Software Educacional para Prática do *Scrum*

Felipe Siller, Juliana Cristina Braga

Centro de Matemática, Computação e Cognição – Universidade Federal do ABC (UFABC)
Santo André – SP – Brasil

felipe.siller@aluno.ufabc.edu.br, juliana.braga@ufabc.edu.br,

Abstract. *Within the theme of Software Engineering Agile Scrum methodology has been widely used in the labor market. Show the student how this process works in practice is complicated, due to the peculiarities of the process (daily meetings, weekly deliveries, etc.). Thus, to minimize this problem, we present software, more precisely an educational game for learning and practicing Scrum. We used INTERA methodology for developing the educational software.*

Resumo. *Dentro da temática da Engenharia de Software a metodologia ágil Scrum vem sendo amplamente utilizada no mercado de trabalho. Mostrar para o aluno como esse processo funciona na prática é um desafio para o professor, devido as peculiaridades do Scrum como: reuniões diárias, entregas semanais, etc). Assim, para minimizar essa dificuldade, é apresentado um software, mais precisamente um jogo educacional, para ensino e prática do Scrum. O processo de desenvolvimento foi norteado pela metodologia INTERA específica para o desenvolvimento de objetos de aprendizagem.*

1. Introdução

Dentro das áreas de conhecimento da Engenharia de Software estão, entre outras, os requisitos, a qualidade e os processos de desenvolvimento software. Um dos conhecimentos abordados por este último são as metodologias de desenvolvimento ágeis. Dentre as metodologias ágeis existentes, uma das mais recentemente utilizadas no mercado de trabalho é o *Scrum*. O *Scrum* que é um processo ágil para gerenciamento de projetos dentro do qual se pretende resolver problemas complexos e adaptativos, e ao mesmo tempo entregar produtos com o mais alto valor possível, de maneira a maximizar a produtividade [Schwaber e Sutherland 2011].

Apesar da forte demanda do mercado, o ensino prático do *Scrum* é um grande desafio da área acadêmica, já que não é trivial para o professor replicar no ambiente acadêmico, situações do mercado como: promover reuniões diárias, cobrar um planejamento prévio do projeto de software, estabelecer metas, gerenciar os requisitos, cobrar que as entregas sejam feitas parcialmente e dentro do planejado, entre outros. Essa dificuldade acarreta problemas de adaptação dos alunos quando eles saem do escopo acadêmico para entrarem no mercado de trabalho. Observa-se nesse ponto que existe uma lacuna entre os alunos que estão sendo formados e o que o mercado

DOI: 10.5753/CBIE.WCBIE.2013.152

necessita. Uma possível solução para preencher essa lacuna é a utilização de softwares que possam simular e gerenciar a aplicação dos conceitos teóricos do *Scrum* em sala de aula.

Diante desse contexto, o objetivo principal desse trabalho é desenvolver um jogo para o ensino prático da metodologia ágil *Scrum* nas disciplinas de Engenharia de Software e correlatas. Esse objetivo visa contribuir para o aumento do aprendizado do aluno consequente redução do o *gap* existente entre as necessidades do mercado e o ensino do *Scrum*. O objetivo secundário desse trabalho é utilizar a metodologia INTERA pra direcionar o desenvolvimento do software proposto e verificar a aderência dessa metodologia nesse processo.

2. Referencial Teórico

2.1. Ensino de Engenharia de Software

A Engenharia de Software (ES) é uma área da Ciência da Computação que contém conhecimento suficiente para auxiliar na criação de diversos tipos de sistemas, sejam eles simples ou complexos. Tipicamente, segundo Possa [2011], o ensino tradicional de ES é baseado em dois componentes: aulas teóricas, nas quais as teorias de ES e conceitos relacionados são apresentados; e projetos, nos quais os estudantes devem trabalhar em grupos para desenvolver parte de um software. Porém, o ensino de ES ainda é um desafio, como mostra a pesquisa feita por Sargent [2004]. A pesquisa revela que apenas 40% dos profissionais da área de informática dos Estados Unidos possuem formação nessa área; 40% desses conhecem os principais campos da ES, tais como: requisitos, arquitetura, testes, gestão de projetos; a maior parte é qualificada em programação e/ou são especialistas em determinados tipos de produtos. Apesar de não terem sido encontrados dados estatísticos a respeito no Brasil, acredita-se que a realidade dos profissionais de ES neste país não deve ser diferente [Braga 2009].

