Um jogo para o ensino de programação em Python baseado na taxonomia de Bloom

Pasqueline Scaico, Diego Lopes Marques, Leandro de Almeida Melo, Max André Azevedo, Sinval V. Mendes Neto, Anderson Oliveira, Josinaldo Alves Júnior, Marcelo Labanca, Alexandre Scaico[1]

Centro de Ciências Aplicadas e Educação – Universidade Federal da Paraíba (UFPB) - Rua da Mangueira, s/n - CEP 58.297-000 – Rio Tinto – PB – Brasil

{pasqueline, diego.lopes, leandro.almeida, max.azevedo, sinval.mendes, alexandre)@dce.ufpb.br

Abstract. One of the key elements for the teacher introductory programming courses is evaluation system used. Also, resources like educational games should be used to challenge students. However, the design of games needs to be supported by pedagogical theory, like Bloom's Taxonomy, which can enable the measurement of learning outcomes. This paper presents an educational game that aims to teach concepts of Python and programming logic, being for the teacher a tool for assessing student learning.

Resumo. Avaliação é um elemento muito importante para o ensino de programação. Por outro lado, melhores recursos, a exemplo de jogos educativos, devem ser utilizados para que os alunos sejam desafiados. Todavia, a concepção de jogos precisa estar respaldada em teorias de aprendizagem, como a taxonomia de Bloom, que permitem possibilidades de medir os resultados de aprendizagem. Este trabalho apresenta um jogo educativo que objetiva o ensino de programação em Python, ao mesmo tempo em que serve para o professor como uma ferramenta de avaliação.

1. Introdução

No mundo inteiro, muitas empresas têm investido em tecnologias que viabilizem o treinamento e o desenvolvimento de competências. O ensino à distância e o treinamento baseado em simulação são exemplos de soluções empregadas nos campos militar, da indústria e dos negócios.

Os jogos também vêm sendo explorados em muitos contextos. Hays (2005) diz que *"um jogo representa uma atividade competitiva, que tem um objetivo bem definido, que é determinado por um conjunto de regras e restrições e que está situado em um contexto específico*". O desenvolvimento de jogos com fins instrucionais tem recebido o investimento de grandes indústrias e a atenção de pesquisadores em função de ser uma alternativa acessível e de baixo custo (Smith et al 2007). Agora, se por um lado, o uso de jogos na educação tem sido relatado como positivo em função de engajar os estudantes em novas experiências de aprendizagem e estabelecer um ambiente mais estimulante, como afirma Prensky (2001) e Blunt (2007), por outro lado, a eficiência dos seus resultados tem sido questionada. Wilson et al (2009) traz um alerta em relação à insuficiência de evidências científicas que comprovem quais os atributos de um jogo são capazes de gerar resultados para a aprendizagem. Van Staalduinen (2011) alerta que

é preciso estabelecer uma dialética entre a dimensão pedagógica e os elementos de um jogo, caso contrário o jogo pode se tornar apenas um mecanismo de diversão, problemática também apontada no trabalho de Thomas (2003). Yue (2009) aponta que falhas no projeto da interface de um jogo educacional pode torná-lo desinteressante e incapaz de reter o jogador tempo suficiente para que ele aprenda.

No ensino de programação o uso de jogos tem sido uma prática comum. Inúmeros trabalhos registram a utilização de jogos de raciocínio lógico como um mecanismo auxiliar para o desenvolvimento de competências de programação, por exemplo. Porém, alguns deles são frágeis em atender requisitos relacionados ao projeto instrucional, assim como, dos requisitos de *game design*. Considerando as dificuldades existentes no processo de aprender a programar, a existência de recursos de qualidade que possam instruir e motivar os iniciantes é muito importante.

Este trabalho apresenta o relato sobre a construção de um jogo para o ensino de Python, cujo desenvolvimento está respaldado em teorias de aprendizagem, e que persegue características de *game design*. Além disso, o jogo está sendo projetado para também ser um mecanismo de avaliação para o professor.

O artigo está organizado da seguinte maneira: a Seção 2 apresenta uma revisão da literatura sobre as questões relacionadas à importância de incorporar no projeto de jogos educativos, atributos provenientes do *game design* e do *design instrucional.* Para ilustrar a complexidade em desenvolver jogos com essa coesão, na Seção 3 são discutidas as características de alguns jogos voltados para o ensino de programação. Na Seção 4 os detalhes da concepção do jogo, seu processo de desenvolvimento e sua arquitetura são apresentados. Na Seção 5, o estado atual do trabalho e algumas considerações finais são apresentados.

