Uma Arquitetura pedagógica para apoiar a aprendizagem de programação introdutória no ensino superior

Jefferson Ribeiro de Lima - PPGIE / UFRGS - jlima@ifes.edu.br,
https://orcid.org/0000-0001-6677-0970
Crediné Silva de Menezes - PPGIE / UFRGS - credine@gmail.com,
https://orcid.org/0000-0002-2709-7135

Resumo: A aprendizagem de programação é frequentemente apontada na literatura acadêmica como uma das principais dificuldades enfrentadas por estudantes ingressantes em cursos superiores de informática, resultando em altas taxas de reprovação e evasão. Para enfrentar essa problemática, propõe-se um arcabouço conceitual que visa aprimorar a qualidade desta aprendizagem. A proposta fundamenta-se na concepção e integração de cinco Arquiteturas Pedagógicas, baseadas na resolução de problemas e na teoria construtivista de Piaget, proporcionando dinâmicas que favorecem a mobilização do conhecimento prévio, a interação e a cooperação entre os estudantes. Como resultado, esta pesquisa apresenta os fundamentos conceituais do arcabouço, além de como realizar a sua aplicação prática.

Palavras-Chaves: Aprendizagem de programação, Arquitetura Pedagógica, Resolução de Problemas.

A Pedagogical Architecture to Support Introductory Programming Learning in Higher Education

Abstract: Programming learning is often highlighted in academic literature as one of the main challenges faced by students entering undergraduate computer science programs, resulting in high failure and dropout rates. To address this issue, a conceptual framework is proposed to improve the quality of programming learning. The proposal is based on the design and integration of five Pedagogical Architectures, grounded in problem-solving and Piaget's constructivist theory, offering dynamics that encourage the mobilization of prior knowledge, as well as interaction and cooperation among students. As a result, this research presents the conceptual foundations of the framework, along with guidelines for its practical implementation.

Keywords: Programming Learning, Pedagogical Architecture, Problem Solving.

1. Introdução

Em um mundo onde lidar com tecnologia é cada vez mais importante para participar da vida social e econômica (Wing, 2006), muitas pessoas buscam adquirir uma nova habilidade que se torna cada vez mais importante no cenário contemporâneo, a programação de computadores. Esse fato é especialmente evidente nos cursos superiores que oferecem esse tipo de formação. Entretanto, a diferença entre a demanda por profissionais qualificados e as persistentes taxas de insucesso acadêmico revelam lacunas estruturais nos modelos educacionais vigentes.

Os estudantes iniciantes de programação na graduação enfrentam diversos obstáculos, como entender o raciocínio lógico e aplicar o que estão aprendendo na prática (Soloway e Spohrer, 1989). Além disso, a ausência de estratégias diferenciadas que propiciem um ensino mais ativo pode estagnar a evolução do raciocínio destes aprendizes (Bennedsen e Caspersen, 2007). Para enfrentar esse problema, é necessário repensar o modo como aprendemos programação de computadores, indo além do conteúdo técnico e das regras de sintaxe das linguagens. Como afirmou Papert (1980), os estudantes aprendem melhor quando estão em ambientes que se assemelham a situações reais, onde

errar faz parte do processo e a cooperação é incentivada.

Com o objetivo de contribuir para superar algumas dessas dificuldades, propomos uma alternativa para apoiar a aprendizagem de programação através da concepção de um arcabouço conceitual. Para facilitar a compreensão desta proposta, este artigo foi estruturado em cinco seções, a partir da introdução. A Seção 2 apresenta os fundamentos teóricos desta abordagem. A Seção 3 descreve a metodologia utilizada no estudo. A Seção 4 detalha o arcabouço conceitual elaborado para apoiar a aprendizagem de programação, além de orientar sobre a sua instanciação. A Seção 5 apresenta as considerações finais desta pesquisa.

2. Fundamentação

Esta seção apresenta os fundamentos teóricos que embasam a proposta do arcabouço conceitual proposto, vislumbrando a promoção da aprendizagem de programação de computadores. Cada subseção apresenta um aspecto essencial para a construção do modelo, conectando as teorias e práticas educacionais.

2.1 O Ensino de Programação em Cursos Superiores no Brasil

A formação em computação e áreas correlatas no Brasil apresenta em sua estrutura curricular componentes essenciais, como lógica de programação e algoritmos. Esses elementos são fundamentais na formação dos estudantes e estão alinhados às Diretrizes Curriculares Nacionais (DCNs), que estabelecem padrões para a construção dos Projetos Pedagógicos de Curso (PPCs), garantindo coerência e qualidade na educação superior.

