ENTRE O CÓDIGO E A SALA DE AULA: MEDIAÇÃO PEDAGÓGICA DIGITAL, REVISÃO TÉCNICA E FORMAÇÃO DO PROFESSOR DE COMPUTAÇÃO

BETWEEN CODE AND THE CLASSROOM: DIGITAL PEDAGOGICAL MEDIATION, TECHNICAL REVIEW, AND COMPUTING TEACHER TRAINING

https://doi.org/10.63330/aurumpub.062-082

José Ivan Vitor Cordeiro
Especialista em Gestão de Pessoas
Universidade Estadual do Ceará
E-mail: jose.ivan@ifce.edu.br
ORCID: https://orcid.org/0009-0002-2545-700X

Resumo

O estudo investiga a coerência entre os compromissos pedagógicos declarados por uma ferramenta digital de mediação para escrita acadêmica, o Orientador Digital de TCC, e sua implementação técnica real. Trata-se de pesquisa aplicada, de natureza qualitativa, fundamentada nos conceitos de escrita acadêmica como prática social situada, de zona de desenvolvimento proximal e no framework de conhecimento tecnológico, pedagógico e de conteúdo, que orientam a análise de um sistema client-side estruturado em torno de geração aumentada por recuperação. A metodologia combinou inspeção de código-fonte, uma primeira rodada de testes automatizados que reimplementavam a lógica do sistema de forma isolada e, posteriormente, a execução real do código em ambiente de navegador simulado, o que permitiu comparar a capacidade de detecção de falhas de cada abordagem. Os resultados revelaram cinco falhas silenciosas relacionadas ao armazenamento e ao processamento de dados, entre elas a sobrescrita de conteúdo entre duas estruturas de documento distintas, além de duas barreiras de acesso ligadas à configuração técnica antes exigida do usuário. Todas as falhas foram corrigidas e revalidadas, e a mudança de método de verificação revelou dois defeitos adicionais que a primeira abordagem não havia detectado. A partir desses achados, propõe-se um roteiro de dez pontos de verificação, reaproveitável na auditoria de tecnologias educacionais semelhantes. Conclui-se que compromissos pedagógicos declarados por sistemas educacionais baseados em inteligência artificial exigem verificação técnica contínua, e que essa prática constitui exercício formativo relevante para a formação do professor de computação.

Palavras-chave: Auditoria de software educacional; Formação de professores de computação; Inteligência artificial na educação; Mediação pedagógica digital; TPACK.

Abstract

This study investigates the coherence between the pedagogical commitments declared by a digital mediation tool for academic writing, the Orientador Digital de TCC, and its actual technical implementation. It is an applied, qualitative study grounded in the concepts of academic writing as a situated social practice, the zone of proximal development, and the technological pedagogical content knowledge framework, which guide the analysis of a client-side system structured around retrieval-augmented generation. The methodology combined source-code inspection, an initial round of automated tests that reimplemented the system's logic in isolation, and, subsequently, real execution of the code in a simulated browser environment, allowing a comparison of the fault-detection power of each approach. Results revealed five silent failures related to data storage and processing, including content overwriting between two distinct document structures, along with two access barriers linked to technical configuration previously required from the user. All failures were corrected and revalidated, and the shift in verification method revealed two additional defects that the first approach had failed to detect. Based on these findings, a ten-point verification checklist is proposed, reusable for auditing similar educational technologies. The study concludes that pedagogical commitments declared by AI-based educational systems require continuous technical verification, and that this practice constitutes a relevant formative exercise for computer science teacher education.

Keywords: Artificial intelligence in education; Computer science teacher education; Digital pedagogical mediation; Educational software auditing; TPACK.

1 INTRODUÇÃO

Escrever um trabalho de conclusão de curso costuma ser descrito, por quem já passou pela experiência, como uma travessia solitária: não falta conteúdo a dizer, falta clareza sobre como dizê-lo dentro das convenções que a escrita científica exige. Street (2014) oferece uma chave de leitura útil para essa dificuldade ao tratar a escrita acadêmica não como habilidade técnica neutra, mas como prática social situada, aprendida por meio da inserção numa comunidade discursiva com regras próprias e raramente explicitadas a quem está entrando nela. Freire (2011), por ângulo complementar, sustenta que transformar experiência em discurso científico exige mediação, diálogo e tempo, recursos que se tornam escassos em contextos de turmas numerosas e orientação limitada.

É desse cenário de mediação escassa que nasce a justificativa para ferramentas digitais de apoio à escrita, não como substitutas do orientador humano, mas como dispositivos capazes de ampliar os poucos espaços de acompanhamento disponíveis. Vygotsky (2007) contribui com o conceito de zona de desenvolvimento proximal, segundo o qual a aprendizagem se beneficia de apoio temporário que se retira

progressivamente à medida que o aprendiz ganha autonomia. Aplicado a sistemas digitais, esse conceito distingue um mediador pedagógico genuíno, capaz de explicar o motivo de suas sugestões, de uma automação que apenas entrega texto pronto.

