Revista Pleiade – Centro Universitário Descomplica UniAmérica
ISSN 1980-8666 (Impresso) – ISSN 2674-8231 (Eletrônico) – Classificação Capes B3 (2017-2020)

Inteligência Artificial e Educação Personalizada: Análise da Plataforma Descomplica sob a Perspectiva da Engenharia de Software

Artificial Intelligence and Personalized Education: Analysis of the Descomplica Platform from a Software Engineering Perspective

Mateus Davi Rodrigues Alcantara Pinto[1] e Guilherme Bezzon[2]

1. Acadêmico de Bacharelado em Engenharia de Computação. Pós-graduado em Engenharia de Software e Data Science no Centro Universitário UniAmérica Descomplica. 2. Graduado em Bacharelado em Engenharia Mecânica. Mestre e Doutor em Planejamento de Sistemas Energéticos. Coordenador do curso de graduação Engenharia de Software do Centro Universitário UniAmérica Descomplica. https://orcid.org/0000-0002-1211-0974.

mateusdavi@hotmail.com e *guilherme.bezzon@descomplica.com.br*

Palavras-chave
Educação Digital
Engenharia de Software
Inovação
Inteligência Artificial

Keywords
Digital Education
Software Engineer
Innovation
Artificial Inteligence

Resumo:
Propõe-se estudo da Engenharia de Software sob a perspectiva científica e aplicada, destacando fundamentos conceituais, processos de desenvolvimento, engenharia de requisitos, algoritmos, estruturas de dados e ferramentas computacionais, com ênfase no uso da Inteligência Artificial (IA) como suporte à detecção de falhas, automação de tarefas e aprimoramento de soluções. A pesquisa adota abordagem exploratória, baseada em revisão bibliográfica e análise de experiências práticas, utilizando como estudo de caso o modelo educacional digital do Centro Universitário Uniamérica Descomplica. O modelo integra metodologias ativas, laboratórios remotos, monitorias, tutoriais semanais e recursos de IA voltados à aprendizagem personalizada. A integração entre teoria e prática favorece o desenvolvimento da autonomia, do raciocínio lógico e da capacidade de organização de requisitos e validação de soluções computacionais. Sistemas de IA podem auxiliar usuários sem formação aprofundada em programação a desenvolver aplicações funcionais, desde que haja clareza comunicacional e estruturação adequada das instruções fornecidas. Conclui-se, IA atua como ferramenta de aceleração cognitiva e operacional, mas permanece dependente da supervisão humana, do planejamento, da modelagem e da validação técnica, reafirmando princípios clássicos da Engenharia de Software.

Abstract:
This study explores Software Engineering from both a scientific and applied perspective, highlighting conceptual foundations, development processes, requirements engineering, algorithms, data structures, and computational tools, with an emphasis on the use of Artificial Intelligence (AI) to support fault detection, task automation, and solution improvement. The research adopts an exploratory approach, based on a literature review and analysis of practical experiences, using the digital educational model of the Uniamérica Descomplica University Center as a case study. This model integrates active methodologies, remote laboratories, tutoring, weekly tutorials, and AI resources geared towards personalized learning. The integration of theory and practice fosters the development of autonomy, logical reasoning, and the ability to organize requirements and validate computational solutions. AI systems can assist users without in-depth programming training in developing functional applications, provided there is clear communication and adequate structuring of the instructions provided. In conclusion, AI acts as a tool for cognitive and operational acceleration, but remains dependent on human supervision, planning, modeling, and technical validation, reaffirming classic principles of Software Engineering.

Artigo recebido em: 05.03.2026.
Aprovado para publicação em: 06.05.2026.

INTRODUÇÃO

Desde períodos remotos, a humanidade tem buscado estratégias para otimizar o tempo, reduzir esforços e ampliar a eficiência na execução de tarefas. À medida que a capacidade cognitiva evoluiu, foram desenvolvidas ferramentas, técnicas e tecnologias voltadas à resolução de problemas práticos. O avanço tecnológico resultou no surgimento de máquinas, circuitos eletrônicos, computadores e linguagens de programação, como Java, Python e C++, além de sistemas baseados em Inteligência Artificial, ampliando significativamente as possibilidades de automação e processamento de informações. Associadas a bibliotecas e *frameworks*, essas tecnologias contribuíram para a abstração de complexidades técnicas, permitindo que profissionais de desenvolvimento concentrem esforços em aspectos funcionais e nas regras de negócio dos sistemas.