No meio acadêmico, dificuldades no ensino de ES já foram relatadas por Soares, [2000], Castro *et al.* [2000] e Hazzan e Dubinsky, [2003]. Dentre essas dificuldades estão: muito conteúdo sendo dado em pouco tempo; baixa motivação que os alunos de ciência da computação possuem para estudar os conceitos teóricos de ES; dificuldades em preparar os estudantes para a prática profissional dentro de ambientes acadêmicos. Diferentes abordagens têm sido adotadas na tentativa de melhorar o ensino de ES, dentre elas: metodologias de ensino Soares, [2000], Castro *et al*. [2000] e Hazzan e Dubinsky, [2003]; jogos e/ou ferramentas Tao e Qing, [2009] Baker *et al*. [2003].

2.2. Jogos Educativos para ensino de Engenharia de Software

Nos últimos anos, novas práticas para o ensino da ES foram adotadas, dentre elas a análise de estudos de caso de grandes projetos, treinamentos que integram a sala de aula com a indústria, e a utilização de jogos de simulação. Segundo Adams, [2010], jogo é o tipo de atividade lúdica realizada no contexto de uma realidade simulada, no qual os participantes tentam alcançar pelo menos um objetivo de forma arbitrária e não trivial, agindo de acordo com regras. Sua definição, segundo o autor, não menciona competição ou conflito, nem entretenimento ou recreação.

Uma das vantagens do uso de jogos educacionais, de acordo com Feitosa, [2010], é que estes contêm elementos lúdicos e desafiadores. Dessa forma, a aprendizagem prolonga-se fora da sala de aula, pelo cotidiano, em um desenvolvimento muito mais rico do que algumas informações que o aluno decora. No caso específico do ensino da ES, os requisitos para o desenvolvimento de jogos, segundo Fernandes e Werner, [2009] poderiam incluir: colaboração, isto é, desenvolvimento de um projeto em equipe; conteúdo/desafio, em que uma importante característica do jogo seria o relacionamento entre os desafios do mesmo e o conhecimento que se deseja transmitir; flexibilidade, pois existe uma grande necessidade de que o jogo possa acompanhar evolutivamente a disciplina abordada; participação do professor, a fim de acelerar, modificar ou direcionar o jogo; foco em processo, uma vez que no mundo real existem problemas de escopo, prazo, custo e qualidade; níveis, sendo uma prática comum em jogos digitais a utilização de níveis de dificuldade para permitir aos jogadores maiores desafios; lúdico, pois esse fator é de vital importância para a aceitação do jogo pelo aluno; multimídia, fator importante para atrair a atenção dos alunos; local x remoto, a medida que permite aos estudantes treinar fora do período e do recinto escolar e em seus horários disponíveis.

2.3. Scrum

O *Scrum* é um processo ágil e simples para gerenciamento de projetos criado para suportar o desenvolvimento e manutenção de produtos complexos, sendo formado por quatro elementos: equipe; eventos; artefatos; regras Schwaber e Sutherland, [2011]. A equipe *Scrum* é responsável por entregar produtos de forma iterativa e incremental, maximizando as oportunidades de realimentação. Possui três papéis principais: i) *Product Owner*, que é o responsável por maximizar o valor do produto e do trabalho da equipe; ii) Equipe de Desenvolvimento: formada por profissionais que realizam o trabalho de entregar uma versão pronta para uso; e iii) *Scrum Master*, responsável por garantir que o *Scrum* seja entendido e aplicado.