Toda tecnologia que se propõe a preencher essa lacuna de mediação carrega, ao mesmo tempo, uma promessa e um risco: a promessa de democratizar o acesso a um acompanhamento historicamente dependente da disponibilidade de um bom orientador, e o risco de que a automação, mal calibrada, substitua a mediação por uma aparência de mediação. Um recurso arquitetural relevante para conter esse risco é a geração aumentada por recuperação, que restringe as respostas de um modelo de linguagem aos materiais fornecidos pelo próprio usuário, preservando parte do esforço intelectual de leitura e síntese crítica que caberia ao estudante.

A análise de sistemas que integram esses compromissos pedagógicos com decisões concretas de engenharia de software encontra respaldo no framework de conhecimento tecnológico, pedagógico e de conteúdo, o TPACK, proposto por Mishra e Koehler (2006), segundo o qual o professor eficaz no uso de tecnologia não domina isoladamente os três conhecimentos, mas a interseção dinâmica entre eles. Grande parte dos currículos de licenciatura em computação ainda trata esses conhecimentos como disciplinas paralelas, o que reforça a relevância formativa de exercícios em que o próprio licenciando examina o código de um sistema que concebeu, confrontando-o com os compromissos pedagógicos que esse código deveria sustentar.

Diante desse quadro, este trabalho tem como objetivo geral verificar, por meio de inspeção de código e execução real do sistema, se a implementação técnica do Orientador Digital de TCC, uma ferramenta de mediação pedagógica baseada em inteligência artificial para apoio à escrita de trabalhos de conclusão de curso, sustenta os compromissos pedagógicos declarados em sua concepção conceitual. Como objetivos específicos, busca-se: identificar inconsistências técnicas capazes de comprometer a autoria e a integridade do conteúdo produzido pelo estudante; comparar a capacidade de detecção de falhas entre testes automatizados isolados e execução real do sistema; e sistematizar um roteiro de verificação reaproveitável para auditoria de tecnologias educacionais semelhantes. Justifica-se a pesquisa pela lacuna, ainda pouco explorada na literatura, entre o discurso pedagógico que acompanha tecnologias educacionais baseadas em inteligência artificial e a verificação técnica sistemática desses mesmos compromissos.

2 METODOLOGIA

Trata-se de pesquisa aplicada, de abordagem qualitativa, conduzida sob o formato de estudo de caso técnico único, tendo como objeto o Orientador Digital de TCC, sistema inteiramente client-side construído em HTML5, CSS3 e JavaScript puro, sem uso de frameworks de front-end, com extração de PDF via biblioteca PDF.js, importação de DOCX via Mammoth.js, exportação para Word e PDF via docx e jsPDF,

persistência de dados via Web Storage API e um backend mínimo em Cloudflare Workers responsável por mediar as chamadas ao modelo de inteligência artificial.

A verificação técnica seguiu dois níveis, mantidos deliberadamente distintos, conforme prática consolidada de inspeção de software (Fagan, 1976): o primeiro nível consistiu na leitura direta do código-fonte, orientada pela busca de inconsistências entre o comportamento implementado e os compromissos pedagógicos declarados na concepção do sistema, em especial a preservação da autoria do estudante e a ampliação do acesso à escrita científica. O segundo nível consistiu na validação funcional das correções propostas.

A validação funcional foi conduzida em duas etapas sucessivas, que constituem, elas próprias, um resultado metodológico deste trabalho. Na primeira etapa, a lógica corrigida foi reimplementada de forma isolada em scripts de teste executados em ambiente Node.js, seguindo prática comum em contextos onde o sistema real depende de um navegador completo e é custoso de isolar para testes automatizados (Wohlin et al., 2012). Na segunda etapa, adotada após um relato de falha em uso manual motivar nova investigação, o próprio arquivo do sistema foi carregado e executado por meio da biblioteca jsdom, que recria em Node.js uma implementação do DOM e das principais APIs de navegador, permitindo chamar e verificar diretamente as funções reais do sistema, sem reescrevê-las.

Como instrumento de sistematização dos achados, foi elaborado um roteiro de dez pontos de verificação, construído a posteriori a partir das categorias de falha efetivamente encontradas, e apresentado neste trabalho como resultado reaproveitável para auditoria de tecnologias educacionais semelhantes. Por se tratar de auditoria conduzida por uma única pessoa, também autora do sistema analisado, reconhece-se como limitação metodológica o risco de viés de expectativa, retomado na seção de resultados e discussão.

