO-APRENDIZAGEM DE TESTE DE SOFTWARE

DISSERTAÇÃO DE MESTRADO

Tarcila Gesteira da Silva

Santa Maria, RS, Brasil
2012

JOGOS SÉRIOS EM MUNDOS VIRTUAIS: UMA ABORDAGEM PARA O ENSINO-APRENDIZAGEM DE TESTE DE SOFTWARE

por

Tarcila Gesteira da Silva

Dissertação apresentada ao Curso de Mestrado em Computação do Programa de Pós-Graduação em Informática, Área de Concentração em Computação Aplicada, da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do grau de
Mestre em Computação.

Orientador: Prof. Dr. Felipe Martins Müller

Santa Maria, RS, Brasil

2012

Ficha catalográfica elaborada através do Programa de Geração Automática
da Biblioteca Central da UFSM, com os dados fornecidos pelo(a) autor(a).

Gesteira da Silva, Tarcila
Jogos sérios em mundos virtuais: uma abordagem para o ensino-aprendizagem de teste de software / Tarcila Gesteira da Silva.-2012.
88 p.; 30cm

Orientador: Felipe Martins Müller
Tese (doutorado) - Universidade Federal de Santa Maria, Centro de Tecnologia, Programa de Pós-Graduação em Informática, RS, 2012

1. Jogos sérios 2. Mundos virtuais 3. OpenSim 4. Teste de software 5. Moodle I. Martins Müller, Felipe II. Título.

Universidade Federal de Santa Maria
Centro de Tecnologia
Programa de Pós-Graduação em Informática

A Comissão Examinadora, abaixo assinada,
aprova a Dissertação de Mestrado

JOGOS SÉRIOS EM MUNDOS VIRTUAIS: ABORDAGEM PARA O ENSINO-APRENDIZAGEM DE TESTE DE SOFTWARE

elaborada por
Tarcila Gesteira da Silva

como requisito parcial para obtenção do grau de
Mestre em Computação

COMISÃO EXAMINADORA:

Felipe Martins Müller, Dr.
(Presidente/Orientador)

Adriano Canabarro Teixeira, Dr. (UPF)

Roseclea Duarte Medina, **Dr.** (UFSM)

Santa Maria, 4 de abril de 2012.

RESUMO

Dissertação de Mestrado
Programa de Pós-Graduação em Informática
Universidade Federal de Santa Maria

JOGOS SÉRIOS EM MUNDOS VIRTUAIS: UMA ABORDAGEM PARA O ENSINO-APRENDIZAGEM DE TESTE DE SOFTWARE

AUTORA: TARCILA GESTEIRA DA SILVA
ORIENTADOR: FELIPE MARTINS MÜLLER
Data e Local da Defesa: Santa Maria, 4 de abril de 2012.

Este trabalho apresenta uma abordagem de aprendizagem baseada em jogos utilizando o jogo sério denominado Jogo da Equipe de Teste de *Software* (JETS), que é um jogo em três dimensões, *multiplayer* e foi desenvolvido no mundo virtual OpenSim. Esse jogo simula o setor de Teste de *Software* de uma empresa de desenvolvimento de sistemas e visa a proporcionar conhecimentos relacionados com Estratégias de Teste de *Software*, motivar os estudantes e estimular habilidades como comunicação e resolução de problemas. O principal diferencial do JETS é sua integração com o Ambiente Virtual de Ensino-Aprendizagem (AVEA) Moodle, que permite ao professor editar os desafios das fases por meio da ferramenta questionário, de modo que o jogo se adapte aos diferentes currículos dos cursos de graduação em computação. O JETS também permite a exportação dos casos de teste criados no jogo para a ferramenta de gerenciamento de Teste de *Software* TestLink, proporcionando aos estudantes o contato com uma ferramenta utilizada por profissionais e a aplicação direta dos conhecimentos adquiridos. A avaliação do JETS foi realizada em uma turma do curso de Engenharia da Computação na disciplina de Engenharia de *Software*. Essa avaliação teve resultados qualitativos muito positivos, pois, segundo os próprios estudantes, os objetivos de aprendizagem foram atingidos. Além disso, os estudantes relataram, em sua maioria, gostar de jogar o JETS e disseram, ainda, preferir esse tipo de atividade (jogo sério) à tradicional – neste trabalho, representada por uma lista de exercícios. Portanto, foi possível perceber que a utilização do JETS pode ser uma alternativa promissora no processo de ensino-aprendizagem do conteúdo de Teste de *Software*. De modo que os resultados positivos obtidos na avaliação do JETS incentivam a realização de outros experimentos.

Palavras-chave: Jogos Sérios; Mundos Virtuais; OpenSim; Teste de *Software*; Moodle.

ABSTRACT

Master's Dissertations
Programa de Pós-Graduação em Informática
Universidade Federal de Santa Maria

SERIOUS GAMES IN VIRTUAL WORLDS: AN APPROACH TO SOFTWARE TESTING TEACHING AND LEARNING

AUTHOR: TARCILA GESTEIRA DA SILVA
ADVISOR: FELIPE MARTINS MÜLLER
Date and Location: Santa Maria, April 4, 2012.

This work presents an approach of game-based learning using a serious game called Software Testing Team Game. It is a three-dimensional multiplayer game developed in the virtual world OpenSim. This game simulates the software testing division of a software development company. It aims to provide knowledge related to software testing strategies, motivate students, and encourage skills such as communication and problem solving. The main difference between this game and others is its integration with the Learning Management System Moodle, which allows teacher to edit game challenges throughout each of the levels, using the Questionnaire tool, so that the game adapts to different undergraduate computing curricula. The game also enables students to export test cases created by them in the game to the test management tool TestLink, in order that they are given the opportunity to interact with a tool used by professionals and also to directly apply the knowledge acquired. The assessment of the game was performed in a class of Computer Engineering undergraduate students in the Software Engineering Discipline. This assessment showed positive results, considering that learners succeed to achieve learning objectives. Students enjoyed playing the game more than learning in a traditional way, which in this study corresponds to a list of exercises. So, we conclude that the use of the game can collaborate on software testing education. These positive results got in the assessment motivate other experiments.

Key-words: Serious Games; Virtual Worlds; OpenSim; Software Testing; Moodle.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

2D	Duas Dimensões
3D	Três Dimensões
ACM	*Association for Computer Machinery*
ADL	*Advanced Distributed Learning*
AIS	*Association for Information Systems*
AVEA	Ambiente Virtual de Ensino-Aprendizagem
EAD	Educação à Distância
EC	Engenharia da Computação
ES	Engenharia de *Software*
CC	Ciência da Computação
CISL	Comitê Técnico de Implementação do *Software* Livre
GB	Gigabyte
GHz	Gigahertz
GUI	*Graphical User Interface*
HTTP	*Hypertext Transfer Protocol*
IEEE	*Institute of Electrical and Electronics Engineers*
INEP	Instituto Nacional de Estudos e Pesquisas Educacionais
JETS	Jogo da Equipe de Teste de *Software*
LSL	*Linden Scripting Language*
MMO	*Massive Muliplayer Online*
MOODLE	*Modular Object-Oriented Dynamic Learning Environment*
MUVE	*Multi-User Virtual Environment*
NRC	*National Research Concil*
OpenSim	*Open Simulator*
OSSL	*Open Simulator Scripting Language*

PDA	*Personal Digital Assistant*
PHP	*Hypertext Preprocessor*
RPG	*Role-Playing Game*
SCORM	*Sharable Content Object Reference Model*
SBC	Sociedade Brasileira de Computação
SGI	*Serious Games Initiative*
SI	Sistemas de Informação
SJSU	*San José State University*
SL	*Second Life*
SLOODLE	*Simulation Linked Object Oriented Dynamic Learning Environment*
SMTP	*Simple Mail Transfer Protocol*
TS	Teste de *Software*
UML	*Unified Modeling Language*
UFSM	Universidade Federal de Santa Maria
URL	*Uniform Resource Locator*
UWS	*University of the West of Scotland*
VoIP	*Voice over Internet Protocol*
XML-RPC	*Extensible Markup Language – Remote Procedure Call*

SUMÁRIO

1 INTRODUÇÃO

A área de conhecimento de Engenharia de *Software* (ES) é ampla, complexa e é de grande importância no desenvolvimento de sistemas. De modo geral, a ES fornece toda a estrutura para o desenvolvimento e manutenção do *software*. A área de ES pode ser dividida em diferentes subáreas de conhecimento como: Requisito de *Software*, Projeto de *Software*, Qualidade de *Software*, Teste de *Software*, entre outras (IEEE, 2004). A área de Qualidade de *Software* destaca-se devido ao forte impacto no desenvolvimento de um projeto, pois um produto que não está de acordo com requisitos de qualidade pode causar muitos prejuízos tanto para o cliente, quanto para a própria empresa de desenvolvimento. O Teste de *Software* é um elemento de um tópico mais amplo, muitas vezes conhecido como verificação e validação, que abrange diversas atividades de Garantia de Qualidade de *Software* (Pressman, 2011). Tópicos relacionados a Teste de *Software* são encontrados em diversos currículos de referência na área de computação. No entanto, a pesquisa de Wangenheim e Silva (2009) indica que os conhecimentos de ES não estão sendo devidamente abordados na graduação. Especificamente em relação a Teste de *Software*, a pesquisa indica que, além do conhecimento atual dos profissionais ser inferior ao adquirido na graduação, esse conhecimento ainda está abaixo do que eles consideram importante.

Ao analisar os anais de eventos como o Fórum de Educação em Engenharia de *Software* (anos de 2009, 2010 e 2011) e *Conference on Software Engineering Education and Training* (ano de 2011), observa-se uma tendência à adoção de práticas educacionais alternativas à tradicional (entendida neste trabalho como unicamente expositiva), entre as quais está o uso de jogos. A aprendizagem baseada em jogos digitais é uma abordagem inovadora no domínio das universidades e está se tornando uma nova forma de conteúdo interativo, que merece ser explorada (Pivec, 2007). O relacionamento entre jogos e educação é bem anterior à criação dos jogos digitais (Mattar, 2010). Apesar da ampla discussão do tema na atualidade, o uso de jogos na educação não é nenhuma novidade, principalmente na educação infantil. Vários autores, entre eles Mattar (2010), Papastergiou (2009), Münz *et al.* (2007) e Garris, Ahlers e Driskel (2002), destacam que o uso de jogos na educação pode ser um fator motivacional no processo de ensino-aprendizagem. A tecnologia de interação em tempo real − a mesma utilizada para desenvolver os jogos − pode servir, ainda, como apoio para uma maior interação interpessoal em situações presenciais ou virtuais (Azevedo, 2005). Logo, o uso de jogos para treinar, aprender e executar atividades reais em ambientes

realísticos pode melhorar o desempenho dos estudantes, pois possibilita a vivência de experiências de aprendizagem produzidas individualmente de acordo com o estilo do estudante (Prikladnicki e Wangenheim, 2008). No contexto educacional, os jogos podem ser denominados jogos sérios (*serious games*), que são jogos que possuem a finalidade instrucional explícita. Existem vários jogos na literatura que abordam diferentes áreas da ES, sendo que, alguns deles foram descritos e avaliados em Wangenheim, Kochanski e Savi (2009). Especificamente na área de Teste de *Software*, ressaltam-se os jogos os jogos U-TEST (Silva, 2010) e Jogo das 7 Falhas (Diniz e Dazzi, 2011), no entanto, os mesmos se atêm apenas aos ensinos de teste de unidade e de teste de caixa preta, respectivamente.

Jogos podem estar inseridos em mundos virtuais, sendo explorados principalmente em jogos do gênero *Massive Muliplayer Online* (MMO). O uso de mundos virtuais é uma possibilidade recente no apoio à educação (Peachey *et al*., 2010). Logo, os mundos virtuais em três dimensões (3D) podem ser vistos como uma nova tecnologia educacional, podendo exercer diferentes funções como jogo, rede social, simulador, entre outros. Esses mundos possibilitam interações próximas às interações reais entre indivíduos, tendo em vista que a partir de uma personificação do indivíduo (*avatar*) no mundo virtual, o usuário pode se comunicar via canal de voz e exibir expressões corporais, travar conversas via texto e realizar ações impossíveis no mundo real como, por exemplo, voar. Esse tipo de interação é bem diferente da ocorrida na utilização de outras tecnologias educacionais e apresenta um grande potencial na educação superior (Wankel e Kingsley, 2009). Nesse contexto, mundos virtuais podem ser definidos como ambientes compartilhados, persistentes e *online*, que permitem que vários usuários criem conteúdos e experiências, colaborem e se comuniquem com outros usuários por meio de um *avatar* (Fox, Kelly e Patil, 2010).

Ao se falar em mundos virtuais, é importante destacar o papel do Second Life[1] (SL) na popularização deste tipo de ambiente. Por volta de 2006 / 2007, o SL atingiu seu auge em relação ao número de usuários, acessos, estabelecimento de empresas e exposição na mídia. Contudo, em 2008, esses indicadores baixaram significativamente (Taurion, 2009). Valente e Mattar (2007) destacam como fatores que podem ter colaborado para o declínio na popularidade do SL a necessidade de máquinas modernas com boa capacidade de processamento e de banda larga. Comentam, ainda, sobre as altas expectativas dos usuários, principalmente na área comercial, pois alguns usuários acreditavam que o SL poderia ser uma verdadeira revolução na interação humana. Assim, algumas empresas realizaram grandes

[1] http://www.secondlife.com.br

investimentos (estimulados pela extrema divulgação nos meios de comunicação), mas, por não encontrarem retorno financeiro imediato, simplesmente abandonaram o ambiente. O SL foi visto como um ambiente para divulgação de produtos e marcas, embora ele não fosse orientado à comunicação de massa (Taurion, 2009). É importante destacar que, a cada nova mídia, novos paradigmas são construídos e outros derrubados. Portanto, é necessário um esforço adicional de adequação para poder extrair resultados satisfatórios desses ambientes inovadores (Valente e Mattar, 2007). Segundo Taurion (2009), atualmente, observa-se o amadurecimento no uso do SL, deixando de ser uma moda, para ser usado de forma mais consistente, pois muitas empresas estão investindo em novos projetos, mais maduros do que os que foram anteriormente lançados. Trabalhos como de Nevo e Nevo (2011) e Dreher *et al.* (2009) confirmam essa tese, pois indicam inovações na utilização de mundos virtuais.

Apesar das controvérsias relacionadas aos mundos virtuais, inúmeros trabalhos destacam seu potencial educacional, dentre eles estão Bainbridge (2010), Peachey *et al.* (2010), e Wankel e Kingsley (2009), que discutem o uso de mundos virtuais na educação superior, ferramentas, oportunidades e desafios; e, ainda,Valente e Mattar (2007), que mostram como diferentes áreas do conhecimento (Ciência da Computação, História, Administração, entre outros) podem ser trabalhadas no SL, destacando experiências nacionais e internacionais. No entanto, ainda não existe consenso sobre a melhor forma de utilizar os mundos virtuais, ou resultados concretos e garantidos. Pontos negativos, como requisitos de *hardware* e banda larga, hoje, são minimizados devido aos avanços nessas áreas. No caso deste trabalho, como o público alvo é formado por estudantes de graduação na área de computação, presume-se que a maioria tenha acesso à banda larga e a equipamentos de melhor desempenho que os dos usuários convencionais. Espera-se, ainda, que a maioria das instituições de ensino superior com cursos na área de computação, possua laboratórios (ou pelo menos um laboratório) compatíveis com as especificações de *hardware* e de rede necessárias para visualização do mundo virtual 3D.

Apesar de ter o SL como uma de suas principais referências, este trabalho prioriza a utilização de tecnologias de código aberto. *Software* de código aberto é um *software* que deve ser distribuído com código fonte incluído ou disponibilizado de alguma maneira. O código fonte deve permitir que outro programador possa realmente utilizá-lo e mantê-lo. A licença desse tipo de *software* não deve restringir a distribuição ou a modificação do código ou, ainda, trabalhos derivados nos mesmos termos (Kavanagh, 2004).

Desde 2003, o governo brasileiro está pondo em prática a estratégia de adoção do *software* livre nas instituições governamentais (CISL, 2011) e muitas empresas privadas

utilizam *software* de código aberto, principalmente em se tratando de aplicações *Web* (Kavanagh, 2004). Algumas vantagens podem ser destacadas na adoção de *software* de código aberto como: a possibilidade de alteração do código fonte, permitindo a adaptação às necessidades de uma pessoa ou organização; o apoio de uma comunidade de desenvolvedores; e a redução de custos (muitos desses *softwares* são gratuitos). Neste trabalho, optou-se pela utilização do mundo virtual OpenSim, que é muito semelhante ao SL, mas que, no entanto, é um *software* de código aberto. Também foram utilizados o AVEA Moodle, o Sistema de Gerenciamento de Banco de Dados (SGBD) MySQL, o *framework* CodeIgniter e o servidor *Web* Apache.

1.1 Objetivos

Este trabalho tem como objetivo geral desenvolver e avaliar um jogo sério, inserido em um mundo virtual, para apoiar o ensino-apredizagem de estratégias de Teste de *Software* em cursos de graduação na área de computação.

Os objetivos específicos são os seguintes:

- Elaborar o *design* instrucional;
- Realizar a modelagem do jogo;
- Preparar o ambiente de desenvolvimento;
- Implementar o jogo sério para apoio Teste de *Software* em um mundo virtual;
- Preparar o experimento de validação do jogo;
- Aplicar o jogo junto a estudantes de graduação, na área de computação; e
- Analisar os resultados obtidos.

1.2 Organização do trabalho

O trabalho está organizado da seguinte forma: o capítulo 2 aborda o tema de Teste de *Software* em seu contexto profissional e educacional, destacando, ainda, o tópico de Estratégias de Teste de *Software* e apresenta algumas ferramentas livres de apoio ao Teste de *Software*; no capítulo 3, são apresentados conceitos de jogos no contexto educacional,

englobando conceitos gerais de jogos, jogos sérios e aprendizagem baseada em jogos; mundos virtuais são descritos no capítulo 4, destacando as características do mundo virtual OpenSim, a integração de mundos virtuais com o AVEA Moodle e também a relação entre mundos virtuais, jogos e simulações; o capítulo 5 relaciona os trabalhos correlatos, que foram divididos em jogos para Engenharia de *Software* e aplicações educacionais em mundos virtuais; já o capítulo 6 apresenta o jogo sério desenvolvido, denominado JETS, para apoio ao ensino-aprendizagem de Teste de *Software*; no capítulo 7, ~~é~~ são relatados o experimento de aplicação do jogo e a análise da execução do mesmo; por fim, o capítulo 8 apresenta as considerações finais e os trabalhos futuros.

2 TESTE DE SOFTWARE

Engenharia de *Software* é uma disciplina que relaciona todos os aspectos da produção do *software* desde os estágios iniciais de especificação do sistema, até depois do mesmo entrar em operação (Sommerville, 2007). A Qualidade de *Software* pode ser definida como a gestão de qualidade efetiva aplicada no desenvolvimento de um produto útil que forneça valor para os que produzem e para os que utilizam (Pressman, 2011). Por sua vez, os requisitos de *software* definem as características de qualidade exigida do *software* e da influência dos métodos de medição e de critérios de aceitação para avaliação dessas características (IEEE, 2004). A aplicação de testes em um programa é a maneira mais comum de verificar se ele atende a sua especificação e realiza o que o cliente deseja (Sommerville, 2007). O Teste de *Software* pode ser definido como o processo de executar um programa com a finalidade de encontrar erros (Myers *et al*., 2004) ou como uma atividade realizada para avaliar e melhorar a qualidade do produto, identificando defeitos e problemas (IEEE, 2004).

Neste capítulo o Teste de *Software* será abordado no contexto profissional e educacional, bem como será detalhado o tópico de Estratégias de Teste de *Software* e de Ferramentas de Teste de *Software*.

2.1 Contexto profissional

A atividade de Teste de *Software* é um elemento crítico de Garantia de Qualidade de *Software*. Os altos custos associados às falhas de *software* forçam as organizações a executar a atividade de teste de forma cuidadosa e bem planejada (Pressman, 2011). Um dos motivos de testar o *software* é adicionar valor a ele. Logo, agregar valor através de testes significa aumentar a qualidade ou a confiabilidade do *software*, e aumentar a confiabilidade do *software* significa encontrar e remover erros (Myers *et al.*, 2004).

Atualmente, *softwares* atingem potencialmente milhões de pessoas, possibilitando o desempenho de suas funções com eficácia e com eficiência ou causando frustração e perdas de trabalho e de negócios (Myers *et al*., 2004). Sem dúvida, o impacto causado pela utilização de *softwares* no dia a dia dos usuários é muito alto, o que torna fundamental que os mesmos possuam aderência aos fatores de qualidade de *software*. Segundo Myres *et al*. (2004), em um

típico projeto de *software*, aproximadamente, cinquenta por cento do tempo e mais de cinquenta por cento do custo total do projeto são gastos unicamente com Teste de *Software*. Já Pressman (2011), estima que uma organização gaste entre trinta e quarenta por cento do esforço de projeto total em testes.

Nesse contexto, as exigências da indústria de *software* estão crescendo, assim como os *softwares* estão se tornando cada vez mais complexos e com isso o número potencial de falhas cresce, bem como os custos para corrigi-las (Hass, 2008). Os serviços de teste estão sendo terceirizados para empresas especializadas e a maioria dos grandes projetos de desenvolvimento de *software* já envolve o uso de testadores especializados, de modo que a performance e os ganhos financeiros dessas empresas dependem do desempenho dos profissionais de teste (Kanij, Merkel e Grundy, 2011). Logo, é importante que os profissionais formados em cursos de graduação em Computação e áreas afins estejam preparados para atender às exigências do mercado.

2.2 Contexto educacional

A área de conhecimento de Engenharia de *Software* (ES) é ampla, e é um assunto recorrente nos currículos dos cursos de graduação na área de computação. No entanto, a pesquisa de Wangenheim e Silva (2009) confirma que os profissionais, bacharéis da área de Ciência da Computação (CC), aprendem mais sobre tópicos de ES depois da graduação, ou seja, as competências de ES podem não estar sendo adequadamente abordadas nos cursos de CC. Stroustrup (2010) também discute as contradições dos currículos do curso de CC em relação às necessidades da indústria. Muitos graduados ingressam no mercado de trabalho com habilidades de desenvolvimento excepcionais, mas com deficiências em teste e depuração (Astigarraga *et al.*, 2010). Na pesquisa de Wangenheim e Silva (2009), bacharéis em Ciência da Computação, que atuam na área de Computação, responderam a um questionário com pontuação de zero a cinco. A Figura 1 ilustra essas respostas. Nesse gráfico é representado o conhecimento aprendido na graduação, o conhecimento adquirido desde a graduação e o conhecimento atual. Ainda se destaca a importância de cada uma dessas disciplinas na vida profissional. É possível observar que em todas as disciplinas existe uma lacuna entre o conhecimento adquirido na graduação e o conhecimento atual. No caso específico da disciplina de Teste de *Software*, pode-se observar que, apesar de grande parte do

conhecimento ter sido obtido na graduação, o conhecimento atual ainda está abaixo da importância do mesmo.