Com o tempo, surgiu a necessidade de profissional de software com foco em organização, qualidade, funcionabilidade, manutenção, disponibilidade do sistema, eficiência algorítmica, robustez ao projeto com abordagem flexível em relação à demanda do cotidiano. Nesse contexto, a engenharia de software tornou-se atividade essencial para garantir qualidade, organização e eficiência na criação de soluções computacionais.

Este estudo tem como objetivo analisar o desenvolvimento de sistemas computacionais sob a ótica da Engenharia de Software, enfatizando o papel do fator humano na concepção, modelagem e validação de soluções, bem como a relevância de algoritmos, estruturas de dados e processos sistematizados de desenvolvimento. Investiga-se, ainda, a Inteligência Artificial (IA) como tecnologia de apoio ao ciclo de vida do software, com potencial para auxiliar na detecção de falhas, identificação de inconsistências, automação de tarefas e proposição de melhorias de eficiência. Como complemento empírico, incorpora-se o estudo de caso do modelo educacional digital da Descomplica, no qual práticas pedagógicas mediadas por tecnologias e recursos de IA são aplicadas à formação em Engenharia de Software, evidenciando a utilização dessa ferramenta como instrumento de suporte ao raciocínio técnico, e não como substituição da análise humana no processo de desenvolvimento.

METODOLOGIA

O estudo caracteriza-se como uma pesquisa exploratória, de abordagem qualitativa, fundamentada em revisão bibliográfica. O objetivo metodológico consistiu em analisar o papel do ser humano na criação de sistemas computacionais, com ênfase na Engenharia de Software, algoritmos, estruturas de dados e na Inteligência Artificial como ferramenta de apoio ao desenvolvimento de soluções tecnológicas.

As fontes de dados foram obtidas por meio de acesso remoto à biblioteca digital institucional, constituindo a população de referência publicações científicas e técnicas relevantes na área. A amostra foi composta por obras clássicas e autores reconhecidos, como Somerville (2018), Pfleeger (2004), Luger (2013), Ascencio (2012), entre outros, que serviram de base teórica para a análise dos conceitos abordados.

Os procedimentos adotados incluíram levantamento, seleção e análise crítica das referências, além de análises comparativas entre diferentes abordagens de desenvolvimento de software, com ênfase na integração da Inteligência Artificial como suporte aos processos de criação, depuração e manutenção de sistemas. Como etapa complementar, realizou-se análise descritiva do modelo educacional digital da Descomplica como estudo de caso aplicado ao contexto acadêmico de formação em Engenharia de Software. Para organização e produção textual, foram utilizadas ferramentas digitais do pacote Microsoft 365, permitindo sistematizar as informações e atender aos objetivos propostos no estudo.

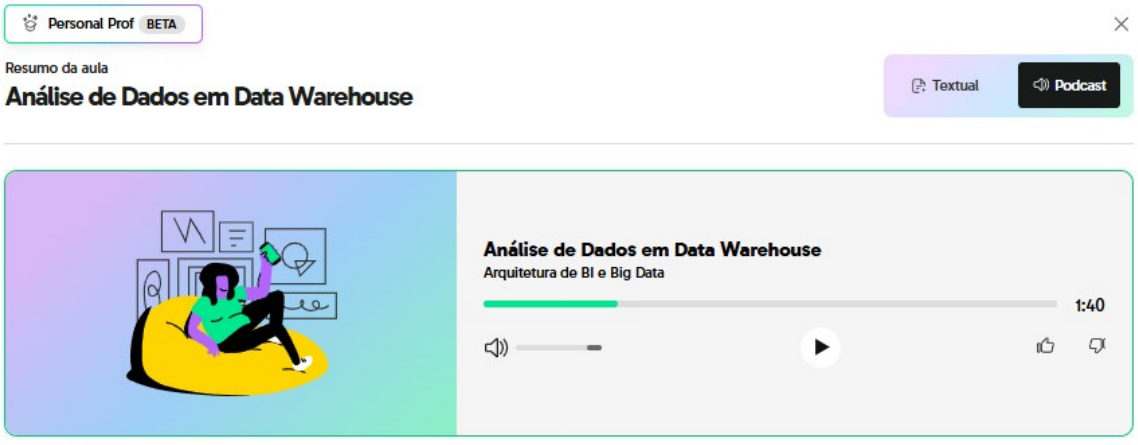
RESULTADOS E DISCUSSÕES

O curso de Engenharia de Software deve proporcionar formação sólida e abrangente, capacitando o estudante com conhecimentos teóricos e práticos essenciais para o uso eficiente de ferramentas tecnológicas. Além disso, desenvolve competências técnicas e analíticas fundamentais para profissionais que almejam atuar na área de desenvolvimento de software, abrangendo desde sistemas baseados em inteligência artificial, servidores e aplicações *desktop*, até interfaces visuais voltadas à experiência do usuário.

