# Arquiteturas Pedagógicas e Saúde Mental no Ensino de Programação: Uma Revisão Sistemática da Literatura

Maurício Rodrigues Lima
*Instituto de Informática (INF)*
*Universidade Federal de Goiás (UFG)*
Goiânia, Brasil
k20x@outlook.com

Elisângela Silva Dias
*Instituto de Informática (INF)*
*Universidade Federal de Goiás (UFG)*
Goiânia, Brasil
elisangelasd@ufg.br

***Abstract*—High failure and dropout rates in introductory programming courses coexist with significant levels of student anxiety, stress, and burnout. Due to this issue, it is necessary to map, classify, and synthesize evidence on Pedagogical Architectures (PAs) applied to computing education, identifying instructional principles, learning strategies and technological supports; delivery levels and modalities; learning and mental health indicators; and existing assessment methods. This study presents a systematic literature review conducted in accordance with the PRISMA 2020 protocol and registered in Parsifal. Eight databases were searched (ACM DL, Compendex, IEEEXplore, Web of Science, ScienceDirect, Scopus, Sol SBC, and Springer Link). After screening 97 records, 10 primary studies (2012–2025) met the inclusion criteria. The results show a predominance of constructionist, constructivist, and project-based PAs, mostly in on-site or hybrid undergraduate programs.**

***Keywords*—programming education; pedagogical architectures; mental health.**

***Resumo*—Altas taxas de reprovação e evasão em disciplinas introdutórias de programação coexistem com níveis relevantes de ansiedade, estresse e *burnout* discente. Devido a tal problemática, se faz necessário mapear, classificar e sintetizar evidências sobre Arquiteturas Pedagógicas (APs) aplicadas ao ensino de computação, identificando princípios instrucionais, estratégias de aprendizagem e suportes tecnológicos; níveis e modalidades de oferta; indicadores de aprendizagem e saúde mental e métodos de avaliação existentes. Este estudo apresenta uma revisão sistemática da literatura conduzida de acordo com o protocolo PRISMA-2020, registrada no *Parsifal*. Foram pesquisadas oito bases (ACM DL, Compendex, IEEEXplore, Web of Science, ScienceDirect, Scopus, Sol SBC e Springer Link). Após triagem de 97 registros, 10 estudos primários (2012-2025) atenderam aos critérios de inclusão. Os resultado demonstram predominância de APs construcionistas, construtivistas e de aprendizagem por projetos, majoritariamente em cursos presenciais ou híbridos de graduação.**

***Palavras-chave*—aprendizagem de programação; arquiteturas pedagógicas; saúde mental.**

## I. Introdução

O insucesso persistente nas disciplinas introdutórias de programação em cursos de computação, expresso por índices de reprovação e evasão que podem superar 50% em alguns contextos [1], [2], é acompanhado por forte sofrimento psicológico, ansiedade e estresse [3], [4]. Apesar desse cenário, este estudo revelou que nenhuma das dez Arquiteturas Pedagógicas (AP) publicadas até o mês de junho de 2025 empregou instrumentos psicométricos validados, como DASS-21 [5], GAD-7 [6] ou K10 [7], para monitorar a saúde mental discente. Paralelamente, apenas 8% das propostas analisadas exploraram recursos de Inteligência Artificial (IA) ou *Learning Analytics* (LA) para personalização ou análise de engajamento [8], [9]. Esses achados expõem uma lacuna em um contexto curricular reconhecidamente marcado por sobrecarga cognitiva e emocional [10]. Este estudo se propõe a avançar o estado da arte por meio de uma AP que integre LA e cuidados explícitos de saúde mental ao ensino de programação.

## II. Justificativa

Há dissonância entre a ênfase no protagonismo discente das APs construcionistas e a ausência de métricas psicométricas sistemáticas, o que limita comparações e a consolidação de evidências sólidas [11]–[13].

No plano socioeconômico, a evasão em cursos de computação gera perdas para instituições e reduz a formação de profissionais de Tecnologia da Informação e Comunicação (TIC), agravando disparidades regionais em inovação [14], [15].

Intervenções eficazes precisam considerar não apenas o conteúdo técnico, mas também fatores de bem-estar e engajamento emocional [3], [16]. LA com modelagem preditiva pode detectar padrões de engajamento e antecipar evasões [17],

enquanto práticas de *mindfulness* em sala contribuem para o bem-estar e o desempenho acadêmico [18], [19]. Todavia, combinações integradas de LA e intervenções de saúde mental permanecem pouco exploradas [9], [20].

## III. Objetivos

O objetivo geral deste estudo é investigar, por meio da realização de uma revisão sistemática da literatura conduzida segundo o protocolo PRISMA-2020 [21], como as APs têm sido concebidas, implementadas e avaliadas no ensino de computação, com ênfase na aprendizagem de programação e na consideração da saúde mental discente. Os objetivos específicos são:

1) Mapear e classificar as APs segundo seus princípios instrucionais, estratégias de aprendizagem e suportes tecnológicos;
2) Caracterizar os níveis de ensino, modalidades de oferta e áreas/disciplina de computação a que as APs se aplicam;
3) Sintetizar as evidências de efetividade educacional relatadas, contemplando resultados de aprendizagem (desempenho, engajamento, retenção, satisfação) e indicadores de saúde mental (ansiedade, estresse, *burnout*);
4) Descrever os métodos, instrumentos e métricas utilizados para avaliar as APs, destacando práticas e lacunas metodológicas;
5) Comparar APs desenvolvidas especificamente para o ensino de programação com aquelas voltadas a outras disciplinas de computação, identificando características distintivas e diferenças nos resultados;
6) Identificar lacunas na literatura, especialmente a ausência de métricas psicométricas validadas para saúde mental; o uso de IA para personalização e monitoramento em tempo real e averiguação de de estudos com desenho experimental robusto e acompanhamento de longo prazo;
7) Propor recomendações para futuras pesquisas e práticas pedagógicas que integrem cuidados de bem-estar discente e recursos de IA em turmas de programação.