De acordo com o Sistema Nacional de Avaliação da Educação Superior (Brasil, 2014), os PPCs devem ser elaborados de forma participativa, envolvendo a comunidade acadêmica, docentes e a coordenação do curso. Além disso, precisam estar articulados ao Plano Pedagógico Institucional (PPI) e ao Plano de Desenvolvimento Institucional (PDI), assegurando uma gestão pedagógica e administrativa eficiente. Essa estrutura visa não apenas à transmissão de conteúdo, mas ao desenvolvimento de competências essenciais para os futuros profissionais.

As DCNs desempenham um papel crucial na organização curricular dos cursos de computação, orientando a definição de conteúdo, carga horária e metodologias de ensino. Essa normatização contribui para a padronização da formação em todo o país, garantindo que os estudantes adquiram uma base sólida de conhecimento. No entanto, conforme Ranali e Lombardo (2006), a implementação dessas diretrizes deve ser flexível o suficiente para permitir a adaptação às necessidades específicas de cada instituição e aos avanços da área tecnológica.

Além das DCNs, a Sociedade Brasileira de Computação (SBC) também fornece diretrizes específicas para a formação em computação, apresentadas nos Referenciais de Formação para os Cursos de Graduação em Computação (SBC, 2017). Esse documento substituiu o antigo Currículo de Referência de 2003 e trouxe mudanças significativas, organizadas em três eixos principais:

2.1.1 Ênfase no Desenvolvimento de Competências

- Substituição do foco rígido em disciplinas por uma abordagem baseada em competências essenciais, como resolução de problemas, comunicação e trabalho em equipe.
- Maior flexibilidade curricular, permitindo que as instituições adaptem seus programas às demandas regionais e tecnológicas.

2.1.2 Integração Interdisciplinar e Formação Integral

- Articulação de conhecimentos de diversas áreas, como matemática, física, humanidades e ciências sociais, promovendo uma formação mais abrangente.
- Desenvolvimento de uma visão holística para que os profissionais compreendam melhor os desafios do mundo real e se tornem cidadãos responsáveis.

2.1.3 Transformações na Abordagem da Programação

- Maior foco na aplicação prática e na resolução de problemas reais, utilizando múltiplas linguagens de programação.
- Ênfase no pensamento computacional e em conceitos fundamentais, preparando os estudantes para inovações futuras no campo da computação.

O incentivo ao uso de tecnologias digitais no ensino da programação também é um aspecto essencial. O PPC deve contemplar estratégias que possibilitem a experimentação prática por meio de projetos, atividades extracurriculares e pesquisa aplicada. A flexibilidade curricular, somada à abordagem interdisciplinar, permite que os alunos explorem diferentes áreas da computação e desenvolvam habilidades essenciais para o mercado de trabalho. Portanto, a estruturação do ensino de programação nos cursos superiores brasileiros se baseia em diretrizes nacionais e recomendações acadêmicas que visam equilibrar rigor conceitual e inovação metodológica. O desafio contínuo é manter essa formação alinhada às transformações tecnológicas e às demandas do setor, garantindo profissionais preparados para os desafios contemporâneos.

2.2 Construtivismo

O construtivismo, conforme Piaget (1970), define que o conhecimento é construído ativamente pelo sujeito por meio da interação entre a biologia e o meio social. O desenvolvimento cognitivo é resultado de um processo contínuo de assimilação e acomodação, em que o indivíduo reorganiza suas estruturas mentais a partir da experiência. Esse movimento dinâmico ocorre por meio da interação com o ambiente, com os objetos e com outras pessoas, o que favorece a tomada de consciência e a construção de sentidos para a realidade vivida (Piaget, 1998).

A teoria construtivista é especialmente relevante para o ensino de programação, pois enfatiza a importância de contextos que estimulem a reflexão, a experimentação e a interação social. Esses elementos são fundamentais para que os estudantes desenvolvam o pensamento computacional e superem os desafios iniciais da aprendizagem de programação, conectando-se diretamente às práticas propostas na resolução de problemas de um modo geral.

A aprendizagem, nesse contexto, é vista como um processo ativo e único para cada indivíduo, que envolve a criação de construções mentais próprias, independentemente da forma como o conteúdo é apresentado. O estudante deve ser incentivado a abandonar a postura passiva e reconstruir seu conhecimento, inclusive por meio do desequilíbrio cognitivo, que promove novas adaptações e amplia o desenvolvimento intelectual. Assim, o construtivismo se refere a um conjunto de teorias sobre como se aprende, influenciando a pedagogia, mas não sendo uma proposta didática em si (Piaget, 1998).

2.3 Arquiteturas Pedagógicas

As Arquiteturas Pedagógicas (AP) surgem como uma resposta às transformações

educacionais impulsionadas pelo avanço das tecnologias digitais, propondo práticas que integram tecnologia e educação de forma dinâmica e contextualizada.