A estrutura curricular contempla unidades curriculares das ciências exatas e de lógica de programação, que são cruciais para o desenvolvimento do raciocínio lógico necessário à criação de soluções robustas, dinâmicas e seguras. Nesse contexto, o engenheiro de software deve considerar não apenas o desenvolvimento da aplicação, mas também aspectos relacionados à arquitetura de dados, como o armazenamento em servidores, a estruturação e o gerenciamento eficiente das informações.

Na plataforma de ensino da Descomplica, utilizada nos cursos do Centro Universitário Uniamérica Descomplica, a metodologia adotada em Engenharia de Software favorece a formação de profissionais aptos a atuar tanto no mercado quanto em pesquisa acadêmica. O modelo pedagógico inclui recursos como aulas laboratoriais, tutoriais semanais, monitorias e atividades práticas, todos acessíveis remotamente, permitindo ao estudante aprender no seu próprio ritmo e no conforto de sua residência (Imagem 1). O discente pode optar em ler o resumo inteligente ou ouvir um podcast sobre o conteúdo.

Imagem 1. Ferramenta de podcast de conteúdo



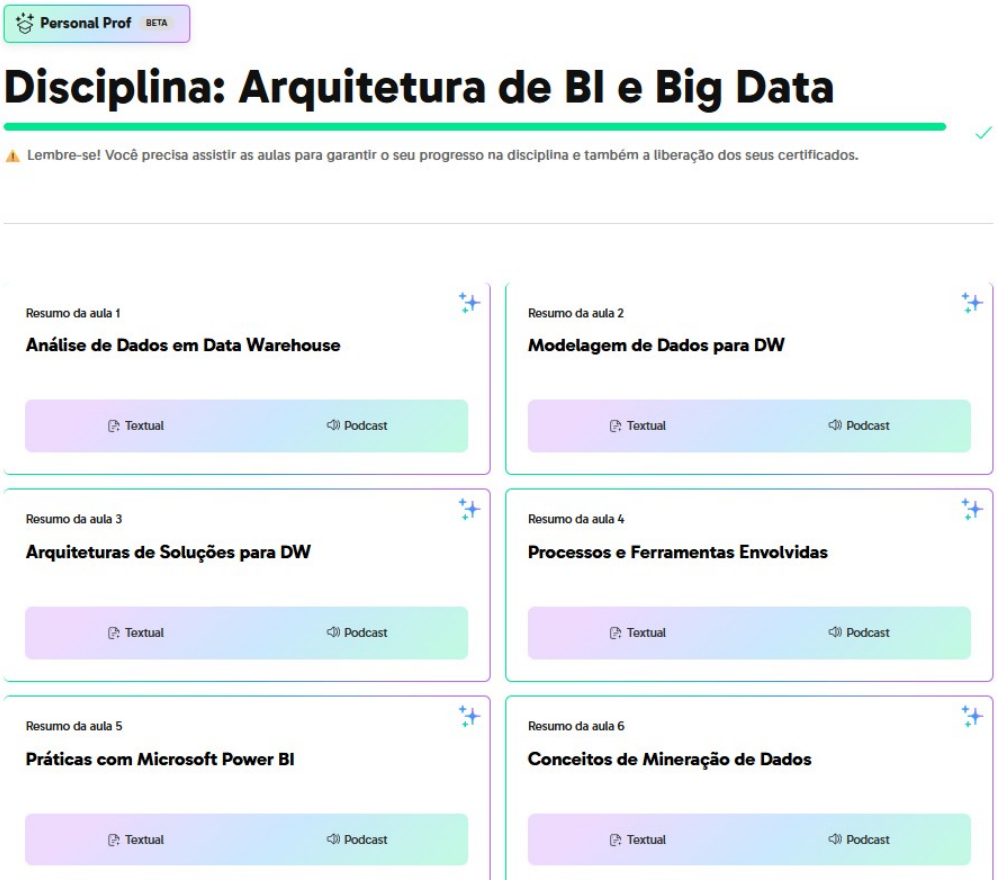
Fonte: Adaptada da plataforma da Descomplica.

