UNIVERSIDADE ESTADUAL DO NORTE DO PARANÁ

CAMPUS LUIZ MENEGHEL - CENTRO DE CIÊNCIAS TECNOLÓGICAS

BACHARELADO EM SISTEMAS DE INFORMAÇÃO

LICENCIATURA EM COMPUTAÇÃO

WESLLEY TONETO DE OLIVEIRA

DIRETRIZES PARA O DESENVOLVIMENTO DE SIMULADORES DE APOIO AO ENSINO DE PROGRAMAÇÃO

Bandeirantes

2019

WESLLEY TONETO DE OLIVEIRA

DIRETRIZES PARA O DESENVOLVIMENTO DE SIMULADORES DE APOIO AO ENSINO DE PROGRAMAÇÃO

Trabalho Final do curso de Bacharelado em Sistemas de Informação e Licenciatura em Computação da Universidade Estadual do Norte do Paraná.

Orientadora: Profa. Dra. Daniela de Freitas Guilhermino Trindade

Bandeirantes
2019

WESLLEY TONETO DE OLIVEIRA

DIRETRIZES PARA O DESENVOLVIMENTO DE SIMULADORES DE APOIO AO ENSINO DE PROGRAMAÇÃO

Trabalho de Conclusão de Curso submetido à Universidade Estadual do Norte do Paraná, como requisito parcial para obtenção do grau de Bacharel em Sistemas de Informação e Licenciado em Computação.

COMISSÃO EXAMINADORA

_____
Profa. Dra. Daniela de Freitas Guilhermino Trindade
UENP – *Campus* Luiz Meneghel

_____
Prof. Me. Fabio de Sordi Junior
UENP – *Campus* Luiz Meneghel

_____
Prof. Me. José Reinaldo Merlin
UENP – *Campus* Luiz Meneghel

Bandeirantes, 10 de junho de 2019

AGRADECIMENTOS

Agradeço primeiramente a Deus que permitiu que tudo isso acontecesse ao longo de minha vida e não somente nestes anos como universitário.

À Instituição pelo ambiente criativo e amigável que proporciona.

A Universidade Estadual do Norte do Paraná – CAMPUS LUIZ MENEGHEL pela oportunidade de realizar o curso, adquirindo o grau de Licenciado em Computação e Bacharel em Sistemas de Informação.

Aos meus pais Lúcia e Wellington e ao meu irmão Weldrey que sempre me apoiaram nas minhas decisões e estiveram comigo em todos os momentos.

A minha orientadora, Professora Dra. Daniela de Freitas Guilhermino Trindade, pela paciência na orientação, pela oportunidade e apoio que tornaram possível a conclusão deste trabalho.

Aos membros da banca examinadora, Professor Me. Fábio de Sordi Junior e Professor Me. José Reinaldo Merlin pelas sugestões e orientações.

A todos os professores que participaram da minha formação acadêmica durante esses anos de curso, por me proporcionarem conhecimento e conselhos.

A todos os meus amigos que me acompanharam durante essa jornada, em especial Vitória, Rafael, Giovanni, Osmar, Igor, Vitor, Mayke, Gian e Luiz Henrique que foram meus companheiros de estudos e irmãos, amigos que fizeram parte da minha formação e que vão continuar presentes em minha vida.

RESUMO

Levando em consideração os altos níveis de reprova e abandono dos cursos de tecnologia por conta das disciplinas correlatas à disciplina de programação de computadores, as dificuldades e problemáticas envolvendo esse tema tem se tornado alvo de muitos estudos e discussões que resultam em propostas de metodologias e ferramentas que auxiliam na compreensão desse conteúdo. Normalmente, a disciplina de programação é realizada por meio da exposição do conteúdo pelo professor, que repassa o conhecimento, de forma teórica, com exemplos básicos e proposição de exercícios, sem levar em conta as necessidades dos alunos e sem considerar suas aptidões e dificuldades. Pensando nestas dificuldades, o uso de simuladores tem se apresentado como uma solução e tem demonstrado resultados satisfatórios na aprendizagem de programação, principalmente, por auxiliar no processo de abstração que esses conteúdos requerem. Neste sentido, o objetivo deste trabalho é propor diretrizes para o desenvolvimento de simuladores que auxiliem os processos de ensino e aprendizagem na área de programação e disciplinas correlatas. Pretende-se, desta forma, contribuir para a criação e disseminação de simuladores que apoiem o desenvolvimento cognitivo e a aquisição de competências pelos educandos relativos à programação.