2. Jogos e aprendizagem

No processo de desenvolvimento de um jogo, o *game design* é a etapa que diz respeito à especificação das características e do conteúdo do jogo que se apresentarão através da interface, e que colaboram para a garantia do envolvimento do jogador (Kiili 2005). Um exemplo da importância do *game design* é ressaltado por Csikszentmihalyi (1990) ao afirmar que se o desafio do jogo estiver aquém das habilidades do jogador, ele ficará entediado, porém se estiver além, ele ficará frustrado e desistirá do que lhe foi proposto.

Para que se estabeleça o nível ideal de estímulo, chamado de *fluxo de experiência*, o jogo deve apresentar objetivos claros e *feedback*. Um estudo apresentado por Van Staalduinen (2010) indica que 25 elementos de *game design* podem ser importantes para acelerar ou prolongar a memória sobre uma instrução, o que pode gerar um efeito positivo para a aprendizagem. Todavia, Garris (2002) aponta seis atributos essenciais para potencializar o aprendizado: 1. fantasia (que envolve o imaginário do jogador, fazendo com que ele imagine situações e assuma papéis com os quais se identifica), 2. representação (que representa a fidelidade do ambiente ou da situação que o jogo se propõe a reproduzir), 3. controle (que é capacidade dada ao jogador para controlar situações do jogo), 4. estímulo sensorial (se refere à provocação dos sentidos do jogador), 5. desafio (que estabelece a tensão motivacional no jogador, ou seja, a criação de incertezas em relação ao alcance das metas) e 6. avaliação (medição do desempenho do usuário – pontuação, precisão, classificação diante de outros jogadores e a visão sobre as ações do jogador).

Na literatura, o uso de jogos na educação fomenta uma grande discussão. Alguns autores afirmam que os jogos colaboram numa perspectiva informal para a aprendizagem, e não intencional. Conforme relata Burgos et al (2007), *"[...] não está claro o que um estudante aprende a partir de um jogo, ou qual é a influência que os facilitadores individuais do jogo têm nos resultados de aprendizagem"*. Assim, algumas abordagens têm sido propostas para incorporar características de jogos eletrônicos em aplicações com fins educacionais, para que se possa tornar o software envolvente e didático, aumentar a memória do aprendizado e, principalmente, para que seja possível estabelecer métricas que dimensionem os efeitos de aprendizagem.

Muitos trabalhos mostram que o desenvolvimento de jogos educacionais deve considerar exploração, emoção e envolvimento, e que desafios complexos deveriam ser reforçados por diálogos com o jogador. Kiili (2005) apresenta um modelo que reforça a aprendizagem experimental, que é aquela que define pequenos ciclos de aprendizagem, como uma maneira de manter o *fluxo de experiência.* Já de Freitas (2006), através do seu framework quadri-dimensional agrega elementos pedagógicos e de *game design* através de uma abordagem de especificação centrada nos *stakeholders* do jogo. Van Staalduinen (2011) congrega os elementos mais importantes de outras teorias na composição do seu processo. Após a definição dos objetivos de aprendizagem, o projeto do *design instrucional*, onde são planejados os ciclos de aprendizagem, incorporam alguns elementos do *game design*, como é o caso da representação e avaliação (*feedback*), para favorecer, por exemplo, o engajamento do jogador.

A aprendizagem é o processo de transformar experiências em conhecimento. Diversos teóricos da aprendizagem, como Bloom (1956), advogam que o saber acontece a partir da organização das capacidades em níveis cognitivos. A hierarquia é cumulativa, sempre do domínio cognitivo mais simples para o mais complexo. Várias revisões aconteceram na taxonomia proposta por Bloom. O trabalho de Jesus (2010) apresenta uma proposta da adequação da taxonomia para o ensino de programação, bem como os verbos que podem ser explorados nas avaliações construídas pelo professor (lembrar, entender, aplicar, analisar, avaliar e criar).

Assim, acredita-se que a concepção de um jogo educativo deve atender requisitos que ajudem o aprendiz alcançar certos objetivos pedagógicos. Porém, se estes não estiverem bem determinados, o resultado da aprendizagem pode ser subjetivo e reduzido a fatores apenas relacionados à diversão.