O principal evento do *Scrum* é a *Sprint*, durante a qual uma versão incremental potencialmente utilizável do produto é criada. Possui duração aproximada de um mês e contém a definição do que é para ser construído. É composta por: reunião de planejamento da *Sprint*; reuniões diárias; trabalho de desenvolvimento; revisão da *Sprint*; retrospectiva da *Sprint*.

Um dos artefatos do *Scrum* é o *Backlog* do Produto, que é uma lista ordenada de tudo aquilo que deve ser necessário ao produto. O *Backlog da Sprint,* outro artefato, é a previsão da equipe de Desenvolvimento sobre qual funcionalidade estará no próximo incremento e do trabalho necessário para entrega-la.

2.4. Trabalhos Relacionados

Para uma melhor compreensão sobre o tema foram realizados estudos de trabalhos relacionados a jogos digitais específicos para o ensino do *Scrum*. A tabela 1 mostra a descrição dos jogos. Nela, estão indicados os nomes dos jogos, os autores, os tipos dos jogos, isto é, quais os objetos utilizados para jogá-los como, por exemplo, cartas, tabuleiro, computador, e uma pequena descrição sobre o que os jogos abordam.

Tabela 1. Revisão de jogos existentes para o ensino do *Scrum.*

Jogo	Referência	Tipo de Jogo	Descrição
PlayScrum	Fernandes e Souza, [2010]	Cartas	Cada aluno joga exercendo o papel de *Scrum Master* e gerencia o projeto, por meio de cartões, seguindo as práticas do *Scrum.*
Scrumia	von Wangenheim *et al* [2013]	Papel e caneta	Os jogadores devem planejar e executar uma *Sprint*, em um projeto hipotético.
Scrum Game	Gkritsi [2011]	Computador	O sistema guia os alunos pelo ciclo de vida do *Scrum*, implementando projetos.
Scrum Simulation with LEGO Bricks	Krivitsky [2009]	Papel, caneta e LEGO	Os jogadores, nas *Sprints*, constroem casas e veículos de LEGO a partir de histórias de usuário.
Scrumming	Neto [2008]	Computador	Foca na definição de uma *Sprint* e na simulação de sua ocorrência.

Os trabalhos encontrados, em geral, focam no aprendizado de conceitos-chave do *Scrum.* Dois desses jogos, *'Scrumming'* e *'Scrum Game',* possuem ênfase na prática da metodologia ágil, entretanto limitam a experiência do aluno a algumas condições. No jogo *'Scrumming'* uma dessas condições é a escolha automática do recurso a ser endereçado no desenvolvimento. Essa tarefa, em situações reais, é feita pelo *Scrum Master*. Outro ponto importante a ser mencionado é que a essa ferramenta permite a simulação de apenas um projeto por vez e não permite o salvamento de histórico, ou seja, uma vez iniciada a simulação, a mesma deve ser realizada até o seu fim. Já o jogo *'Scrum Game'*, apesar de permitir a realização de mais de uma *Sprint*, limita sua duração em dois dias. O papel de *Product Owner* também não é acessível aos alunos, cabendo exclusivamente ao administrador. No jogo educacional desenvolvido por esse trabalho as limitações mencionadas são eliminadas, a fim de que as condições fornecidas pelo jogo sejam as mais próximas possíveis das que ocorrem em situações reais.

3. Metodologia

O desenvolvimento desse trabalho foi baseado na metodologia INTERA, que é recomendada para o desenvolvimento de qualquer conteúdo digital que possa ser reutilizado na educação (Braga, et al [20012], Dotta *et al* [2012]) A seguir um detalhamento de cada uma das etapas da metodologia dentro do contexto do desenvolvimento do jogo educacional.

