15º Congresso de Inovação, Ciência e Tecnologia do IFSP - 2024

Evolução e avaliação de um objeto de aprendizagem gamificado para ensino de Computação

GUSTAVO G. CONTIERO[1], LUCAS B. R. OLIVEIRA[2]

[1] Graduando em Engenharia de Software, Bolsista PIBIFSP, IFSP, Campus São Carlos, contiero.gustavo@aluno.ifsp.edu.br
[2] Professor de Computação, IFSP, Campus São Carlos, lucas.oliveira@ifsp.edu.br.

Área de conhecimento (Tabela CNPq): 1.03.03.04-9 Sistemas de Informação

RESUMO: Em cursos de Computação no Brasil, dois em cada três estudantes não concluem a graduação, representando um problema sistemático e crítico para uma área que prevê um déficit de meio milhão de profissionais até 2025. Investigações científicas realizadas nos últimos anos indicam que as causas para a evasão são diversas, envolvendo problemas pessoais e financeiros, o nível de dificuldade dos cursos e questões relacionadas à didática dos professores. Nesse contexto, torna-se fundamental a incorporação de tecnologias que apoiem o ensino de Computação para engajar e motivar os estudantes durante o processo de aprendizagem. A gamificação tem recebido atenção por integrar elementos de jogos, que engajam os participantes, em contextos originalmente não lúdicos. A literatura já apresenta relatos de resultados positivos em diversas áreas da Computação, como o ensino de programação e teste de software. Este projeto visa aprimorar um jogo de cartas atualmente utilizado para o ensino de Computação no câmpus São Carlos, com o intuito de promover sua disseminação em outros cursos de graduação e em diferentes componentes curriculares. Além disso, pretende-se realizar avaliações para fornecer evidências sobre a eficácia do jogo no engajamento de estudantes de programação e teste de software.

PALAVRAS-CHAVE: Computação; Gamificação; Ensino; Programação;

Evolution and assessment of an gamified learning object to computing teaching

ABSTRACT: In Brazil computing courses, two out of three students do not complete their degree, representing a systematic and critical problem for an area that is expecting a deficit of half a million professionals until 2025. Scientific research carried out in recent years indicates that the causes of dropout are diverse, involving personal and financial problems, the level of difficulty of the courses and issues related to professors' didactics. In this context, it is essential to incorporate technologies that support computing teaching in order to engage and motivate students during the learning process. Gamification has received attention for integrating elements of games, which engage participants, into originally non-playful contexts. The literature already reports positive results in various areas of Computing, such as teaching programming and software testing. This project intends to improve a card game currently used for teaching Computing at the São Carlos Campus, with the goal of promoting its dissemination in other undergraduate courses and in different curricular components. In addition, evaluations will be carried out to provide evidence of the game's effectiveness in engaging programming and software testing students.

KEYWORDS: Computing; Gamification; Teaching; Programming;

INTRODUÇÃO

O Brasil apresentará um déficit de meio milhão de profissionais entre os anos de 2021 e 2025, resultando em um impacto estimado de 1,7% no Produto Interno Bruno - PIB (Google, 2023). Dentre

as causas desse cenário, está a grande taxa de evasão nos cursos superiores de tecnologia. Na última década, a taxa de evasão de cursos presenciais de Computação tem sido constantemente maior do que a média dos demais cursos (SEMESP, 2023). Segundo o Mapa do Ensino Superior, 2023 (SEMESP, 2023), a taxa de desistência acumulada média na área de Tecnologia da Informação - TI é de 65,5%. Estudos como o de Fukao *et al.* (2023) apontam que as principais dificuldades relatadas pelos desistentes estão relacionadas a questões pessoais ou associadas à instituição (curso, grade curricular, apoio estudantil e corpo docente), sendo que a principal dificuldade associada à instituição foi a didática dos professores. Outras dificuldades relatadas pelos desistentes incluem a falta de motivação pessoal, exigência das disciplinas, falta de foco e organização pessoal. Nesse contexto, torna-se fundamental a incorporação de tecnologias que apoiem o ensino de Computação, para engajar e motivar os estudantes durante o processo de aprendizagem.