3. Trabalhos relacionados: jogos para o ensino de programação

Muitos trabalhos registram o uso de jogos de raciocínio lógico como apoio ao ensino de algoritmos (Rapkiewicz 2006). Neste contexto, aplicações com a finalidade de prover a simulação da execução do código também são muito utilizadas. Este é o caso do Robocode (Santos 2002), CeeBot[1] e Furbot (Vahldick 2009).

O Robocode é um jogo multiusuário em que os competidores escrevem códigos para controlar um robô que luta com outros robôs, também programados. Nas duas outras aplicações se propõe uma forma interessante de simular as decisões tomadas por um programador a partir das ações de um robô, que deve cumprir algumas missões do jogo. No jogo ProGame, o jogador é conduzido por um ambiente em que pode desenvolver suas habilidades para solucionar problemas (Dantas 2011). O WuCastle é

[1] http://www.ceebot.com/ceebot/index-e.php

um jogo de RPG onde o jogador cria de maneira iterativa exércitos de bonecos de neve. O jogo fornece retorno imediato e ajuda o jogador a visualizar a execução de seu código (Eagle 2008).

Os jogos mencionados podem motivar o estudo de alguns conceitos de programação, mas possuem fragilidades do ponto de vista instrucional, já que poucas informações são fornecidas para que o jogador aprenda enquanto joga. Nenhum dos jogos possui também qualquer mecanismo de auto regulação, que é um elemento importante para que o jogador assuma o ritmo do seu processo de aprender. Nesta perspectiva, o jogo TALENT (Maragos 2007) apresenta uma proposta baseada em um processo de tutoria em que agentes pedagógicos fornecem dicas e guias de ajuda durante os desafios que aparecem. Todavia, este jogo requer a aprendizagem de uma nova linguagem de programação, própria do ambiente. Sobre este assunto muitos autores afirmam que o uso de várias sintaxes em cursos de introdução à programação não é uma prática adequada, e sugerem, inclusive, que a linguagem de programação deve possuir sintaxe simples. Tanto o Robocode quanto o Furbot exploram os conhecimentos que o jogador possui sobre a linguagem Java. O Ceebot utiliza uma linguagem própria, semelhante à linguagem C#, para que o jogador realize a implementação.

Para ilustrar o nível de conformidade em relação ao *game design* e ao *design instrucional* dos seis jogos educacionais mencionados, uma análise foi realizada pela equipe de desenvolvimento do jogo que é tema deste artigo. A avaliação se baseou na análise dos critérios apontados no modelo de Van Staalduinen (2011) e na identificação dos níveis de aprendizagem de Bloom que são atendidos nos jogos.

A metodologia utilizada para a avaliação dos jogos na perspectiva de *game design* aconteceu em duas etapas. Na primeira delas, seis avaliadores participaram. Cada um deles avaliou um mesmo critério para os seis jogos, indicando a presença ou ausência daquele atributo nos mesmos. Na segunda etapa os seis avaliadores se reuniram e cada um deles apresentou aos demais o resultado da avaliação e os motivos que os levaram as suas conclusões. A equipe dialogou na intenção de dirimir possíveis inconsistências. A avaliação dos critérios relacionados à taxonomia de Bloom foi realizada também neste momento, em grupo. A síntese dos resultados é apresentada na Tabela 1. Com base na avaliação preliminar realizada, o jogo aqui descrito procura atender três perspectivas: pedagógica, de jogabilidade e avaliativa.

Do ponto de vista pedagógico, o jogo propõe um conjunto de recursos instrucionais (textuais, visuais e auditivos) que auxiliarão o aprendiz a entender aspectos de programação e da linguagem Python. Além disso, o nível das instruções é progressivo de acordo com os níveis da hierarquia de Bloom. Para assegurar o envolvimento do usuário, características de jogabilidade que preservem, principalmente, o desafio serão implementadas. Por fim, as decisões tomadas pelo jogador serão transformadas em informações importantes para o professor, logo podendo o mesmo ser uma ferramenta útil para apontar a evolução na aprendizagem da turma. A abordagem do jogo, orientada a desafios, permite também que o professor o incorpore como um recurso pedagógico para as suas aulas.