A etapa de Gestão de Projetos definiu postos-chaves para orientar a realização do trabalho, como papel dos envolvidos e tempo de desenvolvimento. O custo inicial estimado foi igual a zero, uma vez que não foram utilizados equipamentos especiais nem softwares pagos e o tempo da equipe também não foi remunerado. A tabela 2 mostra os membros da equipe envolvidos na construção do jogo e os papéis desempenhados por eles em concordância com a metodologia INTERA.

Tabela 2. Relação dos membros da equipe e respectivos papéis.

Membro	Papéis
Aluno	Programador, arquiteto, testador, analista

Professor	Coordenador, analista, designer pedagógico, demandante, conteudista

Na etapa de Contextualização foi levantamento o contexto pedagógico em que o jogo deveria ser aplicado. O público-alvo definido nessa etapa foram alunos de graduação cursando disciplinas relacionadas à ES. Na etapa Requisitos foram levantados requisitos pedagógicos e técnicos, tudo isso baseado no que foi definido etapa de contextualização. Os requisitos técnicos foram separados em duas categorias: geral e avaliação. A categoria geral indica os pontos de partida para guiar o desenvolvimento do jogo. Já a categoria avaliação indica os critérios a serem avaliados na determinação da pontuação. Os requisitos pedagógicos foram baseados em um mapa conceitual do *Scrum*. Esse mapa serviu de entrada de dados para outras etapas, e o seu principal papel foi na hora de relacionar os conceitos do *Scrum* com cada funcionalidade do jogo e garantir que nenhum conceito ficaria de fora. Além de contribuir para o aprendizado da própria equipe de desenvolvimento, fato este importante para garantir a integridade conceitual do produto final.

Uma vez que o software desenvolvido possui cunho acadêmico e também um viés de jogo, houve a necessidade de basear o design do software unindo a área de design instrucional e design de jogos, sendo esse um diferencial do trabalho. Essa união de áreas é sugerida e permitida pela metodologia INTERA. Assim, durante a etapa de Design, a modelagem conceitual foi baseada no mapa conceitual mostrado pela Figura 1, que apresenta o conteúdo abordado pelo jogo. Visando o reuso dos componentes do jogo, nesta etapa foi realizada a modelagem UML do software.

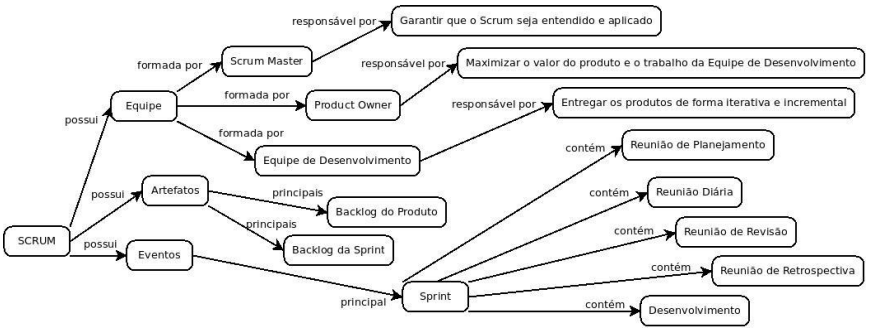

Figura 1. Mapa conceitual do conteúdo abordado pelo jogo.

Ainda na etapa de design foi realizada a modelagem da jogabilidade. Esta pode ser entendida como a maneira em que o jogador interage com a mecânica do jogo. A tabela 3 apresenta os elementos da jogabilidade para o jogo educacional desenvolvido.

Tabela 3. Elementos que fazem parte da jogabilidade.