A gamificação permite utilizar elementos baseados em jogos, como mecânica, estética ou pensamento de jogo em contextos originalmente não lúdicos, buscando engajar pessoas, motivar ações, aprimorar o aprendizado e/ou solucionar problemas (Deterding *et al.*, 2011). Essa abordagem foi prontamente adotada na educação devido ao seu potencial de estimular a motivação e o envolvimento dos estudantes em sala de aula. Dessa forma, a gamificação oferece oportunidades valiosas para os professores transformarem o ambiente de aprendizado, utilizando narrativas envolventes e sistemas de recompensas, para manter os estudantes imersos e motivados. Desde os anos 80, pesquisadores investigam os benefícios de abordagens baseadas em jogos na educação. No entanto, nos últimos anos, o interesse pelo tema aumentou em um ritmo acelerado. Por exemplo, potenciais benefícios da adoção da gamificação no ensino de programação e teste de software vêm sendo investigados em diferentes estudos da literatura (Clegg *et al*., 2017; Garcia *et al.*, 2017; Laurent *et al.*, 2017; Parizi, 2016).

Para aprimorar o processo de ensino-aprendizagem no Instituto Federal de São Paulo — IFSP, vem sendo desenvolvido um jogo de Truco para uso em disciplinas de Programação Orientada a Objetos, Projeto de Software e Teste de Software. O jogo, disponível como Software Livre[1], já tem sido utilizado como material de apoio em disciplinas da graduação, fornecendo exemplos de boas práticas de programação e projeto. Entretanto, é nas disciplinas sobre Programação Orientada a Objetos e Teste de Software que ele tem se destacado. O sistema permite que estudantes desenvolvam módulos autônomos, chamados de bots, capazes de competir contra jogadores humanos ou contra outros bots. Por se tratar de um jogo com regras bastante conhecidas, a criação de campeonatos de Truco permite o ensino do Desenvolvimento Guiado por Testes - TDD (*Test-Driven Development*) (Beck, 2002), engajando os estudantes ao longo dos últimos semestres.

Entretanto, o jogo ainda precisa de aprimoramentos que permitam sua ampla adoção, principalmente como um sistema Web. Além disso, embora o retorno dos estudantes tenha sido positivo quanto a adoção de jogos como objetos de ensino, não foram conduzidas avaliações sistemáticas para mensurar possíveis benefícios relacionados à incorporação de tal tecnologia. Dessa forma, o principal objetivo deste trabalho é evoluir o CTruco para fomentar sua adoção no ensino de disciplinas de Computação, bem como obter evidências sobre os benefícios da incorporação dessa tecnologia no processo de ensino-aprendizagem de estudantes de graduação.

MATERIAL E MÉTODOS

O CTruco foi desenvolvido em Java 17, de acordo com princípios de projetos definidos na Arquitetura Limpa (Martin, 2018), conforme ilustrado na Figura 1. Uma das vantagens desse formato de projeto é separar as regras de negócio, mais estáveis, dos detalhes técnicos de implementação, que possuem maior propensão a mudanças. Nesse modelo de projeto, não há dependências diretas entre as regras de negócio e detalhes como comunicação com o usuário, viabilizando a criação de múltiplas interfaces gráficas de forma independente. Atualmente, existem três formas de se utilizar o jogo: i)

[1] Link para o repositório do projeto: https://github.com/lucas-ifsp/ctruco.

linha de comando; ii) interface desktop em JavaFx[2]; e iii) sistema Web feito com React[3] e Spring Boot[4]. A persistência do sistema utiliza o PostgreSQL[5] como banco de dados relacional, além de um banco de dados não-relacional MongoDB[6], ambos fornecidos por meio de contêineres Docker[7]. O MongoDB é responsável por guardar dados temporários, como partidas em andamento e seus respectivos jogadores. Já o PostgreSQL armazena os dados permanentes, como usuários cadastrados, resultados de partidas concluídas e endereços de *bots* remotos.

Um usuário do CTruco pode realizar partidas contra um *bot* ou simular diversos embates entre dois *bots*, que jogam entre si de forma autônoma. A criação de um *bot* por um estudante começa com a obtenção do projeto no Github[8], seguido da implementação de quatro métodos previstos em uma interface. O primeiro método responde se o *bot* aceita ou não jogar a "*mão de onze"*, caso o jogo se encontre nesse estado; enquanto o segundo responde se o *bot* deseja pedir para aumentar a pontuação da mão atual. O terceiro método decide qual carta deve ser jogada na atual rodada e o último responde a pedidos de aumento de ponto, aceitando, rejeitando ou propondo um valor ainda maior. Todos os métodos recebem como parâmetro um objeto descrevendo o estado atual do jogo.