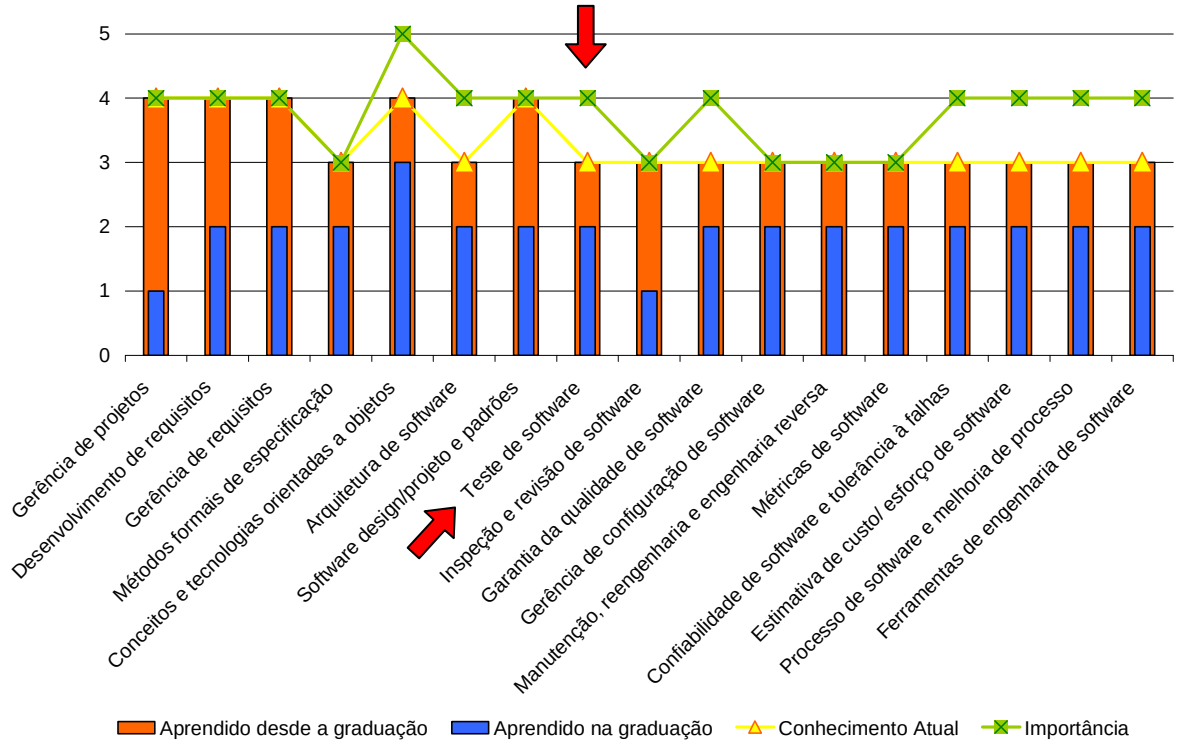


Figura 1 – Importância e quantidade de conhecimentos aprendidos por profissionais da área de *software* (Fonte: Wangenheim e Silva, 2009)

Uma das causas dessa deficiência na área de ES pode estar relacionada à carência de práticas educacionais alternativas à tradicional (unicamente expositiva). As Diretrizes Curriculares dos Cursos de Bacharelado em Ciência da Computação, Engenharia de Computação, Engenharia de *Software* e Sistemas de Informação e dos cursos de Licenciatura em Computação destacam o seguinte sobre a metodologia de ensino:

> O professor deve mostrar as aplicações dos conteúdos teóricos, ser um mediador, estimular a competição, a comunicação, provocar a realização de trabalho em equipe, motivar os alunos para os estudos e orientar o raciocínio e desenvolver as capacidades de comunicação e de negociação. O projeto pedagógico deve prever o emprego de metodologias de ensino e aprendizagem que promovam a explicitação das relações entre os conteúdos abordados e as competências previstas para o egresso do curso. A metodologia de ensino deve desenvolver uma visão sistêmica para resolução de problemas (BRASIL, 2003).

Um professor, ministrando uma aula unicamente expositiva, pode não conseguir estimular a comunicação, a negociação e a competição e ainda realizar trabalhos em equipe e

desenvolver a visão sistêmica do estudante para resolução de problemas. Logo, o uso de estratégias educacionais alternativas torna-se fundamental. Prikladnicki *et al*. (2009) destaca as seguintes estratégias para o ensino de ES: jogos, dinâmicas de grupo, EAD (educação à distância), *capstone projects* (atividade em grupo em que estudantes executam um projeto do início ao fim) e atividades lúdicas no aprendizado por analogia. Existem vários trabalhos na literatura que abordam o uso de jogos para o ensino-aprendizagem de diferentes áreas de ES, sendo que maiores detalhes sobre jogos e trabalhos correlatos serão abordados, respectivamente, nos capítulos 3 e 5.

2.2.1 Análise de currículos de referência

A tarefa de analisar os currículos dos cursos de graduação na área de computação não é simples, tendo em vista que, apenas no Brasil, existem 1916 cursos, sendo 1881 presenciais e 35 à distância (INEP, 2010). Por essa razão, foi feita a análise de currículos de referência, que têm como objetivo servir como base para criação desses cursos. A Sociedade Brasileira de Computação (SBC) propõe dois currículos de referência: um para os cursos de Bacharelado em Ciência da Computação (CC) e Engenharia de Computação (EC) (SBC, 2005), e outro para cursos de Bacharelado em Sistemas de Informação (SI) (SBC, 2003). No currículo proposto pela SBC para os cursos de CC e EC, as matérias da área de computação estão organizadas em dois núcleos: Fundamentos da Computação e Tecnologia da Computação. O núcleo de Tecnologia da Computação compreende, entre outras, a matéria de Engenharia de *Software*, sendo que o assunto Teste de *Software* é abordado em apenas um dos tópicos dessa matéria, que é Verificação, Validação e Teste. No currículo do curso de SI, as matérias de computação são organizadas em três núcleos: Fundamentos da Computação, Tecnologia da Computação e Sistemas de Informação. No núcleo de Tecnologia da Computação, assim como no currículo de CC e EC, é definida a matéria de Engenharia de *Software*, no entanto, a SBC (2003) recomenda que a matéria de ES seja abordada com profundidade e deve cobrir, entre outros, o tema de Teste de *Software*.

A *Association for Computing Machinery* (ACM), em conjunto com outras associações, também propõe currículos de referência para cursos de graduação na área de computação, sendo que esses cursos são os seguintes: Ciência da Computação (ACM e IEEE, 2008), Engenharia da Computação (IEEE e ACM, 2004a), Sistemas de Informação (ACM e AIS,

2010), Tecnologia de Informação (ACM e IEEE, 2008) e Engenharia de *Software* (IEEE e ACM, 2004b). Nesses currículos de referência, com exceção do currículo de SI, são definidos números mínimos de horas necessárias para cobrir determinado assunto. Maiores detalhes sobre como o conteúdo referente a Teste de *Software* é distribuído nos currículos da ACM podem ser observados em Silva, Müller e Bernardi (2011). A análise desses currículos indica que todos os cursos abordam assuntos relacionados a Teste de *Software*. Contudo, dentre esses cursos, há variações no que concerne à relevância do assunto nos currículos de graduação. O curso que menos aborda Teste de *Software* é o curso de Tecnologia da Informação, seguido de Ciência da Computação e de Engenharia da Computação. O curso de Engenharia de *Software* é o que aborda o tema de forma mais completa, inclusive, aparece de forma complementar em várias disciplinas.

2.3 Estratégias de Teste de Software

Por ser um dos focos deste trabalho, o tópico Estratégia de Teste de *Software* (Pressman, 2011; Sommerville, 2007) será detalhado nesta seção. A Estratégia de Teste de *Software* fornece um roteiro que descreve os passos a serem executados como parte do teste, define quando esses passos são planejados e então executados, e quanto trabalho, tempo e recursos serão necessários (Pressman, 2011). Portanto, qualquer estratégia de teste deve incorporar planejamento dos testes, projeto de casos de teste, execução dos testes, coleta e avaliação dos dados resultantes (Pressman, 2011). Basicamente, a Estratégia de Teste de *Software* é um *template* que define um conjunto de etapas nas quais são especificadas as técnicas e os métodos de teste. No entanto, não existe uma estratégia única, pois diferentes estratégias podem ser utilizadas para testar um *software*, desde aplicar os testes apenas ao final de toda a implementação (o que não é nada aconselhável) até o teste diário de tudo aquilo que é desenvolvido (o que é muito caro). Portanto, deve-se procurar um equilíbrio. Ainda, a estratégia depende do tipo de aplicação desenvolvida, ou seja, a estratégia varia se a aplicação é convencional, orientada a objeto ou a uma aplicação *Web* (Pressman, 2011). Logo, ao se definir uma estratégia, deve-se levar em consideração questões como orçamento, cronograma, recursos humanos e materiais, e tipo de aplicação desenvolvida.

O Plano de Teste é o principal documento que norteia a Estratégia de Teste de *Software*. O Plano de Teste descreve o escopo, a abordagem e os recursos, bem como a

agenda de atividades de teste previstas (IEEE, 1998). Os papéis na atividade Teste de *Software* podem ser divididos em:

- Testador de *Software*: responsável pela execução de teste;
- Analista de Teste de *Software*: responsável pela modelagem e elaboração do teste;
- Arquiteto de Teste de *Software*: responsável pelas atividades relacionadas com a infraestrutura do teste; e
- Líder da Equipe de Teste: responsável pela liderança de um projeto de teste específico relacionado a um projeto novo de desenvolvimento ou manutenção.

As atividades de teste são executadas por uma equipe, sendo que o número de membros e a respectiva distribuição dos papéis na equipe dependem do tipo e do tamanho do projeto. Cada papel exige habilidades específicas, contudo, um indivíduo pode assumir mais de um papel na equipe (Hass, 2008).

2.4 Ferramentas livres de apoio ao Teste de Software

Ferramentas podem ser utilizadas em diferentes aspectos do Teste de *Software* para apoiar o gerenciamento do processo de teste, a automação de testes de unidade e de testes funcionais, também para realizar testes de performance, segurança, entre outros. Entre as ferramentas de gerenciamento de teste livres mais populares estão FitNesse[2] e TestLink[3]. FitNesse é uma ferramenta mais simples e com menos recursos, que funciona basicamente como uma *wiki*, que é um conjunto de páginas *Web* que podem ser editadas de forma colaborativa, de modo que a documentação de teste é criada a partir dessas páginas editáveis. Ela permite, ainda, a execução de testes de aceitação por meio de tabelas, logo, a curva de aprendizagem dessa ferramenta é menor do que a do TestLink. No entanto, a ferramenta TestLink é mais completa, dá suporte a vários documentos, como Projeto de Teste, Especificação de Teste, Plano de Teste, Caso de Teste, entre outros. Ainda permite o

[2] http://fitnesse.org/

[3] http:// www.teamst.org/

gerenciamento de perfis de usuários e gera relatórios e gráficos, também possibilitando a importação e a exportação de documentos em diferentes formatos.

2.5 Conclusões do capítulo

O Teste de *Software* é uma das áreas de conhecimentos relacionadas à Engenharia de *Software*. No contexto de desenvolvimento de sistemas, o Teste de *Software* está diretamente relacionado à qualidade do produto, questão de grande impacto tanto nos custos do projeto, quanto na satisfação do cliente. Já no contexto educacional, apesar do conteúdo de Teste de *Software* ser abordado em todos os currículos de referência analisados, pesquisas indicam que esse conteúdo não está sendo devidamente abordado na graduação. Ainda, as Diretrizes Curriculares dos Cursos de Graduação na área de computação (Brasil, 2003), em relação à metodologia de ensino, recomendam que o professor deve: (a) mostrar as aplicações dos conteúdos teóricos; (b) atuar como um mediador; (c) estimular a competição e a comunicação; (d) realizar trabalhos em equipe; (e) motivar os estudos e orientar o raciocínio; (f) desenvolver as capacidades de comunicação e de negociação; e (g) desenvolver uma visão sistêmica para resolução de problemas. Essas recomendações indicam a necessidade da adoção de estratégias pedagógicas alternativas, como o uso de jogos − o que será abordado no próximo capítulo.

3 JOGOS NO CONTEXTO EDUCACIONAL

Neste capítulo, serão abordados conceitos referentes a jogos. A primeira seção contextualiza e apresenta definições e classificações gerais relacionadas a jogos. Em seguida, jogos sérios são detalhados, bem como a abordagem de aprendizagem baseada em jogos.

3.1 Conceitos gerais sobre jogos

Os jogos estão atrelados à história e à cultura da humanidade. Desde a infância, os indivíduos se deparam com jogos recreativos, de tabuleiro ou de *video game*, sendo amplamente utilizados na educação infantil (Kishimoto, 2008). Na vida adulta, alguns indivíduos continuam utilizando os jogos, de forma recreativa ou séria. A Teoria de Jogos (Neumann e Morgenstern, 2007), que teve início em 1944, é aplicada em diferentes áreas como matemática, economia e computação. Apesar da recorrência desse tema, não é uma tarefa simples definir e classificar jogos devido às suas diferentes abordagens e aplicações. Portanto, serão destacadas algumas definições voltadas para o contexto deste trabalho:

- Um jogo é um sistema no qual os jogadores se envolvem em um conflito artificial, definido por regras, que resultam em algo quantificável (Salen e Zimmerman, 2004);
- Um jogo é uma atividade de resolução de problemas, conduzida com uma atitude lúdica (Schell, 2008); e
- Um jogo é uma atividade lúdica, realizada no contexto de uma realidade simulada, na qual os participantes tentam alcançar, pelo menos, uma meta arbitrária, não trivial, agindo de acordo com as regras (Adams, 2010).

Alguns jogos são desenvolvidos especificamente para dispositivos eletrônicos como computadores, consoles de *video-game*, PDAs (*Personal Digital Assistant*), entre outros. Esses jogos são conhecidos por diversos nomes, como jogos digitais, jogos eletrônicos, jogos computacionais, jogos de *video-game* ou, simplesmente, *games*.

3.1.1 Elementos

Existem diferentes abordagens na literatura referentes às características ou aos elementos que compõem os jogos, entre elas estão as abordagens de Adams (2010), Fullerton, Swain e Hoffman (2008) e Schell (2008). Neste trabalho, especificamente, serão destacadas as ideias de Fullerton, Swain e Hoffman (2008). Os autores propõem que jogos são compostos por elementos formais e por elementos dramáticos. Os elementos formais são essenciais e constituem a estrutura do jogo. Esses elementos são os seguintes: (a) Jogadores: jogos são experiências projetadas para os jogadores e eles devem aceitar voluntariamente as regras e as limitações do jogo para poder jogar; (b) Objetivos: definem o que os jogadores estão tentando realizar dentro das regras do jogo; (c) Procedimentos: são os métodos do jogo e as ações que os jogadores podem realizar para alcançar os objetivos; (d) Regras: definem os objetivos do jogo e as possíveis ações dos jogadores; (e) Recursos: são ativos que podem ser usados para realizar determinadas metas; (f) Conflitos: emergem dos jogadores enquanto tentam cumprir as metas do jogo dentro de suas regras e de seus limites; (g) Limites: são referências (físicas ou conceituais) que delimitam o domínio do jogo; e (h) Resultado: determina quem ganha ou quem perde o jogo, portanto, o resultado deve ser incerto para prender a atenção dos jogadores − se eles puderem antecipar o resultado, podem abandonar o jogo; os resultados são geralmente mensuráveis, mas isso não é uma condição obrigatória.

Já os elementos dramáticos são responsáveis por envolver os jogadores emocionalmente, criando um contexto dramático para os elementos formais. Esses elementos são os seguintes: (a) Desafio: é uma ou mais tarefas que exige certa quantidade de trabalho para criar um sentimento de realização e de prazer, quando a tarefa é completada − o desafio é determinado pela habilidade do jogador; (b) *Play*[4]: pode ser considerada a liberdade de movimento dentro de uma estrutura mais rígida − no caso de jogos, as restrições das regras e dos procedimentos são essa estrutura e o jogo é a liberdade dos jogadores para atuar dentro dessas regras; (c) Premissa: estabelece a ação do jogo dentro de uma configuração ou metáfora. Sem uma premissa, muitos jogos podem ser muito abstratos para os jogadores tornarem-se emocionalmente envolvidos com os resultados; (d) Personagem: é um agente da

[4] Este elemento não foi traduzido, pois não existe uma palavra em português que englobe todas as características da palavra play, que está relacionada à liberdade de movimento e ao ato de jogar, brincar, divertir-se e interpretar.

história − proporciona uma maior identificação e envolvimento por parte do jogador; e (e) História: em geral, é limitada a uma história de fundo, que dá contexto para o conflito do jogo.

3.1.3 Classificações

Existem diferentes formas de classificar um jogo. No entanto, vários autores propõem classificações de jogos por gênero, sendo que gênero, nesse contexto, pode ser entendido como uma categoria de jogos definida por um conjunto específico de características, independente do conteúdo do jogo. A Tabela 1 apresenta um apanhado de tipos de jogos encontrados na literatura. Essa tabela destaca o gênero, sua respectiva descrição e os autores que a citaram.

Tabela 1 – Classificação de jogos por gênero

Gênero	**Descrição**	**Autores**
Ação	São jogos em tempo real, nos quais o jogador deve responder com velocidade ao que está ocorrendo na tela, exigindo habilidades como reflexos rápidos e coordenação olho-mão.	Adams (2010); Azevedo (2005); Bates (2004); Finney (2004); Rollings e Adams (2003)
Aventura	São jogos baseados em uma história e está relacionada à exploração, em que o personagem sai em jornada para encontrar coisas e resolver enigmas.	Azevedo (2005); Bates (2004); Finney (2004); Rollings e Adams (2003)
Casual	São adaptações de jogos tradicionais como xadrez, paciência e damas; são jogos simples e fáceis de aprender, que não exigem muito do jogador.	Azevedo (2005); Bates (2004)
Construção e Gerenciamento	Neste tipo de jogo o objetivo é construir algo dentro do contexto de um processo contínuo, sendo que, quanto mais o jogador compreende e controla o processo, maior sucesso terá.	Adams (2010); Rollings e Adams (2003)
Educacional	São jogos que tem o objetivo de ensinar um conteúdo de conhecimento específico.	Azevedo (2005); Bates (2004)

Enigma ou Labirinto	São jogos puramente voltados para o desafio intelectual na solução de problemas.	Adams (2010); Azevedo (2005); Bates (2004); Finney (2004); Rollings e Adams (2003)
Esporte	São jogos que representam esportes coletivos ou individuais.	Adams (2010); Azevedo (2005); Bates (2004); Finney (2004)
Estratégia	São jogos no qual o jogador deve gerenciar recursos para atingir um determinado objetivo.	Azevedo (2005); Finney (2004); Rollings e Adams (2003); Adams (2010); Bates (2004)
God Games ou S*oftware Toy*	São jogos nos quais não existe a condição de ganhar; em geral, esses jogos proporcionam atividades atraentes e divertidas, em que o jogador tem liberdade para escolher ou criar.	Azevedo (2005); Bates (2004)
Luta	São jogos para um ou dois jogadores, em que o personagem é controlado de modo a efetuar ações de ataque e de defesa contra o personagem oponente.	Azevedo (2005); Bates (2004)
Massive Multiplayer Online	São jogos, de qualquer gênero, jogados por meio da *Internet*, seja por um ou por milhares de jogadores.	Azevedo (2005); Adams (2010); Bates (2004); Rollings e Adams (2003)
RPG	São jogos no qual o jogador dirige um ou mais personagens, geralmente configurados por ele, em alguma missão, em diversas tramas e cenários.	Adams (2010); Azevedo (2005); Bates (2004); Finney (2004); Rollings e Adams (2003)
Simulador	São jogos que reproduzem situações e/ou condições do mundo real, de modo que possibilitam ao jogador fazer coisas que ele dificilmente poderia fazer na vida real, em um ambiente controlado e seguro.	Adams (2010); Azevedo (2005); Bates (2004); Finney (2004); Rollings e Adams (2003)
Vida Artificial	São jogos que simulam criaturas que podem interagir com o ambiente, outros jogadores ou agentes.	Adams (2010)

Os jogos digitais podem ser classificados, em relação aos gráficos, como 2D (duas dimensões) ou 3D. Quanto ao número de jogadores, os jogos podem ser classificados como single player (um único jogador) ou multiplayer (vários jogadores). Outra forma de classificar os jogos, relevante para este trabalho, é pelo tipo de interação. Fullerton, Swain e Hoffman

(2008) apresentam sete padrões de interação do jogador: (a) Jogador único *versus* jogo: um único jogador concorre com um sistema de jogo; (b) Múltiplos jogadores individuais *versus* jogo: vários jogadores competem contra um sistema de jogo na companhia uns dos outros; (c) Jogador *versus* jogador: dois jogadores competem diretamente; Competição unilateral: dois ou mais jogadores competem contra um jogador; (d) Competição multilateral: três ou mais jogadores competem diretamente; (e) Jogo cooperativo: dois ou mais jogadores cooperam contra o sistema de jogo; e (f) Competição por equipe: dois ou mais grupos competem entre si.

Na literatura, os jogos usados no contexto educacional são denominados de várias formas, entre elas, jogos educacionais, jogos educativos, jogos instrucionais e jogos sérios. Basicamente, todas essas nomenclaturas se referem a jogos cujo objetivo é o ensino-aprendizagem de algum conteúdo instrucional. Atualmente, observa-se na literatura que a nomenclatura jogos sérios tem sido adotada, com frequência, para descrever jogos utilizados no contexto profissional e no ensino superior. Logo, por ter uma conotação mais adulta, essa nomenclatura foi adotada neste trabalho por apresentar uma abordagem de aprendizagem baseada em jogos para estudantes de graduação. Na próxima seção, jogos sérios serão detalhados.

3.2 Jogos sérios

Apesar do termo *serious games* (jogos sérios) ser utilizado desde 1970, na primeira edição de Abt (1987), apenas na última década esse termo adquiriu maior visibilidade. Um dos responsáveis foi a *Serious Games Initiative*[5] (SGI) fundada em 2002 pelo *Woodrow Wilson International Center for Scholars*[6]. Essa iniciativa visa estabelecer uma colaboração entre as indústrias de jogos eletrônicos e de projetos envolvendo o uso de jogos na educação, treinamento, saúde e políticas públicas. A expressão jogos sérios une a seriedade do pensamento e da resolução de problemas com o experimental e com a liberdade emocional do jogo. Portanto, os jogos sérios combinam as concentrações analítica e questionadora, do ponto de vista científico, com a liberdade intuitiva e com a recompensa construtiva dos atos

[5] http://www.seriousgames.org/

[6] http://www.wilsoncenter.org/

artísticos (Abt, 1987). A principal característica do jogo sério é sua finalidade educacional explícita e cuidadosamente pensada, não tendo como intenção principal ser utilizado por diversão, mas o que não significa que o jogo sério não deva ser divertido (Michael e Chen, 2006). Os jogos sérios possibilitam representações dramáticas de um assunto ou de um problema que está sendo estudado, e permite que os jogadores assumam papéis realísticos, encarem problemas, formulem estratégias, tomem decisões e tenham rápido *feedback* das consequências de suas ações. Tudo isso sem o custo das consequências dos erros do mundo real (Michael e Chen, 2006).