Elemento	Descrição
Jogadores:	Equipes formadas por alunos que desempenham um papel na equipe *Scrum*.
Objetivo a ser alcançado:	Desenvolver o projeto proposto utilizando o *Scrum*, de forma a somar maior quantidade de pontos que outras equipes.
Regras do jogo:	Uma equipe desenvolve apenas um projeto; Um mesmo jogador não pode estar em equipes diferentes; O cliente que solicita o projeto é o professor; É permitido apenas um *Product Owner* e um *Scrum Master* por equipe; O *Product Owner* deve criar o

	Backlog do Produto, sob a condição de ganhar ou perder pontos com esse artefato *Scrum*; O *Scrum Master* deve definir as *Sprints* para o desenvolvimento do projeto; As reuniões são agendadas pelo *Scrum Master* durante a *Sprint* corrente; A não participação da equipe ou de algum integrante da equipe em alguma reunião gera perda de pontos. Ao final de cada *Sprint* deve ser entregue um módulo executável, com os requisitos do *Backlog* do Produto prometidos para essa *Sprint*; A pontuação final é dada pelo professor.
Condição de vitória:	Somar maior número de pontos do que outras equipes participantes.
Condição de derrota:	Somar menor número de pontos do que equipe vencedora.

Na etapa de Desenvolvimento foi iniciada a implementação do software, sendo utilizadas as seguintes tecnologias: linguagem de programação Java para web, com os frameworks JSF 2.0, Primefaces 3.4 e Hibernate 3.5.1, a IDE Eclipse 3.7.2, o banco de dados MySQL 5.5.24, e o servidor GlassFish 3.1.2. A etapa de Testes e Qualidade ocorreu paralelamente a etapa de Desenvolvimento sendo realizados testes de funcionalidade para todo o jogo.

A etapa de Disponibilização do software está em andamento nesse trabalho, uma vez que está sendo gerada a documentação de uso e a instalação do jogo.

Na etapa de Testes de qualidade foram realizados testes funcionais e testes de usabilidade ainda estão sendo realizados. Uma simulação, considerando alunos e professores fictícios, foi realizada. Também nessa etapa foi realizada a validação de conceitos, ou seja, um comparativo entre os conceitos abordados em cada tela do jogo com os conceitos presentes no mapa conceitual (Figura 1).

A etapa de Avaliação, na qual o jogo deverá ser experimentado em sala de aula, não faz parte do escopo desse artigo, mas será realizada em um trabalho futuro.

4. Resultados

No jogo, inicialmente o professor, que desempenha o papel de um cliente no *Scrum*, deve propor um projeto a ser trabalhado pelos alunos. Estes, na figura do *Product Owner*, devem elaborar o *Backlog* do Produto. O *Scrum Master* deve então definir a quantidade de *Sprints* e seus prazos e agendar as reuniões. Cada *Sprint* agendada deve ser iniciada pela Reunião de Planejamento da *Sprint*. Ao término dessa atividade deve ser criado o *Backlog da Sprint*. Durante a fase de programação devem ser feitas as Reuniões Diárias. Ao final, devem ser realizadas as Reuniões de Revisão e de Retrospectiva da *Sprint* e o *Product Owner* deve entregar o módulo gerado para o cliente, que deve avaliá-lo. Caso o projeto não esteja terminado o ciclo deve ser repetido. Caso contrário o professor deve realizar a avaliação final. A fim de que todas essas atividades fossem realizadas dividiu-se o jogo em duas áreas de acesso: professor e aluno.

4.1. Área de acesso do professor

Na área de acesso do professor, mais especificamente na área de projetos, o usuário dessa categoria pode formalizar um projeto a ser desenvolvido, cadastrando um arquivo com a especificação do que deve ser desenvolvido. Ele também pode cadastrar vídeos e links para auxiliar o entendimento da proposta, conforme mostra a Figura 2.

