Wanda 2.0: Jogos de Cartas Digitais para o Ensino de Programação com Feedback por IA

Wanda 2.0: Digital Card Games for Programming Education with AI-Powered Feedback

Lucas Ibiapino[1], **Djefferson Maranhão**[1], **Carlos de Salles S. Neto**[1]

[1] Departamento de Informática (DEINF) – Universidade Federal do Maranhão (UFMA)
São Luís – MA – Brasil

Abstract. Introduction*: Teaching programming in undergraduate computer science courses is a persistent challenge, frequently associated with high dropout rates in Brazilian universities, driven by low motivation and difficulty relating abstract content to practical activities.* ***Objective****: This paper presents Wanda 2.0, a modernized and fully web-based version of the Wanda framework, originally proposed in 2014, which allows teachers to create card-based digital games in which students must program virtual agents to compete against each other, combining algorithmic practice with competitive engagement.* ***Methodology****: The new version introduces a web interface, an elimination tournament system, an AI-powered pedagogical assistant in three communication profiles, and a containerized infrastructure deployable with a single command, addressing the main limitations of the original framework.* ***Results****: An evaluation with 76 undergraduate students showed a 91% activation rate, with 67% completing the full Run→Feedback→Submit cycle and the same proportion using the AI assistant, whose Explanatory style accounted for 42.4% of all interactions. Competitive participation reached 37%, revealing an engagement gap in the challenge phase that informs future improvements to the platform.*
Keywords *Card Games, Programming Education, Game-Based Learning, Software Architecture.*

Resumo. Introdução*: O ensino de programação em cursos de Ciência da Computação apresenta elevadas taxas de evasão nas universidades brasileiras, associadas à baixa motivação e à dificuldade de relacionar conteúdos abstratos à prática.* ***Objetivo****: Este artigo apresenta o Wanda 2.0, uma versão modernizada e inteiramente web do framework Wanda, proposto originalmente em 2014, que permite a criação de jogos de cartas digitais nos quais alunos programam agentes autônomos que competem em torneios, combinando prática algorítmica com motivação competitiva.* ***Metodologia****: A nova versão introduz interface web, sistema de torneios eliminatório, assistente pedagógico com IA em três perfis de comunicação e infraestrutura containerizada implantável com um único comando, superando as principais limitações do framework original.* ***Resultados****: Uma avaliação com 76 alunos de graduação revelou taxa de ativação de 91%, com 67% percorrendo o ciclo completo Run→Feedback→Submit e igual proporção utilizando o assistente de IA, cujo estilo Explicativo concentrou 42,4% das interações. A participação competitiva atingiu 37%, revelando um gargalo na fase de desafios que orienta melhorias futuras.*

Palavras-Chave *Jogos de Cartas, Ensino de Programação, Aprendizagem Baseada em Jogos, Arquitetura de Software.*

1. Introdução

Um dos problemas mais persistentes nos cursos de graduação em Ciência da Computação no Brasil é a elevada taxa de evasão nos períodos iniciais [Carvalho et al. 2019]. A dificuldade de aprender programação é frequentemente apontada como um dos principais fatores dessa evasão [Kinnunen e Malmi 2006, Petersen et al. 2016]: programar exige raciocínio abstrato, decomposição de problemas e persistência diante de erros, competências que levam tempo para se desenvolverem e que, sem o estímulo adequado, podem frustrar o estudante precocemente.

Jogos digitais têm sido explorados como mecanismo de engajamento em contextos educacionais, com resultados positivos em motivação, retenção de conhecimento e redução da ansiedade diante do aprendizado [Prensky 2003, Dicheva et al. 2015]. Jogos que exigem que o estudante escreva código como parte da mecânica são especialmente eficazes, pois integram de forma orgânica a prática de programação ao entretenimento [Bayliss e Strout 2006].

Em 2014, [Drumond et al. 2014] propuseram o Wanda, um *framework* para criação de jogos de cartas digitais voltados ao ensino de algoritmos. A proposta do *framework* é simples: o professor configura um jogo de cartas e o disponibiliza à turma; cada aluno deve escrever um *agente virtual*, isto é, um programa que joga as partidas de forma autônoma. Em seguida, os agentes disputam um torneio, e o desempenho do programa reflete diretamente a qualidade do algoritmo implementado. Essa dinâmica estimula a competição saudável, o pensamento algorítmico e a melhoria contínua das estratégias.