Tabela 1. Análise dos jogos

Atributo	Ceebot	Furbot	Robocode	WuCastle	ProGame	TALENT
Fantasia				✓	✓	
Representação	✓					
Estímulo sensorial			✓	✓	✓	✓
Desafio	✓					
Avaliação				✓		✓
Controle	✓	✓	✓	✓		✓
Bloom						
Lembrar						
Entender					✓	
Aplicar				✓	✓	
Avaliar				✓		
Criar	✓	✓	✓			

Jogo	Objetivo pedagógico
Ceebot	Executar simulação de algoritmos
Furbot	Executar simulação de algoritmos
Robocode	Executar simulação de algoritmos
WuCastle	Executar simulação de algoritmos
ProGame	Testes de mesa e interpretação de algoritmos
TALENT	Reforçar a sintaxe do C, executar algoritmos com matrize

Jogo	Linguagem de Programação
Ceebot	Similar a C
Furbot	Java
Robocode	Java
WuCastle	C
ProGame	Java e Python
TALENT	Linguagem própria

4. A proposta do jogo

A proposta de desenvolver este jogo surgiu a partir de várias tentativas em melhorar o ensino de programação em um curso de Licenciatura em Computação. Além de fatores individuais que influenciavam o desempenho discente, percebeu-se a fragilidade dos mecanismos de avaliação para medir o desempenho dos alunos, fato presente em muitos cursos de Computação em todo o mundo, como registra o trabalho de Whalley et al (2006).

Para motivar mais os estudantes a estudarem programação, o uso de jogos também foi incentivado. Todavia, percebeu-se a existência de poucos softwares educativos para o ensino de programação em Python. Dessa maneira, iniciou-se a produção de um jogo educativo com este fim, que pudesse ser interessante para todos os estudantes da turma, já que o projeto instrucional visa explorar cada conteúdo curricular de diferentes maneiras.

Dessa maneira, o jogo é organizado por bloco de desafios. Cada bloco explora um conteúdo a partir de seis perspectivas distintas, que são compatíveis com as habilidades cognitivas mencionadas pela taxonomia de Bloom. Assim, há desafios em cada bloco que exploram a habilidade de lembrar, outros de entender, e assim, sucessivamente. É importante ressaltar que o *game* possui características que o diferenciam da grande maioria dos jogos educacionais para ensino de programação. Nos desafios estão presentes elementos e mecanismos que favorecem o ensino de algoritmos e também da linguagem de programação Python. Além disso, os desafios são curtos, o que significa que podem ser cumpridos no tempo de uma aula, se assim o professor desejar. Ao final da execução do desafio o jogo envia um relatório de desempenho dos alunos, que poderá auxiliar o professor a perceber as dificuldades de aprendizagem remanescentes.

4.1. Arquitetura do jogo

A arquitetura do jogo está organizada em duas camadas lógicas: apresentação e negócio. O jogo é executado via Web. Na camada de apresentação estão os componentes da interface, que executam através de um *browser*. Ao completar um desafio, um arquivo, que contém o registro dos principais eventos do comportamento do jogador, é gerado para ser tratado na camada de negócio. Nesta camada, um módulo chamado Gerente de avaliação realiza a leitura do arquivo e agrupa as decisões tomadas pelo jogador (um mecanismo simples de autenticação é utilizado). Um componente chamado Gerente de relatório é responsável por montar os relatórios (de desempenho individual ou por turma) que serão enviados ao professor. Para desacoplar a forma de envio, um Gerente de envio está sendo implementado.

Dois outros componentes são responsáveis por características importantes do jogo. Um deles é o Gerente de feedback. Para atender requisitos relacionados ao *game design*, informações relacionadas à pontuação, classificação do jogador em relação aos outros jogadores e informações de ajuda são gerenciadas por este módulo. Por fim, o Gerente de configuração permite que o professor alimente o jogo com elementos técnicos para os desafios e também, que defina *metadados* para estes desafios, os quais também importantes para o processamento realizado pelo Gerente de avaliação.

A visão física da arquitetura do jogo está organizada em três camadas e está ilustrada na Figura 1. O jogo é executado no lado servidor através de uma *engine* para jogos, a Construct 2. O jogador acessa o *game* através de um navegador web e, após a sua autenticação, tem acesso aos desafios disponíveis. Neste modelo descentralizado, que difere de um jogo de fases, o aluno é quem escolhe os desafios que deseja cumprir, fazendo com que tenha controle sobre o ritmo de instruções que deseja receber e da complexidade dos desafios que deseja resolver.