As AP, conforme proposto por Carvalho, Nevado e Menezes (2005), intercalam tecnologia e educação para promover um ensino interativo, onde o estudante assume um papel protagonista no processo de aprendizagem. Segundo Aragón, Michels e Araujo (2018), a integração das tecnologias digitais com uma metodologia sistematizada possibilita a criação de arquiteturas pedagógicas flexíveis e adaptáveis a diferentes contextos. Essa abordagem enfatiza a colaboração não apenas entre professores e alunos, mas também entre os próprios pares na aprendizagem.

Para Menezes et al. (2013), a proposta das arquiteturas pedagógicas não é simplesmente inserir tecnologia em práticas tradicionais de ensino, mas repensar os aspectos pedagógicos a partir das possibilidades tecnológicas. Assim, o suporte telemático é utilizado para estimular o diálogo, a pesquisa e a resolução de problemas, permitindo uma reconstrução contínua do conhecimento.

Além disso, as AP se baseiam em cinco princípios fundamentais, conforme descrito por Carvalho, Nevado e Menezes (2007): i) ensino voltado à resolução de problemas reais; ii) desenvolvimento da capacidade de transformar informações em conhecimento; iii) incentivo à autoria e ao uso de diferentes linguagens; iv) estímulo à investigação e à criação; e v) formação para autonomia e cooperação. Diante dessas perspectivas, percebe-se que a utilização das AP na educação contribui para a inovação pedagógica, promovendo uma aprendizagem significativa e alinhada às necessidades contemporâneas.

2.4 A resolução de problemas para apoiar a aprendizagem

A resolução de problemas é um elemento central no processo de aprendizagem de programação, promovendo reflexão e autonomia nos estudantes. Segundo Bersanette e Francisco (2021), a dificuldade na aprendizagem da programação de computadores vai além do entendimento do desenvolvimento de códigos, estando fortemente relacionada à capacidade de resolver problemas. Sem a promoção dessa habilidade, superar os desafios iniciais na aprendizagem de programação torna-se mais difícil.

A implementação pedagógica da abordagem de resolução de problemas, segundo Lago et al. (2019), promove uma ruptura com modelos instrucionais passivos, transformando estudantes em agentes ativos de sua formação. Nesse cenário, a exposição a problemas não se limita à obtenção de respostas, mas estimula a reconstrução crítica da realidade e o desenvolvimento de estratégias metacognitivas (Jonassen, 2000).

2.4.1 Abordagens Teóricas para Resolução de Problemas

A conceituação de "problema" e seus processos resolutivos tem gerado amplo debate acadêmico. Pérez et al. (1988) definem-no como uma situação carente de soluções imediatas, geradora de tensão cognitiva. Perales (1993) avança nessa perspectiva, caracterizando-o como um fenômeno que exige mobilização de conhecimentos prévios para transpor estados de incerteza.

Para Mayer (1998), a resolução de problemas envolve uma visão multidimensional, enfatizando a interdependência entre fatores motivacionais, cognitivos e contextuais. Paralelamente, Pozo e Echeverría (1998) definem a resolução de problemas como um ciclo metacognitivo quadrifásico: representação mental do problema; planejamento estratégico; execução monitorada; e avaliação regulatória. Dentre os modelos influentes, destaca-se o framework conceitual de Polya (1977), que é organizado

em quatro estágios interdependentes: i) compreensão profunda do problema; ii) seleção de estratégias; iii) implementação sistematizada; iv) e reflexão pós-solução.

Com base nos referenciais teóricos citados, a resolução de problemas configura-se como um eixo estruturante no processo de aprendizagem, especialmente em áreas como a programação, que exigem autonomia, pensamento crítico e raciocínio lógico.

2.5 Trabalhos Relacionados

A seguir, destacamos algumas abordagens existentes para o ensino e aprendizagem de programação relacionadas ao contexto deste estudo, apresentando as suas principais contribuições. Na abordagem proposta por Bennedsen e Caspersen (2007) enfatiza o ensino de programação por meio de práticas guiadas e exercícios estruturados, mas carece de uma integração explícita com teorias construtivistas, limitando o estímulo à autonomia do estudante.

No estudo de Lye e Koh (2014), foram revisadas abordagens baseadas no construtivismo para o ensino de programação, com ênfase na ferramenta *Scratch*. Além disso, os autores destacam que o construtivismo favorece a construção ativa do conhecimento. Contudo, esta abordagem não organiza a aprendizagem em etapas, dificultando assim a sua aplicação.

Já Funke e Geldreich (2020) analisaram o impacto de abordagens ativas, como a programação em pares e a resolução de problemas colaborativa, no ensino de programação introdutória. Seus resultados indicam que essas estratégias aumentam a motivação e o desempenho dos alunos, mas a ausência de uma base construtivista limita a profundidade do desenvolvimento cognitivo dos participantes deste estudo.