## IV. Trabalhos Relacionados

O estudo [22] realizou uma revisão de literatura baseada em critérios específicos, selecionando 86 artigos publicados entre 2017 e 2022 nas bases ACM, Scopus, Web of Science e o Portal de Periódicos da Capes. Foram utilizadas *strings* de busca em português e inglês com foco em termos como "ensino de programação", "aprendizagem em rede", "cooperação" e sinônimos. Os artigos selecionados foram analisados segundo sete categorias que compõem as APs: uso da internet, educação a distância, inteligência artificial, *softwares* educacionais, abordagens pedagógicas, concepção de tempo e espaço, e cooperação. Os resultados apresentaram que a maioria das propostas analisadas utiliza abordagens pedagógicas (92%) e *softwares* educacionais (72,4%), enquanto apenas uma pequena parte faz uso de inteligência artificial (8%). A concepção de tempo e espaço aparece de forma ampla, abarcando contextos presenciais, híbridos e *on-line*, sendo o ensino presencial o mais recorrente (54%). A cooperação entre alunos foi identificada em muitas das propostas, mas nem sempre alinhada ao conceito piagetiano [23] de cooperação como operação racional conjunta. Além disso, observou-se que o foco das iniciativas ainda está majoritariamente no ensino superior (51%), com menor presença nos níveis fundamental e médio.

Diferentemente do estudo [22], que mapeia propostas de APs para o ensino de programação unicamente sob a perspectiva do uso de tecnologias digitais e de princípios gerais de psicologia da aprendizagem, sem avaliar qualquer indicador de saúde mental discente, o presente estudo amplia o foco ao incorporar, como critério central, a averiguação de ansiedade, estresse e *burnout* por meio de escalas psicométricas validadas. Além disso, enquanto o trabalho anterior [22] examina publicações concentradas no quinquênio 2017-2022 e resulta em um levantamento exploratório de 86 estudos dispersos, o presente estudo segue o protocolo PRISMA 2020 [21], restringe-se a dez estudos primários publicados até junho de 2025 e avalia, de forma crítica, não só resultados de aprendizagem, mas também a presença (ou ausência) de métricas de bem-estar e o uso de IA e *Learning Analytics* para monitoramento em tempo real. Dessa forma, o presente estudo preenche a lacuna evidenciada na literatura ao integrar *performance* acadêmica e saúde mental em turmas de programação.

## V. Metodologia

Adotou-se a metodologia PRISMA em sua versão de 2020 [21] e o *Parsifal* para planejamento, execução, condução e documentação da pesquisa [24]. Trata-se de uma Revisão Sistemática da Literatura (RSL), que é caracterizada por uma metodologia replicável, envolvendo uma busca abrangente para localizar os trabalhos publicados e não publicados sobre um determinado tema; uma integração sistemática dos resultados dessa busca; e uma análise crítica da abrangência, natureza e qualidade das evidências em relação a uma específica questão de pesquisa; sintetizando estudos para extrair conclusões teóricas amplas sobre o significado de uma literatura, conectando teoria e evidência [25].

Esta revisão sistemática da literatura foi guiada pelas seguintes questões principais: **(QP1)** quais arquiteturas pedagógicas têm sido empregadas no ensino de computação e como elas podem ser classificadas segundo seus princípios instrucionais, estratégias de aprendizagem e suporte tecnológico; **(QP2)** em quais níveis de ensino (fundamental, médio/técnico,

graduação, pós-graduação, formação continuada) e modalidades (presencial, híbrida, à distância, de curta duração ou campeonatos) as arquiteturas são implementadas, e quais disciplinas ou áreas da computação elas abrangem; **(QP3)** quais evidências de efetividade educacional são relatadas, incluindo resultados de aprendizagem, como desempenho, aquisição de competências, engajamento, retenção e satisfação, e impactos em saúde mental, tais como ansiedade, estresse, bem-estar e *burnout* em cursos de computação; **(QP4)** que métodos, instrumentos e métricas de avaliação são utilizados para aferir os resultados de aprendizagem associados às diferentes arquiteturas pedagógicas em computação; e **(QP5)** sobre a existência de arquiteturas pedagógicas criadas ou adaptadas especificamente para o ensino de programação, identificando suas características distintivas e comparando seus resultados com os observados em arquiteturas aplicadas a outras disciplinas da área.

**String de busca:** *("pedagogical architecture" OR "arquitetura pedagógica") AND ("computing" OR "programming" OR "computação" OR "programação" ).*

**Fontes:** *ACM Digital Library, El Compendex, IEEE Digital Library, ISI Web of Science, Science@Direct, Scopus, Sol SBC, Springer Link.*

Foram incluídos na análise apenas estudos que descrevem ou avaliam empiricamente uma AP implementada no contexto de ensino ou aprendizagem de computação. Foram excluídos estudos duplicados, estudos secundários (revisões, meta-análises), publicações de acesso indisponível aos autores desta revisão, literatura cinzenta (como dissertações, teses ou relatórios técnicos), trabalhos que não tratam de AP, estudos que não se referem a disciplinas da área de computação, bem como versões anteriores de trabalhos já previamente selecionados para a revisão.

### A. Execução

O processo de triagem e seleção dos estudos resultou na seguinte distribuição por base de dados: 14 (quatorze) resultados na ACM Digital Library, seis na El Compendex, um na IEEE Digital Library, 17 (dezessete) na ISI Web of Science, dez na Science@Direct, sete na Scopus, 27 (vinte e sete) na Sol SBC e 10 (dez) na Springer Link. A análise culminou na seleção final de 10 (dez) estudos considerados relevantes e aderentes aos critérios estabelecidos.

Durante o processo, cinco estudos foram descartados por duplicidade, quatro por se tratarem de revisões sistemáticas ou estudos secundários, nove por indisponibilidade de acesso ao texto completo e seis por pertencerem à literatura cinzenta. Além disso, 41 (quarenta e um) estudos foram eliminados por não tratarem de ensino em disciplinas de computação e 17 (dezessete) por não apresentarem propostas de AP. Todo o processo de execução está representado na Figura 1.