Os resultados iniciais do Wanda foram promissores, mas o *framework* tinha limitações que dificultavam sua adoção: era uma aplicação *desktop* que exigia instalação manual em cada computador; não tinha *interface web* e não suportava múltiplas turmas. Após mais de uma década, com o amadurecimento das tecnologias de desenvolvimento, tornou-se viável superar essas limitações sem abdicar da proposta pedagógica original.

Este artigo apresenta o Wanda 2.0, uma versão refatorada do *framework* original, acessível via navegador, com sistema de torneios, *replay* de partidas e um assistente de *feedback* com inteligência artificial que orienta os alunos durante o desenvolvimento de seus agentes. O artigo está organizado da seguinte forma: a Seção 2 apresenta trabalhos relacionados; a Seção 3 descreve o Wanda original; a Seção 4 apresenta o Wanda 2.0; a Seção 5 discute a avaliação; e a Seção 6 apresenta as conclusões.

2. Trabalhos Relacionados

A utilização de jogos digitais no ensino de computação tem amplo registro na literatura, com abordagens que variam de motores de jogos a plataformas web e, mais recentemente, ao uso de modelos de linguagem para *feedback*. Esta seção revisa as mais relevantes para contextualizar o Wanda 2.0.

Chien2D [Radtke e Binder 2010] é um motor de jogos 2D usado para ensinar programação desafiando estudantes a criar jogos, porém exige conhecimentos prévios e

planejamento extenso, sendo menos adequado como atividade de curta duração. *PyGame Educacional* [Rebouças et al. 2010] utilizou a biblioteca *PyGame* com estudantes do ensino médio com resultados motivacionais positivos, mas com demanda de tempo similar.

Plataformas modernas como *Code Combat* oferecem ambientes web onde estudantes escrevem código real para controlar personagens. [Kroustalli e Xinogalos 2021] avaliaram a plataforma em estudo controlado e concluíram que, embora o grupo experimental tenha apresentado desempenho superior, a diferença não foi estatisticamente significativa, e a plataforma não permite que o professor configure torneios ou acompanhe o desempenho da turma de forma integrada.

A gamificação de plataformas de ensino tem mostrado impacto positivo em motivação [Dicheva et al. 2015]: [de Pontes et al. 2019] demonstraram que *leaderboard* e *badges* levaram estudantes a resolver significativamente mais exercícios. Contudo, [Ribeiro et al. 2019] observaram que elementos competitivos não são suficientes para garantir maior aprendizagem, evidenciando a necessidade de suporte adicional ao estudante durante a prática.

Com o avanço dos modelos de linguagem de grande escala, [Zhang et al. 2024] investigaram percepções de iniciantes sobre feedback gerado por LLM em exercícios de programação, identificando divergência significativa nas preferências de tom, parte valorizava orientações encorajadoras, outra preferia avaliações diretas, evidenciando que abordagens uniformes de feedback são insuficientes para atender à diversidade de perfis de aprendizes.

A Tabela 1 sintetiza o comparativo entre as abordagens.

Tabela 1. Comparativo entre trabalhos relacionados e o Wanda 2.0

Trabalho	Web	Prog. Básica	Agentes	Feedback IA
Chien2D	Não	Sim	Não	Não
PyGame Educ.	Não	Sim	Não	Não
Code Combat	Sim	Sim	Não	Não
Gamificação	Sim	Sim	Não	Não
LLM em Prog.	Sim	Sim	Não	Sim
Wanda 2.0	**Sim**	**Sim**	**Sim**	**Sim**

O diferencial do Wanda 2.0 frente às demais abordagens está na combinação de jogos acessíveis via navegador, exigência de que o aluno programe um agente autônomo, competição direta entre esses agentes em torneios configurados pelo professor, e suporte com inteligência artificial durante o desenvolvimento.

3. *Framework* Wanda (Versão Original)

A versão original do Wanda foi concebida como um *framework Python* para criação de jogos de cartas eletrônicos de uso educacional [Drumond et al. 2014]. Sua arquitetura divide-se em três camadas: *Mechanics and Interface* (MaI), *Rules* e *Strategies*.

A camada *Mechanics and Interface* é o núcleo do framework, responsável pelas funções comuns a todos os jogos: embaralhamento, distribuição de cartas, controle de

turnos e exibição das partidas. A camada *Rules* define as regras de conflito: qual combinação de cartas resulta em vitória, empate ou derrota a cada rodada, permitindo ao professor conectar a mecânica do jogo ao conteúdo da disciplina. A camada *Strategies* é a interface que o aluno implementa: uma função *Python* que, dado o estado atual do jogo, decide qual carta jogar, e cuja competição entre agentes é o núcleo motivacional do sistema.