Jogos sérios possuem inúmeras aplicações e podem ser utilizados na educação (básica e superior), em corporações, organizações governamentais e não governamentais. O Departamento de Defesa dos Estados Unidos da América é uma das instituições que utilizam jogos sérios, tendo iniciado esse processo em 1997 (NRC, 2010). Nesse contexto, os jogos sérios oferecem um novo mecanismo para ensino e para treinamento, combinando jogos digitais com educação. Logo, jogos sérios podem ir além das vídeo-aulas e dos livros, permitindo aos jogadores não apenas o aprendizado, mas também a demonstração e a aplicação dos conhecimentos aprendidos (Michael e Chen, 2006). No entanto, a utilização de jogos sérios, assim como de qualquer outro tipo de jogo, requer planejamento, pois nenhum jogo pode ser bem sucedido se os jogadores não entenderem as regras, seus objetivos no jogo e as consequências de suas ações.

Nos últimos anos, o desenvolvimento de jogos sérios está crescendo rapidamente (SGI, 2011). Esse crescimento pode ser observado no estado da arte de jogos sérios para empresas e indústrias apresentado por Riedel e Hauge (2011) e, ainda, nos anais de eventos como *International Conference on Games and Virtual Worlds for Serious*[7]*, International Conference on Fun and Games*[8] e *Serious Play Conference*[9]. O crescimento dos jogos sérios é impulsionado não apenas pelos ganhos obtidos no *design* de jogos, alcance de audiência e de tecnologia de jogos, mas também pelos avanços nas principais tecnologias, como a *Internet* e as redes sociais (NRC, 2010). Percebe-se, ainda, a valorização das ideias, habilidades, tecnologias e técnicas utilizadas em jogos de entretenimento, pois vários jogos comerciais como SimCity[10], Civilization[11] e Hidden Agenda[12] têm sido utilizados como ferramentas de

[7] http://www.vs-games.org/

[8] http://fng2012.org/

[9] http://www.seriousplayconference.com/

[10] http://www.facebook.com/SimCity

ensino-aprendizagem em escolas e em universidades (SGI, 2011). Freitas e Liarokapis (2011) acreditam que o potencial dos jogos sérios irá provocar uma mudança de paradigma na forma como a educação e o treinamento são realizados. Esse novo paradigma é denominado aprendizagem baseado em jogos. Na próxima seção, conceitos relacionados à aprendizagem baseada em jogos serão detalhados.

3.3 Aprendizagem baseada em jogos

A aprendizagem baseada em jogos é uma abordagem que utiliza o jogo como uma ferramenta para que os alunos se engajem no aprendizado enquanto jogam (Sidhu, 2010). Segundo Prensky (2001), a aprendizagem baseada em jogos digitais tem como base duas premissas. A primeira é a de que os estudantes mudaram em alguns aspectos de fundamental importância. A segunda premissa é a de que esses indivíduos são de uma geração que, pela primeira vez na história, cresceu experimentando uma forma radicalmente nova de jogos (jogos de computador e de *video game*). Essa nova forma de entretenimento moldou as preferências e as habilidades, e ofereceu um enorme potencial para a aprendizagem, tanto a das crianças, como a dos adultos. A aprendizagem baseada em jogos impulsiona a inovação através da educação, por isso a importância de envolver todos os interessados como educadores, *designers* e desenvolvedores de jogos, pesquisadores e estudantes em todas as fases do processo de concepção e de implementação da abordagem (Freitas e Liarokapis, 2011).

O modelo de aprendizagem proposto por Garris, Ahlers e Driskell (2002) apresentado na Figura 2, ilustra como ocorre o processo de aprendizagem baseada em jogos. O início desse processo ocorre com a entrada de um sistema que combina conteúdo instrucional com característica de jogo, desencadeando um ciclo que inclui decisões ou reações do usuário (como diversão ou interesse), o comportamento do usuário (como maior persistência ou tempo na tarefa) e o *feedback* do sistema. Com o engajamento do usuário no jogo, ele alcança os objetivos educacionais e o ciclo termina com resultados de aprendizagem.

---

[11] http://www.civilization.com/

[12] http://www.mcli.dist.maricopa.edu/proj/sw/games/hidden-agenda.html

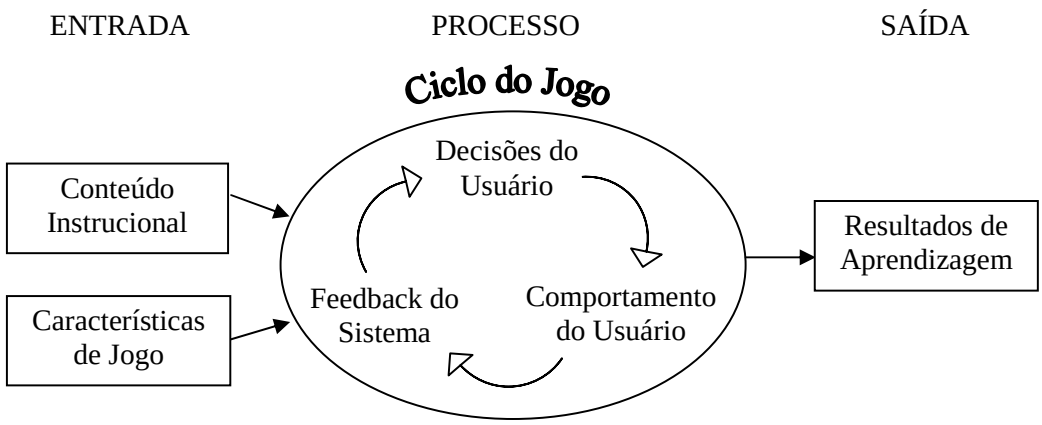


Figura 2 – Modelo de aprendizagem baseada em jogos (Fonte: Garris, Ahlers e Driskel, 2002)

Ao utilizar jogos de computador ou jogos em geral para fins educacionais, vários aspectos do processo de aprendizagem são apoiados: os estudantes são estimulados a combinar o conhecimento de diferentes áreas para tomar uma decisão ou escolher uma solução entre várias; os estudantes podem testar como o resultado do jogo muda com base em suas decisões e ações; e, por fim, os mesmos são incentivados a interagir com outros membros da equipe, discutir e negociar as etapas subsequentes, melhorando assim, entre outras coisas, suas habilidades sociais (Pivec e Dziabenko, 2004).

Mattar (2010) destaca que o uso de jogos pode favorecer a aprendizagem tangencial. Essa aprendizagem está relacionada ao conhecimento adquirido pelo indivíduo quando exposto às atividades em um contexto no qual está envolvido e não ao conhecimento que está sendo diretamente ensinado. Ou seja, na atividade de jogar, o indivíduo desenvolve habilidades e aprende coisas como, por exemplo, uma língua estrangeira, mesmo que esse não seja o objetivo do jogo.

Ainda, segundo Mattar (2010), outra questão relevante em relação aos jogos é a forma de lidar com o erro, pois o fracasso na educação tradicional é muito diferente do que ocorre nos *video games*, no qual o custo do fracasso é normalmente mínimo, tendo em vista que, quando os jogadores fracassam, têm a possibilidade imediata de recomeçar. Essas características do fracasso nos jogos permitem que os jogadores se arrisquem e experimentem hipóteses que seriam muito difíceis de serem testadas em situações reais, em que o custo do fracasso é maior (Mattar 2010). Portanto, é possível concluir que os jogos possibilitam o aprendizado através dos erros cometidos, sem que esse tipo de experiência gere muita frustração nos estudantes, estimulando novas tentativas.

3.4 Conclusões do capítulo

Os conceitos relacionados a jogos são amplos e variam de acordo com o enfoque. Os jogos são constituídos de elementos formais e dramáticos. De modo geral, os jogos podem ser classificados quanto ao gênero, aos gráficos, ao número de participantes e ao tipo de interação. Jogos na educação (digitais ou não) são discutidos há décadas e são amplamente utilizados na educação infantil (Kishimoto, 2008). No entanto, ao se tratar do ensino de jovens e adultos, a utilização de jogos ainda é um assunto controverso. No contexto deste trabalho, os jogos são chamados de jogos sérios, por terem o objetivo formal de ensinar, vinculado ao lúdico dos jogos. A aprendizagem baseada em jogos visa o engajamento do estudante na atividade de tal modo que ele alcance os objetivos de aprendizagem. Favorece, ainda, a aprendizagem tangencial e permite ao estudante aprender com os erros sem os custos negativos que teriam no mundo real. Em relação aos jogos sérios e à abordagem de aprendizagem baseada em jogos, além de ser fundamental uma elaboração efetiva do conteúdo educacional, também é importante considerar questões relacionadas aos elementos do jogo e às tecnologias utilizadas em seu desenvolvimento (Bachen e Raphael, 2011).

A difusão dos jogos, o uso generalizado da *Internet* e a necessidade de criar práticas educacionais mais atraentes têm levado ao surgimento de jogos sérios como uma nova forma de educação e de formação (Freitas e Liarokapis, 2011). Isso sugere uma grande expectativa dos estudantes em relação aos jogos. Os jogos sérios mesmo tendo finalidade educacional explícita, não excluem a necessidade de diversão. Dessa forma, não só as questões relacionadas aos aspectos pedagógicos devem ser consideradas, mas também as questões relacionadas às características do jogo em si, de modo que se aproximem ao máximo dos jogos de entretenimento. Isso, é claro, não é um objetivo trivial, já que a indústria de *games* já superou a do cinema em termos de receitas (Fullerton, Swain e Hoffman, 2008). Apesar de haver custos altos nesse tipo de empreendimento, existem tecnologias gratuitas e de código aberto que podem ser utilizadas no desenvolvimento de jogos, reduzindo assim os custos de produção. É importante ressaltar, ainda, que o avanço das pesquisas e a obtenção de resultados positivos possibilitam mais investimentos na área. Entre as tecnologias para o desenvolvimento de jogos está a *game engine* ou motor de jogo, que é um *framework* ou sistema que pode ser reusado para produzir vários jogos diferentes (Thorn, 2010); outra opção são os mundos virtuais 3D. Alguns desses mundos fornecem ferramentas de construção de objetos, que suportam *scripts*, permitindo o desenvolvimento de jogos com a utilização da

engine do próprio mundo. Por ser foco deste trabalho, mundos virtuais serão detalhados no próximo capítulo.

4 MUNDOS VIRTUAIS

Os mundos virtuais, também conhecidos na literatura como Metaversos e MUVEs (*Multi-User Virtual Environments*), são ambientes *online* persistentes gerados por computador, onde as pessoas podem interagir de maneira comparável ao mundo real, seja para o trabalho ou para o lazer (Bainbridge, 2010). Os mundos virtuais combinam gráficos 3D interativos, tecnologia de simulação, realidade virtual, *Voice over Internet Protocol* (VoIP), e mídia digital para fornecer aos usuários habilidades ilimitadas para se comunicar, colaborar e explorar (Hodge, Collins e Giordano, 2011). Esse conjunto de tecnologias fornece uma interface para um mundo tridimensional, de modo que o usuário acredita estar realmente nesse mundo e, intuitivamente, passa a interagir com esse ambiente imersivo e dinâmico (Valente e Mattar, 2007).

Os mundos virtuais possibilitam a realização de uma série de atividades, entre elas, atividades de cunho educacional e de treinamento. As qualidades únicas dos mundos virtuais 3D permitem oportunidades para experiências sensoriais imersivas, contextos e atividades para o aprendizado experimental, simulação, modelagem de cenários complexos, entre outros, com oportunidade de colaboração e co-criação que não podem ser facilmente experimentadas em outras plataformas (Valente e Mattar, 2007). Assim, instituições acadêmicas e empresas têm explorado os benefícios desses ambientes, onde os participantes podem interagir uns com os outros, bem como, com o professor, mesmo que estejam a milhares de quilômetros de distância um do outro (Wankel e Kingsley, 2009). O surgimento e o uso de ambientes virtuais para desenvolver e para promover a aprendizagem na educação é um fenômeno novo, que está crescendo em ritmo acelerado (Hodge, Collins e Giordano, 2011).

Existem diversos mundos virtuais 3D, mas alguns podem ser destacados, como o *Open Wonderland*[13], *Second Life*[14] e *Open Simulator*[15]. O *Open Wonderland* é um conjunto de ferramentas de código aberto, desenvolvidas em Java, para criação de mundos virtuais. Essa iniciativa teve origem no Projeto *Wonderland* da *Sun*, mas atualmente é desenvolvido pela organização sem fins lucrativos *Open Wonderland Foundation*. O *Open Wonderland* ainda é

[13] http://www.openwonderland.org/

[14] http://www.secondlife.com/

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

uma tecnologia experimental que está em seus estágios iniciais de desenvolvimento, mas, que, no entanto, apresenta grande potencial e seus avanços devem ser acompanhados.

O *Second Life* (SL) é um mundo virtual 3D estável e muito difundido, desenvolvido pelo *Linden Lab*. O SL é utilizado por diversas instituições educacionais, tais como Universidade do Vale do Rio dos Sinos (Brasil), Universidade de Trás-os-Montes e Alto Douro (Portugal), *Boise State University* (Estados Unidos da America) e *University of St Andrews* (Reino Unido). Ainda, vários autores, entre eles Wankel e Kingsley (2009), Peachey *et al.* (2010) e Hodge, Collins e Giordano (2011), destacam o potencial educacional do SL. No entanto, não é uma plataforma de código aberto e apesar do ingresso no mundo virtual ser gratuito, o SL é baseado em uma economia própria, cuja moeda é o dólar *Linden*.

4.1 Open Simulator

O *Open Simulator* (OpenSim) é uma solução de código aberto desenvolvida em C#, muito semelhante ao SL, pois suporta o núcleo do protocolo de mensagens SL, *Linden Scripting Language* (LSL) e os visualizadores do SL. Pode ser instalado em sistemas operacionais baseados em Unix ou *Windows* juntamente com o *framework* Mono[16] e suporta os gerenciadores de bancos de dados SQLite[17] (padrão), MySQL[18] (plenamente) e Microsoft SQL[19] (parcialmente). O *software* é distribuído sob a Licença BSD (*Berkeley Software Distribution*). O OpenSim é desenvolvido e mantido pelos membros da comunidade e o código está no estado de maturidade alpha, portanto, ainda não possui uma versão estável.

Uma questão importante em relação aos mundos virtuais são os requisitos de *hardware* da máquina do usuário, mas esses requisitos não foram encontrados na documentação do OpenSim. No entanto, devido a sua semelhança com o SL, é possível tomar como base os requisitos de *hardware* do SL, que são descritos na tabela 2. Com relação aos requisitos de *hardware* da máquina servidora, também não existe uma documentação definindo requisitos mínimos, contudo, na *wiki* do projeto são compartilhadas configurações de diferentes servidores que rodam o OpenSim.

[16] http://www.mono-project.com/

[17] http://www.sqlite.org/

[18] http://www.mysql.com/

[19] http://www.microsoft.com/sql

Tabela 2 – Requisitos de *hardware* (Fonte: Hodge, Collins e Giordano, 2011)

	Requisitos Mínimos	**Requisitos Recomendados**
Processador	Pentium III ou Athlon de 800 MHz	1,5 GHz ou mais
Memória	512 MB	1 GB ou mais
Placa de Vídeo	NVIDIA GeForce 6600 ou ATI Radeon 8500, 9250	NVIDIA: 6800, 7600, 7800, 8800 ATI: 4850, 4870 ou melhor

O OpenSim possibilita, a partir do console do servidor, que todos os objetos criados, bem como os dados necessários para carregar o terreno, as texturas e os inventários sejam salvos em um arquivo de extensão *.oar*, de modo a permitir a restauração em outros servidores. Isso é importante para compartilhar o mundo desenvolvido com outros indivíduos ou organizações. Basicamente, o OpenSim pode ser configurado para rodar de duas formas diferentes: *standalone* (autônomo) e *grid* (grade). A configuração do OpenSim no modo *standalone* (Figura 3) é a mais simples. Nela o simulador de região e todos os serviços de dados são executados em um único processo quando o arquivo *OpenSim.exe* é executado. No modo *standalone*, é possível executar várias regiões em apenas uma única máquina.

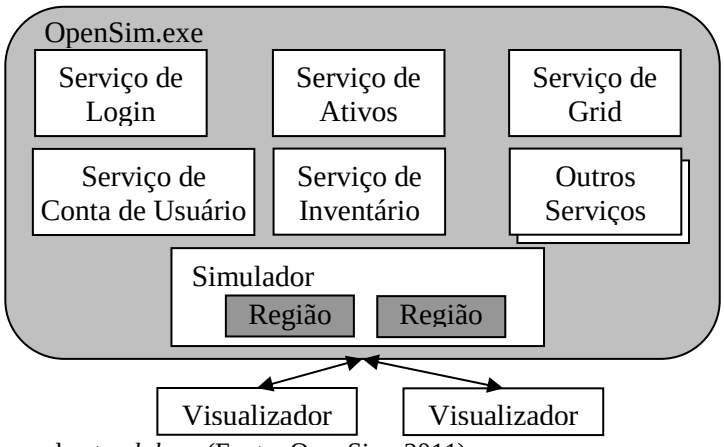


Figura 3 – OpenSim no modo *standalone* (Fonte: OpenSim, 2011)

Já no modo *grid* (Figura 4), os serviços de dados rodam em um arquivo executável chamado *Robust.exe*, enquanto que o simulador de região continua rodando no arquivo *OpenSim.exe*. Um *shell Robust* pode executar todos os serviços ou dividi-los entre várias instâncias *Robust*, o que permite que ele seja executado em máquinas separadas. No modo

grid, o *OpenSim.exe* atua apenas como o servidor de região, permitindo que uma ou mais regiões se comuniquem com os serviços de dados separadamente. Portanto, é possível executar vários simuladores de região em máquinas diferentes.

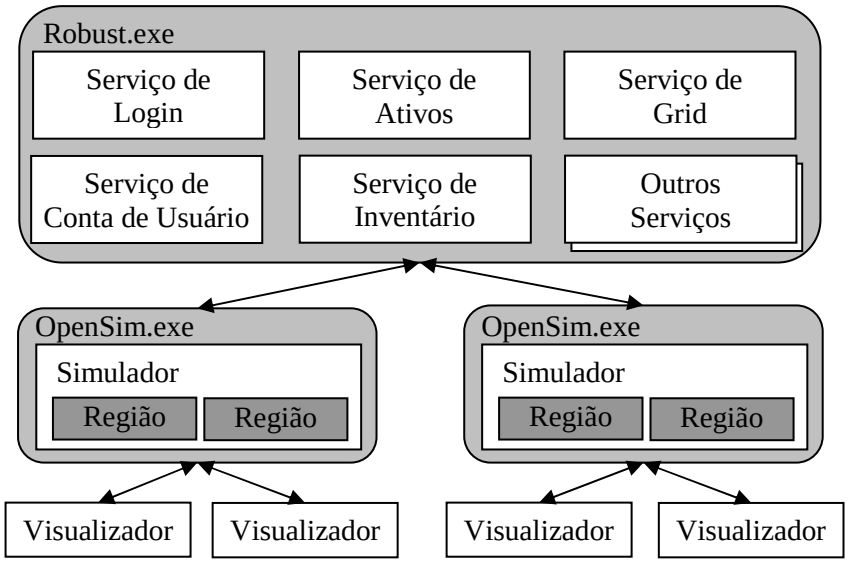


Figura 4 – OpenSim no modo *grid* (Fonte: OpenSim, 2011)

Além da possibilidade de *download* e instalação em uma máquina local ou um servidor, existem mundos virtuais baseados no OpenSim que podem ser utilizados gratuitamente, entre eles estão o OSGrid[20], Avination[21] e MyOpengrid[22]. A lista completa encontra-se no *website* do OpenSim.

4.1.1 Desenvolvimento e comunicação de dados

No momento da criação de uma região, um usuário (*avatar*) é atribuído como seu proprietário. Inicialmente, uma região corresponde a uma pequena ilha circular no ambiente 3D. Essa ilha pode ser expandida e objetos podem ser criados a partir de ferramentas de construção disponíveis nos visualizadores. Todo e qualquer objeto é construído através da

[20] http://www.osgrid.org/

[21] https://www.avination.com/

[22] http://www.myopengrid.com/

combinação e edição de primitivas, chamadas *prims*. São elas: cubo, prisma, cilindro, esfera, torus, tubo, anel e esculpido. A caixa de ferramenta de construção possibilita mover, girar, esticar, aplicar cor e textura e configurar os *prims*. O proprietário da região pode atribuir ou remover permissões de criação e edição de objetos.

Os objetos criados no OpenSim podem receber *scripts*, sendo que tais *scripts* podem ser criados em LSL, OSSL (*OpenSimulator Scripting Language*) e C#. Entre outras coisas, esses *scripts* podem ser usados para realizar a comunicação do OpenSim com o mundo exterior. Essa comunicação ocorre de forma semelhante ao *Second Life* (SL) e suporta os protocolos: URL (*Uniform Resource Locator*) para exibição de páginas *Web*; SMTP (*Simple Mail Transfer Protocol*) para o envio de *e-mails*; HTTP (*Hypertext Transfer Protocol*) para comunicação com páginas *Web*, por meio de requisições GET e POST; e XML-RPC (*Extensible Markup Language - Remote Procedure Call*) para comunicação com aplicações externas, usando o XML para codificação das chamadas e HTTP para transporte. Frequentemente a requisição HTTP é utilizada para armazenamento persistente de dados, sendo, então, os dados enviados ao servidor e recuperados quando necessário (Moore, Thome e Haigh, 2008). Já o XML-RPC permite que sistemas externos ao SL invoquem *scripts* de objetos dentro do mundo virtual (Valério, 2007), no entanto, exige a permanência de um canal de comunicação dedicado. Logo, o OpenSim pode ser integrado com sistemas externos por meio de diferentes protocolos. Na próxima seção, será abordado como integrar os mundos virtuais SL e OpenSim com o AVEA Moodle, por meio do módulo Sloodle.

4.2 Integração entre mundos virtuais e o Moodle

Os Ambientes Virtuais de Aprendizagem (AVEAs) podem ser definidos como mídias que utilizam o ciberespaço para veicular conteúdos e para permitir interação entre os atores do processo educativo (Pereira, 2007). Algumas das características básicas de um AVEA, destacadas por Gomes *et al*. (2010), são as seguintes: interatividade; controle das atividades; compatibilidade com as especificações existentes de conteúdos; sistema colaborativo de aprendizagem; e customização. Os AVEAs são alternativas que conduzem ao aprendizado, principalmente pela capacidade de proporcionar aos professores meios de acompanhar e assessorar constantemente o estudante, e, assim, poder entender o que é feito por ele, bem como propor desafios sobre o que está sendo estudado (Gomes *et al.*, 2010). Ainda, os

AVEAs podem proporcionar um suporte maior ao estudante no processo de ensino-aprendizagem devido à diversidade de ferramentas que podem ser utilizadas, de modo a adequar-se às necessidades e aos estilos cognitivos do estudante, juntamente com o constante apoio e acompanhamento do professor.