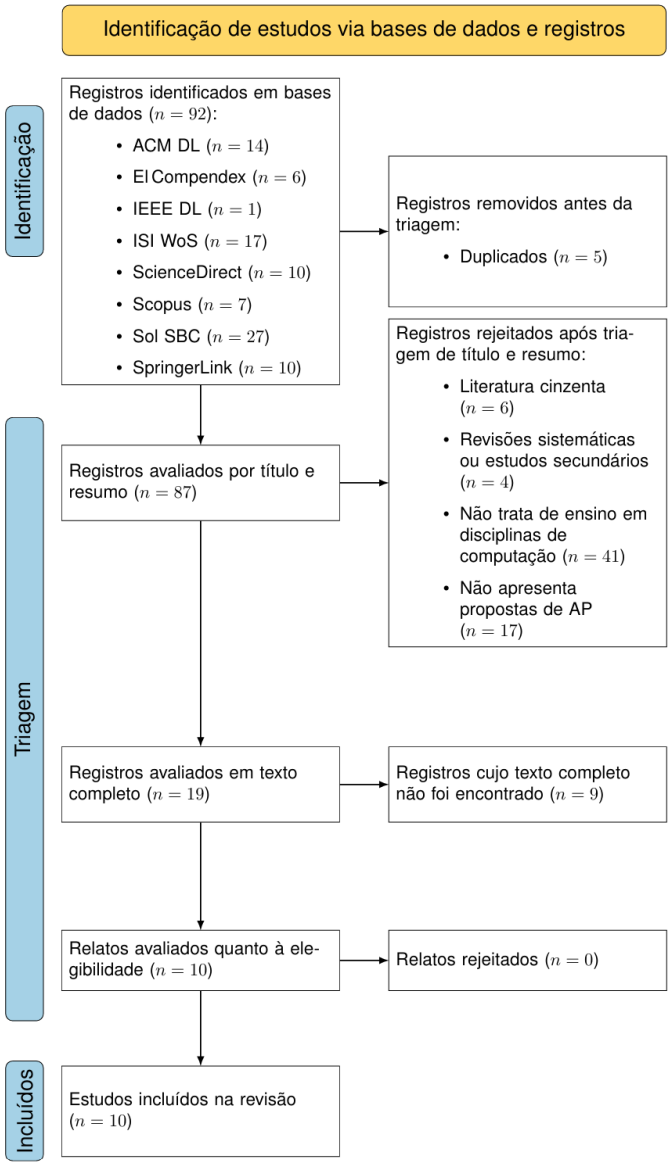


Fig. 1. Processo de seleção de estudos seguindo Prisma 2020 [21].

## VI. Resultados

Os resultados apresentam uma predominância de abordagens baseadas no construcionismo [26], construtivismo [27], aprendizagem por projetos [28] e metodologias ativas [29], que valorizam o protagonismo discente e o desenvolvimento do pensamento computacional [30]. As AP contemplam diferentes níveis e modalidades de ensino, com ênfase na graduação presencial ou híbrida, embora também incluam ações na educação básica e na formação continuada. A saúde mental dos estudantes, embora seja um aspecto importante no contexto educa-

cional, não é abordada em nenhuma das iniciativas revisadas. Além disso, poucos estudos se dedicaram a monitorar os efeitos educacionais após o término das intervenções ou seus impactos a longo prazo.

### A. Respondendo à QP1:

As APs analisadas concentram-se em quatro matrizes teóricas predominantes que orientam suas propostas instrucionais. A Tabela I sintetiza as APs encontradas na literatura analisada; em seguida, é apresentada uma discussão acerca dos eixos de classificação.

TABELA I
CLASSIFICAÇÃO DAS APS SEGUNDO PRINCÍPIOS INSTRUCIONAIS, ESTRATÉGIAS DE APRENDIZAGEM E SUPORTE TECNOLÓGICO

| **Estudo** | **Princípios instrucionais** | **Estratégias de aprendizagem** | **Suporte tecnológico** |
|---|---|---|---|
| [31] | Construcionismo, CT | Oficinas itinerantes com robôs e desafios incrementais | *Lego NXT, Raspberry Pi, Scratch, RobotC* |
| [32] | Aprendizagem baseada em projetos (ABP) | *Design* de processador para criar jogos; motivação por produto final | FPGA, teclado PS/2, saída VGA |
| [33], [34] | Construtivismo (pedagogia da incerteza) | Método MCP (6 etapas) e *wiki* colaborativo | *IDE Haskell, wiki*, fórum e *chat* |
| [35] | *Design-Based Learning* | Simplificação de arquitetura de processador; comparação pré/pós | *Pseudo-assembly*, *SmartPLS*, *NVivo* |
| [36] | Heutagogia, autonomia | *Grouped trail*: ordem livre de módulos | *Moodle* e *plugin Onetopic*, PDFs interativos |
| [37] | Ecossistemas de aprendizagem | Oficina colaborativa com postagens e comentários | *Padlet*, *Google Classroom* |
| [38] | Metodologias ativas | Minicurso extensionista com *Google Colab* | Página *Web* (*GitHub Pages*) e *Colab* |
| [39] | Pensamento computacional transversal | Criação de aplicativos com blocos em disciplinas gerais | *Scratch/BloP*; *tablets* / computador pessoal |
| [40] | Aprendizagem cooperativa | Dinâmica em 6 etapas, revisão por pares | *Trello*, vídeos, planos de aula digitais |

Destacam-se abordagens baseadas no construcionismo [26] e no construtivismo [27], em que o conhecimento emerge da construção de artefatos significativos, como programas computacionais ou robôs. Essa perspectiva enfatiza o aprendizado ativo por meio da materialização de ideias em objetos tangíveis, conforme evidenciado em iniciativas como as descritas por [31] e [33].

Além disso, observa-se a presença da aprendizagem baseada em projetos (ABP) e do *design-based learning* (DBL), nos quais os estudantes são desafiados a resolver problemas autênticos como o projeto de processadores ou o desenvolvimento de jogos, sendo estimulados a refletir criticamente sobre suas decisões e processos. Essa abordagem é recorrente em propostas como as de [32] e [35]. Complementando essas duas matrizes, surgem metodologias ativas centradas no estudante, que envolvem práticas interacionistas, trilhas formativas, heurísticas de resolução e mecanismos de autorregulação. Tais metodologias valorizam a autonomia e o protagonismo dos aprendizes, como discutido por [36] e [37].

Por fim, identifica-se a influência crescente da pedagogia computacional, que utiliza o pensamento computacional como um meio de ensinar conteúdos diversos, e não apenas habilidades de programação. Essa proposta amplia as possibilidades educativas ao integrar raciocínio lógico e resolução de problemas em diferentes contextos disciplinares [39].

No campo das estratégias de aprendizagem, prevalecem abordagens colaborativas e baseadas na autoria dos estudantes. Uma estratégia amplamente utilizada é o ciclo *design–make–share*, no qual os alunos projetam e prototipam soluções em *hardware* ou *software* e compartilham seus resultados em apresentações públicas, como exemplificado por [32]. Além disso, práticas de revisão por pares e socialização de soluções por meio de ferramentas como *wikis*, *Trello* ou *Padlet* promovem um ambiente de crítica construtiva e de refatoração contínua, como discutido em [33] e [40]. Outra estratégia relevante é a organização do conteúdo em trilhas personalizadas e micromódulos autônomos, permitindo que os estudantes avancem de forma flexível de acordo com seu próprio ritmo e interesse [36].