Na versão inicial, dois jogos foram aplicados em turmas de calouros: o *Jo-Ken-Po*, inspirado no Pedra-Papel-Tesoura, e o *Divide-and-Conquer*, com cartas numéricas e regras que remetem ao paradigma algorítmico homônimo, ambos com resultados positivos em motivação. Apesar do sucesso, limitações como instalação manual, ausência de interface web e falta de suporte automatizado aos alunos dificultavam sua adoção mais ampla.

4. Wanda 2.0 (Versão Refatorada)

O Wanda 2.0 é uma reimplementação completa da plataforma, construída sobre uma arquitetura de microsserviços e acessível inteiramente via navegador web. A nova versão mantém o princípio pedagógico original, isto é, a programação de agentes competitivos, e o expande com novas funcionalidades que respondem diretamente às limitações identificadas.

4.1. Arquitetura Geral

A plataforma adota uma arquitetura de microsserviços, composta por três serviços independentes que se comunicam via `API REST`, além de uma infraestrutura de suporte, conforme ilustra a Figura 1. O *frontend* web (React) oferece a interface para alunos e professores; o serviço de negócio (Spring Boot/Java) atua como motor de partidas, gerenciando usuários, torneios e a persistência dos dados em um banco PostgreSQL; e o serviço de agentes (FastAPI/Python) é responsável pela validação, teste e execução do código submetido pelos alunos. Todo o ambiente é containerizado com Docker e exposto por um *proxy* reverso Nginx, permitindo a implantação completa com um único comando.

4.2. Interface Web

A interface do Wanda 2.0 é acessível por qualquer navegador moderno, sem necessidade de instalação. Ela oferece visões e funcionalidades distintas para professores e alunos.

O aluno pode visualizar os jogos disponibilizados pelo professor, acessar o editor de código para desenvolver e submeter seu agente, acompanhar o histórico de suas partidas com suporte a *replay*, verificar sua posição no *ranking*, participar de torneios e desafiar seus amigos para partidas individuais.

O professor pode gerenciar usuários, lançar torneios, acompanhar resultados em tempo real, visualizar o desempenho e a interação dos alunos com o sistema.

A Figura 2 ilustra a tela de submissão de agentes da plataforma.

4.3. Jogos Suportados

A nova versão mantém o *Jo-Ken-Po* com regras aprimoradas. Cada duelo entre dois agentes consiste em 21 partidas, cada uma com três rodadas. O aluno desenvolve a