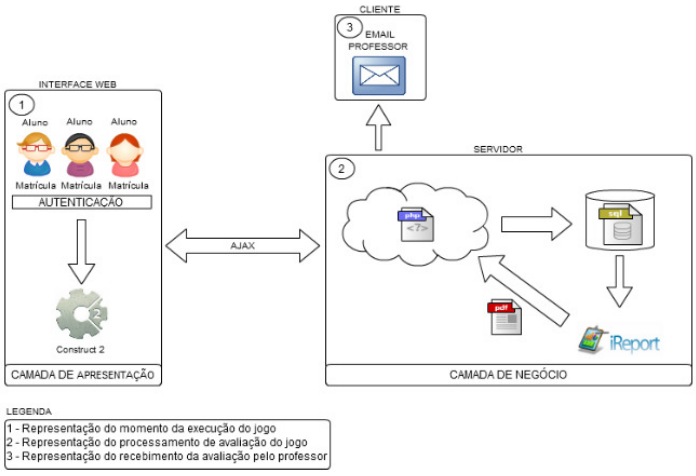

Figura 1. Arquitetura física do jogo

Ao completar um determinado desafio, o registro de alguns eventos do jogo, a exemplo da quantidade de acertos e erros do jogador, é enviado através de uma requisição, via AJAX, para ser tratado por uma aplicação em PHP (onde estão implementados os gerentes da camada de negócio), que recebe os registros, faz o seu tratamento, e os envia para um banco de dados MySQL.

Quando a turma de alunos completa um desafio, outro evento é disparado pela *engine* para que o relatório seja gerado. Ao perceber esse novo evento, o servidor faz uma requisição para uma ferramenta de geração de relatórios, a iReport, que recupera as informações no banco de dados e as organiza em tabelas e gráficos, a fim de exportá-las em um documento no formato pdf. Finalizado este processo, o documento é reenviado ao servidor, que por sua vez, encaminha para o e-mail do professor.

4.2. Processo de desenvolvimento do desafio Entrando pelo cano

A equipe de desenvolvimento é composta por um professor, alunos da Licenciatura em Computação e alunos de um curso de Design. Para planejar um desafio, primeiro são definidos os objetivos de aprendizagem, o conteúdo que será tratado e o nível da taxonomia de Bloom que será utilizado para guiar o desafio. Em geral, o professor de programação conduz estas questões. Em seguida, a equipe inicia o *game design* que, através de um processo de *brainstorming* decide o cenário, personagens e regras, documentadas posteriormente através de *storyboards*.

A partir desses dois direcionamentos, inicia-se o planejamento do ciclo de aprendizagem do desafio, momento em que se planeja um roteiro instrucional para o desafio. Em geral, uma animação ou a narração de uma instrução antecede o início do desafio propriamente dito. Este recurso é interessante porque além de reforçar a informação ensinada pelo professor se pode agregar informações e curiosidades sobre Python. O processo de planejar este ciclo de aprendizagem procura contemplar sempre alguns atributos relacionados ao *game design* do jogo. Também se define nesta etapa os critérios de avaliação que alimentarão os relatórios que serão enviados para o professor e os efeitos na tela que deverão ser causados para que o jogador perceba o erro e aprenda a partir dele. Para efeito de ilustração, um dos desafios do jogo, que trabalha o conteúdo ***tipos de dados*****,** mostra-se descrito na Tabela 2, inclusive com os seus *metadados*.

Tabela 2. Requisitos do desafio Entrando pelo cano

Game Design	
Objetivo do desafio	O jogador é um operário de uma fábrica maluca. Várias caixas de mercadorias devem ser organizadas durante a noite.
Regras do jogo	Um grande sistema de engrenagem derruba as caixas que serão organizadas pelo operário. O jogador deve pegar uma caixa por vez, passá-la pelas esteiras, que estão quebradas e mudam de direção todo o tempo, e guardar a caixa corretamente nos canos de escoamento de mercadoria da fábrica. Se o operário tentar colocar uma caixa no cano incorreto o operário perde pontos *(score)*.
Atributos do jogo (fantasia, estímulo sensorial, *avaliação*, controle e desafio)	Cada cano recebe um tipo de caixa específica. Se o jogador se engana uma luz vermelha é acesa e um sinal sonoro é emitido para avisar o jogador que tem algo errado. O jogador precisa ser produtivo. Para isso deve correr contra o tempo e direcionar corretamente as caixas. O fato das esteiras estarem quebradas e sua posição mudar aleatoriamente dificulta a tarefa do operário, gerando um desafio interessante para o jogo.
Design Instrucional	
Conteúdo curricular	Tipos de dados e alocação de memória em Python
Instruções	O jogador é instruído, a partir de uma animação apresentada antes do desafio, que em Python a definição dos tipos das variáveis não é responsabilidade do