Palavras-chave: Diretrizes. Aprendizagem. Programação. Simuladores.

ABSTRACT

Due to the high levels of failure and abandonment of technology courses due to the disciplines related to the discipline of computer programming, the difficulties and problems that surround it have become the subject of many studies and discussions that result in many proposals and methodological tools to help in understanding this content. Normally, in the programming discipline, the teacher passes the knowledge theoretically with basic examples and propositions of exercises, without taking into account the needs of the students and without considering their aptitudes and difficulties. Considering these difficulties, the use of simulators has presented itself as a solution and has demonstrated satisfactory results in programming learning, mainly because it helps in the abstraction process that these contents require. In this sense, the objective of this work is to propose guidelines for the development of simulators that support the teaching and learning processes in the programming area and related disciplines. The intention is to contribute to the creation and dissemination of simulators that support the cognitive development and the acquisition of competences by the students related to programming.

Keywords: Guidelines. Learning. Programming. Simulators.

LISTA DE QUADROS

Quadro 1 - Área Cognitiva (TELLES, 2004, apud BRANDÃO E MARQUES, 2006) 20
Quadro 2 - Fases de aprendizagem segundo Van Hiele.....24
Quadro 3 - Fases das Taxonomias de Bloom e Marzano.....33
Quadro 4 - Conteúdos de programação com um maior grau de Dificuldade..... 35
Quadro 5 - Diretrizes.....37

LISTA DE FIGURAS

Figura 1 - Dificuldades em relação aos conceitos e práticas de programação.....30
Figura 2 - Soluções e melhorias no processo de ensino e aprendizagem de programação…31
Figura 3 - Conteúdos disponíveis no simulador VISUALGO.....32
Figura 4 - Exemplo, Definição dos conceitos e Abordagem Facilitadora.....42
Figura 5 - Exemplo, Interface amigável.....44
Figura 6 - Exemplo, Comece as simulações com um resumo.....45
Figura 7 - Exemplo, Fornecer *Feedback*.....46

LISTA DE SIGLAS

CBIE	Congresso Brasileiro de Informática na Educação
INACSL	*International Nursing Association for Clinical Simulation and Learning*
ITiCSE	*Innovation and Technology in Computer Science Education*
SBIE	Simpósio Brasileiro de Informática na Educação
TISE	Conferência Internacional sobre Informática na Educação
INEP	Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira

SUMÁRIO

1. INTRODUÇÃO 12

1.1. Objetivo geral 14

1.2. Objetivos específicos 14

1.3. Justificativa 14

2. METODOLOGIA 16

3. FUNDAMENTAÇÃO TEÓRICA 17

3.1 Dificuldades no ensino das disciplinas relacionadas à programação 17

3.2 Taxonomias de ensino-aprendizagem 18

3.2.1 Taxonomia de bloom 19

3.2.2 Taxonomia de marzano 21

3.2.3 Taxonomia de van hiele 22

3.3 Simulação 25

3.4 Trabalhos relacionados 28

3.4.1 Inacsl *standards of best practice: simulation*^{sm} *simulation design*. 28

3.4.2 Um estudo sobre os problemas e dificuldades relacionados ao ensino e aprendizagem de programação 30

3.4.3 Visualgo.net 31

4. DIRETRIZES PARA O DESENVOLVIMENTO DE SIMULADORES DE APOIO AO ENSINO DE PROGRAMAÇÃO 33

4.1 Relacionamento das taxonomias de bloom, marzano e van hiele 33

4.2 As taxonomias de ensino-aprendizagem como forma de minimizar as dificuldades no ensino de programação 34

4.3 Composição das diretrizes para o desenvolvimento de simuladores de apoio ao ensino de programação 36

4.4 Exemplo de aplicação das diretrizes 41

4.4.1 Realize uma avalição das necessidades dentro da disciplina de programação, para fornecer uma experiência baseada em simulação bem projetada. 41

4.4.2 Construa objetivos mensuráveis de aprendizagem. 41

4.4.3 Defina os conceitos e uma abordagem facilitadora. 42

4.4.4 Crie um cenário ou caso que forneça o contexto necessário para a experiência baseada em simulação, sempre promovendo a criatividade. 43

4.4.5 Projete uma interface gráfica amigável. 43

4.4.6 Comece as simulações com um resumo. 44

4.4.7 Forneça *feedback* 45

4.4.8 Forneça os recursos necessários para que os participantes possam atingir seus objetivos, aumentando seu conhecimento e encontrando resultados diferentes das experiências baseada em simulação. 46