A metodologia de ensino utilizada nessa plataforma de ensino exerce influência significativa na qualidade da educação oferecida ao aluno. Segundo Moran (2013, p. 39-40), a educação deve ser compreendida como um processo integrador, centrado nas necessidades individuais dos alunos. Essa abordagem proporciona maior flexibilidade no aprendizado e prepara os estudantes para lidar com as incertezas do mundo, especialmente por meio do uso de tecnologias. Com a crescente democratização dos equipamentos eletrônicos e da internet, mais pessoas têm acesso à educação, podendo aprender no seu próprio ritmo e conciliar os estudos com outras atividades, como o trabalho. A Imagem 2 apresenta o acesso a uma seção de resumos inteligentes da disciplina Arquitetura de BI e Big Data.

Para colaborar com a evolução individual dos estudantes na plataforma da Descomplica, verifica-se a adoção gradual de recursos como resumos inteligentes por disciplina e professores particulares integrados à

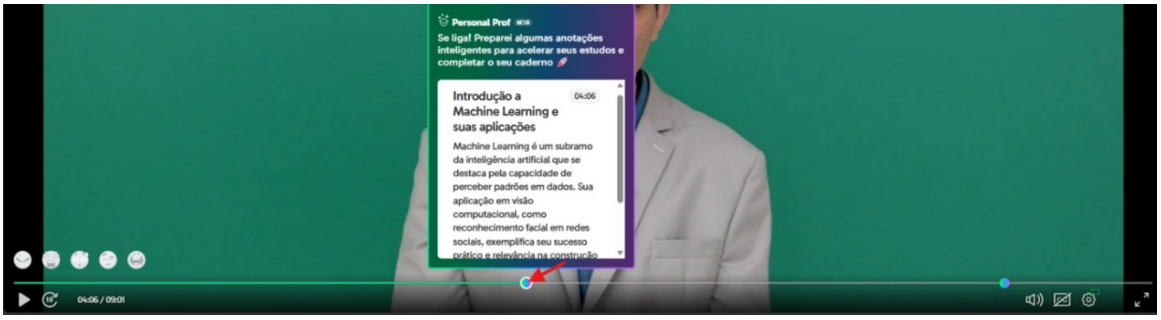
inteligência artificial, promovendo um ambiente de aprendizagem mais confortável e personalizado. Nesse sentido, os autores Mello, Almeida Neto e Costa (2024, p. 43-44) defendem que a educação inteligente não visa substituir os métodos tradicionais, mas sim, otimizar por meio da aplicação de tecnologias de IA, ampliando as possibilidades de ensino e aprendizagem. (Imagens 3 e 4)

Imagem 2. Seção de resumo inteligente aplicada no curso de Pós-graduação em Engenharia de Software



Fonte: Imagem adaptada da plataforma da Descomplica.

Imagem 3. Resumo inteligente na linha do tempo de vídeo-aula do curso de Engenharia de Computação



Fonte: Imagem adaptada da plataforma da Descomplica.

Imagem 4. Resumo inteligente de disciplina do curso de Engenharia de Computação

Fonte: Imagem adaptada da plataforma da Descomplica.

De acordo com os autores Mello, Almeida Neto e Costa (2024, p. 48), a adoção da inteligência artificial nas salas de aula não tem como objetivo substituir os professores, mas sim oferecer suporte às suas atividades. Com isso, é possível que os docentes atendam a um número maior de alunos sem sobrecarga. A inteligência artificial já é uma realidade em grande parte das salas de aula e na vida cotidiana. Sua aplicação no ambiente educacional promove equilíbrio entre as expectativas dos alunos e a entrega de um ensino mais eficiente, reforçando que a educação não tem limites.

A tecnologia nos auxilia em tarefas, fazendo com que a capacidade humana possa transcender as nossas limitações e atingir a perfeição esperada tanto na área científica quanto em outros segmentos. Entretanto, o funcionamento adequado dos sistemas computacionais não ocorre de forma autônoma ou isenta de limitações. Os dispositivos físicos, em geral, compostos por estruturas metálicas e componentes eletrônicos dependem da intervenção humana para sua concepção, programação, operação e manutenção, a fim de que apresentem desempenho condizente com os objetivos para os quais foram projetados, como afirmam Ascencio e Campos (2012, p. 17):

> … o computador pode auxiliá-lo em qualquer tarefa. É consciente, trabalhador, possui muita energia, mas não tem iniciativa, nenhuma independência, não é criativo nem inteligente, por isso, precisa receber instruções nos mínimos detalhes.

Com a inteligência artificial não é diferente, ela necessita do ser humano para fornecer instruções nos mínimos detalhes, para performar e executar tarefas da forma que esperamos. Essa ferramenta precisa receber parâmetros, objetivos e validações. LUGER (2013, p. 3) destaca que:

> … a inteligência artificial, em seu interesse muito direto no dom de Prometeu, tem sido aplicada a todas as áreas de seu legado - medicina, psicologia, biologia, astronomia, geologia - e a muitas áreas de empreendimento científico que nem Ésquilo poderia ter imaginado.