	programador. Além disso, é informado como o Sistema Operacional realiza a alocação de memória para que o conteúdo das variáveis seja armazenado durante a execução do programa. Ao final, é apresentada outra animação que mostra como outra linguagem, no caso, Java, realiza a alocação de memória.
Metáfora utilizada	O cenário da fábrica traz à tona o papel que é realizado pelo Sistema Operacional (SO) quando o programa em Python é executado. A cada instrução processada, o SO, que no desafio é representado pelo operário, precisa alocar na memória o espaço específico para cada tipo de dados e armazenar na área correta de memória o seu conteúdo.
Nível de Bloom	Nível 1: lembrar
Objetivo pedagógico	1. Reconhecer o tipo string, inteiro, float e boolean em Python 2. Informar o conceito de uma variável armazena um conteúdo por vez 3. Perceber a relação do SO com a alocação de memória
Feedback	Ao final do desafio o jogo apresentará como a memória ficou alocada. Caso o jogador tenha tentado alocar o conteúdo de uma variável incorretamente, o jogo mostrará uma explicação sobre aquele tipo de dado.

Na Figura 2 podem ser observadas algumas características do desafio, cujo objetivo pedagógico é reforçar o conhecimento sobre os tipos de dados no Python e alocação de memória, e que explora a capacidade lembrar, descrita na taxonomia de Bloom, útil também para prover a avaliação dos resultados de aprendizagem do jogador feita pelo sistema.

Na parte superior da tela, um bloco de instruções de um código em Python cai um a um, através de caixas. O jogador deve passar pelas esteiras (as setas indicam a direção em que elas estão funcionando no momento) e alocar a caixa no cano pelo cano correto. No exemplo, a caixa contendo o string "João" aparece sendo sugada pelo cano.

Figura 2. Tela do desafio Entrando pelo cano

Aqui o estudante é estimulado a trabalhar com os tipos da linguagem Python, sua sintaxe e de perceber como o Sistema Operacional colabora na alocação dos espaços de memória necessários ao funcionamento do código. É possível com este desafio iniciar a discussão sobre linguagens que são fortemente tipadas.

5. Considerações finais e trabalhos futuros

O jogo apresentado neste trabalho é um recurso indicado para disciplinas introdutórias de programação que utilizam a linguagem Python. Diferente de outros jogos para o ensino de programação, este jogo está sendo concebido para que o estudante possa praticar o desenvolvimento de algoritmos e aprender mais sobre o Python através de um projeto de software que considera elementos essenciais do *game design*. O jogo oferece várias possibilidades de desafios, que estão organizados em função da taxonomia de Bloom. Assim, ele será útil tanto para instruir, quanto para avaliar individualmente os alunos da turma, que assumem níveis de conhecimento diferentes ao longo da disciplina. Entende-se que o jogador deve ser um sujeito da sua aprendizagem. O jogo permite que cada aluno construa a seu tempo o conhecimento necessário para amadurecer a competência de desenvolver algoritmos.

Sabendo que um dos grandes desafios para o professor é a avaliação. Outra característica marcante do jogo é o fato de ser uma ferramenta potencialmente útil para auxiliar o professor a medir os resultados de aprendizagem de cada estudante da turma, inclusive, ainda durante a aula. Sendo possível para o docente definir estratégias e práticas de ensino mais eficazes.

Os trabalhos futuros para o jogo incluem a validação dos desafios já implementados, visando o refinamento do sistema de avaliação do jogo e das características referentes à jogabilidade, assim como, a implementação de novos desafios que contemplem mais níveis da taxonomia de Bloom. Também é meta do projeto desenvolver um esquema na interface do jogo que detalhe mais informações sobre os desafios, a exemplo do conhecimento prévio recomendado para jogar os desafios. Além do mais, alguns experimentos estão sendo montados para que se possa analisar o potencial instrucional dos jogos para iniciantes em programação, bem como sua contribuição para os resultados de aprendizagem alcançados.

Referências

Bloom, B. S. (Ed.). (1956). Taxonomy of educational objectives, Handbook 1: Cognitive domain. New York: David McKay.