3. Metodologia

A metodologia deste estudo ancorou-se nos princípios da Design Science Research (DSR), paradigma voltado à geração sistemática de artefatos inovadores que respondam a desafios complexos do mundo real (Gregor & Hevner, 2013). Conforme descrito por Hevner et al. (2004), pesquisas nesta abordagem devem produzir artefatos funcionais com propósito claramente delimitado, os quais podem manifestar-se como modelos teóricos ou instanciações práticas.

Essa estrutura opera mediante três ciclos dinâmicos interconectados (Hevner, 2007): Ciclo de Relevância: articula o problema de pesquisa com demandas contextuais emergentes, garantindo aplicabilidade prática e impacto social; Ciclo de Rigor: estabelece diálogo crítico com o corpus científico existente, assegurando fundamentação teórica robusta e contribuições epistemológicas originais; e Ciclo de Design: orienta a construção iterativa do artefato através de prototipação e avaliação cíclica, permitindo refinamentos progressivos até atingir maturidade conceitual.

Conforme destacado por Vaishnavi (2007), os frameworks originados deste processo transcendem meras ferramentas, convertendo-se em estruturas de mediação cognitiva que reconfiguram práticas profissionais. Essa natureza dialética entre teoria e prática posiciona a DSR como mecanismo privilegiado para implementar inovação.

3.1 Etapas da pesquisa

A seguir, são apresentadas as etapas utilizadas na construção deste arcabouço conceitual. O desenvolvimento metodológico consiste na adaptação do modelo de DSR proposto por Hevner et al. (2004).

1. Identificação do Problema

Define-se o problema com base no contexto em que está inserido, considerando pessoas, organizações e tecnologias. Neste estudo, a dificuldade no ensino de programação motivou a investigação.

2. Relevância do Problema

   O alto índice de reprovação em disciplinas de programação, segundo o Brasil (2024), está variando entre 30% e 40%, evidencia a necessidade de novas abordagens pedagógicas para facilitar o aprendizado.

3. Conscientização do Problema

   A revisão sistemática da literatura identificou dificuldades no ensino de programação. Foram analisadas publicações recentes nas bases ACM, IEEE e Scopus para fundamentar o estudo.

4. Elaboração do Artefato

   A concepção do artefato baseia-se nas informações levantadas, definindo requisitos e funcionalidades. A solução proposta utiliza um Ambiente Virtual de Aprendizagem para suporte.

5. Contribuições da Pesquisa

   A pesquisa propõe a APAAP, uma Arquitetura Pedagógica para apoiar o pensamento computacional antes da codificação, promovendo autonomia e conexão com a realidade do aprendiz.

6. Avaliação do Artefato

   O artefato foi avaliado por estudantes de um curso de extensão, analisando a sua operacionalização e eficácia preliminar, com foco na aplicação das primeiras quatro APs.

7. Comunicação e Resultado

   Os resultados são disseminados na comunidade acadêmica por meio de artigos científicos apresentados em eventos, congressos e periódicos nacionais e internacionais.

4. Arcabouço conceitual - APAAP

A Arquitetura Pedagógica para Apoiar a Aprendizagem de Programação (APAAP) foi criada para favorecer a compreensão dos estudantes ingressantes em disciplinas introdutórias de programação, no ensino superior. Este arcabouço foi concebido por meio de três pilares fundamentais: i) a teoria construtivista de Piaget, que mostra como os desafios ajudam no processo de reflexão e autoconhecimento; ii) a aprendizagem baseada na resolução de problemas, que transforma as dificuldades dos alunos em oportunidades para se desenvolver cognitivamente e iii) a utilização das arquiteturas pedagógicas (AP), que são suportes estruturantes de aprendizagem que utilizam a tecnologia como ferramental para promoção da aprendizagem, valorização e protagonismo do estudante. A partir destes pilares, esta proposta se concretiza por meio da disponibilização de cinco módulos de AP, pensadas não apenas para apoiar as práticas pedagógicas, mas também, para permitir o desenvolvimento intelectual a partir de um ciclo contínuo de pensamento que envolve o desequilíbrio, a reflexão e a reconstrução. As distribuições destes módulos buscam utilizar um viés interacionista e dialético.

A estrutura desta proposta é interconectada a partir dos seus módulos (Figura 1), visando promover a descoberta de habilidades e competências, com foco no desenvolvimento cognitivo através das práticas realizadas pelos estudantes. A seguir, detalhamos estes módulos de AP e suas contribuições para apoiar a construção da aprendizagem.