4.4.9 Avalie os resultados obtidos. 47

5. CONSIDERAÇÕES FINAIS 48

REFERÊNCIAS 49

1. INTRODUÇÃO

O desenvolvimento da computação e o seu emprego nas atividades cotidianas mudaram a vida não só das empresas, mas de todas as pessoas que as utilizam. Hoed (2016), apresenta em seu trabalho a evasão na computação em relação à média nacional com base nos dados do INEP no ano de 2014, onde em instituições públicas, é de 15,88%, e em instituições privadas, de 21,89%. Um dos grandes problemas enfrentados nesses cursos é justamente a dificuldade com as disciplinas ligadas à programação, e as disciplinas similares, como estrutura de dados e algoritmos, em que grande parte dos estudantes apresenta dificuldade em entender os conceitos iniciais, aumentando, cada vez mais, o índice de evasão e reprovação nas disciplinas desta área (PRIETCH e PAZETO, 2010).

Enfatizando a dificuldade em compreender o nível de abstração que requer a programação, Borges (2000) ressalta a importância de ambientes e *software* que auxiliem no aprendizado, que são um recurso a mais para auxiliar na prática e progredir no desenvolvimento exercitando a lógica.

Para Raabe e Silva (2005), são várias as origens das dificuldades enfrentadas pelos alunos durante o processo de ensino aprendizagem de programação, como: a exigência lógico-matemático preeminente na disciplina, a dificuldade de apreensão, por parte do professor ou até mesmo o ritmo de aprendizagem de cada aluno. Além disso, Robins (2010) destaca que fatores como capacidade cognitiva, estilos cognitivos, motivação e atitude contribuem para este cenário.

Segundo Pereira e Rapkiewicz (2004), com um número consideravelmente grande de desistências e reprovações em programação, é necessário viabilizar métodos, recursos e ferramentas diferenciadas para alcançar a aprendizagem e atrair mais pessoas para a área.

Sabendo das dificuldades de abstração desses conteúdos por parte de alguns alunos, a simulação é considerada, por muitos, a solução dos vários problemas que os professores enfrentam ao tentar explicar para seus alunos fenômenos demasiadamente abstratos para serem “visualizados” através de uma descrição em palavras, e ao mesmo

tempo complicados demais para serem representados através de uma única figura (HECKLER *et al*. 2007).

Segundo Dourado e Giannella (2014), aproveitar o erro como oportunidade de aprendizado além de favorecer a visualização, manipulação e interpretação de situações complexas é uma das potencialidades do simulador. Ao vivenciarem os cenários simulados, os alunos podem reforçar e corrigir conhecimentos/habilidades previamente aprendidos e, ao mesmo tempo, projetar novas situações (DOURADO E GIANNELLA, 2014).

Com isso, nesse trabalho será proposto diretrizes para o desenvolvimento de simuladores que auxiliem o professor no processo de ensino e aprendizagem das disciplinas de programação e correlatas.

O restante do trabalho está estruturado da seguinte forma: na subseção 1.1 e 1.2 serão apresentados os objetivos gerais e específicos desta pesquisa; na subseção 1.3 apresenta-se a justificativa pela qual se dá a importância do desenvolvimento do trabalho; na 2 seção é apresentada a metodologia e técnicas utilizadas; em seguida na seção 3 está apresentado todo o embasamento teórico para o desenvolvimento do trabalho; na seção 4 será apresentado o desenvolvimento das diretrizes para o desenvolvimento de simuladores de apoio ao ensino de programação; e por último na seção 5 se encontra as considerações finais deste trabalho.

1.1 OBJETIVO GERAL

O objetivo geral deste trabalho é a proposição de diretrizes para o desenvolvimento de simuladores de apoio ao ensino e aprendizagem na área de programação e disciplinas correlatas.

1.2 OBJETIVOS ESPECÍFICOS

Para alcançar o objetivo geral são estabelecidos os seguintes objetivos específicos:

- Elencar as dificuldades que afetam compreensão de conteúdos ligados à programação;
- Elencar e analisar as taxonomias de aprendizagem;
- Estudar as metodologias de apoio ao ensino da programação e as metodologias de apoio ao desenvolvimento de simuladores nas diversas áreas de conhecimento;
- Elencar as características e requisitos, a partir dos trabalhos selecionados, para o desenvolvimento de simuladores;
- Elaborar as diretrizes de apoio ao desenvolvimento de simuladores para o ensino da programação.