Blunt, R. (2007). Does game-based learning work? Results from three recent studies. In Proceedings of the Interservice/Industry Training, Simulation, & Education Conference (pp. 945-955). Orlando, FL: National Defense Industrial Association.

Burgos D., van Nimwegen, C., et al. (2007). Game-based learning and the role of feedback: A case study. Advanced Technology for Learning.

Csikszentmihalyi, M. (1990). Flow: The psychology of optimal experience.New York: Harper.4.

Dantas, V.; Freitas, P., Alencar, L. . ProGame: Um jogo para apoiar o ensino-aprendizagem de programação. In: First Workshop on Applications to Provide Learning and Teaching Support (APPLETS), 2011, Aracaju – SE. Anais do XXII SBIE – XVII WIE, 2011.

de Freitas, S., Oliver, M. (2006). How can exploratory learning with games and simulations within the curriculum be most effectively evaluated? Computers & Education, 46, 249–26

Eagle M., Barnes T. (2008) Wu's castle: teaching arrays and loops in a game. Proceeding ITiCSE '08 Proceedings of the 13th annual conference on Innovation and technology in computer science education

Garris, R., Ahlers, R. & Driskell, J. E. (2002). Games, motivation and learning: A research and practice model. Simulation & Gaming, 33(4), 441-467.

Hays, R. T. (2005). The effectiveness of instructional games: A literature review and discussion (Technical Report No. 2005-004). Naval Air Warfare Center Training Systems Division, Orlando, FL.

Jesus E. A., Raabe A. L. A. (2009) Interpretações da Taxonomia de Bloom no Contexto da Programação Introdutória. Anais do XX Simpósio Brasileiro de Informática na Educação.

Kraiger, K., Ford, J., & Salas, E (1993) Application of cognitive, skill-based, and affective theories of learning outcomes to new methods of training evaluation. Journal of Applied Psychology, 78(2), 311-328.

Kiili K. (2005) Educational Game Design: Experiential gaming model revised. Computer and Information Science. Volume: 3, Publisher: Tampere University.

Maragos K., Grigoriadou M. (2007) Designing an Educational Online Multiplayer Game for learning Programming. Proceedings of the Informatics Education Europe II Conference. IEEII 2007.

Prensky, M. (2001). Digital game-based learning. New York: McGraw-Hill.

Rapkiewicz C. E., Falkembach G., Seixas L., Rosa N. S., Cunha V. V., Klemann M. (2006). Revista RENOTE – Novas Tecnologias na Educação, v. 4, n. 2.

Santos, A., Hamerski Jr., E. Robocode: Uma maneira simples e divertida de aprender java. Java Magazine,1 (3):43–45, 2002.

Smith, P., Sciarini, L., Nicholson, D. (2007) The utilization of low cost gaming hardware in conventional simulation. In Proceedings of the Interservice/Industry Training, Simulation, & Education Conference (pp. 965-972). Orlando, FL: National Defense Industrial Association.

Thomas S., Schott G., Kambouri M. (2003) Designing for learning or designing for fun? Setting usability guidelines for mobile educational games. MLEARN 2003 Learning with Mobile Devices (2003). Learning and Skills Development Agency, Pages: 173-181.

Van Staalduinen, Freitas J. P. (2011) A game-based learning framework: Linking game design and learning outcomes. Learning to Play: Exploring the Future of Education with Video Games. Springer.

Van Staalduinen, J. P. (2010). A first step towards integrating educational theory and game design. In P. Felicia (Ed.), Improving learning and motivation through educational games. Hershey, PA: IGI Global.

Yue W. S, Zin N. A. M. (2009) Usability evaluation for history educational games. Proceedings of the 2nd International Conference on Interaction Sciences Information Technology Culture and Human ICIS 09. ACM, pages: 1019-1025

Whalley, Jacqueline L. et al. (2006) "An Australasian Study of Reading and Comprehension Skills in Novice Programmers, using the Bloom and SOLO Taxonomies", In: VIII Australasian Computing Education Conference. (ACE2006),Computer Society, p. 243-252.

Wilson K. A, Bedwell W. L., Lazzara E. H., Salas E., Burke C. S., Estock J. L., Orvis K. L., Conkey C. (2009) Relationships Between Game Attributes and Learning Outcomes. Journal Simulation and Gaming.Volume 40 Issue 2.