Quanto ao suporte tecnológico, observa-se uma grande diversidade de recursos, com variações que se alinham intencionalmente aos objetivos didáticos das APs. Em ambientes virtuais, destacam-se ferramentas como *Moodle*, *Trello*, *Google Colab* e *Padlet*, que promovem a coautoria, o versionamento colaborativo e o fornecimento de *feedback* em tempo real. Em termos de linguagens e técnicas, opta-se frequentemente por ferramentas de baixa barreira de entrada, como linguagens em blocos (*Scratch*, *BloP*), *pseudo-assembly* e ambientes *wiki*, os quais reduzem a complexidade sintática e favorecem a concentração na lógica do pensamento computacional.

### B. Respondendo à QP2:

A análise dos estudos demonstra que as APs investigadas abrangem uma faixa ampla de níveis de ensino e modalidades, porém concentram-se majoritariamente em ofertas presenciais ou híbridas na graduação e em ações extensionistas ou de formação continuada. A Tabela II organiza cada AP segundo nível de ensino, modalidade de oferta e disciplina ou área de computação contemplada.

TABELA II
NÍVEL, MODALIDADE E ÁREA DE COMPUTAÇÃO COBERTA POR CADA AP

| Estudo | Nível de ensino | Modalidade | Disciplina / área |
|---|---|---|---|
| [31] | Fundamental e Médio | Presencial (oficinas itinerantes) | Fundamentos de ciência da computação e programação introdutória |
| [32] | Graduação | Presencial em laboratório | Arquitetura de Computadores e Design de *Hardware* |
| [33], [34] | Graduação | Presencial e *wiki*/fórum | Lógica e Técnicas de programação |
| [35] | Graduação | Presencial (com recursos *on-line*) | Arquitetura de Processadores simplificada |
| [36] | Formação continuada | EaD (*Moodle* e *plugin*) | Robótica Educacional e Pensamento Computacional |
| [37] | Graduação | Híbrida (oficina *on-line*) | Tecnologias Digitais na Educação |
| [38] | Ensino Médio | Híbrida (*web* e *Colab*) | Lógica de programação (*Python*) |
| [39] | Fundamental e Médio | Presencial (laboratório) | Pensamento Computacional com *Scratch* |
| [40] | Graduação | Híbrida (*Trello*, vídeos) | Engenharia de Requisitos de *Software* |

No contexto da educação fundamental e média, destacam-se apenas três iniciativas: o *Turing Project*, que promove oficinas itinerantes de programação e robótica em escolas [31]; a proposta de Pegagogia Computacional com base em linguagens em blocos, voltada para o ensino fundamental [39]; e uma experiência de extensão universitária voltada ao ensino de lógica com Linguagem de programação *Python* em uma escola pública [38]. No entanto, o foco predominante recai sobre a graduação, abrangendo cinco arquiteturas distintas. Essas abordagens incluem disciplinas como Arquitetura de Computadores [32], [35], programação [33], Tecnologias Educacionais em cursos de licenciatura [37], e Engenharia de Requisitos [40]. A formação continuada aparece com apenas um estudo explicitamente voltado à capacitação docente em serviço, utilizando ensino a distância (EaD) como estratégia para formação em Robótica Educacional [36].

Quanto às modalidades de oferta, as APs analisadas distribuem-se entre formatos presenciais, híbridos e exclusivamente a distância. A modalidade presencial em laboratório é essencial em contextos que demandam manipulação de *hardware* específico, como FPGA ou kits de robótica, uma vez que o aprendizado requer interação física com os dispositivos [31], [32]. Já o modelo híbrido é amplamente adotado, combinando encontros presenciais com atividades remotas por meio de ambientes virtuais de aprendizagem (AVAs), *wikis* e repositórios colaborativos, como *Trello*, *Padlet* e *Google Colab*, visando potencializar a colaboração assíncrona e ampliar a autoria dos estudantes [33], [37], [40]. A modalidade EaD pura aparece em apenas um estudo, centrado em uma trilha de aprendizagem implementada no *Moodle*, cujo desenho permite navegação não linear e autonomia total do cursista [36]. Notavelmente, não foram identificadas implementações em *MOOCs* ou *bootcamps*, o que sugere uma preferência dos autores por turmas reduzidas, que viabilizem acompanhamento próximo e avaliação processual mais cuidadosa. Em relação à abrangência disciplinar, as APs demonstram diversidade. Elas abrangem desde conteúdos como Arquitetura de Processadores [35] e Lógica de programação [38], até áreas da engenharia de *software*, como Engenharia de Requisitos [40], além de competências transversais relacionadas ao pensamento computacional [39]. Essa variedade evidencia a flexibilidade do conceito de AP, que se adapta tanto a aspectos fundamentais do *hardware* quanto a competências sócio interacionais, especialmente relevantes para a formação docente.

### C. Respondendo à QP3:

A literatura analisada, conforme Tabela III, apresenta evidências positivas quanto à eficácia das APs no ensino de computação, embora os indicadores de saúde mental (ansiedade, estresse, bem-estar, burnout) sejam ausentes.

TABELA III
EVIDÊNCIAS DE EFETIVIDADE EDUCACIONAL RELATADAS PARA CADA AP

| Estudo | Indicadores / método | Principais resultados de aprendizagem | Saúde mental |
|---|---|---|---|
| [35] | Comparação de notas (pré × pós), survey TAM, rating, entrevistas | Aumento significativo no desempenho, alta utilidade/facilidade percebidas, satisfação e intenção de uso elevadas | Não reportado |
| [32] | Observação qualitativa de projetos e relatórios de curso | Maior motivação, projetos mais complexos e compreensão mais profunda de arquitetura de *hardware* | Não reportado |
| [33] | Questionários, análise de 18 exercícios, gráfico de desempenho | Engajamento elevado, desenvolvimento de habilidades de cooperação e resolução de problemas; recuperação de alunos antes reprovados | Não reportado |
| [34] | *Logs* de interação e análise de listas | *feedback* contínuo, maior consciência sobre processo de programação, diagnósticos individuais | Não reportado |
| [31] | Relatos de oficinas móveis | Forte motivação e participação em robótica/programação; sem métricas objetivas | Não reportado |
| [36] | Reflexões dos cursistas | Complementação de aprendizagem e desenvolvimento de novas capacidades em EaD | Não reportado |
| [37] | Questionário de percepção | Experiência "interativa e dinâmica", colaboração e facilidade de uso reconhecidas | Não reportado |
| [38] | Questionário pós-ação | Tendência positiva na percepção do uso das páginas *Web*; alto envolvimento do público-alvo | Não reportado |
| [39] | Três experimentos controlados | Retenção a longo prazo; alunos que programaram obtêm melhores resultados seis meses depois | Não reportado |
| [40] | Depoimentos de 4 semestres e *feedbacks* em *Trello* | Alta satisfação, engajamento sustentado e valorização dos *feedbacks* na prática de *soft skills* | Não reportado |

Observa-se uma predominância de evidências relacionadas ao engajamento e à satisfação dos estudantes. Seis estudos relatam aumento na motivação ou descrevem uma "experiência positiva" por parte dos participantes, o que indica um impacto afetivo importante. No entanto, apenas duas iniciativas, em [35] e [39], apresentaram comparações quantitativas sistematizadas de desempenho acadêmico, o que aponta para uma limitação na generalização dos resultados.