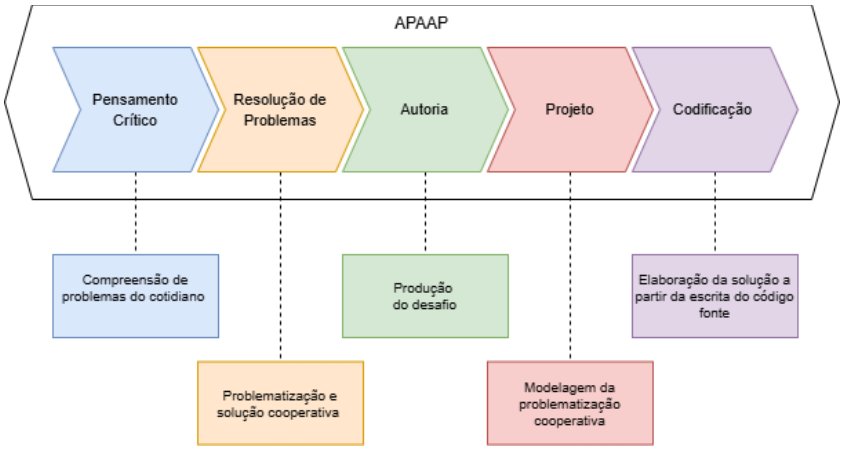

Figura 1 – Módulos de aprendizagem da APAAP (autor, 2024)

AP - Pensamento Crítico

Desenvolver a capacidade de reflexão e resolução de problemas simples, estimulando o pensamento lógico e crítico. Inicialmente, os estudantes são incentivados a propor e solucionar problemas simples do próprio dia a dia, após alguns exemplos apresentados pela mediação. Assim, cada estudante tenta propor e solucionar os problemas do seu cotidiano, levantando hipóteses e possíveis soluções com ajuda de todo o grupo. Além disso, este módulo utiliza uma ferramenta para simulação, objetivando gerar engajamento para a aprendizagem de programação. No segundo momento, os estudantes compartilham com o grupo suas experiências após a utilização do simulador. Por fim, esta AP promove a motivação e a autonomia, visando incentivar a aquisição de novas habilidades. Esse módulo relaciona-se ao conceito de operações mentais, descrito na teoria de Piaget (1977), sobre a capacidade de manipular mentalmente objetos. Complementa-se ao conceito de “práxis” de Freire (1985), onde há uma integração da teoria e da prática.

AP - Resolução de Problemas

Esse módulo capacita os estudantes a identificar, descrever e resolver problemas cotidianos mais complexos, utilizando conhecimentos prévios e trabalho em equipe. Piaget (1970) destaca a importância das interações dos sujeitos com o ambiente social, assim aprendem e desenvolvem habilidades cognitivas. Neste caso, os estudantes são incentivados a trabalhar em grupos para propor e avaliar soluções. Assim, aprendem a resolver problemas mais complexos em grupo utilizando conceitos do pensamento computacional, como decomposição, reconhecimento de padrões, abstração e algoritmos. Deste modo, desenvolvem habilidades de cooperação, problematização e pensamento estruturado, preparando os estudantes para enfrentar situações desafiadoras ou imprevistos da vida real.

AP - Autoria

Estimula a autonomia e a criatividade, permitindo que os estudantes criem seus próprios desafios descritivos e os compartilhem com os colegas. Essa prática incentiva

a autoria, pois a tarefa exige que sejam elaborados desafios (inspirados em jogos educacionais, com regras e dicas), que, no segundo momento, são resolvidos e revisados por pares. A ideia consiste em promover autonomia, raciocínio, a prática da resolução de problemas, além das trocas a partir das interações entre os colegas, com o objetivo de apoiar a reconstrução do pensamento. A maior contribuição é o fortalecimento da criatividade, capacidade analítica e crítica, além de promover a cooperação a partir da revisão por pares.

AP - Projeto

Esse módulo foi construído para reforçar a aprendizagem cooperativa. Os grupos iniciais (AP Resolução de Problemas) são retomados para revisitar o trabalho elaborado em grupo, tentando desenvolver soluções alternativas que ainda não envolvam codificação, como diagramas ou protótipos. Os grupos devem trabalhar na transição das soluções para modelos visuais ou diagramáticos que possam ser posteriormente implementados computacionalmente. Neste módulo, os grupos podem utilizar ferramentas para modelagem ou geração de interfaces, com a finalidade de colocar em prática as ideias que inicialmente eram apenas descritivas, objetivando a aquisição de habilidades que envolvem revisão, planejamento e reconstrução. Para Piaget (1972), a construção ativa do conhecimento incentiva a experimentação e reflexão para uma compreensão mais profunda.