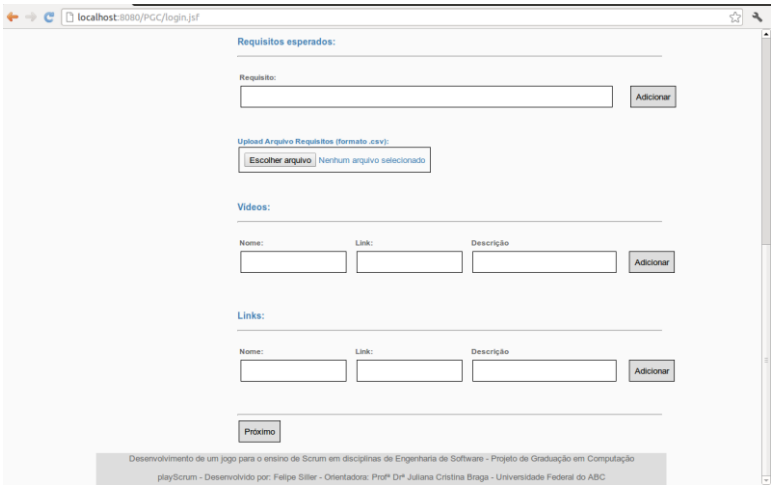

Figura 2. Parte da interface para o professor cadastrar um projeto.

A fim de associar projeto e equipe foi criado o conceito de grupo. Para criar um grupo o professor escolhe um projeto a partir de sua lista de projetos cadastrados e também as equipes, cadastradas pelos próprios alunos, que desenvolverão esse projeto. O grupo, dessa maneira, dá liberdade ao professor para reaproveitar um mesmo projeto para diferentes salas, além de permitir escolher a dificuldade dos projetos de acordo com a classe / equipe.

Outra funcionalidade do software educacional disponibilizada na área do professor é a avaliação. Nela é realizada a avaliação das equipes durante as *Sprints* bem como suas avaliações finais. O sistema do software é responsável por verificar a realização das diversas reuniões agendadas para a *Sprint* e descontar pontos de acordo com os critérios de avaliação definidos, caso seja necessário.

4.2. Área de acesso do aluno

Na área de acesso do aluno, logo após ter feito seu cadastro, o usuário da categoria aluno pode optar por criar sua própria equipe, ou entrar em uma já existente. Em ambos os casos ele deve optar entre os papéis de *Scrum Master*, *Product Owner* ou Equipe de Desenvolvimento, respeitada a restrição de haver somente um usuário por equipe no papel de *Scrum Master* e um no papel de *Product Owner.*

Para acesso às interfaces que são disponibilizadas na área do aluno, o jogo fornece um menu lateral esquerdo, listando todas as opções. Apesar de todos os integrantes da equipe *Scrum* terem acesso a esse menu, o redirecionamento a determinadas interfaces é condicionado ao papel do jogador dentro do *Scrum*. Dessa forma, se um aluno tentar realizar uma tarefa na qual seu papel não é o responsável, ele será direcionado a uma interface informando que a tarefa, no *Scrum*, não é realizada por seu papel. Além disso, essa interface apresenta informações teóricas a respeito dos papéis e atividades no *Scrum*, fornecendo elementos para que o aluno possa aprofundar seus conceitos teóricos sobre essa metodologia ágil. Entretanto, essa funcionalidade ainda está em desenvolvimento.

No menu esquerdo, também são disponibilizados ícones, de cor azul com a letra

"i" ao centro, localizados ao lado direito do nome das áreas do jogo. Esses ícones são links que direcionam o usuário a uma interface com informações teóricas a respeito dessa área no *Scrum*, sendo mais um dos elementos educacionais disponibilizados pelo jogo desenvolvido. Os ícones podem ser visualizados na Figura 4.