O Moodle (*Object-Oriented Dynamic Learning Environment*) é um AVEA de código aberto e possui uma comunidade muito ativa, que está sempre desenvolvendo novas ferramentas e trabalhando para melhorar o AVEA, sendo que a organização *Moodle Trust*[23] coordena esses esforços. O Moodle é um AVEA muito popular, adotado por inúmeras instituições, tanto no Brasil, quando no exterior. Ainda possui características que lhe permitem usabilidade tanto em grande escala, quanto em escala menor. Muitas instituições o utilizam como plataforma para realização de cursos à distância, enquanto outras usam como apoio a cursos presenciais (MOODLE, 2011).

Uma das formas de realizar a integração entre mundos virtuais, como o SL e o OpenSim, com o AVEA Moodle é por meio do módulo de código aberto Sloodle[24] (*Simulation Linked Object Oriented Dynamic Learning Environment*). O projeto Sloodle foi criado e é desenvolvido pela empresa Eduserv, e conta com o apoio da UWS (*University of the West of Scotland*) e da SJSU (*San José State University*). O Sloodle possui a maioria das ferramentas de atividades do AVEA Moodle, tais como: tarefas, fórum, c*hat, blog*, glossário e questionário. A atividade questionário é muito versátil, porque permite a combinação de diferentes tipos de questões, como múltipla-escolha, verdadeiro ou falso, numérica e resposta breve, sendo estas suportadas pelos objetos Sloodle *Quiz Chair* e *Pile On Quiz*. Os objetos Sloodle possuem *scripts* na linguagem LSL (*Linden Scripting Language*), que permitem a comunicação por meio da *Internet* com servidores *Web* externos via *e-mail*, XML-RPC e requisições HTTP. Já o Moodle é implementado com linguagem de programação PHP (*Hypertext Preprocessor*). A Figura 5 apresenta a arquitetura do Sloodle, que exibe como ocorre a integração do Moodle com o mundo virtual. Como se pode observar na figura 6, o Moodle é acessado por meio de um navegador *Web*, que se comunica com o servidor via HTTP; o servidor verifica se o usuário tem as permissões adequadas, e responde. O mesmo acontece quando um usuário do Sloodle interage com os dados do Moodle, ou seja, o objeto Sloodle, no mundo virtual, envia uma requisição HTTP, que é manipulada pelo módulo Sloodle no servidor do Moodle.

[23] http://www.moodle.org.br/

[24] http://www.sloodle.org/

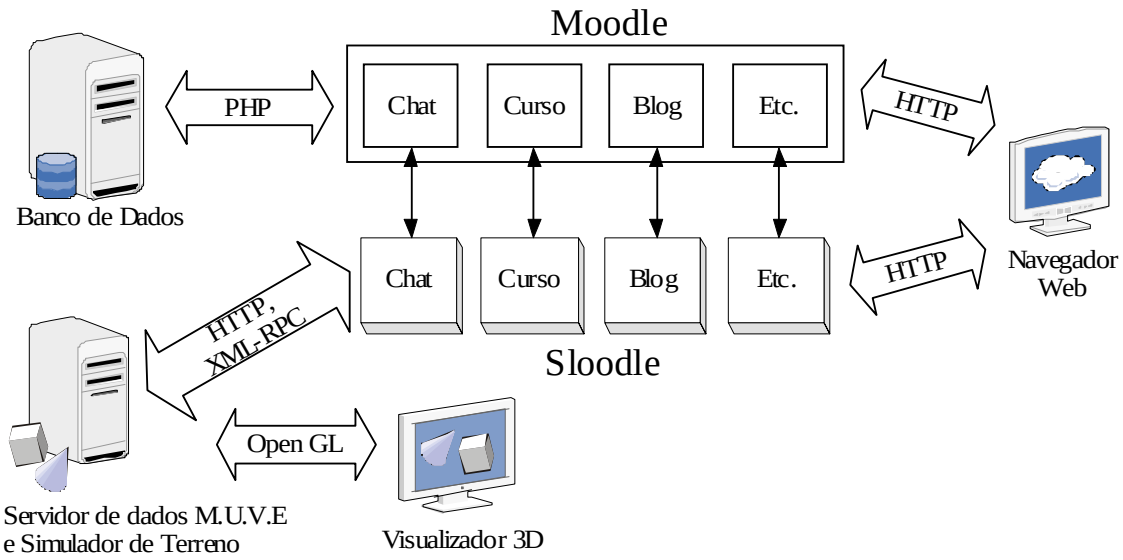


Figura 5 – Arquitetura do Sloodle (Fonte: Livingstone e Kemp, 2008)

4.3 Mundos virtuais e educação

Com as tecnologias educacionais tornando-se cada vez mais diversificadas, os professores são confrontados com novas possibilidades e, acompanhar essa evolução, pode ser um grande desafio. As expectativas dos estudantes também estão aumentando, colocando uma carga maior de responsabilidade sobre os professores para o desenvolvimento de uma gama de oportunidades de aprendizagem, incluindo *podcasts*, *wikis*, redes sociais e atividades em mundos virtuais (Wankel e Kingsley, 2009).

Aplicações de mundos virtuais são cada vez mais populares, especialmente para fins educacionais (Chen *et al*., 2010). Os mundos virtuais oferecem uma gama de possibilidades no contexto educacional, pois diferentes abordagens de ensino podem ser adotadas, como palestras, seminários, demonstrações, exibição de vídeos, simulações, performances virtuais, debates, *podcasts* e interações com *chatbots* (Savin-Baden, 2010). Os motivos para a adoção e para a manutenção de atividades educacionais em mundos virtuais, destacadas por Savin-Baden (2010), são as seguintes: (a) apóia a aprendizagem à distância, flexível e *blended learning;* (b) permite o aprendizado através da imersão; (c) promove a aprendizagem dialógica; (d) ofusca as relações de poder no aprendizado; (e) apóia a criatividade e a diversão em aprender; (f) induz a reconsideração da identidade na aprendizagem; e (g) encoraja a exploração da emoção na aprendizagem. Problemas e falhas comuns na utilização de mundos virtuais estão relacionados a não ter todos os alunos no ambiente ao mesmo tempo; a misturar

alunos experientes com novatos; a longas apresentações de *slides*; e a palestras com mais de 20 minutos, sem discussões (Savin-Baden, 2010).

Os mundos virtuais podem ser uma boa ferramenta na educação à distância (Ritzema e Harris, 2008; Wankel e Kingsley, 2009; Hodge, Collins e Giordano, 2011). Com os mundos virtuais, a educação à distância parece mais substancial e espera-se uma melhor aprendizagem devido à imersão (Savin-Baden, 2010) − além das diversas possibilidades educacionais já citadas, que podem favorecer maior interação e integração dos estudantes. Vários trabalhos destacam as possibilidades e as potencialidades do uso de mundos virtuais na educação superior. Livros, como os de Wankel e Kingsley e (2009), Savin-Baden (2010) e Vincenti e Braman (2011), abordam especificamente esse tema.

Wankel e Kingsley (2009) apresentam um estudo que mostra uma quantidade muito maior de usuários em mundos virtuais direcionados ao público infantil, como Habbo[25], Stardoll[26], entre outros, do que em mundos virtuais para o público jovem e adulto, como There[27], Second Life, entre outros. Isso indica que, apesar da popularização dos mundos virtuais entre o público adulto e jovem ser uma realidade hoje, é possível prever uma demanda ainda maior em um futuro próximo, pois o público infantil está crescendo utilizando os mundos virtuais.

A maioria dos livros e trabalhos analisados relacionados a mundos virtuais e educação, faz referência ao Second Life. No entanto, existe a restrição financeira para o desenvolvimento de espaços e aplicações nesse ambiente. Como já comentado na seção 4.1, existem diversos *grids* OpenSim que podem ser utilizados gratuitamente, mas, recentemente, surgiu a iniciativa *OpenSim Education Grid*[28] (OSEG), que reúne mais de vinte instituições de ensino para discutir a possibilidade do desenvolvimento, de forma colaborativa, de um *grid* OpenSim com o objetivo de melhorar o compartilhamento de informações técnicas, recursos, conteúdos e pesquisas. Com base em todas essas possibilidades, os mundos virtuais têm o potencial de revolucionar as formas de aprender, de ensinar e de conduzir pesquisas (Thomas e Brown, 2009).

[25] http://www.habbo.com/

[26] http://www.stardoll.com/

[27] http://www.there.com/

[28] http://osedugrid.wikispaces.com/

4.4 Relação entre mundos virtuais, jogos e simulações

Aldrich (2009) apresenta uma abordagem que relaciona e, simultaneamente, diferencia mundos virtuais, jogos e simulações. De acordo com Aldrich (2009), jogos são atividades divertidas e envolventes, geralmente usados exclusivamente para o entretenimento, mas que também permitem a exposição a ferramentas e a ideias. Em contrapartida, as simulações usam cenários rigorosamente estruturados e cuidadosamente projetados para desenvolver competências específicas. Já os mundos virtuais são ambientes sociais 3D, multiusuários, de fácil acesso e com recursos de construção. Mundos virtuais têm o ambiente tridimensional em comum com os jogos e as simulações. Entretanto, não têm o foco em um objetivo específico, como avançar para o próximo nível ou navegar com sucesso pelo cenário.

Nesse contexto, os mundos virtuais, jogos e simulações podem ser unificados como conceitos alinhados em ambientes virtuais altamente interativos. A Figura 6 representa esta ideia, em que os jogos acontecem em uma espécie de mundo virtual. Mesmo os jogos físicos são jogados em um mundo sintético estruturado por regras específicas, por mecanismos de *feedback* e por ferramentas de apoio. As simulações compartilham características-chave com os jogos, incluindo o uso de um mundo virtual e o foco em um objetivo particular, mas as simulações usam um conjunto mais refinado de regras, desafios e estratégias para guiar os jogadores no desenvolvimento de determinados comportamentos e habilidades. Aldrich (2009) ainda destaca que os jogadores frequentemente mudam entre os diferentes modos, passando da exploração do mundo virtual para os jogos e depois para uma simulação estruturada, conforme se tornam confortáveis no ambiente.

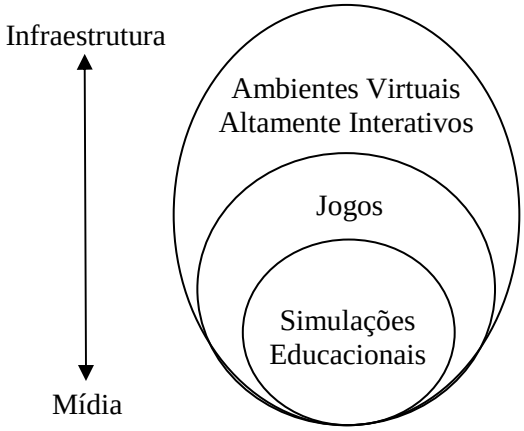


Figura 6 – Relação entre ambientes virtuais, jogos e simulações educacionais (Fonte: Aldrich, 2009)

4.5 Conclusões do capítulo

Os mundos virtuais 3D são ambientes que apresentam inúmeras possibilidades e proporcionam um tipo de interação diferente de qualquer outra na *Web*. O mundo virtual é uma tecnologia educacional inovadora e apresenta um grande potencial na educação, destacando-se, principalmente, no ensino à distância e no ensino superior, pois possibilitam diferentes atividades como, por exemplo, palestras, debates, jogos e simulações, além de favorecer a colaboração e interações sociais entre os estudantes. Existem diversos mundos virtuais (gratuitos e pagos), entre eles está o OpenSim, um mundo virtual 3D livre, que permite a construção de cenários, a customização de *avatares* e o desenvolvimento de aplicações por meio de *scripts*, que, por sua vez, permitem a comunicação com o mundo externo. É possível, ainda, integrar o OpenSim ao AVEA Moodle por meio do módulo Sloodle, que fornece ferramentas para integração dos dois ambientes.

Os mundos virtuais podem ser relacionados com jogos e com simulações educacionais, contudo, vão além deles. Na verdade, Os mundos virtuais abrangem os jogos e as simulações educacionais, pois são plataformas onde ambos se encontram. Essas plataformas são mais especializadas do que os jogos, já que, por serem um tipo de simulacro da realidade, possibilitam ao estudante aproximar-se de forma mais precisa de uma vivência prática. O que, de acordo, com aquilo que foi aferido por esta pesquisa, facilitaria a aprendizagem. No próximo capítulo, serão apresentados trabalhos encontrados na literatura, relacionados a jogos para Engenharia de *Software* e a aplicações em mundos virtuais.

5 TRABALHOS CORRELATOS

De modo a contemplar as duas características principais do trabalho, que são jogos para Engenharia de *Software*, mais especificamente Teste de *Software*, e mundos virtuais, os trabalhos correlatos foram divididos em dois grupos: jogos para Engenharia de *Software* e aplicações educacionais em mundos virtuais.

5.1 Jogos para Engenharia de Software

Como apresentado na seção 2.2.1, a Engenharia de *Software* é um tema recorrente nos currículos de graduação na área de computação, pois a disciplina de Engenharia de *Software* está presente em todos os currículos de referência analisados. Como também já comentado na seção 2.2, a abordagem de aprendizagem baseada em jogos pode ser agregada à estratégia de ensino desse conteúdo. Existem vários trabalhos na literatura sobre jogos digitais voltados para o ensino-aprendizagem de Engenharia de *Software*. A seguir, serão apresentados os mais recentes:

- SimulES-W (Monsalve, Werneck e Leite, 2011): é um jogo de cartas *Web* para ensino de Engenharia de *Software*, no qual os estudantes assumem diferentes papéis em um projeto de construção de *software*;
- SPARSE (Resende *et al*., 2010): é um jogo desenvolvido para auxiliar no ensino e na aprendizagem de Engenharia de *Software*, focando em modelos de processo;
- SimSE (Navarro e Hoek, 2009): simula processos de Engenharia de *Software* em diferentes metodologias de desenvolvimento;
- SE•RPG (Benitti e Molléri, 2008): é um jogo de RPG que proporciona a experiência de gerenciamento e de desenvolvimento de um projeto de *software*;
- XMED v1.0 (Wangenheim *et al*., 2009): aborda o assunto de Medição de *Software*;
- Elicit@ção (Vargas et. al, 2010): jogo em que o estudante assume o papel de analista e simula a elicitação de Requisitos de *Software*;

- A Ilha dos Requisitos (Gonçalves, Thiry e Zoucas, 2011): esse jogo tem como objetivo auxiliar no ensino dos principais conceitos relacionados à Engenharia de Requisitos de *Software*;
- InspectorX (Pötter e Schots, 2011): é um jogo para treinamento e aprendizado de Inspeção de *Software*;
- UTEST (Silva, 2010): um jogo para o ensino de teste de unidade com a aplicação de técnicas e de práticas relacionadas à seleção de dados de entrada; e
- Jogo das 7 Falhas (Diniz e Dazzi, 2011): é um jogo que aborda a técnica de teste de caixa preta.

A Tabela 3 apresenta as principais características dos jogos descritos, destacando a área de conhecimento de Engenharia de *Software* a qual o jogo aborda, a plataforma de execução (*on-line* / *desktop*), a tecnologia que foi desenvolvida, o número de jogadores, o gênero de jogo, o tipo de gráficos e se foi realizada avaliação junto a estudantes.

Tabela 3 – Síntese de características dos jogos para Engenharia de Software

Jogo	**Área de conhecimento de ES**	**Execução**	**Tecnologia**	**Nº de Jogadores**	**Gênero**	**Gráficos**	**Avaliação**
SimulES-W	Processos de Engenharia de Software	*on-line*	Java	*multi-player*	Cartas	2D	não
SPARSE	Processos de Engenharia de Software	*desktop*	Java	*single-player*	Simulador	2D	sim
SimSE	Processos de Engenharia de Software	*desktop*	Java	*single-player*	Simulador	2D	sim
SE•RPG	Gerenciamento de Projeto de Software	*desktop*	Adobe Flash	*single-player*	RPG	2D	sim
XMED v1.0	Medição de Software	*desktop*	Java	*single-player*	Simulador	2D	sim
Elicit@ção	Requisitos de Software	*desktop*	Adobe Flash	*single-player*	Simulador	2D	não
A Ilha dos Requisitos	Requisitos de Software	*on-line*	Adobe Flash	*single-player*	Aventura / Estratégia	2D	sim
InspectorX	Inspeção de Software	*on-line*	Adobe FLEX / C#	*single-player*	não identificado	2D	não
UTEST	Teste de Software	*desktop*	Adobe Flash	*single-player*	Simulador	2D	sim
Jogo das 7 Falhas	Teste de Software	*on-line*	Java	*single-player*	Simulador	2D	sim

5.2 Aplicações educacionais em mundos virtuais

Como discutido na seção 4.3, a maioria das iniciativas do uso de mundos virtuais na área educacional ocorre no Second Life (SL), que é um ambiente estável e com muitos recursos. Entretanto, é uma plataforma fechada e que exige um investimento financeiro para o desenvolvimento em suas terras virtuais. O OpenSim é uma opção ao SL, tendo em vista que é bastante semelhante tanto em características estruturais, quanto em recursos oferecidos, mas não possui ainda uma versão estável. Com o surgimento de iniciativas que adotem o OpenSim, é possível que outras instituições e organizações optem por migrar do SL. Como trabalhos correlatos, serão apresentadas quatro pesquisas. Uma delas utiliza o OpenSim e as outras três o SL.

O trabalho que utiliza o OpenSim é o de Schaf (2011), que propõe uma arquitetura para ambientes computacionais de suporte à colaboração para auxiliar o ensino e o treinamento em áreas multidisciplinares de engenharia de controle e de automação. Essa arquitetura engloba tecnologias e características de ambientes sociais, ambientes imersivos com visualização tridimensional, tutoriamento virtual autônomo baseado em coleta de informações de interação, auxílio à colaboração de usuários, experimentos com componentes intercambiáveis de realidade mista e outras funcionalidades.

Toro-Troconis *et al.* (2010), com apoio da Faculdade de Medicina do *Imperial College London*, propõe atividades de aprendizagem baseada em jogos, que apresentam pacientes virtuais, permitindo a condução de experiências, diagnósticos e atividades de *role-play learning* que apóiam diagnósticos, investigações e tratamento de pacientes.

Já Wang e Zhu (2009) propõem um jogo *multiplayer* e 3D para Engenharia de *Software*, baseado no SimSE (Navarro e Hoek, 2009), que tem como objetivo o ensino de conceitos relacionados a processos de Engenharia de *Software*. No entanto, esse jogo só suporta o Modelo de Prototipagem Rápida. No jogo, o estudante pode assumir diferentes papeis de Engenharia de *Software*. Os jogadores formam uma equipe de desenvolvimento de *software* e podem interagir uns com os outros. A equipe deve entregar o produto no prazo, e a pontuação da equipe é dada ao final do jogo. Para obter uma boa pontuação, os jogadores devem não apenas trabalhar em suas partes, mas também colaborar uns com os outros.

Sturgeon, Allison e Miller (2009) apresentam a versão para mundo virtual do laboratório virtual IEEE 802.11, chamado WiFiVL. O laboratório foi estendido para o Second Life com o objetivo de melhorar a interação com o usuário e dar suporte à aprendizagem

exploratória. Neste laboratório virtual, estudantes e professores podem criar um cenário selecionando nós 802.11 a partir de um repositório no mundo virtual, eles também podem criar conexões e realizar a configuração de *access points*. Eles ainda podem conversar uns com os outros usando texto ou áudio e podem se mover ao redor do cenário permitindo assim a visualização por diferentes perspectivas.

5.3 Conclusões do capítulo

Trabalhos recentes na área de Educação em Engenharia de *Software* indicam uma tendência ao desenvolvimento de jogos para apoio à estratégia de ensino-aprendizagem de diferentes áreas da Engenharia de *Software*. Na fase inicial de revisão bibliográfica deste trabalho, não foram encontradas referências específicas sobre jogos para Teste de *Software* − apenas surgiram publicações em agosto de 2010 e em novembro de 2011. Logo, a utilização de jogos para resolução desse problema é atual e essas publicações colaboram dando maior visibilidade ao tema. Os dois jogos encontrados na literatura que abordam o conteúdo de Teste de *Software*, U-Test (Silva, 2010) e Jogo das 7 Falhas (Diniz e Dazzi, 2011), são jogos 2D, *single player*, que abordam, respectivamente, teste de unidade e teste funcional. Apesar de contemplarem apenas uma pequena parte do conteúdo do tema Teste de *Software*, não deixam de ser interessantes, pois abordam em profundidade dois tipos de teste importantes. No entanto, nenhum desses jogos possibilita a edição do conteúdo do jogo, o que não atende ao problema de heterogeneidade dos currículos de graduação na área de computação.

Ao analisar os jogos para Engenharia de *Software* apresentados neste capítulo, observa-se que todos, com exceção do SimulES-W, são *single player*. Com relação aos gráficos, todos são 2D, no entanto, o SPARSE está em fase de desenvolvimento de uma interface 3D. A principal tecnologia de desenvolvimento utilizada nesses jogos é Java ou Adobe Flash, com exceção do InspectorX, que é desenvolvido com Adobe Flex e C#. Quanto ao gênero, a maioria é do tipo Simulador. Cabe destacar que a maioria dos jogos passou por processo de validação com estudantes e obteve resultados positivos. Quanto às interfaces dos jogos, observa-se uma grande distância da qualidade dos gráficos dos jogos comerciais de entretenimento.

Mundos virtuais 3D são possibilidades recentes e ainda pouco exploradas, mas são destacados por diferentes autores (Wankel e Kingsley, 2009; Savin-Baden, 2010; Vincenti e

Braman, 2011) como um grande potencial na educação superior. No entanto, ainda não existe uma abordagem educacional plenamente consolidada. Em relação às aplicações educacionais em mundos virtuais, encontra-se na literatura muito mais referências a trabalhos que utilizam o Second Life, possivelmente, pelo fato do OpenSim ser mais recente e ainda não possuir uma versão estável. No entanto, o trabalho de Schaf (2011) apresenta uma gama de possibilidades do OpenSim no contexto educacional. Quanto às aplicações no Second Life, o trabalho de Toro-Troconis *et. al* (2010) se assemelha a este por ser um jogo para o ensino superior e utiliza comunicação com uma aplicação *Web* para armazenamento de dados persistente. O jogo proposto por Wang e Zhu (2009) está diretamente relacionado a este trabalho, porém sua publicação foi em forma resumida e não foram encontradas outras referências. A implementação dos jogos de Toro-Troconis *et al.* (2010) e Wang e Zhu (2009) obtiveram resultados positivos com estudantes de graduação. O trabalho de Sturgeon, Allison e Miller (2009) é uma iniciativa da Escola de Ciência da Computação da Universidade de *St Andrews* e apresenta uma aplicação que aborda conceitos de Rede de Computadores para cursos de graduação e de pós-graduação.

No próximo capítulo, será apresentada a abordagem proposta neste trabalho para o ensino-aprendizagem de Teste de *Software* em curso de graduação na área de computação. Essa abordagem prevê o uso de um jogo sério denominado JETS (Jogo da Equipe de Teste de *Software*), que foi desenvolvido em um mundo virtual para apoio às aulas.