AP - Codificação

Essa AP consolida as etapas descritivas anteriores a partir da transição das propostas para uma linguagem codificada (linguagem de programação), transformando as soluções intermediárias em programas funcionais. Os estudantes colocam em prática os trabalhos desenvolvidos a partir da escolha da linguagem. Inicialmente, cada estudante desenvolve um recorte da proposta realizada em grupo. Entre as principais contribuições deste módulo, destacamos o desenvolvimento do raciocínio abstrato e lógico, além da concretização das ideias que envolvem a codificação. Segundo Piaget (1977), no estágio formal, os sujeitos desenvolvem a capacidade de pensar de maneira abstrata e lógica, formulando hipóteses e deduzindo conclusões a partir destas.

4.1 Instanciação do arcabouço

A validação do conceito proposto pode ser realizada a partir da instanciação deste arcabouço, que ocorre a partir da convergência de momentos sócio interativos que emergem das práticas pedagógicas. No momento da instanciação, o mediador apresenta exemplos com cenários de problemas contextualizados a partir do cotidiano, com problemas atuais e de conhecimento da maioria. Além disso, se mobiliza para incentivar a participação e a cooperação para que os estudantes identifiquem, descrevam e solucionem os problemas que serão vivenciados. Os resultados das produções obtidas são compartilhados com os demais participantes, para que possam sugerir alternativas para o aperfeiçoamento da ideia original. O papel do estudante durante o processo é identificar e problematizar situações diversas, que podem ser definidas coletivamente, individual ou em grupos.

Cada problema abordado requer concentração e reflexão na solução, de acordo com o propósito do módulo de aprendizagem explorado. Entretanto, indiferente da situação-problema, as propostas para resolução devem partir do conhecimento prévio. Os resultados obtidos na resolução devem abarcar as habilidades que incentivem o protagonismo dos envolvidos e a cooperação durante o processo de construção da ideia.

Essa autonomia pode ajudar nos momentos de reflexão, apoiando o desenvolvimento do pensamento crítico, essencial para o avanço cognitivo, que pode ajudar na interpretação para a resolução de qualquer tipo de problema, incluindo os que se relacionam com a computação.

Durante a abordagem de construção do conhecimento serão geradas provocações para que o estudante identifique e contextualize a situação, formule hipóteses, teste possibilidades e verifique os resultados. Ao enfrentar problemas, os envolvidos no processo de aprendizagem passam a lidar com os erros, frustrações e a procurem por alternativas. Essa independência objetiva incentivar a autonomia para apoiar o desenvolvimento de outras habilidades, que vão surgir a partir das ações individuais desempenhadas por cada participante. Segundo Piaget, é necessário valorizar mais o "como" aprender do que "quando" aprender, por isso, é necessário respeitar os tempos individuais para um desenvolvimento saudável.

Ao longo de todos os módulos de aprendizagem, a cooperação pode contribuir diretamente para que haja um avanço de outras habilidades, e, consequentemente, das ideias elaboradas, em virtude do olhar diferenciado dos sujeitos sobre o próprio conhecimento adquirido. Além disso, a mediação distribuída a partir da revisão por pares e do trabalho em grupo, podem contribuir para a socialização, a comunicação e a reflexão crítica, a partir de desequilíbrios e reconstruções cognitivas sucessivas.

Acreditamos que mediação junto do apoio e suporte tecnológico pode auxiliar no desenvolvimento da autonomia e reflexão crítica dos estudantes na resolução de problemas, ao mesmo tempo que os alunos podem cooperar com os seus pares em soluções alternativas apoiando a produção autoral do colega, que não necessariamente deve ser inédita, e sim, um modo de reforçar a compreensão ou até mesmo a materialização da ideia, gerando um ciclo permanente de aprendizagem utilizado na APAAP, conforme destacamos na Figura 2.

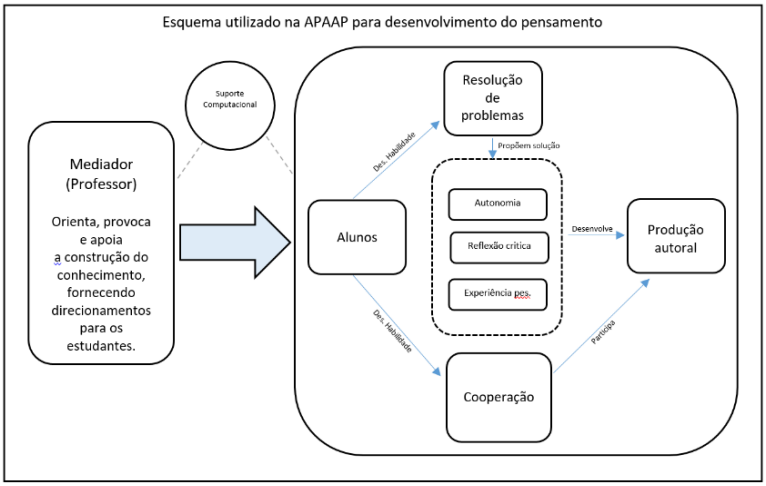

Figura 2 – Instância do ciclo de conhecimento base da APAAP (autor, 2024)

Dentro da estruturação desta esquematização, incluímos também o suporte computacional, que integra às práticas pedagógicas em todos os instantes para aprimorar e enriquecer o aprendizado.