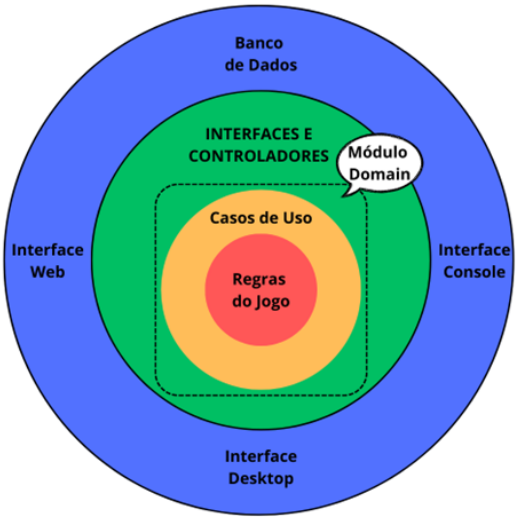

FIGURA 1. Abstração da arquitetura do CTruco. Fonte: autor

Nesta etapa do projeto, foram projetadas e implementadas novas funcionalidades para o CTruco. Primeiramente, foram criadas funcionalidades que permitem: i) avaliar *bots*, gerando estatísticas sobre o desempenho do elemento avaliado; ii) classificar *bots*, criando um *ranking* de todos os *bots* em funcionamento; iii) integração de *bots* rodando remotamente, chamados *remote bots*, possibilitando o desenvolvimento deles em outras linguagens de programação; iv) menu de gestão dos bots remotos, permitindo o usuário cadastrar, visualizar, editar e excluir os próprios *bots*.

Na avaliação de *bots*, são feitas diversas simulações entre o módulo avaliado e todos os outros da plataforma. Tais simulações são feitas em paralelo por meio de *threads* e da API de Streams do Java, o que torna o tempo de simulação mais rápido. O resultado da avaliação de *bots* exibe ao usuário estatísticas do desempenho, como a taxa de vitória contra cada bot existente na plataforma. Já no sistema de classificação de *bots*, são realizadas simulações entre todos os bots. Nesta funcionalidade, o número de partidas simuladas entre bots é menor, por questão de eficiência, pois mesmo executando em paralelo, o tempo de execução pode ser muito alto. Após o término das simulações, o algoritmo exibe uma lista de classificação, dos mais vitoriosos até os menos vitoriosos.

[2]JavaFx: https://openjfx.io. Acesso em 12/08/2024.
[3]React: https://react.dev/. Acesso em 12/08/2024.
[4]Spring Boot:https://spring.io/projects/spring-boot. Acesso em 12/08/2024.
[5]PostgreSQL: https://www.postgresql.org. Acesso em 20/08/2024.
[6]MongoDB:https://www.mongodb.com. Acesso em 12/08/2024.
[7]Docker: https://www.docker.com. Acesso em 12/08/2024.
[8]Github: https://github.com/. Acesso em 22/08/2024.

A implementação de *remote bots* consiste em um consumidor de API seguindo o protocolo HTTP, no qual cada *endpoint* fornece decisões que o *bot* deve tomar durante as partidas. A utilização de *bots* remotos possibilita criar implementações em outras linguagens e *frameworks*, sendo que o usuário pode cadastrá-los na plataforma sem a necessidade de integração ao código base do jogo. A Figura 2 mostra uma partida entre um jogador e um módulo remoto, na qual é perceptível que o fluxo da partida consiste em uma sequência de decisões tomadas pelo *bot* após uma ação do jogador, que o próprio programa reconhece como necessárias e, em seguida, pergunta ao *bot*. Essas perguntas seguem uma lógica fixa. Por exemplo, é perguntado se o *bot* quer aumentar os pontos da rodada, antes de perguntar qual carta ele quer jogar. Se o módulo desejar o aumento, será a vez do jogador responder se aceita essa mudança na rodada ou não. Este processo é muito parecido ao de partidas de *bots* locais, a diferença é que neste caso o *backend* faz requisições, ou perguntas, para uma aplicação externa ao programa.