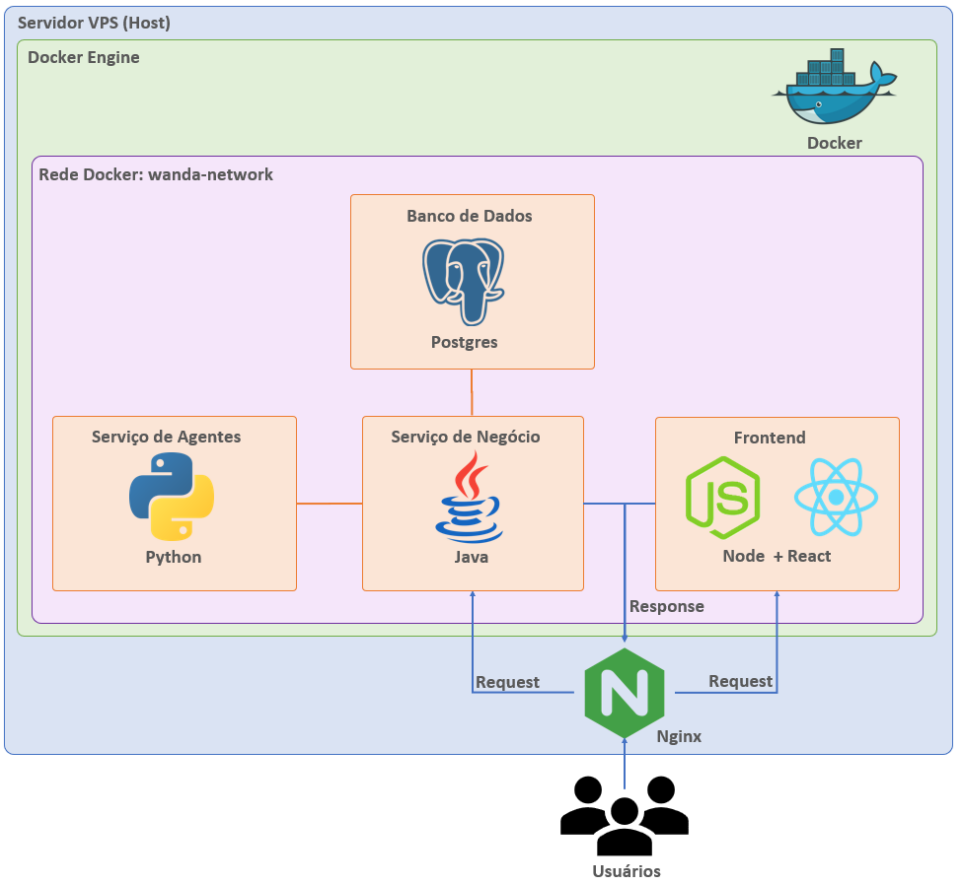

Figura 1. Arquitetura de microsserviços do Wanda 2.0

estratégia de dois agentes, um que escolhe sua carta no primeiro *round*, e um segundo que escolhe a sua jogada no segundo *round*; no terceiro joga a carta restante. A função da segunda rodada recebe também informações sobre as cartas restantes na mão do adversário, introduzindo uma informação extra que incentiva o desenvolvimento de estratégias mais elaboradas.

O novo jogo *Bits* introduz um vocabulário inspirado em redes de computadores: o agente gerencia pacotes de dados (*BIT8*, *BIT16*, *BIT32*) e um mecanismo de proteção (*FIREWALL*), decidindo qual recurso usar em cada rodada com base no estado da mão e na última jogada do adversário. O jogo conecta a mecânica competitiva a conceitos de sistemas computacionais.

A plataforma foi projetada de modo a permitir que novos jogos possam ser adicionados sem modificação do código existente, por meio do registro centralizado de especificações.

4.4. Submissão e Validação de Agentes

O fluxo de trabalho do aluno para criar um agente é dividido em três etapas complementares, disponíveis na interface de submissão (Figura 2):

1. *Feedback*: o aluno submete o código e recebe uma análise pedagógica do assistente, sem que a função seja salva, o que é essencial para uma exploração inicial;

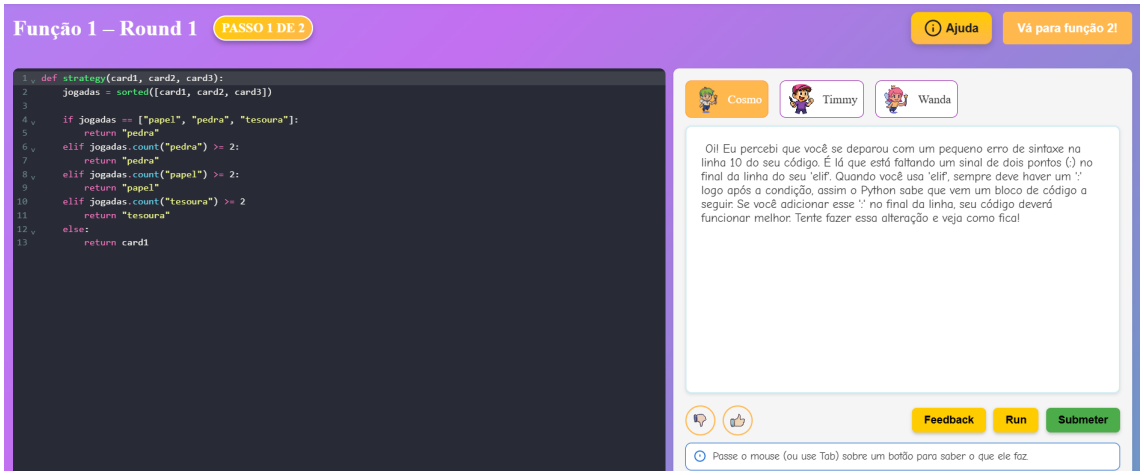

Figura 2. Tela de submissão de agentes com editor de código integrado e assistente de feedback

2. Executar testes (*Run*): o código é executado contra casos de teste predefinidos e o aluno recebe a descrição detalhada dos resultados de sua estratégia para situações reais dos jogos;
3. Enviar (*Submit*): cenário em que ocorre a validação completa. Somente funções que passam em todos os testes são salvas e habilitadas para torneios e desafios individuais.

A separação em etapas incentiva a iteração e reduz a frustração: o aluno pode explorar e corrigir seu código antes de submetê-lo em definitivo.