3 RESULTADOS E DISCUSSÃO

3.1 FALHAS IDENTIFICADAS NA INSPEÇÃO DE CÓDIGO

A inspeção do código-fonte revelou um problema estrutural no gerenciamento de dados do sistema: as duas estruturas de documento oferecidas pela ferramenta, uma para trabalhos de conclusão de curso e outra para artigos científicos, compartilhavam identificadores internos idênticos para seções equivalentes, de modo que alternar entre os dois modos podia sobrescrever silenciosamente o conteúdo já produzido pelo estudante, sem qualquer aviso. Esse achado contraria diretamente o compromisso de preservação da autoria que fundamenta, do ponto de vista ético, a arquitetura do sistema. A correção introduziu um prefixo de modo na chave de armazenamento de cada seção, isolando de forma definitiva as duas estruturas de documento.

A esse achado principal somaram-se outros quatro problemas de menor gravidade individual, mas relevantes em conjunto por compartilharem uma mesma característica: o sistema falhava silenciosamente,

sem comunicar ao usuário que algo havia dado errado. O Quadro 1 sintetiza os cinco achados e as respectivas correções.

Quadro 1 — Falhas identificadas na inspeção de código e correções aplicadas

Falha identificada	Efeito sobre o usuário	Correção aplicada
Chaves de armazenamento sem separação por modo de documento	Conteúdo de TCC e de artigo se sobrescreviam ao alternar modos	Prefixo de modo adicionado à chave de cada seção
Evidências de múltiplos PDFs substituídas, não somadas	Apenas o último documento anexado ficava disponível à IA	Acúmulo de evidências implementado
PDFs digitalizados sem texto extraível reportados como sucesso	Estudante não percebia que nenhum conteúdo havia sido extraído	Aviso explícito de extração vazia
Exportação para Word sem formatação ABNT	Estudante precisava reformatar manualmente o documento	Exportação ajustada às normas ABNT
Ausência de tempo-limite nas chamadas à IA	Interface podia travar indefinidamente diante de falha remota	Cancelamento automático da requisição após 45s

Fonte: elaborado pelo autor (2026), a partir da inspeção do código-fonte do sistema.

3.2 MÉTODO DE TESTE COMO VARIÁVEL DE RESULTADO

A primeira rodada de validação, baseada em testes que reimplementavam isoladamente a lógica corrigida, passou integralmente, o que trouxe uma falsa sensação de segurança: esses testes avaliavam uma reconstrução simplificada do raciocínio do sistema, não o código real em execução. A mudança para execução real via jsdom revelou dois defeitos genuínos que a primeira abordagem não detectava: uma função de extração de texto de PDF que se declarava assíncrona mas retornava antes de a leitura do arquivo terminar, e um problema de normalização do endereço do servidor de inteligência artificial quando salvo sem o prefixo de protocolo. Esse segundo achado foi confirmado também em uso real, com registros do sistema mostrando o endereço salvo corretamente e requisições concluídas com sucesso após a correção.

Esse episódio constitui, em si, um resultado metodológico relevante: evidencia que validar a lógica reimplementada de um sistema é uma forma de evidência mais fraca do que validar o sistema em execução, e que essa diferença pode ocultar defeitos reais mesmo quando todos os testes reportam sucesso. A crescente disponibilidade de ferramentas capazes de simular ambientes de navegador em linha de comando reduz o custo dessa armadilha para aplicações web, tornando mais acessível, mesmo para projetos de pequeno porte, adotar desde o início o padrão de verificação mais rigoroso.

3.3 BARREIRAS DE ACESSO CORRIGIDAS

Além dos defeitos de código, a análise identificou duas barreiras de acesso incorporadas silenciosamente ao design da ferramenta: a ausência de uma função dedicada para iniciar um novo projeto sem apagar a configuração técnica já realizada, e a exigência de que cada novo usuário configurasse manualmente o endereço do servidor de inteligência artificial antes de poder usar qualquer funcionalidade de geração automática. Essa segunda barreira, embora pouco relevante para um usuário com afinidade

técnica, era suficiente para inviabilizar o uso da ferramenta por um estudante de ensino médio ou dos primeiros períodos da graduação. Ambas as barreiras foram corrigidas, a primeira com uma função de reinício que preserva a configuração técnica, e a segunda com a definição de um servidor padrão pré-configurado, mantendo a possibilidade de configuração manual para usuários avançados.

Essas correções relacionam-se diretamente à fundamentação teórica apresentada na introdução: um sistema que exige configuração técnica prévia reproduz, em menor escala, o mesmo problema de mediação escassa que a proposta original do sistema pretendia resolver, apenas deslocando a barreira da falta de um orientador humano disponível para a falta de conhecimento técnico necessário para configurar a ferramenta.