De forma mais ampla, constata-se que as medidas objetivas de aprendizagem ainda são escassas nas avaliações das APs. Apenas três estudos recorreram a instrumentos como notas, testes de conhecimento ou análises estatísticas inferenciais [33], [38], [39], evidenciando a necessidade de protocolos mais robustos. Avaliações com grupos controle, métodos quase-experimentais e acompanhamento prolongado ainda são raramente utilizados, o que compromete a confiabilidade dos resultados reportados.

Outro ponto crítico diz respeito à retenção de longo prazo do conteúdo aprendido. Poucos estudos se dedicaram a monitorar os efeitos educacionais após o término das intervenções. A única exceção identificada é o experimento conduzido por [39], que investigou o uso de blocos visuais no ensino de pensamento computacional e constatou a manutenção do conhecimento adquirido mesmo após um período de seis meses.

Adicionalmente, um aspecto negligenciado nos estudos analisados diz respeito à saúde mental dos estudantes. Nenhum dos artigos revisados incluiu métricas para mensurar ansiedade, estresse ou *burnout*, mesmo sendo esses fatores importantes no contexto da formação em computação. Essa omissão representa uma oportunidade para avanços metodológicos, por meio da incorporação de escalas psicométricas validadas, como a DASS-21, em futuras investigações sobre o impacto das APs.

Além disso, é importante que as avaliações considerem não apenas indicadores cognitivos imediatos, mas também aspectos afetivos e de bem-estar dos aprendizes, monitorando-os ao longo de janelas temporais ampliadas.

### *D. Respondendo à QP4:*

Os estudos sobre APs em computação utilizam uma combinação de métodos quantitativos e qualitativos para aferir resultados de aprendizagem. A Tabela IV sintetiza, para cada AP, o desenho de pesquisa, os instrumentos de coleta de dados e as métricas computadas.

A avaliação das APs no ensino de computação mobiliza um conjunto diverso de instrumentos metodológicos, que visam capturar tanto o desempenho acadêmico quanto aspectos comportamentais e perceptivos dos estudantes. Entre os instrumentos mais comuns estão os testes de desempenho, como provas, *quizzes* e planilhas, que medem a aquisição de conhecimentos conceituais e procedimentais relacionados ao conteúdo abordado [35], [39]. Tais instrumentos são frequentemente utilizados para mensurar o domínio técnico dos aprendizes ao final das intervenções pedagógicas.

TABELA IV
MÉTODOS DE AVALIAÇÃO DE APRENDIZAGEM NAS DIFERENTES APS

| Estudo | Desenho / método | Instrumentos | Métricas reportadas |
|---|---|---|---|
| [35] | Quase-experimental (pré–pós) e modelo estrutural PLS | Testes de curso; *Survey* TAM (*Likert* 7 pontos); entrevistas codificadas em *NVivo* | Média e desvio padrão de notas; percentual de falha; PU, PEOU, SAT, INT; coeficiente de caminho ($\beta$) |
| [32] | Estudo de caso longitudinal | Relatórios de projeto; observação em laboratório | Complexidade do jogo (número de módulos); percentual de equipes concluídas; evidências de motivação qualitativa |
| [33] | Acompanhamento de 18 tarefas | Rubrica de código; questionário de engajamento | Nota média por tarefa; taxa de sucesso (percentual de aprovação); autoavaliação (*Likert*) |
| [34] | Análise de *logs* e listas de exercícios | *Logs* de IDE; *checklists* semanais | Número compilações bem-sucedidas; tempo de resolução; progresso individual |
| [31] | Relato de oficinas | Diário de campo; entrevistas com alunos | Frequência; relatos de entusiasmo (qualitativos) |
| [36] | Relato reflexivo EaD | Questionário pós-curso (*Likert*) | Satisfação global; utilidade percebida |
| [37] | Estudo exploratório | *Survey on-line* (*Likert* 5 pontos) | Cooperação, autonomia, organização (média e desvio padrão) |
| [38] | Extensão escolar; pós-teste | Questionário de percepção | Clareza, interesse, utilidade (escala 1–5) |
| [39] | Experimento cont-role–tratamento em 3 momentos | Testes cognitivos; planilhas | Acurácia imediata; retenção 6 meses; ganho normalizado |
| [40] | Ação em 4 semestres | *feedbacks* no *Trello*; autoavaliação de *soft skills* | Satisfação (percentual positivo); frequência de *posts*; auto-percepção de habilidades |

A avaliação das APs no ensino de computação mobiliza um conjunto diverso de instrumentos metodológicos, que visam capturar tanto o desempenho acadêmico quanto aspectos comportamentais e perceptivos dos estudantes. Entre os instrumentos mais comuns estão os testes de desempenho, como provas, *quizzes* e planilhas, que medem a aquisição de conhecimentos conceituais e procedimentais relacionados ao conteúdo abordado [35], [39]. Tais instrumentos são frequentemente utilizados para mensurar o domínio técnico dos aprendizes ao final das intervenções pedagógicas.

Outro recurso amplamente adotado são as rubricas avaliativas e os relatórios de projeto, que permitem avaliar a qualidade dos artefatos desenvolvidos pelos estudantes, como códigos-fonte, jogos educacionais ou protótipos robóticos [32], [33]. Esses instrumentos são especialmente relevantes em propostas

baseadas em aprendizagem por projetos, nas quais o produto final constitui evidência direta da aprendizagem.

Com o avanço das tecnologias educacionais, também têm ganhado espaço os registros automáticos de interação, os chamados *logs* ou analíticas de aprendizagem, que capturam dados objetivos como o tempo de compilação, o número de submissões em ambientes de desenvolvimento e o padrão de uso de plataformas digitais [34]. Esses dados possibilitam uma análise mais fina da trajetória de aprendizagem e do engajamento efetivo dos estudantes.

Além disso, diversos estudos fazem uso de questionários padronizados para avaliar percepção de utilidade, facilidade de uso, satisfação e intenção de continuidade, baseando-se, por exemplo, no modelo *Technology Acceptance Model* (TAM), ou ainda em escalas do tipo *Likert* voltadas a aspectos como engajamento e desenvolvimento de competências socioemocionais [35], [37]. Por fim, entrevistas semiestruturadas e observações dos participantes também são utilizadas para captar dimensões qualitativas da experiência dos estudantes, fornecendo dados ricos para triangulação metodológica [31], [32].