Seguindo nesse contexto, a engenharia de software surge como uma área de conhecimento fundamental para garantir que sistemas computacionais sejam desenvolvidos de forma estruturada, eficiente e confiável.

Assim como a inteligência artificial, a engenharia de software também enfrenta desafios complexos que exigem uma compreensão profunda do problema antes de aplicar soluções tecnológicas. Ao mergulhar nesse

oceano digital e conforme aprofundamos, surgem uma série de dúvidas como: O que é engenharia de software? Para o que serve a engenharia de software? Como esse segmento organiza e executa todas essas abordagens tecnológicas? A inteligência artificial fará tudo sem a necessidade de um humano? Afinal de contas, essa ferramenta é poderosíssima! Será que o destino seria tão cruel? Será que a consideração de Prometeu foi a ruína de nós mortais? Ao proporcionar o fogo dos deuses, para a iluminação da mente humana foi uma calmaria antes da tempestade, que incidirá em trevas?

Afastando das trevas, a engenharia de software tem como objetivo solucionar problemas que envolvem sistemas e computação por meio do uso de ferramentas, diagramas, técnicas, casos de uso e análise de dados. Ela proporciona qualidade, segurança e soluções para os problemas propostos com metodologias profissionais, mantendo o sistema sempre pronto e operante para o usuário final. Mas, existem problemas muito complexos computacionalmente como NP-hard ou NP-completos, que utilizam muito processamento computacional para resolver um determinado problema, como por exemplo xadrez e o sudoku, que ainda não existe uma solução eficiente. Os problemas NP-hard e NP-completos são classes de problemas na ciência da computação que são complexos de resolver, mas é possível produzir o fenômeno ao custo de hardware. Esses problemas exigem abordagens forçadas que necessitam de muitos recursos computacionais, como tempo de processamento e memória, para mover peças ou adicionar um número em uma posição específica do sudoku. PFLEEGER (2004, p. 2) afirma que:

> Como engenheiros de software, utilizamos nosso conhecimento sobre computadores e computação para ajudar a resolver problemas. Frequentemente, o problema com o qual estamos lidando está relacionado a um computador ou a um sistema computacional já existente, mas, algumas vezes, as dificuldades que são apresentadas não têm relação com computadores. Portanto, é essencial entender primeiro a natureza do problema. Devemos ser muito cautelosos para não impor máquinas e técnicas computacionais a toda questão que aparecer em nosso caminho. Primeiro, devemos resolvê-la. Então, se necessário, utilizar a tecnologia como ferramenta para implementar a nossa solução.

Mesmo sendo datada de 2004, a contribuição de Pfleeger (2004) continua válida, como reforça Sommerville (2018, p. 4), ao afirmar que a engenharia de software é essencial para o funcionamento do governo, da sociedade, das empresas e instituições nacionais e internacionais, destacando que o mundo atual não funciona sem software.

A engenharia de software está presente em todas as fases da construção de sistemas, desde o levantamento de requisitos até a manutenção. Essa abrangência é destacada por Sommerville (2018, p. 7), que considera a engenharia de software uma área que envolve todos os aspectos da produção de software.

A engenharia exige que os resultados atendam aos padrões de qualidade dentro dos limites de tempo e orçamento, o que frequentemente demanda concessões e impede uma postura perfeccionista por parte dos profissionais.

Por conta disso, é de suma importância analisar primeiro a natureza do problema, elaborar artefatos e escolher a metodologia mais adequada ao projeto. Como informa o Sommerville (2018, p. 8):

> Em geral, os engenheiros de software adotam uma abordagem sistemática e organizada para o seu trabalho, já que muitas vezes esse é a maneira mais eficaz de produzir software de alta qualidade. Entretanto, engenharia se trata de escolher o método mais adequado para um conjunto de circunstâncias, de modo que uma abordagem mais criativa e menos formal de desenvolvimento pode ser a mais adequada para alguns tipos de software. Um processo de software mais flexível que acomode a mudança rápida é particularmente adequado para o

> desenvolvimento de sistemas web interativos e de aplicativos para dispositivos móveis, que requerem um conjunto de habilidades de projeto de software e de projeto gráfico.