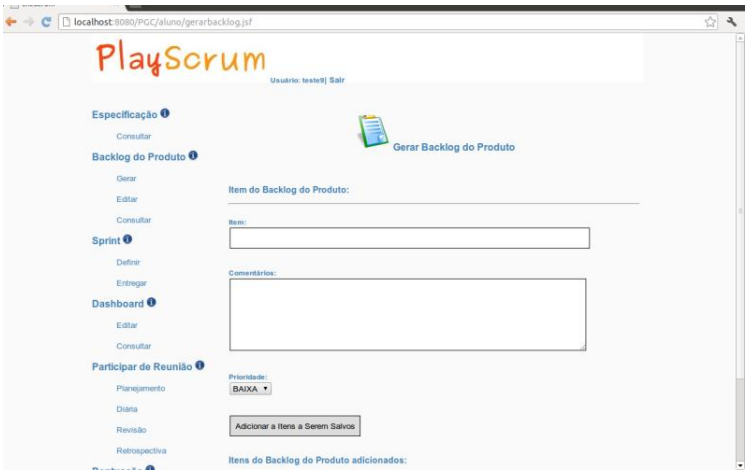

Figura 4. Parte da interface para o aluno cadastrar os itens do *Backlog* do Produto, com destaque para os ícones azuis no menu esquerdo.

5. Análise comparativa entre os jogos relacionados e o jogo desenvolvido

Os jogos educacionais desenvolvidos pelos trabalhos relacionados não abordam alguns aspectos do *Scrum* de forma completa, como mostra a tabela 4. Isso pode distanciar as situações promovidas pelos jogos da prática da metodologia ágil no mundo real. O jogo '*Scrumia'*, por exemplo, apesar disponibilizar diversos recursos não permite que mais de uma *Sprint* seja executada para um mesmo projeto, podendo limitar a dificuldade do mesmo a projetos simples, ou ainda limitar o planejamento da equipe. O jogo desenvolvido, '*Playing Scrum'*, busca preencher os itens faltantes nos demais jogos, de modo que as situações e funcionalidades fornecidas se aproximem o máximo possível daquelas encontradas no mundo real.

Tabela 4. Itens do *Scrum* presentes nos jogos

Item	PlayScrum	Scrumia	Scrum Game	Scrum Simulation with LEGO Bricks	Scrumming	Playing Scrum
Jogadores podem escolher entre os três papéis do *Scrum*		x				x
Jogadores realizam todas as reuniões previstas no *Scrum*		x	x	x		x
Jogadores elaboram *Backlog* do Produto		x		x		x
Jogadores gerenciam a equipe	x	x	x		x	x
Possibilidade de executar mais de uma *Sprint*	x		x	x		x

6. Considerações finais

O desenvolvimento desse jogo educativo para a prática de *Scrum* poderá contribuir para o ensino prático dessa metodologia, fazendo uso das vantagens proporcionadas aos alunos quando é utilizado esse tipo de software. Pelas funcionalidades fornecidas e interações permitidas o jogo desenvolvido pode inclusive ser utilizado em disciplinas à distância.

A aplicação da metodologia INTERA evidenciou a diferença no desenvolvimento de softwares convencionais e desenvolvimento de objetos de aprendizagem. Dentre essas diferenças, pode-se citar: i) Necessidade de avaliar o contexto educacional em que o jogo será inserido antes de iniciar o desenvolvimento. ii) A definição dos requisitos técnicos é mais simples do que em softwares de natureza não educacional, já que o software educacional, em geral, possui um escopo mais reduzido, caso contrário poderá confundir o aprendizado do aluno. iii) Necessidade de não perder o foco no aprendizado durante o desenvolvimento, o que foi permitido através do uso dos mapas conceituais; iv) Foi fundamental durante esse processo ter uma equipe que conhecia os conceitos de *Scrum* envolvidos no jogo, destacando assim o importante papel do professor; v) Necessidade de unir design de jogos com design instrucional, neste caso, a análise de contexto e definição da jogabilidade guiaram todo o processo de desenvolvimento e foram fundamentais para a definição das funcionalidades e restrições de acesso de usuários.

Como trabalho futuro pretende-se avaliar a eficiência do aprendizado do jogo em sala de aula. Esse é um importante trabalho futuro que evidenciaria a validade do jogo proposto e também poderia indicar possíveis pontos que devem ser revisados. Outro trabalho que pode ser desenvolvido é o estudo de acessibilidade do jogo e propostas de modificações para que atenda também a usuários portadores de deficiência visual.