6 O JOGO DA EQUIPE DE TESTE DE SOFTWARE

O JETS – Jogo da Equipe de Teste de *Software* – é um jogo sério, desenvolvido em um mundo virtual que simula o setor de Teste de *Software* de uma empresa fictícia de desenvolvimento de *software*. Esse trabalho compartilha da ideia de Aldrich (2009), descrita na seção 4.4, em que ele propõe que mundos virtuais, Jogos e Simulações Educacionais convergem em uma estrutura chamada de Ambientes Virtuais Altamente Interativos. Nesse contexto, o JETS pode ser identificado como um jogo, mais especificamente, uma Simulação Educacional inserida em um mundo virtual. Neste capítulo, serão apresentados detalhes relacionados ao *Design* Instrucional, classificação, elementos cenários e implantação do jogo.

6.1 Design instrucional

Design Instrucional é o processo de identificar um problema de aprendizagem e projetar, implementar e avaliar uma solução para esse problema (Filatro, 2008). Considerando esse conceito, inicialmente, foi identificada uma carência no conhecimento relacionado a Teste de *Software* de profissionais da área de computação (Wangenheim e Silva, 2009; Astigarraga *et al.*, 2010). Uma possível solução, considerada uma tendência na área de educação em Engenharia de *Software* (discutido na introdução), é a adoção de jogos. Nesse contexto, o JETS tem como objetivo aprimorar o conhecimento de Estratégias de Teste de *Software* do estudante e fornecer uma visão geral de como o Teste de *Software* pode ser implementado em uma empresa, indo ao encontro das recomendações das Diretrizes Curriculares para Cursos de Graduação na Área de Computação (Brasil, 2003), citadas no capítulo 2. As recomendações e a respectiva relação com o jogo são apresentadas a seguir:

- *Mostrar as aplicações dos conteúdos teóricos*: o estudante aplica os conhecimentos, adquiridos em aula, na simulação de uma empresa e ainda pode exportar os casos de teste para ferramenta TestLink;
- *O professor atuar como um mediador*: no jogo, o professor assume o cargo de Gerente de Projetos e pode auxiliar os estudantes nos desafios;

- *Estimular a competição e a comunicação*: no jogo, existe um *ranking*, no qual aparecem os melhores resultados e os estudantes podem se comunicar via mensagem de texto ou de voz;
- *Trabalho em equipe*: os estudantes podem ajudar uns aos outros, podendo assim trabalhar de forma colaborativa;
- *Motivar os estudos e orientar o raciocínio*: como já comentado na Introdução e no capítulo de Jogos, vários autores indicam que o uso de jogos pode ser um fator motivacional;
- *Desenvolver as capacidades de comunicação e de negociação*: os estudantes podem auxiliar uns aos outros a resolver os desafios, logo, eles devem se comunicar e negociar com os colegas, propiciando um ambiente colaborativo melhor;
- *Desenvolver uma visão sistêmica para resolução de problemas*: a visão sistêmica consiste em examinar as mais diferentes soluções e escolher aquela que, em princípio, pode proporcionar uma solução ótima, com a maior eficiência e com um custo mínimo (Martinelli e Ventura, 2006), logo, a atividade de resolução dos desafios do jogo pode favorecer essa habilidade.

Cabe destacar que a proposta é de um jogo de apoio ao ensino-aprendizagem, pois, como destaca (CEEInf, 1999), a disciplina de Engenharia de *Software* deve fornecer uma base teórica consistente, portanto, o jogo não deve excluir a necessidade de um professor. O *Design* Instrucional dessa abordagem segue o modelo proposto por Kochanski (2009) e está representado na Tabela 4. Os elementos desse modelo são os seguintes: contexto, pré-condições, ementa, objetivos gerais de aprendizagem, conteúdos, objetivos gerais, estratégias de ensino, avaliação e referências. O *contexto* apresenta o público alvo e os objetivos a serem alcançados. As *pré-condições* definem os requisitos para aplicação da abordagem de ensino-aprendizagem. A *ementa* apresenta de forma sucinta os assuntos que serão abordados. Os *objetivos gerais de aprendizagem* descrevem os resultados pretendidos da instrução. Os *conteúdos* detalham os assuntos abordados, que podem ser organizados em unidades. Cada conteúdo deve ter um objetivo *de aprendizagem* e *de estratégia de ensino* específicos. A *avaliação* descreve de que forma a abordagem será aferida. Por fim, as *referências* apresentam a bibliografia utilizada na elaboração do *Design* Instrucional.

Tabela 4 – *Design* instrucional da abordagem de aprendizagem baseada em jogos

Contexto		
O público alvo são estudantes de cursos de graduação na área de computação. Esta abordagem visa proporcionar aos estudantes embasamento teórico e prático sobre os conceitos básicos de Teste de *Software*.		
Pré-Condições		
O estudante deve estar cursando ou ter cursado alguma disciplina que aborde o tema Teste de *Software*.		
Ementa		
Fundamentos de Teste de *Software*, casos de teste e Estratégia de Teste de *Software*.		
Objetivos Gerais e de Aprendizagem		
Identificar os principais conceitos de Teste de *Software*. Reconhecer e entender as técnicas de Teste de *Software*. Entender e aplicar as técnicas relacionadas à Estratégia de Teste de *Software*.		
Conteúdos	Objetivos de Aprendizagem	Estratégias de Ensino
Unidade1 – Fundamentos de Teste de *Software* Referências: [1] e [2]	Reconhecer os principais conceitos relacionados a Teste de *Software*	Aula expositiva.
Unidade 2 – Documentação de Teste de *Software* e desenvol-vimento de casos de teste Referências: [3] e [4]	Identificar e compreender a documentação do processo de Teste de *Software*.	Aula expositiva, trabalho em grupo e atividade prática com a ferramenta TestLink.
Unidade 3 – Aplicação de conceitos relacionados à Estratégia de Teste de *Software* Referências: [1], [2], [3] e [4]	Aplicar conceitos de execução e de desenvolvi-mento de casos de teste, de definição de ambientes e de ferramentas de teste; estratégia de teste.	Exercício prático por meio do JETS (Jogo da Equipe de Teste de *Software*).

Avaliação
A avaliação ocorrerá por meio de dois testes divididos em três segmentos: Reconhecimento: questões de múltipla escolha − em que o objetivo é verificar se o aluno é capaz de reconhecer os conceitos básicos de Teste de *Software*. Entendimento: questões de múltipla escolha − em que o objetivo é verificar se o aluno compreende os principais conceitos relacionados à Estratégia de Teste de *Software*. Aplicação: questões de múltipla escolha e discursivas − em que é verificada a capacidade do aluno de aplicação dos conceitos de Estratégia de Teste de *Software*.
Referências
[1] PRESSMAN, R. S. **Engenharia de *software***: uma abordagem profissional. Porto Alegre: AMGH, 7.ed. 2011. [2] SOMMERVILLE, I. **Engenharia de *software***. São Paulo: Pearson Addison Wesley, 8.ed. 2007. [3] INSTITUTE OF ELECTRICAL AND ELECTRONIC ENGINEERS COMPUTER SOCIETY. **SWEBOK**: Guide to the *Software* Engineering Body of Knowledge, 2004. [4] MYERS, G. J. et al. **The art of *software* testing**. New Jersey: John Wiley & Sons, 2.ed. 2004.

O *Design* Instrucional proposto (Tabela 4) é uma sugestão para os professores, pois essa abordagem visa a ser o mais flexível possível. O docente pode adaptá-la de acordo com o currículo do curso, com a ementa da disciplina e com o tempo disponível para trabalhar o tópico de Teste de *Software*. Esse *Design* Instrucional foi elaborado com base nos desafios inseridos no jogo e com o conteúdo trabalhado na turma em que foi feita a avaliação da abordagem. Portanto, os conteúdos, a avaliação, as referências bibliográficas, entre outros, podem variar de acordo com a proposta e planejamento didático do professor.

6.2 Classificações e elementos do jogo

Conforme descrito na seção 3.1.3, os jogos podem ser classificados de várias formas. Quanto ao gênero, o JETS pode ser classificado como um Simulador, tendo em vista que

simula atividades de uma Equipe de Teste de S*oftware*. Em relação aos gráficos, o jogo é classificado como 3D, pois foi desenvolvido em um mundo virtual 3D. Quanto ao número de jogadores ele é classificado como *Multiplayer*, pois pode ser jogado por um ou vários estudantes ao mesmo tempo, com a supervisão, ou não, do professor. Quanto ao tipo de interação, o JETS pode ser classificado como Múltiplos Jogadores Individuais *Versus* Jogo, pois cada jogador deve responder aos desafios do jogo. No entanto, o JETS também pode ser classificado como Jogo Cooperativo, pois os estudantes jogadores podem auxiliar seus colegas a atingirem os objetivos do jogo.

Como comentado na seção 3.1.1, os jogos são formados por elementos específicos, que podem ser classificados como formais ou como dramáticos. Os elementos formais do JETS são os seguintes:

- *Jogadores*: estudantes de graduação da área de computação;
- *Objetivos*: responder os desafios de forma correta, de modo a passar de fase até chegar ao último estágio do jogo;
- *Procedimentos*: os jogadores devem se dirigir à sala correspondente a cada fase, escolher uma mesa de trabalho e sentar na cadeira para responder os desafios;
- *Regras*: o jogo possui quatro fases, que correspondem a quatro cargos de uma equipe de teste: (1) Testador de *Software*, (2) Analista de Teste de *Software*, (3) Arquiteto de Teste de *Software* e (4) Líder de Equipe de Teste. Essas fases são sequenciais e para passar de fase, o jogador deve atingir uma determinada pontuação, que é obtida com base nos erros e nos acertos dos desafios;
- *Recursos*: na sala de reuniões, ficam disponíveis os recursos educacionais de aula e caso o professor esteja *online* no jogo, também pode ser considerado um recurso, no sentido de auxiliar o jogador;
- *Conflitos*: eles podem surgir na interação entre os jogadores; podem tanto auxiliar, quanto atrapalhar os participantes do jogo;
- *Limites*: o domínio do jogo se restringe a um prédio em uma ilha dentro de um mundo virtual e também a uma aplicação *Web*, na qual o usuário cria os casos de teste; ele pode, ainda, acessar a *Internet* para pesquisar outras fontes, de modo a auxiliá-lo na resolução dos desafios; e

- *Resultado*: o jogador ganha quando conclui com sucesso a fase de Líder da Equipe de Teste; ele pode comparar seus resultados em relação ao de outros jogadores, por meio de um *ranking*.

Os elementos dramáticos do JETS são os seguintes:

- *Desafio*: o jogador deve responder as questões de Estratégias de Teste de *Software* relacionadas a cada fase (cargo na equipe de teste);
- *Play*: os jogadores assumem o papel de funcionários de uma grande empresa de desenvolvimento de *software*, atuando e interagindo uns com os outros de modo a atingir o cargo de nível mais alto na Divisão de Teste de *Software*;
- *Premissa*: o jogador é um profissional apto a ingressar como Testador de *Software* em uma empresa de desenvolvimento de *software*;
- *Personagem*: o personagem é o próprio jogador que é contratado para Divisão de Teste de *Software* de uma empresa de desenvolvimento. O jogador pode customizar seu *avatar* de modo a criar uma maior identificação e envolvimento com o jogo; e
- *História*: a DevInfoData é uma empresa (fictícia) de desenvolvimento de *software* e está recrutando funcionários para sua Divisão de Teste de *Software*. Ao ingressar na empresa, o funcionário ocupa o cargo de Testador de *Software* e, conforme seu desempenho, é promovido até chegar ao cargo de Líder da Equipe de Teste.

6.3 Implementação

Na implementação do JETS foram utilizadas diferentes tecnologias, que podem ser visualizadas na arquitetura do mesmo, ilustrada na Figura 8. O servidor de mundos virtuais OpenSim é a plataforma principal do jogo, nele foram construídos os cenários e é onde rodam os *scripts* do JETS e do *Sloodle Quiz Chair* 1.0, que foi adaptado para atender as necessidades do jogo, como retornar a pontuação dos desafios e suas penalidades, bem como realizar o controle das fases. No entanto, para realizar este controle é necessária uma aplicação externa ao OpenSim, pois o mesmo não permite acesso direto ao banco de dados, onde as informações da partida são armazenadas. A aplicação *Web* do JETS (WebAppJETS) é responsável por

essa comunicação entre o mundo virtual e o banco de dados. Além da função de *middleware*, a WebAppJETS foi desenvolvida para permitir a criação de casos de teste tanto pelos professores, quanto pelos estudantes. Os professores acessam a WebAppJETS diretamente pelo *site* da aplicação, onde eles devem efetuar *login* para criar e editar casos de teste. Este *site* também contém informações do jogo, com descrição, contato e *download*. Já os estudantes acessam a aplicação por meio de um *link* presente nos desafios do jogo (dentro do mundo virtual) e têm a possibilidade de gerar um arquivo *.xml*, permitindo assim a exportação do caso de teste para ferramenta Testlink.

Os *scripts* do *Sloodle Quiz Chair* 1.0 são responsáveis pela comunicação com a atividade questionário no AVEA Moodle. Cada fase do jogo corresponde a um questionário e as perguntas do questionário, inseridas pelo professor, correspondem aos desafios que os estudantes devem responder no jogo. As perguntas podem ser do tipo múltipla escolha, verdadeiro ou falso, numérica e resposta breve. Na configuração da pergunta é definida a pontuação de acordo com o grau de dificuldade do desafio, sendo que também pode ser definido um fator de penalidade (de 0 a 100%), ou seja, a cada tentativa incorreta, o estudante perde essa porcentagem na pontuação do desafio. As perguntas do tipo múltipla escolha podem ter inúmeras opções de escolha e em cada opção é possível definir um percentual da nota da pergunta, isto é, uma opção com uma resposta incompleta pode representar um percentual menor que 100% e uma opção de resposta incorreta pode representar, inclusive, um percentual negativo. Nas configurações do questionário, o professor pode definir o número de tentativas, desde apenas uma tentativa até um número ilimitado. Nessa aplicação, os questionários foram configurados para permitir um número ilimitado de tentativas. No entanto, as perguntas foram configuradas com um fator de penalidade de 25%, de modo que após a quarta tentativa incorreta a pontuação recebida no desafio é zero.

A WebAppJETS foi desenvolvida com base no *framework* de código aberto CodeIgniter[29], que foi escolhido por ser um *framework* para PHP, que utiliza o conceito de arquitetura de *software* MVC (*Model, View, Controller*) (Fowler *et al.*, 2006); ou seja, separa o modelo de dados, a interface de usuário e o controle das regras de negócio. Ainda, o CodeIgniter apresenta uma boa documentação e uma comunidade ativa de desenvolvedores. O banco de dados utilizado foi o MySQL, sendo escolhido por sua estabilidade e desempenho e também por ser uma ferramenta de código aberto.

29 http://codeigniter.com/

A opção de desenvolvimento do jogo em um mundo virtual ocorreu devido à sua flexibilidade em relação a *engines*, pois o OpenSim possibilita a criação dinâmica do ambiente virtual, enquanto que utilizando *engines*, o ambiente deve ser totalmente criado previamente. A opção pelo mundo virtual OpenSim foi baseada no fato de ele ser uma opção de *software* de código aberto, grátis e estruturalmente semelhante ao Second Life. O jogo foi implementado utilizando a linguagem de *scripts* LSL. Essa linguagem de *script* foi escolhida devido a sua ampla documentação e também por ser a linguagem utilizada nos objetos Sloodle.

Os cenários do jogo no mundo virtual foram construídos, em sua maioria, com base em objetos prontos (criados e disponibilizados por membros da comunidade do OpenSim), como prédio, mesas, cadeiras e computadores, agilizando assim o desenvolvimento. Os *scripts* do *Sloodle Quiz Chair* 1.0, adaptados ao JETS, foram inseridos nas cadeiras das mesas de trabalho. Esses *scripts* são executados quando o *avatar* do jogador senta nesse objeto. O *script* responsável pelo início da partida foi inserido em monitor no *hall* de entrada do prédio, sendo executado pelo toque do *avatar*. O *script* de *ranking* do jogo foi inserido em um totem no *hall* de entrada, também executado após o toque do jogador. Foram criados ainda diversos pontos de teletransporte com o objetivo de facilitar a movimentação dos jogadores pelos cenários.

O jogo foi desenvolvido para suportar uma turma de até 40 estudantes jogando simultaneamente, mas é possível a replicação do ambiente de jogo em várias ilhas. Neste trabalho foram realizados testes com apenas uma ilha. A Figura 7 apresenta o diagrama UML de Máquina de Estados do JETS, que descreve o fluxo do jogo no mundo virtual.

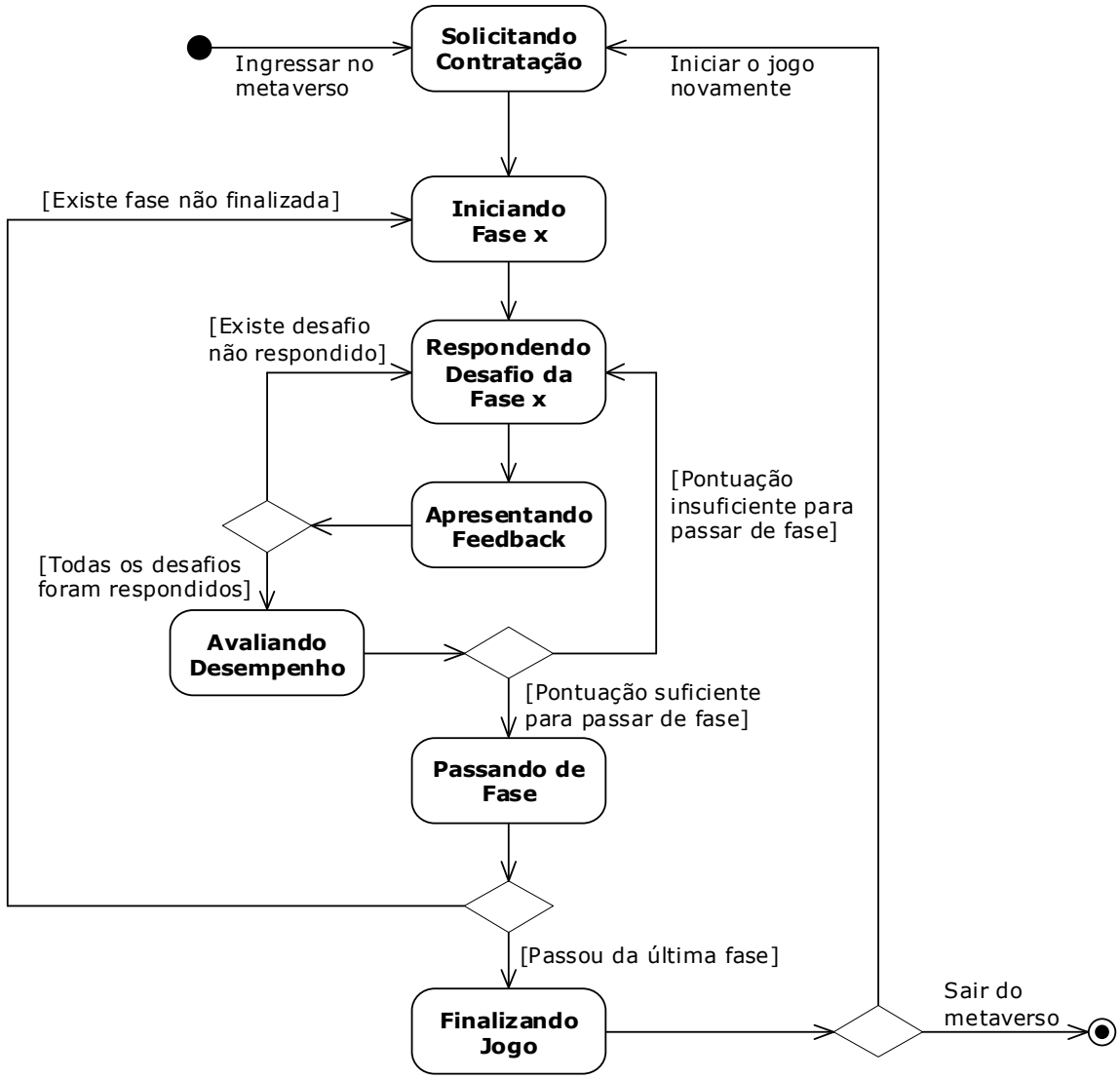


Figura 7 – Diagrama UML de Máquina de Estados do JETS

A arquitetura do JETS (Figura 8) consiste basicamente em uma ou em várias máquinas com um *software* cliente (visualizador de mundos virtuais) OpenSim. Esse *software* cliente acessa o servidor OpenSim, no qual os *scripts* são executados. Entre outras coisas, esses *scripts* fazem requisições HTTP ao servidor *Web* para solicitar informações armazenadas, então, a WebAppJETS faz o acesso ao banco de dados MySQL e retorna a resposta para o servidor OpenSim. A comunicação do OpenSim com o Moodle ocorre por meio do módulo Sloodle (instalado no Moodle) e de objetos Sloodle no OpenSim, que também utilizam requisições HTTP. A WebAppJETS é acessada por meio de um navegador *Web*, os casos de teste criados pelos estudantes podem ser importados no formato *.xml* na ferramenta TestLink, que também é acessada via navegador *Web*. Cabe destacar que todos os serviços citados

podem rodar em um único servidor, que foi o caso desta implementação, mas, dependendo da demanda de acessos, é recomendada a distribuição desses serviços.

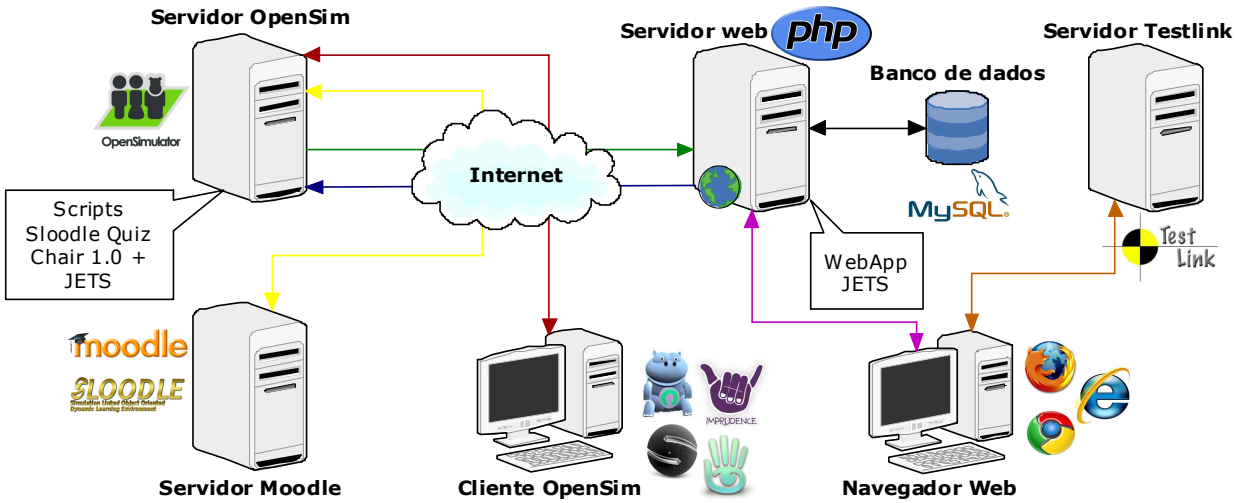


Figura 8 – Arquitetura do JETS

6.4 Especificações do jogo

Por ser um jogo *online*, é necessário um servidor com Sistema Operacional Linux, Apache[30], PHP5[31], Mono, MySQL e OpenSim. É possível utilizar o Sistema Operacional Windows, porém não foram realizados testes nessa plataforma. O visualizador de mundos virtuais utilizado no desenvolvimento e na aplicação do jogo foi o Imprudence, mas o jogo também suporta outros visualizadores, incluindo Hippo Viewer, Singularity e Second Life Viewer. As especificações de *hardware* são descritas na seção 4.1.