1.3 JUSTIFICATIVA

A tecnologia está presente em todos os setores de atividades, e vem adquirindo cada vez mais relevância na vida das pessoas, modificando a forma como interagem e como realizam as tarefas. Face a este cenário, a procura pelos cursos na área de tecnologia tem crescido consideravelmente, porém, a disciplina de programação, de extrema importância para a criação de sistemas computacionais, é uma das disciplinas com maior grau de dificuldade e também, com maior nível de reprova nos cursos da área de tecnologia.

Normalmente, a disciplina de programação é realizada por meio da exposição do conteúdo pelo professor, que detém o conhecimento e o repassa, baseando-se na apresentação de teoria, de exemplos básicos e proposição de exercícios, sem levar em conta as necessidades dos alunos e sem considerar suas aptidões e dificuldades

(GOMES *et al.* 2008). Percebe-se que a dificuldade em compreender o nível de abstração que requer a programação é um fator complicador das disciplinas de programação, o que acaba causando grande evasão nos cursos da área de computação.

Segundo Mercado (2002), a informática quando aplicada ao ensino, neste caso o simulador, pode contribuir para auxiliar os professores em sua tarefa de transmitir o conhecimento e obter uma nova maneira de ensinar cada vez mais de forma criativa e dinâmica, auxiliando novas descobertas.

Segundo Santos (2015), a proposta de uso de ferramentas de simulação como suporte ao ensino e aprendizagem, dispensa o investimento em grandes laboratórios tecnologicamente equipados, proporcionando uma enorme redução de custos e inovações significativas na maneira de ensinar.

A ideia de utilizar um simulador para o ensino de disciplinas com um grau superior de abstração é de grande relevância, pois segundo Santos, & Costa, (2006) só será possível o acompanhamento contínuo e personalizado de cada aluno desta disciplina, através do apoio de sistemas inteligentes, sendo auxiliados pelo computador.

A simulação é baseada na Teoria Humanista de Carl Rogers, que tem como premissa o conhecimento sendo integrado com a vida, onde o aluno deve ter a capacidade de auto conduzir o seu próprio processo de formação e criar experiências que gerem reflexão (JUNIOR DIAS e MERCADO, 2016).

Assim, tendo em vista que o uso de simuladores no ensino e aprendizagem tem sido objeto de pesquisa de vários pesquisadores como alternativa de melhoria neste processo, propõe-se neste trabalho analisar como os simuladores podem auxiliar na compreensão de conteúdos voltados à computação e apresentar diretrizes para a construção de simuladores na área de programação.

2. METODOLOGIA

A pesquisa classifica-se como um estudo exploratório, pois tem como finalidade proporcionar um maior entendimento sobre como os simuladores auxiliam no ensino e aprendizagem de conteúdos de difícil abstração como os das disciplinas de programação e similares, além de entender como são desenvolvidos para que seja possível propor diretrizes de boas práticas para seu desenvolvimento (GIL, 2002).

Para o desenvolvimento deste trabalho foram necessários os seguintes passos metodológicos:

a) Fundamentação teórica com a abordagem dos temas: Dificuldades no Ensino de programação; taxonomias de aprendizagem; metodologias para o ensino da computação e simuladores.
b) Levantamento e diagnóstico de metodologias de apoio à aprendizagem e ao desenvolvimento de simuladores nas diversas áreas de conhecimento;
c) Diagnóstico das características dos simuladores e extração dos requisitos utilizados no desenvolvimento de simuladores;
d) Elaboração das diretrizes de apoio ao desenvolvimento de simuladores para o ensino da programação.

3. FUNDAMENTAÇÃO TEÓRICA

Nesse capitulo é apresentado o embasamento teórico necessário para o entendimento e desenvolvimento do trabalho. O foco são estudos que abordam o uso de simuladores na educação, principalmente nas disciplinas relacionadas ao ensino e aprendizagem de programação. Para isso serão abordados temas como: as dificuldades encontradas no ensino-aprendizagem da programação, as taxonomias de ensino-aprendizagem e simulação.

3.1 DIFICULDADES NO ENSINO DAS DISCIPLINAS RELACIONADAS À PROGRAMAÇÃO

Pears *et al*. (2007) destacam que a programação de computadores é uma disciplina tradicionalmente difícil para estudantes iniciantes e que algumas pesquisas nesta área têm sido desenvolvidas, a fim de superar estas dificuldades.