As métricas derivadas desses instrumentos seguem algumas categorias recorrentes. O desempenho acadêmico costuma ser medido por indicadores como a média de notas, o ganho normalizado de aprendizagem e a taxa de reprovação. O engajamento é mensurado por meio de frequência às aulas ou oficinas, número de contribuições em fóruns e repositórios, bem como tempo ativo nas plataformas utilizadas. A retenção do conteúdo, embora menos explorada, aparece em alguns estudos por meio da aplicação de testes diferidos, como no experimento de [39], realizado seis meses após a intervenção. Já a satisfação e a aceitação das APs são geralmente avaliadas com base em escores de modelos como o TAM ou em coeficientes derivados de análises baseadas em Modelagem de Equações Estruturais, como *Partial Least Squares (PLS)*.

### *E. Respondendo à QP5:*

A revisão identificou três APs explicitamente concebidas ou adaptadas para o ensino de programação (Linha A, Tabela V). Demais APs (Linha B) abordam outras áreas como Arquitetura de Processadores, Engenharia de Requisitos, Pensamento Computacional e Robótica, servindo aqui como referencial comparativo.

As APs voltadas ao ensino de programação apresentam características distintivas que as diferenciam de propostas em outras áreas da computação, como Arquitetura de *Hardware* ou Engenharia de Requisitos. Uma das principais singularidades dessas APs é a nível de detalhe elevado nos dados. Em vez de se apoiarem em projetos extensos, elas estruturam o percurso formativo em exercícios graduais, promovendo a publicação frequente de soluções e possibilitando ciclos rápidos de *feedback*. Esse padrão permite intervenções didáticas mais ágeis, otimizando a assimilação de conceitos e técnicas por parte dos estudantes.

TABELA V
APs FOCADAS EM PROGRAMAÇÃO (LINHA A) VERSUS APs DE OUTRAS DISCIPLINAS DE COMPUTAÇÃO (LINHA B)

| Estudo | Características distintivas | Evidências de resultado |
|---|---|---|
| **Linha A – programação** | | |
| [33] | Método MCP (motivação → construção → publicação); *wiki* colaborativo | Melhora na nota média; taxa de recuperação de reprovados; engajamento elevado |
| [34] | Estratégia *learning by doing*; monitoramento de *logs* de IDE; *feedback* formativo automático | Redução do tempo de compilação com erro; progressão individualizada documentada |
| [38] | Pilares (objetivos, domínio, tecnologia, avaliação); *Google Colab* e página *Web*; foco em ensino médio | Alta percepção de clareza/utilidade; motivação de estudantes não especialistas |
| **Linha B – Outras disciplinas** | | |
| [32] | Design de *hardware* para viabilizar jogos; integração teclado/VGA/FPGA | Projetos mais complexos; Aumento de motivação |
| [35] | Simplificação de processador; análise PLS de utilidade | Melhoria nas notas e redução nas falhas |
| [40] | Dinâmica 6 etapas; *Trello* para revisão em pares | Satisfação e engajamento sustentados |
| [39] | Blocos *Scratch* para conteúdos gerais; experimento longitudinal | Melhora na retenção |

Outro aspecto é o uso intensivo de mecanismos de monitoramento automático. A instrumentação pedagógica via *logs* de compilação, conforme explorado por [34], e a utilização de *wikis* como forma de rastrear o processo de construção das soluções [33], oferecem aos docentes uma visibilidade quase em tempo real do progresso dos alunos, algo ausente nas APs de outras áreas, que dependem mais da avaliação pontual de entregas finais.

Além disso, as APs de programação tendem a adotar tecnologias de baixa barreira de entrada, como *Google Colab*, editores *Web* ou Ambientes Integrados de Desenvolvimento educacionais, que minimizam dificuldades com instalação e configuração de ambientes, reduzindo a fricção inicial no processo de aprendizagem. Essa estratégia contrasta com disciplinas que exigem infraestrutura física dedicada onde o acesso ao *hardware* pode representar uma barreira adicional.

Por fim, essas APs mantêm um foco claro em fundamentos algorítmicos, lógica, estruturas de controle, depuração e análise de código, enquanto outras áreas priorizam o desenvolvimento de competências interpessoais (caso de Engenharia de Requisitos) ou a compreensão sistêmica de componentes físicos e lógicos (como em Arquitetura de Computadores).

No que diz respeito aos resultados alcançados, observa-se que o desempenho acadêmico obtido nas APs de programação é promissor. Tanto o estudo de [35] quanto a proposta de [33] demonstram ganhos estatisticamente significativos nas notas

dos estudantes, embora as diferenças em magnitude se reduzam quando controlado o nível de ensino. Quanto ao engajamento e à motivação, os relatos qualitativos indicam altos níveis de envolvimento dos estudantes, especialmente associados ao *feedback* imediato e à resolução de erros, um efeito comparável ao entusiasmo relatado em projetos baseados em jogos [32].

## VII. Conclusão

A presente revisão sistemática da literatura evidenciou que as APs aplicadas ao ensino de computação concentram-se majoritariamente em matrizes construcionistas, construtivistas e de aprendizagem por projetos, privilegiando o protagonismo discente e o desenvolvimento do pensamento computacional. Embora tais abordagens mostrem ganhos consistentes de engajamento e, em alguns casos, melhoria no desempenho acadêmico, o corpus analisado, restrito a apenas dez estudos primários, apresenta uma baixa oferta de APs em níveis de ensino fundamental e médio, além de virtual inexistência em modalidades massivas, como *Massive Open On-line Course (MOOCs)* e *bootcamps*.

Do ponto de vista metodológico, persistem fragilidades que limitam a generalização dos resultados: predominam desenhos exploratórios ou quase-experimentais sem grupos controle, amostras reduzidas e uso esporádico de análises estatísticas inferenciais. Indicadores cognitivos imediatos prevalecem sobre métricas de retenção de longo prazo, e apenas dois estudos adotaram acompanhamento diferido superior a um semestre. Além disso, nota-se a adoção incipiente de instrumentos padronizados de avaliação de competências, o que dificulta a comparação entre investigações.

Uma lacuna crítica diz respeito à saúde mental dos estudantes, tendo em vista que nenhum estudo incorporou escalas psicométricas validadas, como DASS-21, GAD-7 ou K10, para mensurar ansiedade, estresse ou *burnout*, apesar da reconhecida relevância desses fatores em cursos de computação. Esta omissão abre uma oportunidade substancial para pesquisas futuras que integrem indicadores de bem-estar, possibilitando uma avaliação mais holística do impacto das APs.