O JETS está sob a licença Creative Commons[32], que permite o compartilhamento e a criação de obras derivadas, mas desde que seja para uso não comercial e seja feita a devida atribuição dos créditos. O JETS foi empacotado utilizando o padrão SCORM[33] (*Sharable Content Object Reference Model*) com o objetivo de facilitar o compartilhamento e

[30] http://www.apache.org/

[31] http://php.net/

[32] http://creativecommons.org/licenses/by-nc/3.0/deed.pt_BR

[33] http://www.adlnet.gov/capabilities/scorm

distribuição do jogo. O SCORM é um modelo de referência para conteúdos e para serviços de *e-learning* que integra especificações, normas técnicas e guias para atender aos seguintes requisitos: acessabilidade, reusabilidade, interoperabilidade e durabilidade (ADL, 2011). No pacote do JETS, está incluído o mundo virtual no formato de arquivo *.oar*, o *backup* da disciplina do Moodle no formato *.zip*, uma pasta compactada com os arquivos PHP da aplicação *Web*, os *scripts* para criação do banco de dados MySQL e um arquivo no formato *.txt* com as informações de configuração do jogo. É fato que a preparação do ambiente do jogo não é uma tarefa simples para uma pessoa leiga em tecnologia. Entretanto, é necessário destacar que o público que irá aplicar o jogo é formado, provavelmente, por professores da área de computação, logo, não deverão ter maiores problemas na implantação desse ambiente.

6.5 Descrição e cenários do jogo

Ao entrar no mundo virtual, o jogador é materializado em frente ao prédio da Divisão de Teste de *Software* da empresa DevInfoData (Figura 9). Ao ingressar no prédio, o jogador se depara com um monitor em um balcão no *hall* de entrada (Figura 10). Para iniciar uma partida, o jogador deve clicar na tela do monitor para ser contratado, permitindo, assim, que as informações da partida sejam armazenadas em um banco de dados externo. Após a contratação, o jogador recebe um *notecard* com informações para novos funcionários (instruções do jogo) e ingressa na equipe de teste no cargo de *Testador de Software* (a primeira fase do jogo). Então, ele deve ser dirigir à sala dos testadores e responder aos desafios (Figura 11).

Na fase de *Testador de Software*, o estudante deve responder a perguntas relacionadas a conhecimentos básicos de teste de *software* e procedimentos relacionados ao cargo de testador de *software*. A cada conclusão de desafio, o estudante recebe um *feedback*, caso ele tenha acertado, o sistema retorna a pontuação obtida, caso contrário, o sistema informa a penalidade aplicada aquele desafio. Após responder a todos os desafios da fase, o estudante recebe o *feedback*, caso ele tenha acertado todos os desafios, ele é promovido, caso contrário, ele permanece na fase e deve responder novamente todos os desafios, sendo aplicada as penalidades nos desafios que, anteriormente, foram respondidos de forma incorreta.

Nesta aplicação, foram criadas perguntas de múltipla escolha, sendo que em uma delas o estudante deve executar um caso de teste pronto, criado na WebAppJETS. Nesse caso de

teste o estudante deve executar um teste funcional em um *website* criado exclusivamente para esta atividade, no qual foi inserido um erro. Cabe salientar, conforme descrito na seção 6.1, que esta abordagem foi desenvolvida para adaptar-se a diferentes currículos, de modo que o professor deve modificar ou criar novos desafios de acordo com o seu plano de ensino.

Figura 9 – Cenário do prédio da divisão de Teste de *Software* da DevInfoData

Figura 10 – Cenário do *hall* de entrada do prédio

Figura 11 – Cenário da fase de Testador de *Software*

Após a conclusão da fase de *Testador de Software*, o jogador é promovido a *Analista de Teste de Software* − segunda fase do jogo (Figura 12). Nessa fase o estudante deve elaborar casos de teste conforme as especificações definidas pelo professor. Para isso, o estudante acessa a WebAppJETS (Figura 13) na qual ele deve criar o caso de teste. Após a criação é exibido um *link*, que deve ser copiado e colado na barra de conversa do mundo virtual para responder à pergunta, que é do tipo resposta breve. Cabe destacar que o jogo apenas valida se a resposta foi inserida no formato correto, posteriormente, o professor deve analisar os casos de teste criados pelos estudantes. O estudante também tem a possibilidade de gerar um arquivo com a extensão *.xml* para uma futura exportação para a ferramenta Testlink, como mostrado na Figura 14.

Figura 12 – Cenário da fase de Analista de Teste de *Software*

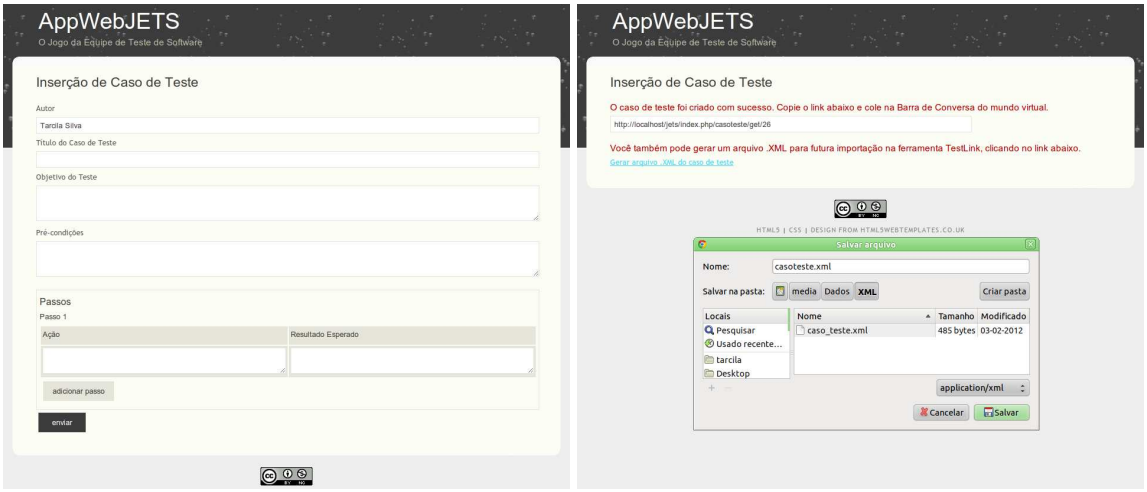


Figura 13 – Interfaces de criação e *download* dos casos de teste da WebAppJETS

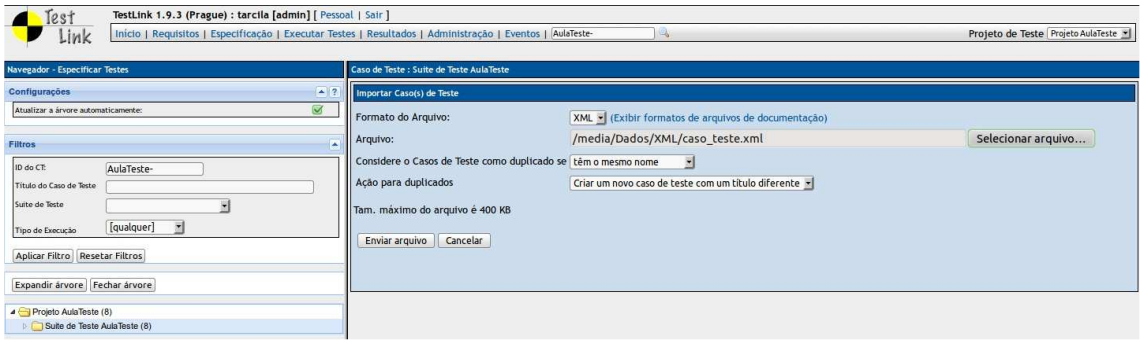


Figura 14 – Interface da ferramenta TestLink para importação dos casos de teste

Após concluir a fase de analista, o jogador é promovido a *Arquiteto de Teste de Software* e deve tomar decisões relacionadas ao ambiente de teste e às ferramentas utilizadas em um determinado projeto − terceira fase do jogo (Figura 15). Nesta aplicação foram elaboradas perguntas de múltipla escolha, com cinco opções de resposta cada, tendo apenas uma opção correta.

Figura 15 – Cenário da fase de Arquiteto de Teste de *Software*

Na quarta e última fase do jogo (Figura 16), o jogador é promovido a *Líder da Equipe de Teste* e deve tomar decisões relacionadas ao cronograma, ao orçamento e à alocação de recursos para os testes de um determinado projeto. Ao final do jogo o estudante terá passado por diversos desafios e visualizado diferentes conteúdos relacionados a teste de *software* de forma lúdica.

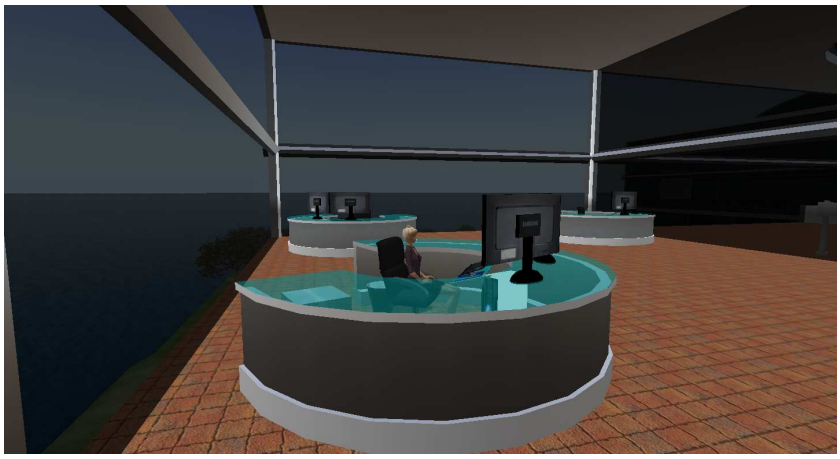


Figura 16 – Cenário da fase de Líder de Equipe de Teste

O objetivo do jogo é progredir na empresa, passando por todos os cargos da equipe de teste, até chegar à líder da equipe. Como já relatado, os desafios, a pontuação dos mesmos, bem como o fator de penalidade, são definidos pelo professor por meio da atividade questionário no AVEA Moodle (Figura 17). Cada fase corresponde a um questionário e o professor pode acompanhar os resultados de cada tentativa do estudante. Logo, o professor

pode analisar o desempenho do estudante em relação aos erros e aos acertos, ao número de tentativas e também quanto ao tempo que ele levou para concluir a fase.

Figura 17 – Interfaces da disciplina e dos resultados do questionário no AVEA Moodle

6.6 Conclusões do capítulo

O jogo proposto, denominado Jogo da Equipe de Teste de *Software* (JETS), visa apoiar a relação de ensino-aprendizagem de Teste de *Software* em cursos de graduação em computação e áreas afins. Mais especificamente, nesse jogo sério, desenvolvido em um mundo virtual, é abordado o tópico Estratégias de Teste de *Software*. O JETS é um jogo de simulação, 3D e *multiplayer*. Apenas essas características já o diferenciam dos trabalhos correlatos relacionados a jogos para Engenharia de *Software*, já que a grande maioria dos jogos analisados é 2D e *single player*. No entanto, o JETS vai além e permite a customização do jogo pelos professores, inserindo questões, de acordo com o conteúdo abordado em aula e o acompanhamento do desempenho dos estudantes no jogo pelo AVEA Moodle. Outro diferencial importante é a possibilidade de exportação de casos de teste para a ferramenta TestLink, de modo a permitir ao estudante experienciar uma ferramenta real da área de Teste de *Software*.

Apesar do OpenSim ainda não possuir uma versão estável e de não ter uma boa documentação, demonstra ser uma boa opção de mundo virtual. No decorrer do desenvolvimento do JETS, não foi identificada nenhuma anomalia ou *bug* relacionado ao OpenSim. Em relação à pouca documentação, essa deficiência foi suprida pela documentação do Second Life. A maior dificuldade encontrada foi a integração de diversas tecnologias distintas. O ponto fraco no desenvolvimento de aplicações no OpenSim é a ausência de

componentes padrões de GUI (*Graphical User Interface*), como botões, caixas de texto etc. A integração com aplicações externas é possível, mas não é simples. Isso é um ponto negativo, tendo em vista o contexto atual de convergência de tecnologias (Kurbanoglu *et al.*, 2010). Logo, para realizar a integração foi criada uma aplicação *Web* (WebAppJETS), que é responsável por armazenar os dados das partidas, fazer o controle das fases e do *ranking* do jogo. A WebAppJETS também oferece uma interface para a criação e edição de casos de teste e a exportação dos casos para o formato *.xml*. Essa aplicação disponibiliza, ainda, um *website* para divulgação do jogo, com descrição, contato e *download*.

7 AVALIAÇÃO E ANÁLISE DE RESULTADOS

Este capítulo apresenta a avaliação do JETS, descrevendo em detalhes a preparação e a aplicação do experimento, bem como o *framework* utilizado como referência para sua realização. Ao final, é apresentada a análise dos resultados.

7.1 Preparação do Experimento

A avaliação segue o *framework* proposto por Kochanski (2009), para apoiar a construção de experimentos na avaliação empírica de jogos educacionais na área de Engenharia de *Software*. Esse *framework* é dividido em cinco partes: (I) definição do experimento, (II) planejamento do experimento, (III) operação do experimento, (IV) análise e interpretação dos dados e (V) apresentação dos resultados. Na parte I do *framework,* são apresentados os elementos que devem ser observados no momento da definição do experimento, que são os seguintes:

- *Estratégia da pesquisa*: qualitativa, no sentido de analisar as percepções do jogo pelos estudantes e quantitativa, no sentido de medir a aprendizagem;
- *Forma de realização*: *in vivo*, pois é realizado no ambiente real;
- *Abordagem de pesquisa*: analítica, pois o propósito da pesquisa é ter um entendimento inicial sobre determinada questão ou fenômeno, no caso, a aprendizagem e as percepções do jogo;
- *Estratégia para seleção de grupos*: aleatória, não havendo nenhuma distinção entre os estudantes;
- *Questionários*: apenas o questionário de percepções do jogo;
- *Pré*-condições: os estudantes devem ter cursado ou estar cursando uma disciplina de graduação que aborda o tema de Teste de *Software*; e
- *Design Instrucional*: descrito na seção 6.2.

Na parte II do *framework,* são descritos os elementos necessários para o planejamento do experimento, sendo esses apresentados a seguir:

- *Seleção do contexto*: o experimento deve ocorrer na sala de aula (aulas expositivas, exercícios, pré-teste e pós-teste) e no laboratório de informática (utilização do jogo);
- *Definição de hipóteses*: (H_0) define que o grupo experimental **não** apresenta aprendizagem superior ao grupo de controle; (H_1) define que o grupo experimental apresenta aprendizagem superior ao grupo de controle; e (H_2) define que a aplicação do jogo sério JETS torna o processo de ensino-aprendizagem mais atrativo que o considerado tradicional;
- *Variáveis de controle*: variável (1) está relacionada à pontuação dos participantes do pré-teste e do pós-teste e a variável (2) está relacionada às respostas do questionário de percepções do jogo;
- *Seleção dos Participantes*: aleatória, sendo divididos em dois grupos: A (experimental) e B (controle);
- *Design de Experimento Utilizado*: descrito no diagrama representado na Figura 18; e
- *Planejamento da Instrumentalização*:
  - *Tratamento a ser realizado*: realização de aulas sobre o conteúdo de Teste de *Software*; aplicação do pré-teste; divisão dos grupos; aplicação da lista de exercício para o grupo de controle; aplicação do jogo para o grupo experimental e do questionário de percepções; e aplicação do pós-teste;
  - *Objetos / Equipamentos*: pré-teste, pós-teste, questionário de percepções do jogo, *data show* e computadores com acesso a *Internet* e com o *software* Imprudence instalado;
  - *Diretrizes*: o experimento deve seguir o *Design* Instrucional (seção 6.1); e
  - *Instrumentos de medição*: pré-teste, pós-teste e questionário de percepções do jogo.

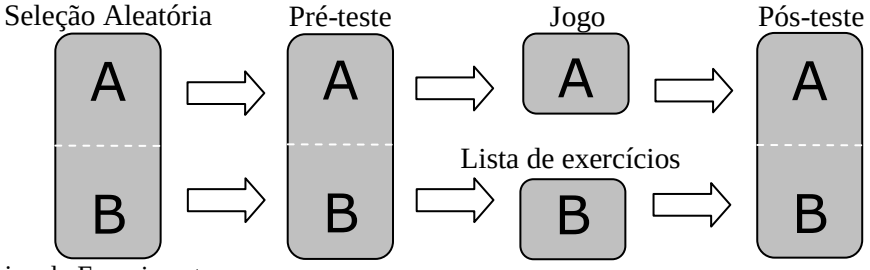


Figura 18 – *Design* do Experimento

A Parte III do *framework* será abordada na próxima seção (seção 7.2) e as Partes IV e V referentes, respectivamente, a análise e a interpretação dos dados, assim como a apresentação dos resultados, são abordadas na seção 7.3.

7.2 Aplicação do experimento

O ambiente do jogo foi preparado em um servidor da Universidade Federal de Santa Maria (UFSM) com a seguinte configuração: sistema operacional Linux Ubuntu, processador Intel Xeon X2460 2.80 GHz (Gigahertz) e memória de 16 GB (Gigabyte). A aplicação do JETS foi realizada em uma turma do curso de Engenharia da Computação da UFSM, na Disciplina de Engenharia de *Software*, no segundo semestre de 2011. A disciplina de Engenharia de *Software* é ofertada no quinto semestre do curso e é a única no currículo que aborda conceitos relacionados à área de Engenharia de *Software*. A turma era constituída de dezessete estudantes, tendo dois desistentes. Logo, o grupo considerado foi de quinze estudantes. O experimento foi realizado no mês de novembro de 2011. A Unidade 1, descrita no *Design* Instrucional (seção 6.1), foi abordada no dia 8, em dois períodos de aula de 50 minutos cada. Já a Unidade 2 foi trabalhada nos dias 16 e 22, em aulas de dois períodos cada. O pré-teste foi aplicado no dia 23 em um período de aula e no outro os estudantes tiveram aula de outro tópico da disciplina de Engenharia de *Software*; a aplicação do jogo foi realizada dia 29; e, por fim, o pós-teste foi aplicado no dia 30, também em um tempo de aula, após a abordagem de outro tópico de Engenharia de *Software*.

Até o início do experimento, o tema de Teste de *Software* não havia sido abordado na disciplina. Na primeira aula, por ser unicamente expositiva, foi observado um pouco de dispersão por parte dos estudantes, em alguns momentos. A aula do dia 16 foi dividida em um momento expositivo e um outro momento em que os estudantes realizaram um trabalho em grupo. Já na aula do dia 22, foi realizada uma atividade prática com a ferramenta TestLink, porém os estudantes apresentaram dificuldades no domínio da ferramenta. Apesar da documentação de Teste de *Software* ter sido abordada na aula expositiva e do manual da ferramenta também ter sido disponibilizado, vários estudantes tiveram dificuldade de entender e de realizar a vinculação dos documentos na ferramenta.

Como previsto na preparação do experimento, a turma foi dividida em dois grupos, seguindo a ordem alfabética do diário de classe da disciplina, fazendo da primeira metade da turma o grupo de controle e da segunda metade o grupo experimental. Com relação ao pré-teste, os resultados dos grupos de controle e experimental foram razoáveis tendo uma média de, respectivamente, 6,6 e 8. No dia 29, o grupo de controle realizou uma atividade tradicional, que foi uma lista de exercícios disponibilizada no ambiente Moodle. O grupo experimental realizou a atividade alternativa, que foi jogar o JETS no laboratório de informática, sendo que, os primeiros 30 minutos, foram destinados a orientações sobre o mundo virtual e o Jogo, e 60 minutos para os estudantes jogarem. Ao final da atividade, eles responderam um questionário de percepções do jogo.

No dia da aplicação do pós-teste, vários estudantes faltaram, o que interferiu na média dos grupos. Ainda, muitos dos estudantes que realizaram o pós-teste tiveram rendimento inferior ao pré-teste. As médias dos grupos de controle e experimental baixaram, respectivamente, para 4,8 e 7,1. Os resultados foram bastante controversos, alguns estudantes tiveram melhor desempenho, outros um pouco abaixo do pré-teste e alguns tiveram notas bem abaixo do que haviam obtido anteriormente. Esses resultados podem estar relacionados ao fato do pós-teste ter sido realizado vinte e dois dias após a aula expositiva, na qual a maior parte do conteúdo foi apresentada, causando um distanciamento entre a discussão dos conteúdos e o momento da avaliação. Outra possibilidade, seria o desinteresse por parte dos estudantes, motivado por não ser uma atividade que valia nota, pois alguns estudantes tendem a valorizar mais a nota do que a aprendizagem em si (Oliveira e Santos, 2005). Ainda, o fato de ser final de semestre e de estarem realizando provas de outras disciplinas, pode ter interferido no foco em relação ao conteúdo de Teste de *Software*.

No entanto, apesar dos resultados não tão satisfatórios do pós-teste, na aplicação do jogo, foi observado grande interesse por parte do grupo. Inicialmente, foi exibido um vídeo do jogo e, em seguida, foi explicado o funcionamento da ferramenta visualizadora de mundos virtuais Imprudence. Os estudantes foram instruídos sobre as formas de visualização e sobre como andar, voar, se comunicar e editar seu *avatar* no OpenSim. No início, os estudantes se dispersaram um pouco (e se divertiram bastante) com as diversas possibilidades de edição do *avatar*, a possibilidade de voar e de interagir com os colegas de uma maneira diferente daquela que ocorre no dia a dia. Mas, após algum tempo, os estudantes se concentraram nas atividades do jogo. Algumas dúvidas sobre os desafios surgiram, mas os estudantes preferiram fazer as perguntas diretamente à professora, apesar de estar *online* no jogo, sendo que a maior parte da comunicação com os colegas também ocorreu fora do mundo virtual.