A programação de computadores é o processo de transformar uma solução abstrata de alto nível em um conjunto de instruções sintaticamente precisas, expressas em uma linguagem formal e avaliar sua execução em um dispositivo de computação (ROBINS *et al*. 2003).

Segundo Gomes *et al*. (2008), para muitos estudantes, as dificuldades começam em uma fase inicial da aprendizagem, quando se precisa compreender e aplicar certos conceitos abstratos de programação, como as estruturas de controle, para criar algoritmos que resolvam problemas concretos.

Desta forma, quando um conceito não é totalmente compreendido pelos estudantes, eles não conseguem entender os próximos conteúdos abordados na disciplina, pois a disciplina de programação possui um conteúdo cumulativo, em que um novo conhecimento necessita de um já existente. Além da dificuldade de assimilação dos conteúdos, é preciso que os estudantes consigam abstrair os conceitos de programação, conseguindo visualizar o que precisa ser feito antes e durante a construção do código fonte, e isso se torna difícil para a maioria deles (VIEGAS, *et al.* 2015).

Esses assuntos são de complexo entendimento para os estudantes da área de tecnologia, primeiramente por se tratar de algo nunca visto por eles, além de faltar a

prática de leitura, fator que ocasiona dificuldades de interpretação dos problemas a serem solucionados e ainda por faltar uma base matemática sólida, ambas oriundas da formação básica (MARTINS, 2015).

Relacionado a isso, vemos que os métodos tradicionalmente usados para ensino dessas disciplinas não são adequados às necessidades da maioria dos alunos por diversas razões. De acordo com Gomes *et al*. (2008), as restrições temporais levam a que seja muito difícil fornecer, em sala de aula, um *feedback* e supervisão adequados e personalizados às necessidades de cada aluno. Ainda segundo Gomes *et al*. (2008), os indivíduos aprendem de diversas formas e, de acordo com os métodos tradicionais todos os alunos são forçados a uma aprendizagem uniforme, devendo aprender ao mesmo ritmo e de acordo com as estratégias pedagógicas do professor. Porém, é importante que o professor consiga contemplar a enorme diversidade de estilos de aprendizagem presentes em sala de aula.

Diante dessas dificuldades, diversas técnicas têm sido desenvolvidas com o intuito de mitigar os problemas nos quais os alunos se deparam na aprendizagem de programação. Nesse contexto, Gomes *et al*. (2008), afirma que alguns indivíduos recorrem a sistemas de animação com o propósito de recorrer ao potencial do sistema visual humano, visando que o formato gráfico animado contribua para uma melhor compreensão de conceitos inerentemente dinâmicos, quando comparado com o formato textual.

3.2 TAXONOMIAS DE ENSINO-APRENDIZAGEM

Taxonomia (do grego taxis, que é ordenação, e nomos, que é sistema, norma) é todo sistema de classificação que possui três características: cumulatividade, hierarquia e eixo comum (KRATHWOHL, 2001).

Em termos gerais a hierarquia é uma ordenação de elementos com valores definidos, e neste caso diz respeito aos níveis de conhecimento, que estão organizados de maneira hierárquica, do nível mais inferior, que representa objetivos educacionais mais básicos, ao mais superior em que o indivíduo já é capaz de compreender e organizar conhecimentos mais complexos (KRATHWOHL, 2001).

A cumulatividade está associada à característica que as taxonomias descrevem de forma que o indivíduo ao passar por um nível “acumula” os conhecimentos obtidos no nível anterior. As taxonomias de ensino-aprendizagem possuem um eixo comum de raciocínio para aquisição do conhecimento, os níveis estão organizados de forma a sempre manter os objetivos e capacidades cognitivas relacionadas (KRATHWOHL, 2001).

Assim, pode-se considerar a partir das afirmações de Ferraz e Belhot (2010) que as taxonomias são instrumentos que apoiam a análise do desenvolvimento cognitivo, englobando a aquisição do conhecimento, competência e atitudes, visando facilitar o planejamento do processo de ensino e aprendizagem.

Pensando nas dificuldades para o ensino de disciplinas relacionadas à programação e no processo de aplicação de objetos de aprendizagem, como os simuladores, foram analisadas algumas taxonomias que são de grande importância no processo de ensino-aprendizagem, são elas, a taxonomia de Bloom, a de Marzano e a de Van Hiele.

3.2.1 TAXONOMIA DE BLOOM

Benjamin Bloom liderou um grupo formado pela *American Psychological Association* para criar uma "classificação de objetivos de processos educacionais" (TELLES, 2004, apud BRANDÃO E MARQUES, 2006).