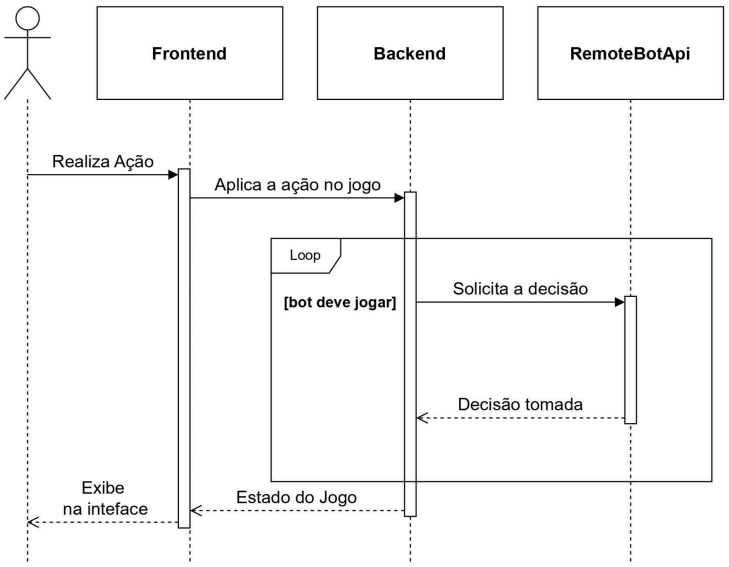

FIGURA 2. Diagrama de sequência de uma partida entre um usuário e um bot remoto. Fonte: autor

RESULTADOS E DISCUSSÃO

A partir deste projeto, o CTruco recebeu diversas melhorias, que visam, principalmente, adicionar maior versatilidade e atratividade durante o processo de ensino-aprendizagem. Estas melhorias incluem refatorações na estrutura do sistema e novas funcionalidades que auxiliam na criação de *bots* e que permitem desenvolvê-los em outras linguagens de programação.

No contexto da refatoração, foram ajustadas rotinas que estavam em camadas mais externas da aplicação, a partir da criação de novos casos de uso reutilizáveis. O mecanismo de persistência de dados foi migrado do Spring Data JPA[9] para o JDBC do próprio Java, uma vez que o modelo anterior acoplava o processo de armazenamento de dados à tecnologia adotada na interface Web do projeto. O código também foi redigido conforme o necessário, deixando-o mais compreensível.

Também foram desenvolvidas rotinas para a avaliação dos *bots*, o que ajuda os estudantes a medir a qualidade do módulo sendo desenvolvido, ao passo que estimula a competitividade durante as atividades. A título de exemplo, no primeiro semestre de 2024 foram feitos dois campeonatos, adicionando mais de 20 *bots* ao sistema. A maioria desses *bots*, a partir da adoção da funcionalidade de avaliação, supera os demais cadastrados anteriormente na plataforma.

Além de funcionalidades que medem o desempenho dos bots, também foram feitas funcionalidades que incluem a criação de *bots* executados remotamente. Com isto, o CTruco se torna

[9] Spring Data JPA: https://spring.io/projects/spring-data-jpa acesso em: 23/08/2024

uma ferramenta consideravelmente mais versátil como instrumento educacional, uma vez que os professores podem ensinar sobre APIs REST e desenvolvê-las em outras linguagens e frameworks como Express[10] e Django[11], abrindo caminho para o desenvolvimento de *bots* com base em aprendizado de máquina, por exemplo.

Atualmente, a interface Web da aplicação está sendo aprimorada, com o intuito de deixá-la mais atrativa. Esta etapa terá como foco a implementação de funcionalidades que já foram feitas no back-end e também incluirá novas funcionalidades, como a criação automática de campeonatos. Na sequência, será aplicado um formulário para avaliar a aceitação do jogo como um objeto de ensino. Tal formulário seguirá o modelo TAM (*Technology Acceptance Model*) (Venkatesh, 2000). O princípio deste modelo é avaliar duas características do sistema: se ele é útil (e.g., o usuário percebe que a tecnologia ajuda nos processos que se propõe a interagir) e se ele é fácil de usar (e.g., o usuário acredita que a ferramenta possui uma curva de aprendizado que não atrapalhe na utilidade dela). O formulário citado será aplicado de forma não obrigatória em duas disciplinas, uma do curso de Bacharelado em Engenharia de Software e outra do curso de Análise e Desenvolvimento de Sistemas, ambas do IFSP campus São Carlos.

CONCLUSÕES

Este artigo apresentou o processo de aprimoramento do CTruco, um objeto ensino-aprendizagem gamificado para ensinar conceitos na programação em cursos de graduação em Computação. A partir das melhorias, foram introduzidas funcionalidades para a avaliação e criação de *ranking* de *bots* e a implementação de *bots* remotos. Foi também realizada uma refatoração na estrutura existente, deixando o código mais coeso e fácil de manter. Entre as funcionalidades em desenvolvimento, destaca-se a criação de campeonatos e o aprimoramento da interface *web* do jogo. As novas funcionalidades já expandem o escopo do projeto, permitindo o ensino de frameworks como o Spring no processo de integração de módulos remotos. Adicionalmente, foi criado um formulário para coletar as opiniões dos estudantes que utilizam o jogo, visando obter evidências sobre sua aceitação no suporte ao ensino de programação e teste de software. Espera-se que a adoção do CTruco possa beneficiar o processo de ensino e aprendizagem em cursos de Computação, impactando positivamente no percurso do estudante no curso e contribuindo pontualmente para a questão da evasão.