4.5. Assistente de Feedback com IA

O assistente pedagógico baseado em *LLM* (modelo de linguagem de grande escala) é uma inovação em relação à versão anterior. Ao solicitar *feedback*, o aluno recebe orientações em linguagem natural sobre os erros no seu código: o *prompt* enviado ao modelo inclui o código submetido, as regras do jogo em execução e o perfil de comunicação selecionado, garantindo orientações contextualizadas ao jogo e ao conteúdo programático, e não apenas mensagens genéricas de compilação. O assistente é instruído a orientar o aluno sem revelar a solução, preservando o esforço de desenvolvimento. O assistente opera em três perfis de comunicação, que o aluno pode escolher:

- explicativo: explanações passo a passo, acolhedoras e com encorajamento;
- intermediário: equilíbrio entre clareza e concisão;
- direto: respostas objetivas e diretas.

Todo o histórico de interações entre o aluno e o assistente fica registrado na plataforma, permitindo ao professor analisar as dificuldades recorrentes da turma e o progresso individual de cada estudante ao longo do semestre.

4.6. Sistema Competitivo e Gamificação

A versão atual estrutura a competição em duas modalidades: torneios e desafios.

Os torneios são criados pelo professor com configuração de nome, jogo, data de início, número máximo de participantes e opção de acesso privado por senha. O sistema

verifica automaticamente se o aluno possui agente aprovado antes de permitir a inscrição, garantindo que apenas estratégias válidas disputem partidas. O chaveamento eliminatório é gerado automaticamente e fica disponível para acompanhamento, com suporte a *replay* de cada partida.

Os desafios possibilitam embates diretos, permitindo a qualquer aluno desafiar um colega a qualquer momento, sem necessidade de torneio formal, tornando a competição contínua e acessível ao longo do período letivo.

Diferentemente da mecânica de jogo que constitui o núcleo da plataforma, a nova versão incorpora também elementos de gamificação, isto é, componentes de jogos aplicados ao redor da atividade para sustentar o engajamento entre partidas: um *ranking* global com a classificação dos alunos baseada nos resultados acumulados, e *badges* concedidas automaticamente por conquistas ao longo do uso da plataforma. Esses elementos estimulam o aluno a revisar e aprimorar continuamente sua estratégia, reforçando o ciclo de desenvolvimento que é o núcleo pedagógico da plataforma.

4.7. Infraestrutura e Implantação

Todos os serviços da plataforma são containerizados com *Docker*, e a implantação completa é realizada com um único comando (*docker compose up*), eliminando a instalação manual de dependências, uma das principais limitações da versão original, e tornando a plataforma portável para qualquer ambiente com Docker instalado. O processo de desenvolvimento conta ainda com um *pipeline* de integração e entrega contínua (CI/CD) via *GitHub Actions*, que automatiza a verificação do código e garante a consistência dos *builds* entre ambientes. Em síntese, a nova versão supera a original ao oferecer acesso via navegador, arquitetura de microsserviços, torneios, desafios diretos, replay de partidas, feedback automatizado com IA e histórico completo de interações, recursos ausentes na implementação legada. A plataforma Wanda 2.0 encontra-se disponível publicamente para uso e avaliação, com o software registrado no repositório eduCAPES [Ibiapino et al. 2025], favorecendo a transparência e a reprodutibilidade da pesquisa.

5. Avaliação

O Wanda 2.0 foi avaliado em uma turma de graduação no mês de abril de 2026. Ao todo, 76 alunos foram cadastrados na plataforma, e 69 (91%) realizaram pelo menos uma interação, indicando alta ativação inicial. No total, foram registradas 1.436 interações, englobando ações de *Run*, *Feedback* e *Submit*, e os dados foram coletados a partir dos registros de log da plataforma.

A análise foi organizada em quatro eixos: engajamento geral, ciclo de uso, estilos do assistente de IA e participação competitiva.

Engajamento Geral

O funil de engajamento revelou uma taxa de ativação expressiva: dos 76 alunos cadastrados, 69 interagiram com o sistema, 63 testaram código, 63 submeteram pelo menos uma solução, e 51 utilizaram o assistente de feedback, todos antes de chegar à

fase competitiva. A participação em competições foi menor: 28 alunos (37% da base cadastrada) disputaram ao menos uma partida, conforme ilustrado na Figura 3.

A distribuição de interações por aluno apresentou mediana de 15 e média de 20,8, indicando distribuição assimétrica: o aluno típico realizou cerca de 15 interações, mas um grupo menor foi consideravelmente mais ativo. Os picos de uso concentraram-se nas terças e sextas-feiras, correspondendo ao calendário das aulas.