4.2 Suporte computacional

Dentre os recursos tecnológicos utilizados na concepção deste arcabouço, destacamos o Ambiente Virtual de Aprendizagem (AVA) – Moodle, que permite registrar atividades, organizar os conteúdos e facilitar a comunicação. Além disso, utilizamos editores de texto, planilhas, geradores de interface (prototipação) e modelos de diagramas, tudo em nuvem, sem necessidade de instalações locais. A proposta também contempla a utilização da ferramenta *Blockly*, que é usada para apoiar a aprendizagem de lógica e resolução de problemas simples, com a expectativa de proporcionar engajamento dos envolvidos nesta aprendizagem.

Adicionalmente, podem ser utilizadas outras ferramentas ao longo dos módulos, como navegadores de internet, linguagem de programação, IDE para desenvolvimento dos códigos, softwares de editoração e qualquer outra ferramenta que possa auxiliar na construção do conhecimento. Nos encontros síncronos, a videoconferência e a principal ferramenta para viabiliza os encontros virtuais. Já em momentos assíncronos, utilizamos o ambiente virtual de aprendizagem como apoio central. O suporte computacional não apenas sustenta as práticas pedagógicas, mas também amplia as possibilidades de aprendizagem, tornando o processo mais organizado, dinâmico e participativo.

4.3 Resultado da instanciação

A fim de aferir a viabilidade do método desenvolvido foi realizado um estudo preliminar com alguns estudantes voluntários de um curso de extensão em 2024. Entretanto, o resultado deste estudo será publicado em outro artigo. Contudo, antecipando-se a essa publicação, destacamos de modo qualitativo parte da análise dos dados do experimento aplicado, como: o aumento do engajamento dos participantes na aprendizagem de programação, o fortalecimento da autonomia, o desenvolvimento da capacidade de planejamento, e a sinalização da evolução cognitiva dos participantes com relação a aprendizagem dos fundamentos da programação de computadores, além de outros benefícios que serão divulgados posteriormente.

5. Considerações Finais

O arcabouço conceitual deste trabalho busca enfrentar as dificuldades relacionadas ao ensino de programação introdutória no ensino superior. Esta proposta está fundamentada nas teorias construtivistas de Piaget, na Resolução de Problemas e nas abordagens das Arquiteturas Pedagógicas. A partir desses conceitos, foram desenvolvidas cinco Arquiteturas Pedagógicas modulares que articulam e favorecem o desenvolvimento do pensamento computacional antes de qualquer tipo de codificação.

Além disso, este trabalho oferece uma abordagem pedagógica sequencial, centrada em um modelo ativo de aprendizagem que, ao longo do percurso educacional, parte da reflexão crítica sobre problemas do cotidiano que podem culminar na implementação de soluções futuras codificadas. Dessa forma, proporciona aos estudantes experiências que incentivam a autonomia, a criatividade e a reconstrução do conhecimento, a partir da mobilização do conhecimento prévio. A instanciação prática deste arcabouço, embora preliminar, indicou resultados promissores quanto ao engajamento e ao desenvolvimento cognitivo dos participantes, fortalecendo o potencial transformador das Arquiteturas Pedagógicas instanciadas.

Conclui-se que a integração da Resolução de Problemas, da integração tecnológica e da construção ativa do conhecimento pode contribuir para superar os entraves que historicamente dificultam a aprendizagem de programação. Espera-se que, com novos estudos e aplicações desta proposta, haja um refinamento para a posterior incorporação nos currículos educacionais, visando à promoção de uma formação mais alinhada com as realidades individuais na aprendizagem de programação.

Referências

ARAGÓN, Rosane; MICHELS, Ana Beatriz; ARAÚJO, Alexandre. Arquiteturas pedagógicas na formação de professores a distância. **Revista Intersaberes Uninter**, v. 13, n. 29, 2018.

BENNEDSEN, Jens; CASPERSEN, Michael E. Failure rates in introductory programming. **ACM SIGCSE Bulletin**, v. 39, n. 2, p. 32–36, 2007.

BERSANETTE, Silvia P.; FRANCISCO, Rosângela M. M. Por que é tão difícil aprender programação? **Revista Brasileira de Informática na Educação**, v. 29, n. 2, p. 48–71, 2021.