3.4 IMPLICAÇÕES PARA A FORMAÇÃO DO PROFESSOR DE COMPUTAÇÃO

Lido à luz do TPACK (Mishra; Koehler, 2006), o processo de identificar uma inconsistência técnica, avaliar seu impacto pedagógico e implementar uma correção validável constitui exemplo concreto da interseção entre conhecimento tecnológico, pedagógico e de conteúdo esperada do professor de computação. Os resultados deste trabalho sugerem uma extensão ao framework original: a interseção entre os três tipos de conhecimento não se esgota no momento de planejar ou conceber uma tecnologia educacional, mas precisa ser mobilizada de forma contínua, na forma de auditoria periódica de tecnologias já em uso, já que um sistema pedagogicamente coerente no momento de sua concepção pode deixar de sê-lo à medida que evolui ou passa a ser usado em contextos diferentes daqueles para os quais foi originalmente pensado.

3.5 LIMITAÇÕES

A verificação funcional realizada permanece unipessoal, conduzida pela mesma pessoa que desenvolveu o sistema analisado, o que introduz risco de viés de expectativa e não substitui a observação de múltiplos usuários em condições variadas de navegador, dispositivo e conexão. A correção que eliminou a barreira de configuração manual, ao expor publicamente o endereço do servidor no código-fonte da página, introduziu ainda um risco operacional de uso indevido, cuja mitigação por limitação de requisições permanece como tarefa de configuração de infraestrutura pendente. Não foi conduzida validação empírica com múltiplos estudantes em contexto real de produção de trabalhos acadêmicos, tampouco testes de usabilidade moderados por terceiros, lacunas retomadas como recomendação de pesquisa futura.

4 CONCLUSÃO

Este trabalho teve como objetivo verificar se a implementação técnica do Orientador Digital de TCC sustenta os compromissos pedagógicos declarados em sua concepção. A resposta, construída por meio de inspeção de código e execução real do sistema, é parcialmente afirmativa: os compromissos declarados

eram sustentados apenas em parte pela implementação original, que apresentava cinco falhas silenciosas e duas barreiras de acesso capazes de comprometer justamente a autoria e a ampliação de acesso que fundamentam, do ponto de vista pedagógico, a existência do sistema. Todas as falhas foram corrigidas e revalidadas.

Um resultado secundário, mas não menos relevante, diz respeito ao próprio processo de verificação: a substituição de testes isolados por execução real do código revelou defeitos que a primeira abordagem não detectava, evidenciando que o rigor de uma verificação técnica depende tanto do que se testa quanto de como se testa. Como contribuição prática, sistematizou-se um roteiro de dez pontos de verificação, reaproveitável na auditoria de tecnologias educacionais semelhantes, tanto por desenvolvedores quanto por educadores sem formação técnica aprofundada.

Conclui-se que compromissos pedagógicos declarados por tecnologias educacionais baseadas em inteligência artificial não podem ser avaliados apenas no plano das intenções, sendo necessário submetê-los a verificação técnica sistemática, sob pena de reproduzir na prática os riscos que o desenho conceitual pretendia evitar. Para trabalhos futuros, recomendam-se testes de usabilidade moderados com estudantes reais, conduzidos por observador externo ao autor do sistema, a implementação de testes automatizados de regressão contínua, e a investigação empírica do impacto da ferramenta sobre a autonomia e a permanência de estudantes ao longo da elaboração de seus trabalhos de conclusão de curso.

REFERÊNCIAS

BOGHI, C.; SHITSUKA, D. M.; SHITSUKA, R. Metodologias ativas: um estudo de caso do fenômeno aprender. **Educação & Linguagem**, v. 20, n. 2, p. 143, 2018.

CORDEIRO, J. I. V. Do código à mediação pedagógica: desenvolvimento de um orientador digital para apoio à escrita do Trabalho de Conclusão de Curso (TCC). **Research, Society and Development**, v. 15, n. 1, e8215150578, 2026. DOI: 10.33448/rsd-v15i1.50578.

FAGAN, M. E. Design and code inspections to reduce errors in program development. **IBM Systems Journal**, v. 15, n. 3, p. 182-211, 1976.

FREIRE, P. **A importância do ato de ler**: em três artigos que se completam. São Paulo: Cortez, 2011.

MISHRA, P.; KOEHLER, M. J. Technological pedagogical content knowledge: a framework for teacher knowledge. **Teachers College Record**, v. 108, n. 6, p. 1017-1054, 2006.

STREET, B. V. **Letramentos sociais**: abordagens críticas do letramento no desenvolvimento, na etnografia e na educação. São Paulo: Parábola Editorial, 2014.

VYGOTSKY, L. S. **A formação social da mente**. 7. ed. São Paulo: Martins Fontes, 2007.

WOHLIN, C. et al. **Experimentation in software engineering**. Berlin: Springer, 2012.