Isso pode ser explicado pelo fato do jogo ter sido aplicado em um laboratório, onde os estudantes e o professor estavam praticamente lado a lado. Os casos de teste criados pelos estudantes na fase de Analista de Teste de *Software* foram salvos com a extensão *.xml*, mas, devido à limitação de tempo, eles foram orientados a não realizar a exportação para ferramenta TestLink naquele momento. Contudo, foram informados de que poderiam, posteriormente, realizar essa exportação. Em geral, os estudantes demonstraram bastante interesse, motivação e envolvimento com o jogo. Alguns estudantes perguntaram sobre a disponibilização do jogo. Inclusive, um dos estudantes comentou que gostaria de jogá-lo nas férias. Uma estudante que era do grupo de controle também perguntou se poderia jogar também. Esse interesse demonstrado pela estudante do grupo de controle, indica que os estudantes que utilizaram o jogo responderam de forma positiva ao mesmo, comentando com os demais sobre a experiência que realizaram. Então, com base nesses relatos, as expectativas da experiência foram excedidas, pois os estudantes mostraram interesse em jogar o JETS também fora do contexto de aula.

7.3 Análise dos resultados

Todo o processo de aplicação do experimento foi realizado em um período de vinte e dois dias, pois, apesar da disciplina ser ministrada duas vezes na semana, em um dos dias de aula foi feriado e também outros tópicos de Engenharia de *Software* foram abordados nesse período. Na aula destinada ao trabalho com a ferramenta TestLink, alguns estudantes tiveram mais dificuldade na vinculação dos documentos. Por ser uma ferramenta profissional, não é possível adquirir pleno domínio da mesma em uma única aula. Entretanto, a criação de tutoriais pode auxiliar aos estudantes na utilização da ferramenta. Como já comentado na seção 2.4, o TestLink possui inúmeras funcionalidades de gerenciamento de processo de teste e contempla diversos documentos. No JETS, apenas é trabalhado o documento Caso de Teste, que pode ser exportado para a ferramenta TestLink, de modo que isso pode incentivar os estudantes a explorarem mais a ferramenta, potencializando seus conhecimentos relacionados a Teste de *Software* e proporcionando uma maior experiência com uma ferramenta profissional. Por ser um *software* de código aberto, os professores podem adaptar o JETS para suportar outras ferramentas relacionadas a Teste de *Software*, de forma a familiarizar melhor os estudantes com esses tipos de ferramentas.

Com relação aos testes realizados para avaliar a aprendizagem dos estudantes, os resultados do pré-teste foram regulares, já os resultados obtidos no pós-teste foram muito controversos. Em função da ausência de vários participantes do experimento e dos resultados bastante heterogêneos dos estudantes, a média do pós-teste foi menor que a do pré-teste. As hipóteses para explicar essa ocorrência são as seguintes: o teste foi realizado muito tempo depois da abordagem do conteúdo; desinteresse por parte dos estudantes, motivado por ser uma atividade que não valia nota; e, por ser final de semestre, o foco dos alunos talvez estivesse nas avaliações finais e nas férias. Em vista disso, esses resultados não puderam confirmar as hipóteses (H_0) (o grupo experimental não apresenta aprendizagem superior ao grupo de controle) e (H_1) (o grupo experimental apresenta aprendizagem superior ao grupo de controle). Já a hipótese (H_2) (a aplicação do jogo sério JETS torna o processo de ensino-aprendizagem mais atrativo que o tradicional) foi avaliada com base no questionário de percepções, aplicado após os estudantes jogarem o JETS. A Tabela 5 apresenta as questões do questionário de percepções do JETS com as respostas dadas pelos estudantes. O questionário original do *framework* possui dez perguntas, sendo que a décima primeira foi adicionada para avaliar a hipótese de que os estudantes preferem uma abordagem alternativa (jogos sérios) ao invés de uma atividade tradicional (lista de exercícios). Como proposto no *framework*, os estudantes responderam a essas perguntas com números de 1 a 4, sendo 1 equivalente a sim e 4 equivalendo a não.

Tabela 5 – Questionário de Percepções dos estudantes em relação ao JETS

Questão	**Não**	**mais para não**	**mais para sim**	**Sim**	**não respondeu**
1 - Você gostou do jogo?		1	1	5	
2 - Os objetivos de aprendizagem foram atingidos com o jogo?			3	4	
3 - O conteúdo do jogo foi relevante para o aprendizado?		1	1	5	
4 - A sequência de tópicos do jogo foi adequada?			3	4	
5 - O jogo forneceu informação suficiente sobre o assunto?	1	1	4	1	
6 - O grau de dificuldade do jogo foi adequado para o aprendizado?		1	3	2	1
7 - A duração do jogo foi adequada?	1		2	4	
8 - O método de ensino do jogo foi adequado?		1	1	5	
9 - A contextualização apresentada pelo jogo foi adequada?			3	4	
10 - Você gostou de jogar o JETS?	1	1	1	4	
11 - Você preferiria realizar uma atividade tradicional, como uma lista de exercícios, ao invés de jogar o JETS?	5	1		1	

Sobre a questão 1 (*Você gostou do jogo?*) pode-se perceber que a maioria dos estudantes respondeu que gostou do JETS e apenas um respondeu mais para não. Os trabalhos correlatos de Diniz e Dazzi (2011), Gonçalves, Thiry e Zoucas (2011), Silva (2010) e Wangenheim *et al.* (2009) apresentam experimentos que mostram que a maioria dos estudantes também gostou de jogar os respectivos jogos, o que confirma o potencial educacional dos jogos na educação.

Todos os jogos possuem objetivos e no caso dos jogos sérios também são incluídos os objetivos de aprendizagem. Com relação à questão 2 (*Os objetivos de aprendizagem foram atingidos com o jogo?*), todos os estudantes deram respostas positivas, sim ou mais para sim", logo, isto sugere que todos os estudantes aprenderam sobre Estratégias de Teste de *Software* com a utilização do jogo. Ainda, resultados positivos em relação aos objetivos de aprendizagem também foram obtidos por Gonçalves, Thiry e Zoucas (2011). Isso é importante, pois os estudantes não gostam de atividades nas quais acham que nunca terão sucesso, logo, eles devem se sentir aptos a alcançar os objetivos do jogo (Bachen e Raphael, 2011).

A questão 3 (*O conteúdo do jogo foi relevante para o aprendizado?*) teve, em sua maioria, respostas positivas − apenas um estudante respondeu "mais para não". Já a questão 4 (*A sequência de tópicos do jogo foi adequada?*) obteve apenas respostas positivas, o que indica que a distribuição das fases do jogo foi acertada. Logo, é possível supor que o *design* instrucional foi bem definido e que o *design* do jogo foi bem elaborado. No desenvolvimento de jogos sérios, é fundamental o equilíbrio entre as exigências do *design* do jogo e dos requisitos educacionais, de modo que ambos são importantes e devem se relacionar afim de que culminem em um jogo divertido e que também permita resultados de aprendizagem (Bachen e Raphael, 2011; Klopfer *et al*., 2009).

Em relação à questão 5 (*O jogo forneceu informação suficiente sobre o assunto?*) houve duas respostas negativas e apenas um "sim". Isso pode indicar que apenas os *slides* da aula disponíveis na sala de reuniões do ambiente de jogo e a presença do professor não sejam suficientes para alguns estudantes. Uma possibilidade é o uso de *chatterbots*, que é um tipo de agente que simula conversas inteligentes com um ou mais usuários humanos (Orlando e Giovanni, 2008). Esses agentes podem ser usados para auxiliar os estudantes, fornecendo informações sobre o conteúdo pedagógico.

Na questão 6 (*O grau de dificuldade do jogo foi adequado para o aprendizado?*) a maioria dos estudantes apresentou respostas positivas, mas houve uma resposta "mais para não" e uma abstenção. De qualquer forma, essa é uma questão que pode ser facilmente

resolvida, pois o jogo permite a edição dos desafios, portanto, o nível de dificuldade pode ser alterado de acordo com a turma. Com relação à questão 7 (*A duração do jogo foi adequada?*), apenas um estudante achou a duração do jogo inadequada, mas isso também pode ser ajustado pelo professor, criando ou removendo desafios nas fases. Alguns estudos destacados por Wangenheim e Shull (2009) indicam que alguns estudantes perdem o interesse no jogo quando ele é muito complexo ou muito longo. O JETS está em conformidade com essas questões.

Sobre a questão 8 (*O método de ensino do jogo foi adequado?*), observa-se que a maioria dos estudantes respondeu "sim" para a pergunta, o que apoia o a adoção da abordagem de aprendizagem baseada em jogos. Segundo Pivec (2007), nos últimos anos, observam-se mudanças no meio educacional, como o emergente método de ensino utilizando jogos, o que indica a vontade e a necessidade de mudar o processo e o ambiente de ensino-aprendizagem.

A questão 9 (*A contextualização apresentada pelo jogo foi adequada?*) apresentou somente respostas positivas por parte dos estudantes, o que indica que a narrativa e o ambiente realístico do jogo fornecem uma contextualização adequada entre o conteúdo de Teste de *Software* e o jogo. O que é muito relevante, pois a narrativa permite que o jogador deixe de ser um agente passivo e passe a ser um participante da trama. Os recursos gráficos e computacionais possibilitam, ainda, o enriquecimento do enredo, criando uma atmosfera de imersão no jogo (Beatriz, Martins e Alves, 2009). Assim, a contextualização do jogo tem um papel fundamental no envolvimento do estudante.

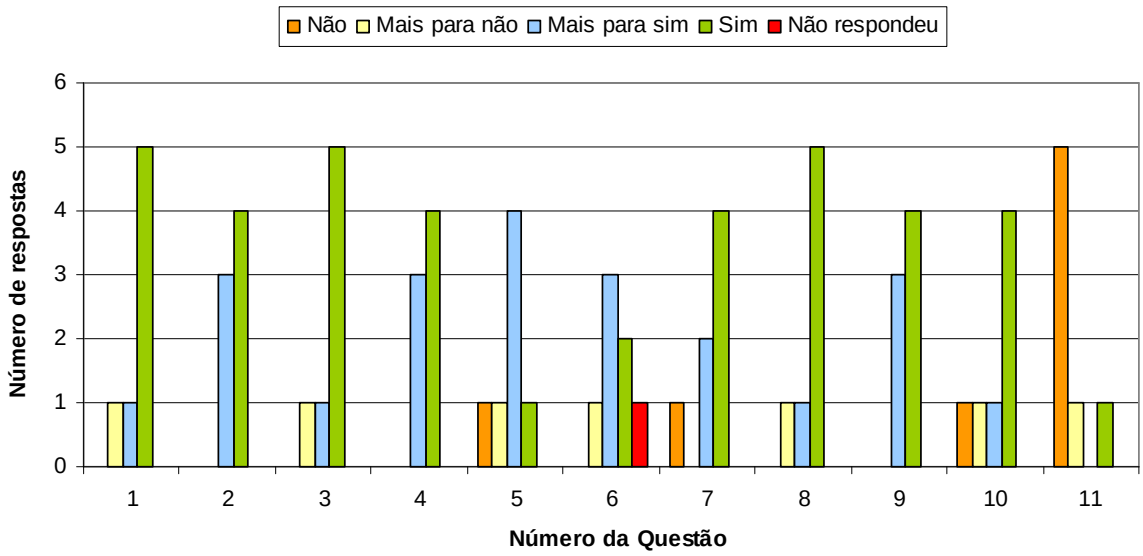
Na questão 10 (*Você gostou de jogar o JETS?*), apenas dois estudantes deram respostas negativas ("não" e "mais para não"). Apenas um respondeu "mais para sim" e todos os demais responderam "sim". Logo, a maioria gostou de jogar o JETS, o que indica que é uma atividade atrativa e divertida, podendo representar um fator motivacional no processo de ensino-aprendizagem. Esse resultado corrobora as afirmações de Mattar (2010), Papastergiou (2009), Münz (2007) e Garris, Ahlers e Driskell (2002) em relação à motivação adquirida pelos estudantes ao realizar atividades de jogos.

Com relação à questão 11 (*Você preferiria realizar uma atividade tradicional, como uma lista de exercícios, ao invés de jogar o JETS?*) a grande maioria respondeu que não preferiria realizar uma atividade tradicional ao invés de realizar uma atividade com o jogo sério JETS. Cabe destacar que o único estudante que respondeu sim a pergunta 11, forneceu respostas positivas na maioria das outras questões. Isso indica que, apesar de ele ter gostado do jogo e ter atingido os objetivos de aprendizagem, provavelmente, possui um estilo de

aprendizagem que se adapta melhor à abordagem tradicional. Entretanto, esse tipo de análise poderia ser mais exata se houvesse um campo de comentários no questionário. Outros estudos como de Freitas e Liarokapis (2011), Hainey *et al.* (2011), Wangenheim e Shull (2009) e Garris, Ahlers e Driskell (2002) mostram que os estudantes preferem atividades de jogo em relação a atividades tradicionais, contribuindo, assim, para a adoção desta recente abordagem de aprendizagem baseada em jogos.

É possível observar no gráfico, ilustrado na Figura 19, que, em geral, os estudantes tiveram uma boa receptividade ao JETS, destacando-se as percepções sobre os objetivos de aprendizagem, a experiência de jogar o JETS, a contextualização do jogo e a predileção dos estudantes à atividade com o JETS em relação à atividade tradicional. Esses resultados positivos indicam que esta é uma abordagem promissora e incentivam a realização de novos experimentos.

Figura 19 – Gráfico de respostas do questionário de percepções dos estudantes em relação ao JETS



No processo de análise dos resultados, foi identificada a necessidade de adaptação do questionário de percepções dos estudantes. Primeiramente, a adição de um campo de comentários para cada questão. Também se verificou que é necessária a inclusão de questões referentes à percepção do estudante relacionada a vários aspectos como: interface gráfica do jogo; interação com os colegas e com o professor; o *avatar*; possibilidade de integração com uma ferramenta de teste utilizada por profissionais; e integração com o AVEA Moodle. Identificou-se, ainda, a necessidade de criação de um questionário de percepções do professor.

8 CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

As atividades de teste são muito importantes no processo de desenvolvimento de *softwares* que atendem a requisitos de qualidade. Com isso, o número de empresas que terceirizam serviços de teste e de equipes de teste dentro das empresas de desenvolvimento está aumentando (Kanij, Merkel e Grundy, 2011). Apesar do Teste de *Software* estar presente em todos os currículos de referência analisados, esse conteúdo é pouco abordado nos cursos de graduação (Chen e Poon, 2004, Elbaum *et al.*, 2007) e não está atendendo às necessidades da indústria (Wangenheim e Silva, 2009; Astigarraga *et al.*, 2010). Tudo isso evidencia a importância do tema e a necessidade de novas abordagens para suprir essa carência do mercado. Outra questão que deve ser considerada são as recomendações relacionadas com as estratégias de ensino descritas nas Diretrizes Curriculares para Cursos de Graduação na Área de Computação (Brasil, 2003), comentadas na seção 2.2. Essas recomendações não podem ser englobadas apenas em aulas tradicionais. Percebe-se que, nesse contexto, os jogos demonstram grande potencial, pois, entre outras coisas, podem mostrar as aplicações dos conteúdos teóricos, estimular a comunicação e motivar os estudantes no processo de ensino-aprendizagem.

O jogo sério desenvolvido em um mundo virtual, apresentado neste trabalho, aborda um tópico amplo relacionado a Teste de *Software*, que são Estratégias de Teste de *Software*. Por ser um tópico amplo, e também devido às diferenças entre os currículos, o jogo denominado Jogo da Equipe de Teste de *Software* (JETS) permite adaptação dos desafios por meio do AVEA Moodle, de modo a contemplar o conteúdo dos diferentes currículos. O fato do jogo ser *multiplayer* possibilita uma maior interação entre os próprios estudantes e também do professor com os mesmos. Essa característica também pode ser interessante para cursos na modalidade à distância e semi-presencial, pois possibilita uma interação diferente de outras mídias e agrega mais um meio de comunicação. Outro grande diferencial do JETS é a possibilidade de exportação dos Casos de Teste desenvolvidos no jogo para ferramenta TestLink, que é uma aplicação *Web* de código aberto para gerenciamento de atividades de Teste de *Software*, o que permite a experiência do estudante com uma ferramenta utilizada por profissionais no mercado de trabalho.

O JETS utiliza um ambiente 3D, proporcionado pelo mundo virtual OpenSim, para construção de cenários realísticos e simula o setor de Teste de *Software* de uma grande empresa, dando uma noção ao estudante sobre o ambiente profissional e possibilitando

aplicação dos conhecimentos de Teste de *Software* em um ambiente controlado, de modo que se pôde cometer erros sem o custo que haveria no mundo real. No mundo virtual, o estudante tem a possibilidade de editar seu *avatar*, de modo que ele fique com suas características físicas (ou as desejadas), o que pode resultar na criação de um vínculo com o personagem, proporcionando, assim, envolvimento e imersão maiores no jogo.

Um dos maiores desafios no desenvolvimento do JETS foi integrar a gama de tecnologias que forma a arquitetura do jogo, o que exigiu a utilização de diferentes linguagens e paradigmas de programação. O fato de utilizar objetos prontos na construção dos cenários agilizou consideravelmente o desenvolvimento, sendo um fator fundamental para conclusão do projeto no prazo. Também é importante destacar que o objeto *Sloodle Quiz Chair* 1.0 não foi a primeira escolha na implementação do JETS. A princípio, o objeto *Sloodle Prim Drop* 1.0 seria utilizado para enviar os resultados do jogo na forma de tarefa. No entanto, ao analisar os demais objetos do Sloodle e estudar as diversas possibilidades da atividade Questionário do Moodle, o *Sloode Quis Chair* 1.0 mostrou-se uma ótima solução para este caso.

O objetivo deste trabalho, enfim, foi apresentar uma abordagem que agregue diferentes estratégias de ensino-aprendizagem, de modo que se adaptem a diferentes cursos de graduação na área de computação. Nesse contexto, esta abordagem visa a ir ao encontro das recomendações relacionadas ao ensino, descritas nas Diretrizes Curriculares para Cursos de Graduação na Área de Computação (Brasil, 2003), utilizando conceitos de aprendizagem baseada em jogos por meio da plataforma de um mundo virtual.

Para avaliação da abordagem proposta neste trabalho, foi utilizado o *framework* de Kochanski (2009) para avaliação empírica de jogos educacionais na área de Engenharia de *Software*. Com base nesse *framework*, foi realizado um experimento com uma turma do Curso de Engenharia da Computação da UFSM na disciplina de Engenharia de *Software*. Inicialmente, foram realizadas aulas em que foram expostos e trabalhados conteúdos relacionados a Teste de *Software*. Em seguida, os estudantes realizaram um pré-teste para avaliar os conhecimentos adquiridos em aula. Posteriormente, os estudantes foram divididos em dois grupos, um experimental e outro de controle. O grupo de controle realizou uma atividade tradicional, que foi uma lista de exercícios, e o grupo experimental utilizou o JETS e respondeu ao questionário de percepções do jogo. Em outra ocasião, foi aplicado o pós-teste para averiguar as diferenças na aprendizagem em relação ao pré-teste.

A hipótese (H_2) foi confirmada, tendo como base as respostas do questionário de percepções do jogo por parte dos estudantes. Corrobora a confirmação dessa hipótese, o

grande interesse, envolvimento e motivação por parte dos estudantes, observados durante a aplicação do jogo. Importante, sobretudo, é que, com base nas percepções dos estudantes em relação ao jogo sério JETS, é possível concluir que a adoção de uma abordagem de aprendizagem baseada em jogos, utilizando o JETS, é promissora, pois os estudantes relataram ter atingido os objetivos de aprendizagem e ter gostado da experiência de jogar o JETS, considerando essa uma atividade mais atrativa do que uma abordagem tradicional. Devido ao não comparecimento de alguns estudantes e ainda a fatores externos ao experimento, que podem ter influenciado nos resultados, não foi possível confirmar o experimento com relação à aprendizagem (hipóteses (H_0) e (H_1)). Logo, os resultados quantitativos que mediriam a aprendizagem por meio de um pré-teste e de um pós-teste, não foram considerados. No entanto, esse foi um experimento inicial e outros devem ser realizados com o objetivo de testar novamente as hipóteses definidas. Algumas lições aprendidas nesse experimento podem ser destacadas como: o período de aplicação do experimento deve ser definido entre o início e a metade do semestre, período em que, a princípio, o estudante tem maior disponibilidade; aplicar o pré-teste, o pós-teste, o jogo e a lista de exercícios como atividades que valem nota para a disciplina; realizar todo experimento de forma contínua, evitando datas como feriados ou aulas com outros assuntos entre as etapas do experimento; disponibilizar uma aula para ambientação no jogo − mesmo que não seja presencial−, de modo que os estudantes adquiram “fluência” no mundo virtual e aprendam o funcionamento do jogo, bem como editem seus *avatares* sem pressa, propiciando maior vínculo com o personagem.

Como trabalhos futuros, pretende-se:

- Refinar o experimento de modo a avaliar melhor as características do jogo, como interface, jogabilidade e integração com outras plataformas, adaptando as questões de percepção e incluindo o professor na avaliação;
- Realizar o experimento em outras turmas, cursos e instituições de modo a confirmar a efetividade do jogo;
- Realizar experimentos em cursos à distância, o que exige um estudo mais direcionado, incluindo testes mais específicos de configuração de *hardware* e desempenho do OpenSim, pois cursos à distância tem o potencial de atingir um número maior de estudantes;

- Implementar melhorias no jogo, como a inclusão de desafios coletivos, de modo que exija a cooperação entre os estudantes para a resolução dos problemas;
- Implementar agentes do tipo *chatterbot* para auxiliar os estudantes, dando informações sobre o jogo e ajudando na resolução dos desafios, pois segundo Orlando e Giovanni (2008), o uso de *chatterbots* pode auxiliar os professores no monitoramento de atividades e também aumentar a satisfação e a eficiência didática dos estudantes no contexto de *e-learning*;
- Implementar melhorias na jogabilidade do JETS, aproximando-o mais dos jogos de entretenimento;
- Integrar o jogo a outras ferramentas relacionadas à atividade de Teste de *Software*;
- Desenvolver um *framework* para desenvolvimento de jogos sérios em mundos virtuais;
- Desenvolver aplicações de aprendizagem baseada em jogos para apoio ao ensino-aprendizagem de outros tópicos de Teste de *Software* e de Engenharia de *Software*; e
- Envolver os estudantes de graduação no processo de desenvolvimento de jogos em mundos virtuais, de modo a propiciar a aprendizagem tangencial (Mattar, 2010).

REFERÊNCIAS BIBLIOGRÁFICAS

ABT, C. C. **Serious games**. Boston: University Press of America, 1987.

ADAMS, E. **Fundamentals of game design**. Berkeley: New Riders, 2nd ed., 2010.

ADVANCED DISTRIBUTED LEARNING. **SCORM Users Guide for instructional designers**, v.8, 2011. Disponível em: http://www.adlnet.gov/wp-content/uploads/2011/12/SCORM_Users_Guide_for_ISDs.pdf . Acesso em: 12 jan. 2012.

ALDRICH, C. Learning **Online with games, simulations, and virtual worlds**: strategies for online instruction. San Francisco: Jossey-Bass, 2009.