CONTRIBUIÇÕES DOS AUTORES

Gustavo G. Contiero contribuiu no desenvolvimento e implementação de melhorias, Lucas B. R. Oliveira atuou na supervisão do projeto e na idealização das funcionalidades desenvolvidas. Gustavo G. Contiero e Lucas B. R. Oliveira trabalharam na escrita e revisão do trabalho submetido.

REFERÊNCIAS

BECK, K. **Test Driven Development: By Example**. Addison-Wesley Longman Publishing Co., Inc., USA, 2002.

CLEGG, B. S.; Rojas, J. M.; Fraser, G. Teaching Software Testing Concepts Using a Mutation Testing Game. In: **2017 ACM/IEEE International Symposium on Software Engineering (ICSE)**, SEER Track, Buenos Aires, Argentina: IEEE, 2017. p. 33-36.

DETERDING, S. *et al*. From game design elements to gamefulness: defining "gamification". In: **15th International Academic MindTrek Conference: Envisioning Future Media Environments**, 2011. p. 9-15.

[10] Express: https://expressjs.com/pt-br/. Acesso em: 20/08/2024
[11] Django: https://www.djangoproject.com/. Acesso em: 20/08/2024

FEICHAS, F. A.; SEABRA, R. D.; SOUZA, A. D. Gamificação no ensino superior em ciência da computação: uma revisão sistemática da literatura. **Revista Novas Tecnologias na Educação**, Porto Alegre, v. 19, n. 1, p. 443–452, 2021.

FUKAO, A. T. *et al.* Estudo sobre evasão nos cursos de Computação da Universidade Estadual de Maringá. In: **Simpósio Brasileiro de Educação em Computação**, 3. ed., 2023, Evento Online. Anais…Porto Alegre: Sociedade Brasileira de Computação, 2023. p. 86-96.

GARCÍA, F. *et al.* A Framework for Gamification in Software Engineering. **Journal of Systems and Software**, v. 132, p. 21–40, 2017.

GOOGLE FOR STARTUPS. **Panorama de talentos em tecnologia:** a escassez de profissionais de tecnologia no Brasil e seu consequente impacto no ecossistema de startups. 2023. Disponível em: https://campus.co/sao-paulo/gap-de-talentos. Acesso em: 23 ago. 2024.

HOED, R. M. **Análise da evasão em cursos superiores: o caso da evasão em cursos superiores da área de computação**. 2016. 164 f. Dissertação (Mestrado Profissional em Computação Aplicada) - Universidade de Brasília, Brasília, 2016.

LAURENT, T. *et al.* Towards a Gamified Equivalent Mutants Detection Platform. In: **ICST Conference - Posters Session**, Tokyo, Japan: IEEE, 2017. p. 382–384.

MARTIN, R. C. **Clean Architecture:** a craftsman's guide to software structure and design. Boston: Prentice Hall, 2018. 404 p.

MARTIN, R. C. **Clean Code:** a handbook of agile software craftsmanship. Boston: Pearson Education, 1. ed. São Paulo: A, 2008. 464 p.

PARIZI, R. M. Sobre a gamificação de tarefas de rastreabilidade centradas no ser humano em testes e codificação de software. In: **IEEE 14th International Conference on Software Engineering Research, Management and Applications (SERA)**, 2016, Towson, MD, EUA. Towson, MD: IEEE, 2016. p. 193-200.

SEMESP. Sindicato das Mantenedoras do Ensino Superior Privado. **Mapa do Ensino Superior no Brasil: edição 2023**. São Paulo: Semesp, 2023. Disponível em: https://www.semesp.org.br/wp-content/uploads/2023/06/mapa-do-ensino-superior-no-brasil-2023.pdf. Acesso em: 23 ago. 2024.

VENKATESH, V.; DAVIS, F. D. A theoretical extension of the technology acceptance model: four longitudinal field studies. **Management Science**, v. 46, n. 2, p. 186-204, 2000.