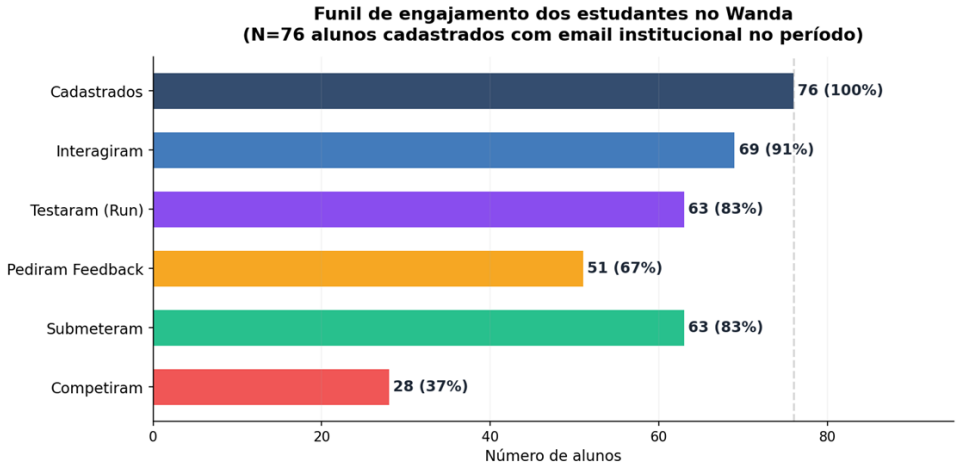

Figura 3. Funil de engajamento dos estudantes no Wanda 2.0 (N=76 alunos cadastrados)

Ciclo de Uso: Run, Feedback e Submit

A análise do ciclo de uso revelou que o padrão mais comum foi o percurso completo *Run* → *Feedback* → *Submit*: 46 dos 69 alunos ativos (67%) passaram pelas três ações (Figura 4). Outros alunos seguiram caminhos alternativos; alguns realizaram *Run* e *Submit* sem solicitar feedback; enquanto outros foram diretamente ao *Submit*; e alguns exploraram o sistema sem chegar a submeter um agente.

Os dados sobre a primeira ação registrada por aluno mostraram distribuição equilibrada: 39% iniciaram pelo *Feedback*, 38% por *Run* e 23% por *Submit*, indicando que os alunos adotaram pontos de entrada distintos. Quanto à preparação antes de submeter, 80% das submissões foram precedidas de ao menos uma interação, e 61% vieram após quatro ou mais interações, sugerindo que a plataforma estimula um processo iterativo de desenvolvimento. Esse comportamento é corroborado pela taxa crescente de ações válidas ao longo do ciclo: enquanto 50% dos *Runs* resultam em código válido, essa proporção sobe para 64% no *Feedback* e 78% no *Submit*.

Estilos do Assistente de IA

O assistente pedagógico *explicativo* foi o mais utilizado, totalizando 609 interações (42,4% do total) e sendo o perfil predominante para 29 alunos (42% da turma ativa), conforme mostrado na Figura 5. Verificou-se ainda diversidade no uso dos estilos: 26 alunos utilizaram apenas um estilo ao longo de toda a atividade, enquanto outros 26 exploraram os três perfis disponíveis.

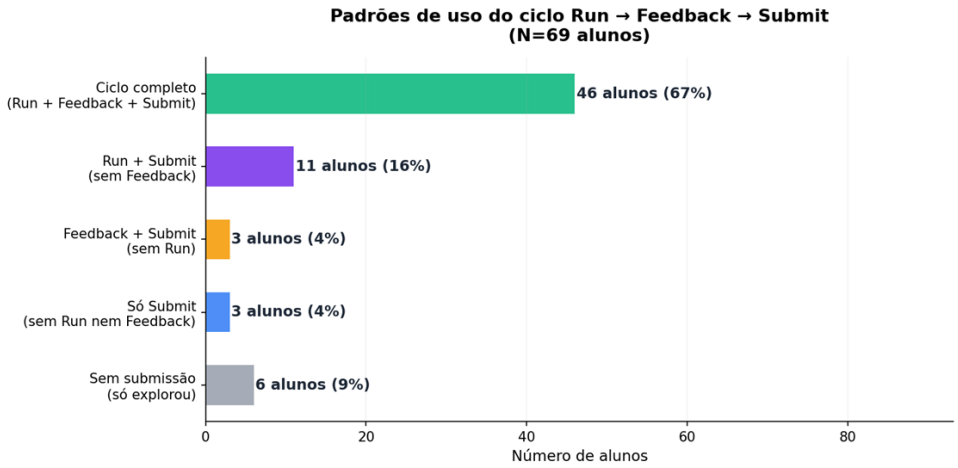

Figura 4. Padrões de uso do ciclo Run→Feedback→Submit (N=69 alunos)

Em todas as configurações de estilo, o *Submit* representou a menor proporção das ações, enquanto *Run* e *Feedback* concentraram a maior parte do uso, reforçando o caráter iterativo da interação com a plataforma.