ASTIGARRAGA, T.; DOW, E. M.; LARA, C.; PREWITT, R.; WARD, M. R. The emerging role of software testing in curricula. In: TRANSFORMING ENGINEERING EDUCATION: CREATING INTERDISCIPLINARY SKILLS FOR COMPLEX GLOBAL ENVIRONMENTS, 1., 2010, Dublin. **Anais**. Dublin: IEEE, 2010.

ASSOCIATION FOR COMPUTING MACHINERY; ASSOCIATION FOR INFORMATION SYSTEMS. **Model curriculum and guidelines for undergraduate degree programs in Information Systems**, 2010. Disponível em: <http://www.acm.org/education/curricula/IS%202010%20ACM%20 final.pdf>. Acesso em: 1 fev. 2012.

ASSOCIATION FOR COMPUTING MACHINERY; INSTITUTE OF ELECTRICAL AND ELECTRONIC ENGINEERS COMPUTER SOCIETY. **Computer Science curriculum 2008**: an interim revision of CS 2001, 2008. Disponível em: <http://www.acm.org/education/curricula/ComputerScience2008.pdf>. Acesso em: 1 fev. 2012.

AZEVEDO, E. (Coord.). **Desenvolvimento de jogos 3D e aplicações em realidade virtual**. Elsevier: Rio de Janeiro, 2005.

BACHEN, C. M.; RAPAHEL, C. Social flow and learning in digital games: a conceptual model and research agenda. In: MA, M.; OIKONOMOU, A.; JAIN, L. C. (Ed.). **Serious games and edutainment applications**. London: Springer, 2011.

BAINBRIDGE, W. S. (Ed.). **Online worlds**: convergence of the real and the virtual. London: Springer, 2010.

BATES, B. **Game design**. Boston: Thomson Course Technology, 2nd ed., 2004.

BEATRIZ, I.; MARTINS, J.; ALVES, L. A crescente presença da narrativa nos jogos eletrônicos. In: BRAZILIAN SYMPOSIUM ON GAMES AND DIGITAL ENTERTAINMENT, 8., 2009, Rio de Janeiro. **Anais…** Rio de Janeiro, 2009.

BENITTI, F. B. V.; MOLLÉRI, J. S. Utilização de um RPG no ensino de gerenciamento e processo de desenvolvimento de software. In: WORKSHOP SOBRE EDUCAÇÃO EM COMPUTAÇÃO, 16., 2008, Belém do Pará. **Anais...** Belém do Pará: SBC, 2008.

BRASIL. **Diretrizes Curriculares dos cursos de Bacharelado em Ciência da Computação, Engenharia de Computação, Engenharia de Software e Sistemas de Informação e dos cursos de Licenciatura em Computação**. 2003. Disponível em: <portal.mec.gov.br/cne/arquivos/pdf/CES0492.pdf>. Acesso em: 1 nov. 2011.

CHEN, B.; HUANG, F.; LIN, H.; HU, M. VCUHK: integrating the real into a 3D campus in networked virtual worlds. In: INTERNATIONAL CONFERENCE ON CYBERWORLDS, 10., 2010, Singapura. **Anais**.
Singapura: IEEE, 2010.

CHEN, T. Y.; POON, P. L. Experience with teaching black-box testing in a computer science/software engineering curriculum. **IEEE Transactions on Education**, v. 47, n. 01, p. 42-50, 2004.

COMISSÃO DE ESPECIALISTAS DE ENSINO DE COMPUTAÇÃO E INFORMÁTICA. **Diretrizes Curriculares de Cursos da Área de Computação e Informática**. 1999. Disponível em: <http://www.inf.ufrgs.br/ecp/docs/diretriz.pdf>. Acesso em: 26 mai. 2011.

COMITÊ TÉCNICO DE IMPLEMENTAÇÃO DO SOFTWARE LIVRE. Website. Disponível em: <http://www.softwarelivre.gov.br/>. Acesso em: 3 ago. 2011.

DREHER, C. REINERS, T.; DREHER, N.; DREHER, H. 3D virtual worlds enriching innovation and collaboration in information systems research, development, and commercialisation. In: INTERNATIONAL CONFERENCE ON DIGITAL ECOSYSTEMS AND TECHNOLOGIES, 3., 2009, Istanbul. **Anais...** Istanbul: IEEE, 2009.

DINIZ, L. L.; DAZZI, R. L. S. Jogo para o apoio ao ensino do teste de caixa-preta. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 22., 2011, Aracaju, SE, **Anais...** Aracaju, SE, 2011.

ELBAUM, S.; PERSON, S.; DOKULIL, J.; JORDE, M. Bug hunt: making early software testing lessons engaging and affordable. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, 29., 2007, Minneapolis. **Anais...** Minneapolis, 2007.

FILATRO, A. **Design instrucional na prática**. São Paulo: Pearson, 2008.

FINNEY, K. C. **3D Game programing: all in one**. Boston: Thomson Course Technology, 2004.

FOWLER, M.; RICE, D. FOEMMEL, M.; HIEATT, E.; MEE, R.; STAFFORD, R. **Padrões de arquitetura de aplicações corporativas**. São Paulo: Bookman, 2006.

FOX, M. R.; KELLY, H.; PATIL S. Medulla: A cyberinfrastructure-enabled framework for research, teaching and learning with virtual worlds. In: BAINBRIDGE, W. S. (Ed.). **Online worlds**: convergence of the real and the virtual. London: Springer, 2010.

FREITAS, S.; LIAROKAPIS, F. Serious games: a new paradigm for education? In: MA, M.; OIKONOMOU, A.; JAIN, L. C. (Ed.). **Serious games and edutainment applications**. London: Springer, 2011.

FULLERTON, T.; SWAIN, C.; HOFFMAN, S. S. **Game design workshop**: a playcentric approach to creating innovative games. Burlington: Elsevier, 2nd ed., 2008.

GARRIS, R.; AHLERS, R., DRISKELL, J. E. Games, motivation, and learning: a research and practice model. **Simulation & Gaming**, v. 33, n. 4, p. 441-467, dez. 2002. Disponível em: <http://sag.sagepub.com/content/33/4/441>. Acesso em: 20 mar. 2011.

GOMES, A. S.; MEDEIROS, F. P. A. ARAÚJO, T. S.; VASCONCELOS, B. Q.; ALBUQUERQUE, F. A.; PAIVA, P. V. F. Instalação, Configuração e uso da plataforma de gestão de aprendizagem Amadeus. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 21., 2010, João Pessoa, PE,. **Anais...** João Pessoa, PE, 2010.

GONÇALVES, R. Q.; M. THIRY; A. ZOUCAS. Avaliação da aprendizagem em experimentos com jogo educativo de engenharia de requisitos. In: SIMPÓSIO BRASILEIRO DE QUALIDADE DE SOFTWARE, 10., 2011, Curitiba, PR,. **Anais...** Curitiba, PR, 2011.

HAINEY, T.; CONNOLLY, T. M.; STANSFIELD, M.; BOYLE, E. A. Evaluation of a game to teach requirements collection and analysis in software engineering at tertiary education level. **Computers & Education**, v. 56, n. 1, p.21-35, 2011.

HASS, A. M. J. **Guide to advanced software testing**. Boston: Artech House, 2008.

HODGE, E.; COLLINS, S.; GIORDANO, T. **The virtual worldsh**: how to use Second Life and other 3D virtual environments. Sudbury: Jones and Bartlett, 2011.

INSTITUTO NACIONAL DE ESTUDOS E PESQUISAS EDUCACIONAIS ANÍSIO TEIXEIRA. **Senso da educação superior 2010**. Disponível em: <http://portal.inep.gov.br>. Acesso em: 19 out. 2011.

INSTITUTE OF ELECTRICAL AND ELECTRONIC ENGINEERS COMPUTER SOCIETY. **SWEBOK**: guide to the software engineering body of knowledge, 2004. Disponível em: <http://www.computer.org/portal/web/swebok/htmlformat>. Acesso em: 17 abr. 2010.

_____. **Std 829**: Standard for software test documentation. New York, 1998.

INSTITUTE OF ELECTRICAL AND ELECTRONIC ENGINEERS COMPUTER SOCIETY; ASSOCIATION FOR COMPUTING MACHINERY. **Curriculum guidelines for undergraduate degree programs in Computer Engineering**, 2004a. Disponível em: <http://www.acm.org/education/education/curric_vols/CE-Final-Report.pdf>. Acesso em: 1 fev. 2012.

_____. **Curriculum guidelines for undergraduate degree programs in Software Engineering**, 2004b. Disponível em: <http://sites.computer.org/ccse/SE2004Volume.pdf>. Acesso em: 1 fev. 2012.

_____. **Curriculum guidelines for undergraduate degree programs in Information Technology**, 2008. Disponível em: <http://www.acm.org//education/curricula/IT2008%20 Curriculum.pdf>. Acesso em: 1 fev. 2012.

KANIJ, T.; MERKEL, R.; GRUNDY, J. A preliminary study on factors affecting software testing team performance. In: INTERNATIONAL SYMPOSIUM ON EMPIRICAL SOFTWARE ENGINEERING AND MEASUREMENT, 5., 2011, Banff. **Anais…** Banff, 2011.

KAVANAGH, P. **Open source software**: implementation and management. Oxford: Elsevier, 2004.

KISHIMOTO, T. M. **O jogo e a educação infantil**. São Paulo: Thonson, 2008.

KLOPFER, E.; OSTERWEIL, S.; SALE, K; HAAS, J.; GROFF, J.; ROY, D. **Moving learning games forward: obstacles, opportunities e openness**. 2009. Disponível em: <http://education.mit.edu/papers/MovingLearningGamesForward EdArcade.pdf>. Acesso em: 4 jan. 2012.

KOCHANSKI, D. **Um framework para apoiar a construção de experimentos na avaliação empírica de jogos educacionais**. 2009. 224f. Dissertação (Mestrado em Computação Aplicada) – Universidade do Vale do Itajaí, São José, 2009.

KURBANOGLU, S.; Al, U.; ERDOGAN, P. L.; TONTA, Y.; UCAK, N. (Ed.). Technological convergence and social networks in information management. **Communications in Computer and Information Science**, v. 96, 2010.

LIVINGSTONE, D.; KEMP, J. Integrating web-based and 3D learning environments: Second Life meets Moodle. **Upgrade**, v. 9, n. 3, jun. 2008.

MARTINELLI, D.P.; VENTURA, C. A. A. (Org.). **Visão sistêmica e administração: conceitos, metodologias e aplicações**. São Paulo: Saraiva, 2006.

MATTAR, J. **Games em educação**: como nativos digitais aprendem. São Paulo: Person, 2010.

MICHAEL D.; CHEN, S. **Serious games**: games that educate, train and inform. Boston: Thomson Course Technology, 2006.

MONSALVE, E. S.; WERNECK, V. M. B.; LEITE, J. C. S. P. Teaching software engineering with SimulES-W. In: IEEE-CS CONFERENCE ON SOFTWARE ENGINEERING EDUCATION AND TRAINING, 24., 2011, Honolulu. **Anais...** Honolulu, 2011.

MOODLE. Website. Disponível em: <http://www.moodle.org>. Acesso em: 17 dez. 2011.

MOORE, D.; THOME, M.; HAIGH, K. **Scripting your world**: the official guide to second life scripting. Indianapolis: Wiley, 2008.

MÜNZ, U. SCHUMM, P.; WIESEBROCK, A.; ALLGOWER, F. Motivation and learning progress through educational games. **IEEE Transactions on Industrial Electronics**, n. 6, v. 54 dez. 2007.

MYERS, G. J.; BADGETT, T.; THOMAS, T. M.; SANDLER, C. **The art of software testing**. New Jersey: John Wiley & Sons, 2.ed. 2004.

NATIONAL RESEARCH CONCIL. **The rise of games and high-performance computing for modeling and simulation**. Washington: The National Academies Press, 2010.

NAVARRO, E.; HOEK, A. Multi-site evaluation of SimSE. In: TECHNICAL SYMPOSIUM ON COMPUTER SCIENCE EDUCATION, 40., 2009, New York. **Anais…** New York: ACM, 2009.

NEUMANN J.; MORGENSTERN, O. **Theory of games and economic behavior**. New Jersey: Princeton University Press, 2007.

NEVO, S.; NEVO D. Re-invention of applicable innovations: the case of virtual worlds. In: HAWAII INTERNATIONAL CONFERENCE ON SYSTEM SCIENCES, 44., 2011, Hawaii. **Anais…** Hawaii: IEEE, 2011.

OLIVEIRA, K. L.; SANTOS, A. A. A. Compreensão em leitura e avaliação da aprendizagem em universitários. **Psicologia: Reflexão e Crítica**, v.18, n.1, p. 118-124, 2005. Disponível em: <http://www.scielo.br/pdf/prc/v18n1/24825.pdf>. Acesso em: 15 fev. 2012.

OPEN SIMULATOR. Website. Disponível em: <http://opensimulator.org>. Acesso em: 30 mai. 2011.

ORLANDO, P; GIOVANNI, F. An integrated system, with natural language management, for the monitoring activities in e-learning environments. In: INTERNATIONAL CONFERENCE ON COMPLEX, INTELLIGENT AND SOFTWARE INTENSIVE SYSTEMS, 2., 2008, Catalonia .**Anais…** Catalonia: IEEE, 2008.

PAPASTERGIOU, M. Digital Game-Based Learning in high school computer science education: impact on educational effectiveness and student motivation. **Computers & Education**, v. 52, n. 1, p.1-12, 2009.

PEACHEY, A.; GILLEN, J.; LIVINGSTONE, D.; SMITH-ROBBINS, S. (Ed.). **Researching learning in virtual worlds**. London: Springer, 2010.

PEREIRA, A. T. C. (Org.). **Ambientes virtuais de aprendizagem em diferentes contextos.** Rio de Janeiro: Ciência Moderna, 2007.

PIVEC, M. Play and learn: potentials of game-based learning. **British Journal of Educational Technology,** v. 38, n. 3, p. 387-393, 2007.

PIVEC, M.; DZIABENKO, O. Game-based learning in universities and lifelong learning: "UniGame: social skills and knowledge training" game concept. **Journal of Universal Computer Science**, n. 1, v. 10, p.14-26, 2004.

PÖTTER, H.; SCHOTS, M. InspectorX: um jogo para o aprendizado em inspeção de software. FÓRUM DE EDUCAÇÃO EM ENGENHARIA DE SOFTWARE, 4., 2011, São Paulo. **Anais...** São Paulo: USP, 2011.

PRENSKY, M. **Digital game-based learning**. New York: McGraw-Hill, 2001.

PRESSMAN, R. S. **Engenharia de software**: uma abordagem profissional. Porto Alegre: AMGH, 7.ed. 2011.

PRIKLADNICKI, R.; ALBUQUERQUE, A. B.; WANGENHEIM, C. G.; CABRAL, R. Ensino de engenharia de software: desafios, estratégias de ensino e lições aprendidas. In: FÓRUM DE EDUCAÇÃO EM ENGENHARIA DE SOFTWARE, 2., 2009, Fortaleza. **Anais eletrônicos**... Fortaleza: UFC, 2009. Disponível em: <http://fees.inf.puc-rio.br/FEESArtigos/ FEES09/ >. Acesso em: 1 mar. 2010.

PRIKLADNICKI, R.; WANGENHEIM, C. G. O Uso de jogos educacionais para o ensino de gerência de projetos de software. In: FÓRUM DE EDUCAÇÃO EM ENGENHARIA DE SOFTWARE, 1., 2008, Fortaleza. **Anais eletrônicos...** Rio de Janeiro: PUC, 2008. Disponível em: <http://fees.inf.puc-rio.br/FEESArtigos/ FEES08/ >. Acesso em: 1 mar. 2010.

RESENDE, R.; SOUZA, M.; FRANCO, E. F.; CARVALHO, F.; SCHMOELLER, L.; RODRIGUES, A. SPARSE: um ambiente de ensino e aprendizado de engenharia de software baseado em jogos e simulação. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 21., 2010, João Pessoa, PB,. **Anais…** João Pessoa, PB, 2010.

RIEDEL, J. C. K. H.; HAUGE, J. B. State of the art of serious games for business and industry. In: INTERNATIONAL CONFERENCE ON CONCURRENT ENTERPRISING, 17., 2011, Aachen. **Anais…** Aachen: IEEE, 2011.

RITZEMA, T.; HARRIS, B. The use of Second Life for distance education. **Journal of Computing Sciences in Colleges**, v.23, n.6, jun. 2008.

ROLLINGS, A; ADAMS, E. **Andrew Rollings and Ernest Adams on game design**. Berkeley: New Riders, 2003.

SALEN, K; ZIMMERMMAN, E. **Rules and play**: game design fundamentals. Cambridge: The MIT Press, 2004.

SAVIN-BADEN, M. **A practical guide to using Second Life in higher education**. New York: McGraw-Hill, 2010.

SCHAF, F. M. **Arquiteura modular para ambientes virtuais de ensino de automação com suporte a realidade mista e colaboração**. 2011. 153 f. Tese (Doutorado em Engenharia Elétrica) – Universidade Federal do Rio Grande do Sul, Porto Alegre, 2011.

SCHELL, J. **The art of game design**: a book of lenses. Burlington: Elsevier, 2008.

SIDHU, M. S. **Technology-assisted problem solving for engineering education**: interactive multimedia applications. Hershey: Engineering Science Reference, 2010.

SILVA, A. C. **Jogo educacional para apoiar o ensino de técnicas para elaboração de testes de unidade**. 2010. 179f. Dissertação (Mestrado em Computação Aplicada) – Universidade do Vale do Itajaí, São José, 2010.

SILVA, E. L.; MENEZES, E. M. **Metodologia da Pesquisa e Elaboração de Dissertação**. 3.ed., 2001. Disponível em:<http://projetos.inf.ufsc.br/arquivos/Metodologia%20da%20Pesquisa%203a%20edicao.pdf >. Acesso em: 31 mai. 2011.

SILVA, T. S.; MÜLLER, F. M.; BERNARDI, G. Panorama do ensino de engenharia de software em cursos de graduação focado em teste de software: uma proposta de aprendizagem baseada em jogos. **Renote**, v. 9, n. 2, 2011.

SERIOUS GAMES INITIATIVE. Website. Disponível em: < http://www.seriousgames.org>. Acesso em 11 dez. 2011.

SOCIEDADE BRASILEIRA DE COMPUTAÇÃO. **Currículo de referência da SBC para cursos de graduação em bacharelado em Ciência da Computação e Engenharia de Computação**, 2005. Disponível em: <http://portal.sbc.org.br/educacao/lib/exe/fetch.php?media= documentos:cr2005.pdf >. Acesso em: 1 fev. 2012.

_____. **Currículo de referência para cursos de bacharelado em Sistemas de Informação**, 2003. Disponível em: <http://www.sbc.org.br/index.php?option=com_jdownloads&Itemid=195&task=finish&cid=52&catid=36>. Acesso em: 1 fev. 2012.

SOMMERVILLE, I. **Engenharia de software**. São Paulo: Pearson Addison Wesley, 8.ed. 2007.

STROUSTRUP, B. What should we teach new software developers? Why?. **Communications of the ACM**, v.53, n.1, jan. 2010.

STURGEON, T.; ALLISON, C.; MILLER, A. Exploring 802.11: real learning in a virtual world. In: FRONTIERS IN EDUCATION CONFERENCE, 39., 2009, San Antonio. **Anais…** San Antonio, 2009.

TAURION, C. **Mundos virtuais, pessoas reais**. v.2. Ebook, 2009. Disponível em: <http://www.smashwords.com/books/view/3186>. Acesso em: 12 dez. 2011.

THOMAS, D.; BROWN, J. S. Why virtual worlds can matter. **International Journal of Learning and Media**, v.1, n.1, p.37-49, mar. 2009. Disponível em: <http://www.mitpressjournals.org/doi/pdf/10.1162/ijlm.2009.0008>. Acesso em: 12 dez. 2011.

THORN, A. **Game engine design and implementation**. London: Jones & Bartlett, 2010.

TORO-TROCONIS, M.; MEERAN, K.; HIGHAM, J.; MELLSTRÖM, U.; PARTRIDGE, M. Design and delivery of game-based learning for virtual patients in Second Life: initial findings. In: PEACHEY, A. *et al*. (Ed.). **Researching learning in virtual worlds**. London: Springer, 2010.

VALENTE, C.; MATTAR, J. **Second Life e web 2.0 na educação**: o potencial revolucionário das novas tecnologias. São Paulo: Novatec, 2007.

VALÉRIO, S. R. S. **Sistema de informação no Second Life® interligando serviços instant messaging e SMS**. 2007. 61 f. Dissertação (Mestrado em Engenharia Electrotécnica e de Computadores) – Universidade de Trás-os-Montes e Alto Douro, Vila Real, 2007.

VARGAS, D. P.; MORO, T.; DAMBROSIO, G. M.; CASSAL, M. L.; BERNARDI, G.; CORDENONSI, A. Z. Desenvolvimento de um jogo de empresa baseado em agentes de software e instituições eletrônicas para simulação de elicitação de requisitos de software. In:

WORKSHOP SOBRE EDUCAÇÃO EM COMPUTAÇÃO, 18., 2010, Minas Gerais. **Anais...** Minas Gerais: PUC, 2010.

VINCENTI, G.; BRAMAN, J. **Multi-user virtual environments for the classroom: practical approaches to teaching in virtual worlds**. Hershey: Information Science Reference, 2011.

WANG, T.; ZHU, Q. A Software engineering education game in a 3-D online virtual environment. In: INTERNATIONAL WORKSHOP ON EDUCATION TECHNOLOGY AND COMPUTER SCIENCE, 1., 2009, Wuhan. **Anais...** Wuhan: IEEE, 2009.

WANGENHEIM, C. G.; THIRY, M.; KOCHANSKI, D.; STEIL, L.; SILVA, D.; LINO, J. Desenvolvimento de um jogo para ensino de medição de software. VIII SIMPÓSIO BRASILEIRO DE QUALIDADE DE SOFTWARE, 8., 2009, Minas Gerais. **Anais...** Minas Gerais: PUC, 2009.

WANGENHEIM, C. G.; KOCHANSKI, D.; SAVI, R. Revisão sistemática sobre avaliação de jogos voltados para aprendizagem de engenharia de software no Brasil. In: FÓRUM DE EDUCAÇÃO EM ENGENHARIA DE SOFTWARE, 2.,2009, Fortaleza. **Anais eletrônicos...** Fortaleza: UFC. Disponível em: <http://fees.inf.puc-rio.br/FEESArtigos/ FEES09/ >. Acesso em: 1 mar. 2010.

WANGENHEIM, C. G.; SHULL, F. To Game or Not to Game? **IEEE Software**, v.26, n.2, p. 92-94, mar./abr. 2009.

WANGENHEIM, C. G.; SILVA, D. A. Qual conhecimento de engenharia de software é importante para um profissional de software? In: FÓRUM DE EDUCAÇÃO EM ENGENHARIA DE SOFTWARE, 2.,2009, Fortaleza. **Anais eletrônicos...** Fortaleza: UFC. Disponível em: <http://fees.inf.puc-rio.br/FEESArtigos/ FEES09/ >. Acesso em: 1 mar. 2010.

WANKEL, C.; KINGSLEY, J. **Higher education in virtual worlds**. Bingley: Emerald, 2009.