Afinal, como devemos criar um software? Segundo Sommerville (2018, p. 32), não existe um modelo universal para se desenvolver software, temos que ficar atento às necessidades do cliente, às necessidades do sistema e do ambiente para uso. A seleção adequada da metodologia se dá ao tipo de atividade que o sistema ou software irá executar, como por exemplo, sistema que exigem alta confiabilidade e segurança se adota o modelo cascata por conta da sua abordagem estruturada e rigorosa.

Diante desse contexto, surge a questão acerca de qual modelo adotar para iniciar o desenvolvimento de um software ou sistema. O modelo em cascata constitui um dos modelos clássicos de desenvolvimento de software, pois organiza todo o processo de produção de forma estruturada e sequencial. Esse modelo é composto por fases bem definidas, nas quais cada etapa exige a elaboração de documentação específica, sendo a progressão para a fase seguinte condicionada à conclusão e validação da documentação correspondente à etapa anterior. Como consta o Sommerville (2018, p. 35):

> A princípio, o resultado de cada fase no modelo em cascata consiste em um ou mais documentos que são aprovados. A fase seguinte não deve começar até que a fase anterior tenha terminado. No desenvolvimento de hardware, que envolve altos custos de produção, isso faz sentido. No entanto, no desenvolvimento de software, esses estágios se sobrepõem e alimentam uns aos outros com informações. Durante o projeto (design), são identificados problemas com os requisitos; durante a codificação, são encontrados problemas com o projeto, e assim por diante. O processo de software, na prática, nunca é um modelo linear simples, pois envolve feedback entre as fases.

À medida que surgem novas informações em uma etapa do processo, os documentos produzidos nas etapas anteriores devem ser modificados para refletir as mudanças no sistema. Por exemplo, se for descoberto que um requisito é caro demais para ser implementado, o documento de requisitos deve ser modificado para removê-lo. Entretanto, isso exige a aprovação do cliente, o que implica atraso do processo de desenvolvimento como um todo.

Como consequência, tanto clientes quanto desenvolvedores podem congelar prematuramente a especificação do software para não serem feitas outras alterações. Infelizmente, isso significa que os problemas são deixados para depois, ignorados ou contornados por meio de programação. O congelamento prematuro dos requisitos pode significar que o sistema não fará o que o usuário deseja. Também pode levar a sistemas mal estruturados, já que os problemas de projeto (*design*) foram contornados por artifícios de implementação.

Durante a fase final do ciclo de vida (operação e manutenção), o software começa a ser utilizado. Erros e omissões nos requisitos originais do software são descobertos, falhas de programação e de projeto emergem e a necessidade de novas funcionalidades é identificada. Por isso, o sistema deve evoluir a fim de continuar sendo útil. Realizar essas mudanças (manutenção de software) pode envolver a repetição dos estágios de processo prévios. Na realidade, o software precisa ser flexível e acomodar mudanças à medida que for sendo desenvolvido.

Depois de selecionar a metodologia adequada ao projeto, devemos escolher as ferramentas ideais para nos ajudar no processo de desenvolvimento de software, isso também depende das tecnologias utilizadas no projeto chamado de *stack*, como por exemplo a linguagem *Python*, com o seu framework *Django*, podendo-se utilizar o *Visual Studio Code* como ferramenta de edição de código, produção e execução de testes, bem

como também o uso de uma outra ferramenta chamada *GitHub* para produzir versões do código e trabalhar colaborativamente com outros desenvolvedores, como informa o Sommervile (2018, p. 40):

> Os processos de softwares reais são sequenciais intercaladas de atividades técnicas, colaborativas e gerenciais, cujo objetivo global é especificar, projetar, implementar e testar um sistema de software. Geralmente, os processos são apoiados por ferramentas. Isso significa que os desenvolvedores de software podem usar uma gama de ferramentas de software para ajudá-los, como sistemas de gerenciamento de requisitos, editores de modelo de projeto (design), editores de programa, ferramentas de teste automatizadas e depuradores.

Ao lidar com tecnologia, é recorrente a necessidade de considerar ambientes adequados para o armazenamento dos dados gerados pelos produtos desenvolvidos por profissionais de engenharia de software ou desenvolvimento. Esses dados, provenientes dos usuários de aplicações ou sistemas, devem ser armazenados em estruturas que permitam consultas eficientes, tratamento adequado da informação e análise sistemática. Dessa forma, torna-se possível extrair conhecimento relevante e subsidiar a definição de estratégias, como o direcionamento de campanhas de marketing a públicos-alvo específicos, com base nos dados coletados e analisados, como afirma Puga, França e Goya (2013, p. -9):

> Seja para realizar um simples clique em um link aleatório, seja para preencher um cadastro de compra pela internet, dados são gerados a todo instante. Com o advento das mídias sociais, a produção de informações tem aumentado a uma velocidade espantosa, oriunda das mais diversas fontes, como celulares, redes de relacionamentos e demais dispositivos.