BRASIL. **Ministério da Educação. Sistema Nacional de Avaliação da Educação Superior (SINAES).** Brasília: INEP, 2014.

CARVALHO, Marie Jane S.; DE NEVADO, Rosane Aragon; DE MENEZES, Crediné Silva. Arquiteturas pedagógicas para educação à distância: concepções e suporte telemático. In: **Simpósio Brasileiro de Informática na Educação-SBIE**, 2005.

CARVALHO, Marie Jane Soares; NEVADO, Rosane Aragon de; MENEZES, Crediné Silva de. Arquiteturas pedagógicas para educação a distância. In: LENZ, Ricardo (org.). **Aprendizagem em rede na educação a distância: estudos e recursos para formação de professores**. Porto Alegre: Ricardo Lenz, v. 1, p. 36–52, 2007.

FREIRE, Paulo; FAUNDEZ, Antonio. Por uma pedagogia da pergunta**.** Rio de Janeiro: **Paz e Terra**, 1985.

FUNKE, A.; GELDREICH, K. Collaborative problem-solving and pair programming in introductory programming courses: impacts on student motivation and performance. **Computer Science Education**, v. 30, n. 3, p. 287–310, 2020.

GREGOR, Shirley; HEVNER, Alan R. Positioning and presenting design science research for maximum impact. **MIS Quarterly**, v. 37, n. 2, p. 337–355, 2013.

HEVNER, Alan R. **A three cycle view of design science research. Scandinavian Journal of Information Systems**, v. 19, n. 2, p. 87–92, 2007.

HEVNER, Alan R. et al. Design science in information systems research**. MIS Quarterly**, v. 28, n. 1, p. 75–105, 2004.

JONASSEN, David H. **Toward a design theory of problem solving. Educational Technology Research and Development**, v. 48, n. 4, p. 63–85, 2000.

LAGO, Celina B.; CORRÊA, Roberta S.; PRADO, Márcia E. B. B. A resolução de problemas como estratégia pedagógica na formação docente. **Revista Diálogo Educacional**, v. 19, n. 63, p. 887–906, 2019.

LYE, S. Y.; KOH, J. H. L. Review on teaching and learning of computational thinking through programming: what is next for K-12? **Computers in Human Behavior**, v. 41, p. 51–61, 2014.

MAYER, Richard E. **Cognitive, metacognitive, and motivational aspects of problem solving. Instructional Science**, v. 26, n. 1–2, p. 49–63, 1998.

MENEZES, Crediné Silva de; ARAGÓN, Rosane; ZIEDE, Mariangela Kraemer Lenz. Arquiteturas pedagógicas para a aprendizagem em rede no contexto do seminário integrador. **RENOTE: Revista Novas Tecnologias na Educação**, v. 11, n. 2, jul. 2013.

PAPERT, Seymour. Mindstorms: children, computers, and powerful ideas. New York: **Basic Books**, 1980.

PERALES, Francisco J. Resolución de problemas y enseñanza de las ciencias. **Enseñanza de las Ciencias**, v. 11, n. 3, p. 265–276, 1993.

PÉREZ, César C. et al**. La resolución de problemas en la enseñanza de las ciencias. Enseñanza de las Ciencias**, v. 6, n. 3, p. 227–233, 1988.

PIAGET, Jean. Genetic epistemology. New York: **Columbia University Press**, 1970.

PIAGET, Jean. Para onde vai a educação? 13. ed. Rio de Janeiro: **Livros Técnicos e Científicos**, 1972.

PIAGET, Jean. A tomada de consciência. **São Paulo: Melhoramentos**, 1977.

PIAGET, Jean. Seis estudos de psicologia. 20. ed. Rio de Janeiro: **Forense Universitária**, 1998.

POLYA, George. **How to solve it: a new aspect of mathematical method**. 2. ed. Princeton: Princeton University Press, 1977.

POZO, Juan I.; ECHEVERRÍA, María P. Aprendizagem e instrução: a teoria e a prática do ensino. **Porto Alegre: Artmed**, 1998.

RANALI, José; LOMBARDO, Luiz. Projeto pedagógico e currículo: repensando o ensino de graduação. **São Paulo: Cortez**, 2006.

SBC – SOCIEDADE BRASILEIRA DE COMPUTAÇÃO. Referenciais de formação para os cursos de graduação em computação. Porto Alegre: SBC, 2017.

SOLOWAY, Elliot; SPOHRER, James C. Studying the novice programmer. Hillsdale: **Lawrence Erlbaum Associates**, 1989.

VAISHNAVI, Vijay K.; KUECHLER, William. Design science research methods and patterns: innovating information and communication technology. **Boca Raton: CRC Press**, 2007.

WING, Jeannette M. Computational thinking. **Communications of the ACM**, v. 49, n. 3, p. 33–35, 2006.