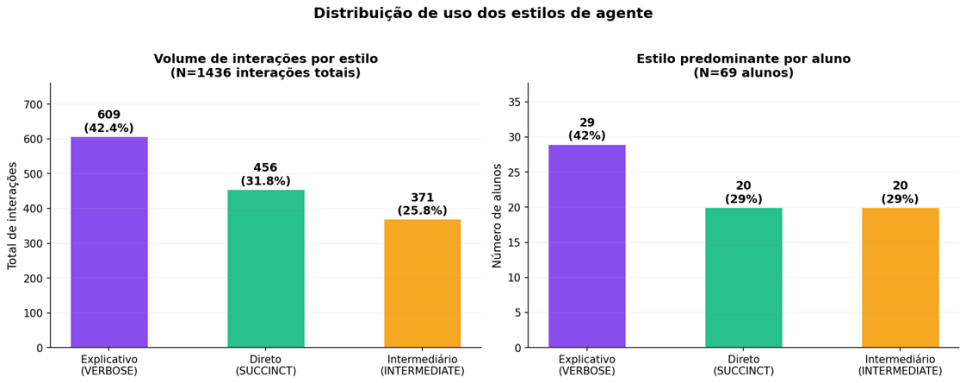

Figura 5. Distribuição de uso dos estilos de agente por volume de interações e por aluno

Participação Competitiva

A fase competitiva apresentou menor adesão em relação ao uso geral da plataforma. Dos 28 alunos que competiram, a maioria participou de poucas partidas: 11 jogaram apenas 1 partida, 12 jogaram de 2 a 4 partidas, e somente 5 jogaram 5 ou mais. A mediana de partidas por competidor foi de 2.

Foram criados 248 desafios diretos, dos quais 196 ficaram sem resposta, 49 foram aceitos e 3 foram recusados. Embora a taxa de aceitação entre os desafios respondidos tenha sido elevada (94%), o alto volume de desafios ignorados indica uma barreira de engajamento na fase de competição direta que merece atenção em iterações futuras da plataforma.

Por fim, comparando competidores e não-competidores quanto ao uso da plataforma, os primeiros apresentaram mediana de 18 interações totais, contra 14 dos não-competidores; e mediana de 7 usos de *Feedback*, contra 4 dos não-competidores,

sugerindo que alunos com maior engajamento geral na plataforma tendem a participar mais ativamente da fase competitiva.

6. Conclusão

Este artigo apresenta uma versão refatorada e inteiramente acessível via web do framework Wanda, para criação de jogos de cartas digitais educacionais. A nova versão preserva a proposta pedagógica original, usando agentes competitivos como motor de motivação intrínseca, e a expande com funcionalidades que respondem diretamente às limitações identificadas ao longo de mais de uma década de uso da plataforma.

Os resultados da avaliação conduzida com 76 alunos de graduação fornecem evidências iniciais encorajadoras sobre a adoção da plataforma. A taxa de ativação de 91% indica que a transição para uma interface web eliminou efetivamente a fricção de instalação que limitava o *framework* original. Mais relevante do ponto de vista pedagógico, 67% dos alunos ativos percorreram o ciclo completo *Run* $\rightarrow$ *Feedback* $\rightarrow$ *Submit*, e 61% das submissões foram precedidas de quatro ou mais interações, evidenciando que a plataforma estimula um comportamento iterativo de desenvolvimento, mesmo sem exigi-lo explicitamente.

O assistente pedagógico com IA demonstrou adoção expressiva: 67% dos alunos utilizaram a funcionalidade de feedback, e o estilo *explicativo* foi o mais demandado, com 42,4% das interações, demonstrando que alunos iniciantes preferem orientações detalhadas a respostas concisas. A diversidade de perfis de uso, com metade da turma alternando entre estilos ao longo da atividade, indica que a possibilidade de personalizar o tom do assistente foi percebida e utilizada pelos estudantes.

O principal ponto de atenção identificado pela avaliação está na fase competitiva: apenas 37% dos alunos cadastrados chegaram a disputar partidas, e 79% dos desafios criados ficaram sem resposta. Esse gargalo não parece decorrer de desinteresse pela competição, alunos que competiram tiveram mediana de 18 interações totais e 7 feedbacks, contra 14 e 4 dos não-competidores, mas possivelmente de barreiras operacionais como a ausência de notificações de desafios recebidos.