Diante desse cenário, atribui-se às tecnologias de bancos de dados e, mais recentemente, ao *Big Data*, compreendido como uma evolução das tecnologias de *data warehouse* e *business intelligence*, voltadas ao armazenamento e à análise de informações estruturadas e não estruturadas em grandes volumes e com elevada variedade, além de possibilitarem processamento em tempo real, a capacidade de permitir que as organizações compreendam tais informações de forma sistemática e as utilizem estrategicamente em benefício de seus próprios negócios.

Antes de se utilizar algoritmos e suas demais estruturas de dados em projeto, se faz necessário produzir primeiro uma modelagem do banco de dados, para guardar os dados e termos noção de onde que os dados vão, com isso podermos programar com base nas tabelas e seus campos dentro do banco de dados, como afirma (Puga, França e Goya, 2013, p. 78):

> A modelagem de dados é um método de análise que, a partir de fatos relevantes a um contexto de negócio, determina a perspectiva dos dados, permitindo organizá-los em estruturas bem definidas e estabelecer regras de dependência entre eles, além de produzir um modelo expresso por uma representação descritiva e gráfica.

Quando utilizamos a lógica de programação e os algoritmos pertinentes à solução desejada, buscamos resolver um problema por meio do uso da computação. Assim que resolvido o problema proposto, podemos reutilizar essa mesma solução para problemas similares em futuro próximo, caso necessário, e podemos traduzir para qualquer linguagem de programação como Python, Java e entre outras. Não só a solução pode ser reutilizada como também alguns trechos e funções, como afirma Forbellone e Eberspächer (2005, p. 3):

> Um algoritmo tem por objetivo representar mais fielmente o raciocínio envolvido na Lógica de Programação e, dessa forma, permite-nos abstrair de uma série de detalhes computacionais, que podem ser acrescentados mais tarde. Assim, podemos focalizar nossa atenção naquilo que é importante: a lógica da construção de algoritmos.
> Outra importância da construção dos algoritmos é que uma vez concebida uma solução algorítmica para um problema, esta pode ser traduzida para qualquer linguagem de programação e ser agregada das funcionalidades disponíveis nos diversos ambientes; costumamos denominar esse processo de codificação.

Para saber se os algoritmos escolhidos são ideais para a solução do problema, devemos levar em conta os passos computacionais e tempo utilizados pela lógica, para solucionar o problema proposto. Para isso Ascencio e Araújo (2010, p. 3-4) explicam:

> O projeto de um algoritmo deve considerar o desempenho que este terá após sua implementação. Quando um problema é estudado e um projeto de algoritmo deve ser feito, várias soluções podem surgir, e aspectos de tempo de execução e espaço ocupado são pontos muito relevantes na escolha.

O custo de utilização ou tempo de execução de um algoritmo pode ser medido por meio de sua execução em um computador real, anotando-se o tempo. Os resultados obtidos podem ser considerados inadequados levando-se em conta que (1) dependem do compilador, que pode privilegiar algumas construções e outras não, (2) dependem do hardware; e (3) grandes quantidades de memória também interferem no tempo de execução do algoritmo.

A inteligência artificial pode ser utilizada tanto como uma ferramenta para agilizar consultas quanto para sugerir aplicações de algoritmos, dependendo de quem a utiliza e com qual finalidade. Como explica Filho, Oscar Gabriel (2023, p. 10):

> No que se refere à IA, o leitor pode se enquadrar simplesmente como usuário, uma classe de pessoas que precisa apenas usufruir dos produtos disponíveis no mercado e saber distinguir a qualidade daquilo que consome no seu dia a dia, ou como um profissional capacitado ou que busca capacitação para desenvolver IA, com a finalidade de fornecer produtos mais atraentes para o mercado de consumo.

CONSIDERAÇÕES FINAIS

Os resultados desse estudo indicam que, embora a inteligência artificial disponibilize recursos tecnológicos avançados, sua efetividade está diretamente condicionada à atuação humana, especialmente na definição de objetivos, no estabelecimento de parâmetros e na validação das soluções geradas. As evidências apresentadas demonstram que o uso consciente e estratégico da inteligência artificial pode potencializar a produtividade e a qualidade dos sistemas desenvolvidos, desde que sua integração ocorra de maneira planejada, criteriosa e acompanhada por profissionais qualificados.