Referências

Adams, E. (2010) “Fundamentals of Game Design”, 2ª edição, New Riders, 2010

Baker, A.; Navarro, E.O.; van der Hoek. (2003), A. “An experimental card game for teaching software engineering”, Software Engineering Education and Training, Proceedings. 16th Conference on Volume , Issue , 20-22 March 2003 p. 216 – 223.

Braga, J. C. (2009) “Diretrizes para o Ensino Interdisciplinar de Engenharia de Software”. Anais do FEES09 - Fórum de Educação em Engenharia de Software, Fortaleza, Outubro.

Castro, J. F. B, Gimenes, I.M.S, Maldonado,J.C. (2000), “Uma proposta de Plano Pedagógico para a matéria Engenharia de Software”. In: II curso de qualidade de cursos de graduação da área de Computação e Informática, Curitiba, pp. 251-270.

Dotta, S., Jorge, E., Braga, J.,; Pimentel, E. (2012) “Relato de Experiência: Processo de Elaboração de um Curso à Distância Utilizando a Metodologia Intera”. ESUD 2012 – IX Congresso Brasileiro de Ensino Superior a Distância. Recife – PE.

Feitosa, A. C. (2010) "AprendES: um jogo educacional para auxiliar o processo de ensino-aprendizagem da Engenharia de Software". Trabalho de Conclusão de Curso de Ciência da Computação. Universidade do Estado do Rio Grande do Norte.

Fernandes, J.M., Sousa, S.M. (2010). "PlayScrum – a card game to learn the scrum agile method". In: Proc. of Second International Conference on Games and Virtual Worlds for Serious Applications, Braga, Portugal

Fernandes, L., Werner, C. M. L. (2009) "Sobre o uso de Jogos Digitais para o Ensino de Engenharia de Software**"**. Universidade Federal do Rio de Janeiro, Rio de Janeiro.

Gkritsi, A. (2011) "Scrum Game: An Agile Software Management Game". Master Thesis, University of Southampton, Electronics and Computer Science, Great Britain

Hazzan,O., Dubinsky,Y. (2003) "Teaching a Software Development Methodology: The case of Extreme Programming", Proceedings of the 16th Conference on Software Engineering Education and Training (CSEE&T 2003), Espanha, p. 176-184.

Krivitsky, A. (2009) "Scrum Simulation with LEGO Bricks". http://agileee.org/wp-content/uploads/2011/12/Scrum-Simulation-with-LEGO-Bricksv2.0.pdf

Neto, E. I., (2008) "Scrumming Ferramenta Educacional para Ensino de Práticas do SCRUM" Pontifícia Universidade Católica do Rio Grande do Sul.

Possa, R. M. (2011) "Um estudo sobre os requisitos de jogos de simulação usados no ensino de Engenharia de Software". Dissertação de mestrado. Departamento de Ciência da Computação da Universidade Federal de Minas Gerais.

Sargent, J. (2004) "An Overview of Past and Projected Employment Changes in the Professional *IT* Occupations". Computing Research News, 16, 3, May, p. 1 - 21.

Soares, M. S. (2008) "Uma experiência de ensino de Engenharia de Software orientada a trabalhos práticos". Fórum de Educação em Engenharia de Software (FEES), Campinas – SP, Outubro.

Tao W., Qing Z., (2009), "A Software Engineering Education Game in a 3-D Online Virtual Environment", vol. 2, pp.708-710, First International Workshop on Education Technology and Computer Science.

von Wangenheim, C. G., Savi, R., Borgatto, A. ,F. (2013) "SCRUMIA—An educational game for teaching SCRUM in computing courses". J. Syst. Software, http://dx.doi.org/10.1016/j.jss.2013.05.030