Por fim, o trabalho de [22] apresenta a falta de APs que usufruam de inteligência artificial. Para trabalhos futuros, recomenda-se investigar a adoção desta tecnologia como camada transversal às APs, não apenas para personalizar percursos de aprendizagem, mas também para monitorar em tempo real indicadores de saúde mental e engajamento.

## Referências

[1] L. E. Margulieux, B. B. Morrison, and A. Decker, "Reducing withdrawal and failure rates in introductory programming with subgoal labeled worked examples," *International Journal of STEM Education*, vol. 7, pp. 1–16, 2020.

[2] N. Guibert, P. Girard, and L. Guittet, "Example-based programming: a pertinent visual approach for learning to program," *Proceedings of the working conference on Advanced visual interfaces*, 2004.

[3] K. E. Nolan and S. Bergin, "The role of anxiety when learning to program: a systematic review of the literature," *Proceedings of the 16th Koli Calling International Conference on Computing Education Research*, 2016.

[4] J. Owolabi, P. Olanipekun, and J. Iwerima, "Mathematics ability and anxiety, computer and programming anxieties, age and gender as determinants of achievement in basic programming," *GSTF Journal on Computing (JoC)*, vol. 3, pp. 1–6, 2014.

[5] P. F. Lovibond and S. H. Lovibond, "The structure of negative emotional states: Comparison of the depression anxiety stress scales (dass) with the beck depression and anxiety inventories," *Behaviour Research and Therapy*, vol. 33, no. 3, pp. 335–343, 1995.

[6] R. L. Spitzer, K. Kroenke, J. B. W. Williams, and B. Löwe, "A brief measure for assessing generalized anxiety disorder: The gad-7," *Archives of Internal Medicine*, vol. 166, no. 10, pp. 1092–1097, 2006.

[7] R. C. Kessler, G. Andrews, L. J. Colpe, E. Hiripi, D. K. Mroczek, S.-L. T. Normand, E. E. Walters, and A. M. Zaslavsky, "Short screening scales to monitor population prevalences and trends in non-specific psychological distress," *Psychological Medicine*, vol. 32, no. 6, pp. 959–976, 2002.

[8] F. J. M. Coelho, E. H. T. Oliveira, F. Pereira, D. B. F. Oliveira, L. S. G. Carvalho, E. Souto, M. Pessoa, R. Melo, M. A. P. de Lima, and F. G. Nakamura, "Learning analytics in introductory programming courses: A showcase from the federal university of amazonas," *Revista Brasileira de Informática na Educ.*, vol. 31, pp. 1089–1127, 2023.

[9] D. Zhidkikh, V. Heilala, C. V. Petegem, P. Dawyndt, M. Järvinen, S. Viitanen, B. D. Wever, B. Mesuere, V. Lappalainen, L. Kettunen, and R. Hämäläinen, "Reproducing predictive learning analytics in cs1: Toward generalizable and explainable models for enhancing student retention," *Journal of Learning Analytics*, vol. 11, no. 1, pp. 132–150, Jan. 2024. [Online]. Available: https://learning-analytics.info/index.php/JLA/article/view/7979

[10] A. P. Ambrosio, F. Costa, L. Almeida, A. R. Franco, and J. Macedo, "Identifying cognitive abilities to improve cs1 outcome," *2011 Frontiers in Education Conference (FIE)*, pp. F3G–1–F3G–7, 2011.

[11] E. S. Puudist, A. Luberg, and K. Aus, "Work-in-progress: Exploring psychological and behavioural differences between university it student segments formed based on dropout time and academic performance," in *Learning in the Age of Digital and Green Transition*, M. E. Auer, W. Pachatz, and T. Rüütmann, Eds. Cham: Springer International Publishing, 2023, pp. 590–596.

[12] Q. Cutts, E. Cutts, S. Draper, P. O'Donnell, and P. Saffrey, "Manipulating mindset to positively influence introductory programming performance," in *Proceedings of the 41st ACM Technical Symposium on Computer Science Education*, ser. SIGCSE '10. New York, NY, USA: Association for Computing Machinery, 2010, p. 431–435. [Online]. Available: https://doi.org/10.1145/1734263.1734409

[13] A. Sørensen, P. Lagestad, and H. K. Mikalsen, "Student teacher experiences of learning and pedagogical involvement using a student-centered learning approach," *Education Sciences*, vol. 13, no. 9, 2023. [Online]. Available: https://www.mdpi.com/2227-7102/13/9/965

[14] A. Yadin, "Reducing the dropout rate in an introductory programming course," *Inroads*, vol. 2, pp. 71–76, 2011.

[15] T. T. Mai, M. Crane, and M. Bezbradica, "Students' learning behaviour in programming education analysis: Insights from entropy and community detection," *Entropy*, vol. 25, 2023.

[16] S. I. Malik and J. Coldwell-Neilson, "A model for teaching an introductory programming course using adri," *Education and Information Technologies*, vol. 22, pp. 1089 – 1120, 2016.

[17] A. Hawlitschek, V. Köppen, A. Dietrich, and S. Zug, "Drop-out in programming courses – prediction and prevention," *Journal of Applied Research in Higher Education*, vol. 12, no. 1, pp. 124–136, 2020. [Online]. Available: https://doi.org/10.1108/JARHE-02-2019-0035

[18] K. Dowling, A. J. Simpkin, and M. M. Barry, "A cluster randomized-controlled trial of the mindout social and emotional learning program for disadvantaged post-primary school students," *Journal of Youth and Adolescence*, vol. 48, no. 7, pp. 1245–1263, Jul. 2019. [Online]. Available: https://doi.org/10.1007/s10964-019-00987-3

[19] Y.-S. Hwang, J.-E. Noh, O. N. Medvedev, and N. N. Singh, "Effects of a mindfulness-based program for teachers on teacher wellbeing and person-centered teaching practices," *Mindfulness*, vol. 10, no. 11, pp. 2385–2402, Nov. 2019. [Online]. Available: https://doi.org/10.1007/s12671-019-01236-1

[20] J. Figueiredo and F. García-Peñalvo, "Teaching and learning strategies for introductory programming in university courses," *Ninth International Conference on Technological Ecosystems for Enhancing Multiculturality (TEEM'21)*, 2021.