A inteligência artificial proporciona agilidade e recomendações relevantes aos indivíduos que a utilizam de forma adequada, configurando-se como uma ferramenta tecnológica de elevado potencial e de uso crescente. Entretanto, os resultados por ela gerados não devem ser aceitos de maneira acrítica, sendo necessária a análise criteriosa, a validação e, quando pertinente, o ajuste das soluções propostas. Para um aproveitamento efetivo, deve ser empregada como um suporte ao estudo ou ao trabalho, atuando de forma complementar ao raciocínio humano: inicialmente, o indivíduo deve buscar resolver o problema de forma autônoma e, apenas diante de limitações, recorrer à ferramenta para obtenção de sugestões ou perspectivas alternativas.

Nesse contexto, o ser humano permanece como agente central do processo cognitivo e decisório, enquanto a inteligência artificial se restringe ao papel de instrumento de apoio, cujos resultados não são, por si só, definitivos ou infalíveis. Assim, torna-se indispensável adotar postura crítica e reflexiva frente as respostas fornecidas, reconhecendo que a inteligência artificial não substitui o julgamento humano, de vez que carece de compreensão contextual, valores e discernimento ético, atributos inerentes à tomada de decisão realizada por pessoas.

REFERÊNCIAS

ASCENCIO, Ana Fernanda Gomes; CAMPOS, Edilene Aparecida Veneruchi de. **Fundamentos da programação de computadores: algoritmos, PASCAL, C/C++ (padrão ANSI) e JAVA**. 3. ed. São Paulo: Pearson, 2012. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 23 fev. 2025.

ASCENCIO, Ana Fernanda Gomes; ARAÚJO, Graziela Santos de. **Estruturas de dados: algoritmos, análise da complexidade e implementações em Java e C/C++**. 1. ed. São Paulo: Pearson, 2010. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 23 fev. 2025.

CORDEIRO, Silmara Terezinha Pires. **Evolução biológica: atualizações na linha do tempo da teoria da evolução**. 1. ed. Curitiba: Intersaberes, 2020. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 23 fev. 2025.

CERVO, Amado Luiz; BERVIAN, Pedro Alcino; SILVA, Roberto da. **Metodologia científica**. 6. ed. São Paulo, SP: Pearson, 2006. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 01 mar. 2025.

FORBELLONE, André Luiz Villar; EBERSPÄCHER, Henri Frederico. **Lógica de programação: a construção de algoritmos e estruturas de dados**. 3. ed. São Paulo: Pearson, 2005. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 08 jun. 2025.

MELLO, Cleyson de Moraes; ALMEIDA NETO, José Rogério Moura de; COSTA, Marcio Martins da. **Inteligencia artificial e educação 6.0:** os caminhos da educação inteligente. Rio de Janeiro: Processo, 2024. *E-book*. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 05 out 2025.

MEDEIROS, Luciano Frontino de. **Inteligência artificial aplicada: uma abordagem introdutória**. Curitiba, PR: Intersaberes, 2018. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 23 fev. 2025.

MORAN, José Manuel. **A educação que desejamos:** novos desafios e como chegar lá. 1. ed. Campinas: Papirus, 2013. *E-book*. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 05 out 2025.

LUGER, G. F. **Inteligência artificial**. 6. ed. São Paulo: Pearson, 2013. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 02 mar. 2025.

FILHO, Oscar Gabriel. **Inteligência artificial e aprendizagem de máquina: aspectos teóricos e aplicações**. 1. ed. São Paulo, SP: Blucher, 2023. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 22 jun. 2025.

PFLEEGER, Shari Lawrence. **Engenharia de software: teoria e prática**. 2. ed. São Paulo: Pearson, 2004. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 23 fev. 2025.

PUGA, Sandra Gavioli; RISSETTI, Gerson. **Lógica de programação e estruturas de dados, com aplicações em Java**. 3. ed. São Paulo, SP: Pearson, 2016. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 23 fev. 2025.

PUGA, Sandra Gavioli; FRANÇA, Edson Tarcísio; GOYA, Milton Roberto. **Banco de dados: implementação em SQL, PL/SQL e Oracle 11g**. São Paulo: Pearson, 2013. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 01 jun. 2025.

SOMMERVILLE, Ian. **Engenharia de software**. 10. ed. São Paulo, SP: Pearson, 2018. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 23 fev. 2025.

SABBAGH, Rafael. Scrum: **gestão ágil para produtos de sucesso**. São Paulo, SP: Casa do Código, 2022. E-book. Disponível em: https://plataforma.bvirtual.com.br. Acesso em: 19 maio 2025.