As principais contribuições da nova versão são: eliminação das barreiras de instalação com acesso inteiramente via navegador; sistema de torneios eliminatório com chaveamento visual e *replay* de partidas; assistente pedagógico com IA em três perfis de comunicação; histórico completo de interações que permite ao professor acompanhar a evolução individual de cada estudante; e arquitetura extensível que facilita a adição de novos jogos sem alteração do código existente.

Como trabalhos futuros, pretende-se: implementar um sistema de notificações para desafios pendentes, reduzindo o gargalo competitivo identificado; conduzir avaliações formais com questionários motivacionais pré e pós-atividade, replicando a metodologia da versão original e permitindo comparação direta dos indicadores; estender o estudo a múltiplas turmas e instituições, a fim de avaliar a escalabilidade e a generalização dos resultados; e investigar o impacto específico do estilo de comunicação do assistente sobre os indicadores de auto-eficácia e percepção de aprendizagem dos estudantes.

Agradecimentos

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) – Brasil – Código de Financiamento 001. Também conta com financiamento no âmbito do projeto "Wanda: Card Games para Ensino de Lógica de Programação", processo 88887.180797/2025-00 contemplado pelo Edital nº 3/2025 CAPES “InovaEducação”.

Referências

Bayliss, J. D. e Strout, S. (2006). Games as a"flavor"of cs1. In *Proceedings of the 37th SIGCSE technical symposium on Computer science education*, pages 500–504, New York, NY, USA. ACM.

Carvalho, L., Santos, A., Nakamura, F., e Oliveira, E. (2019). Detecção precoce de evasão em cursos de graduação presencial em computação: um estudo preliminar. In *Anais do XXVII Workshop sobre Educação em Computação*, pages 233–243, Porto Alegre, RS, Brasil. SBC.

de Pontes, R. G., Guerrero, D. D. S., e de Figueiredo, J. C. A. (2019). Analyzing gamification impact on a mastery learning introductory programming course. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, pages 400–406. ACM.

Dicheva, D., Dichev, C., Agre, G., e Angelova, G. (2015). Gamification in education: A systematic mapping study. *Journal of educational technology & society*, 18(3):75.

Drumond, R. R., Brandao, A., e Salles, C. (2014). Wanda: a framework to develop card based games to help motivate programming students. In *2014 Brazilian Symposium on Computer Games and Digital Entertainment*, pages 158–164. IEEE.

Ibiapino, L., Drumond, R., e Soares Neto, C. (2025). Wanda: card games para o ensino de lógica de programação. Repositório eduCAPES. Software. Disponível em: `https://educapes.capes.gov.br/handle/capes/983894`. Acesso em: 30 jun. 2026.

Kinnunen, P. e Malmi, L. (2006). Why students drop out cs1 course? In *Proceedings of the second international workshop on Computing education research*, pages 97–108.

Kroustalli, C. e Xinogalos, S. (2021). Studying the effects of teaching programming to lower secondary school students with a serious game: a case study with Python and CodeCombat. *Education and Information Technologies*, 26(5):6069–6095.

Petersen, A., Craig, M., Campbell, J., e Tafliovich, A. (2016). Revisiting why students drop cs1. In *Proceedings of the 16th Koli Calling International Conference on Computing Education Research*, pages 71–80.

Prensky, M. (2003). Digital game-based learning. *Computers in entertainment (CIE)*, 1(1):21–21.

Radtke, P. V. e Binder, F. (2010). Chien 2d: A multiplatform library to teach the c language through games programming. *Anais Simpsio Brasileiro de Jogos e Entretenimento Digital*.

Rebouças, A., Marques, D. L., Costa, L. F. S., e Silva, M. d. A. (2010). Aprendendo a ensinar programação combinando jogos e python. In *Anais do Simpósio Brasileiro de Informática na Educação*, volume 1, pages 1–10.

Ribeiro, F., Merlin, B., e Fülber, H. (2019). Avaliação do impacto de ambientes gamificados no processo de ensino-aprendizagem de programação de computadores: uma comparação entre elementos monousuário e multiusuário. In *Anais do Simpósio Brasileiro de Informática na Educação*, pages 803–812, Brasil. SBC.

Zhang, Z., Dong, Z., Shi, Y., Price, T., Matsuda, N., e Xu, D. (2024). Students' perceptions and preferences of generative artificial intelligence feedback for programming. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 23250–23258. AAAI Press.