[21] M. Page, J. McKenzie, P. Bossuyt, I. Boutron, T. Hoffmann, C. Mulrow, L. Shamseer, J. Tetzlaff, E. Akl, S. Brennan, R. Chou, J. M. Glanville, J. Grimshaw, A. Hrõbjartsson, M. Lalu, T. Li, E. Loder, E. Mayo-Wilson, S. McDonald, L. McGuinness, L. Stewart, J. Thomas, A. Tricco, V. Welch, P. Whiting, and D. Moher, "The prisma 2020 statement: An updated guideline for reporting systematic reviews." *Journal of clinical epidemiology*, 2021.

[22] F. Silva, C. Menezes, and A. C. Junior, "Ensino introdutório de programação: Um estudo rumo ao uso das arquiteturas pedagógicas," in *Anais do XXXIV Simpósio Brasileiro de Informática na Educação*. Porto Alegre, RS, Brasil: SBC, 2023, pp. 428–438. [Online]. Available: https://sol.sbc.org.br/index.php/sbie/article/view/26682

[23] J. Piaget, *Piaget's Theory*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1976, pp. 11–23. [Online]. Available: https://doi.org/10.1007/978-3-642-46323-5_2

[24] A. Carrera-Rivera, W. Ochoa, F. Larrinaga, and G. Lasa, "How-to conduct a systematic literature review: A quick guide for computer science research," *MethodsX*, vol. 9, p. 101895, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2215016122002746

[25] A. P. Siddaway, A. Wood, and L. Hedges, "How to do a systematic review: A best practice guide for conducting and reporting narrative reviews, meta-analyses, and meta-syntheses." *Annual review of psychology*, vol. 70, pp. 747–770, 2019.

[26] S. A. Papert, *Mindstorms: Children, computers, and powerful ideas*. Basic books, 2020.

[27] C. Coll, E. Martín, T. Mauri, M. Miras, J. Onrubia, I. Sole, and A. Zabala, "O construtivismo na sala de aula," in *O construtivismo na sala de aula*. Editora Ática, 2004, pp. 221–221.

[28] T. J. Masson, L. F. d. Miranda, A. Munhoz Jr, and A. M. P. Castanheira, "Metodologia de ensino: aprendizagem baseada em projetos (pbl)," in *Anais do XL Congresso Brasileiro de Educação em Engenharia (COBENGE), Belém, PA, Brasil*, vol. 13. sn, 2012.

[29] P. V. Santos, "Metodologias ativas," *Educação e Matemática*, pp. 2–4, 2006.

[30] J. M. Wing, "Pensamento computacional," *Educação e Matemática*, pp. 2–4, 2021.

[31] V. Grout and N. Houlden, "Taking Computer Science and Programming into Schools: The Glyndŵr/BCS Turing Project," *Procedia - Social and Behavioral Sciences*, vol. 141, pp. 680–685, Aug. 2014. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1877042814035435

[32] E. Brunvand, "Games as motivation in computer design courses: I/o is the key," in *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education*, ser. SIGCSE '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 33–38. [Online]. Available: https://doi.org/10.1145/1953163.1953178

[33] O. De Lira Tavares, C. S. De Menezes, and R. A. De Nevado, "Pedagogical architectures to support the process of teaching and learning of computer programming," in *2012 Frontiers in Education Conference Proceedings*. Seattle, WA, USA: IEEE, Oct. 2012, pp. 1–6. [Online]. Available: http://ieeexplore.ieee.org/document/6462427/

[34] O. Tavares, C. Menezes, R. Aragón, and L. Costa, "Uma arquitetura pedagógica auxiliada por tecnologias para ensino e aprendizagem de programação," in *Anais do XXI Workshop sobre Educação em Computação*. Porto Alegre, RS, Brasil: SBC, 2013, pp. 778–787. [Online]. Available: https://sol.sbc.org.br/index.php/wei/article/view/27774

[35] M. Scalera, A. Marengo, V. S. Barletta, D. Caivano, G. Dimauro, and J. Pange, "Rethinking pedagogy for computer architecture: A Simplified Approach to teach a Processor (SAtTaP)," *Education and Information Technologies*, vol. 30, no. 6, pp. 7357–7386, Apr. 2025. [Online]. Available: https://link.springer.com/10.1007/s10639-024-13091-2

[36] I. R. D. S. Fausto, F. R. Leta, R. M. M. Braz, and S. C. C. D. S. Pinto, "Pedagogical Architecture Trail: Virtual Environment of Learning of the Initial and Continuing Training Course in Educational Robotics in Basic Education - Instituto Federal of Education, Science and Technology - RO - IFRO," in *Anais do I Workshop de Pensamento Computacional e Inclusão (WPCI 2022)*. Brasil: Sociedade Brasileira de Computação, Nov. 2022, pp. 33–41. [Online]. Available: https://sol.sbc.org.br/index.php/wpci/article/view/22557

[37] A. Pereira and C. S. D. Menezes, "Formação de Professores para Uso de Tecnologias Digitais na Educação: um experimento com licenciandos em computação," in *Anais do XXIX Workshop de Informática na Escola (WIE 2023)*. Brasil: Sociedade Brasileira de Computação - SBC, Nov. 2023, pp. 1318–1323. [Online]. Available: https://sol.sbc.org.br/index.php/wie/article/view/26412

[38] F. J. Portilho, J. V. F. Amaral, N. A. Rodrigues, C. X. Pereira Junior, and N. T. Costa, "Uma Arquitetura Pedagógica para o Ensino de Lógica de Programação: Lições Aprendidas a partir de um Projeto de Extensão," in *Anais do XXXII Workshop sobre Educação em Computação (WEI 2024)*. Brasil: Sociedade Brasileira de Computação - SBC, Jul. 2024, pp. 329–340. [Online]. Available: https://sol.sbc.org.br/index.php/wei/article/view/29637

[39] S. Federici, E. Sergi, C. Medas, R. Lussu, E. Gola, and A. Zuncheddu, "Computational Pedagogy: Block Programming as a General Learning Tool," in *Computer Supported Education*, H. C. Lane, S. Zvacek, and J. Uhomoibhi, Eds. Cham: Springer International Publishing, 2020, vol. 1220, pp. 211–235, series Title: Communications in Computer and Information Science. [Online]. Available: http://link.springer.com/10.1007/978-3-030-58459-7_11

[40] T. S. Santana, T. N. Kudo, and R. F. Bulcão-Neto, "Requisitos em Ação: Uma Arquitetura Pedagógica para o Ensino de Engenharia de Requisitos de Software," in *Anais do XXXIV Simpósio Brasileiro de Informática na Educação (SBIE 2023)*. Brasil: Sociedade Brasileira de Computação - SBC, Nov. 2023, pp. 210–221. [Online]. Available: https://sol.sbc.org.br/index.php/sbie/article/view/26663