O primeiro passo para a definição dessa taxonomia foi a divisão do campo de trabalho em 3 áreas não mutuamente exclusivas: a cognitiva, ligada ao saber, a afetiva, ligada a sentimentos e posturas; a psicomotora, ligadas a ações físicas.

Nessa taxonomia são definidos seis níveis de desenvolvimento cognitivo para a aquisição de comportamentos e de competências por parte do educando, como mostra no Quadro 1.

ÁREA COGNITIVA		
NÍVEIS	**DEFINIÇÕES**	**AMOSTRA DE VERBOS**
Conhecimento	O aluno irá recordar ou reconhecer informações, ideias e princípios na forma em que foram aprendidos.	Escreva, Liste, Rotule, Nomeie, Diga e Defina.
Compreensão	O aluno compreende ou interpreta informação com base em conhecimento prévio ou novo.	Explique, Resuma, Parafraseie, Descreva.
Aplicação	O aluno seleciona, transfere e usa dados e princípios para completar um problema ou tarefa com um mínimo de supervisão.	Use, Compute, Resolva, Demonstre, Aplique, Construa.
Análise	O aluno distingue, classifica e relaciona pressupostos, hipóteses, evidências ou estruturas de uma declaração ou questão.	Analise, Categorize, Compare, Contraste e Separe.
Síntese	O aluno cria, integra e combinam ideias em um produto, plano ou proposta, novos para ele.	Crie, Planeje, Elabore hipótese (s), Invente e Desenvolva.
Avaliação	O aluno aprecia, avalia ou critica com base em padrões e critérios específicos.	Julgue, Recomende, Critique e Justifique.

Quadro 1 - Divisões dos níveis da taxonomia de Bloom
(Extraído de: ARAÚJO de O. S. L. A. *et al.* 2013)

Além do domínio cognitivo, os resultados de aprendizagem podem ser observados no domínio afetivo (BRANDÃO E MARQUES, 2006). A taxonomia dispõe os objetivos educacionais afetivos num contínuo hierárquico de cinco níveis, descritos como processos de internalização:

- Primeiro nível (mais baixo): chamado de recepção ou de atenção, o indivíduo tem consciência do fenômeno e é capaz de percebê-lo;
- Segundo nível: chamado de emissão de respostas, o indivíduo responde ao fenômeno com sentimentos implicando numa atenção ativa, como por exemplo, de interesse voluntário (BRANDÃO E MARQUES, 2006);
- Terceiro nível: de valoração, ele é capaz de se empenhar em responder ao fenômeno com sentimentos a partir da percepção de valor atribuído;
- Quarto nível: de organização, ele conceitua seu comportamento e seus sentimentos e os organiza em uma estrutura para a determinação das inter-relações entre valores e formação de atitude; e
- Quinto nível (o mais alto da hierarquia): de caracterização, esta estrutura de valores, crenças, ideias e atitudes se tornam parte de sua vida (BLOOM et al. 1983 apud BRANDÃO E MARQUES, 2006).

Os objetivos afetivos, em geral, não são medidos para fins de classificação, tendo sua utilidade na análise de resultados de aprendizagem.

O domínio psicomotor tem sua aplicação em medida e avaliação de atividade física ou de aptidão física. Os processos caracterizados pelas taxonomias representam resultados de aprendizagem, assim, cada categoria taxonômica representa o que o indivíduo aprende e não aquilo que ele já sabe, assimilado do seu contexto familiar ou cultural (BLOOM, 1956 apud BRANDÃO E MARQUES, 2006).

3.2.2 TAXONOMIA DE MARZANO

A nova taxonomia dos objetivos educacionais proposta por Robert Marzano e John Kendall baseia-se na proposta apresentada por Benjamin Bloom em 1956 (CÓRDOVA, 2009). A Taxonomia de Marzano melhora em alguns pontos a proposta apresentada por Bloom há mais de cinco décadas, na Bloom oferece uma estrutura que descreve seis níveis de processamento de informações (CÓRDOVA, 2009).

Segundo Granados (2015), a nova taxonomia de Marzano é composta por três sistemas: o sistema automático, o sistema metacognitivo e o sistema cognitivo além da área do conhecimento, e todos eles são importantes para pensar e aprender. Ainda segundo Granados (2015), o sistema automático decide se continua com o comportamento atual ou se executa uma nova atividade; o sistema metacognitivo estabelece metas e fica ciente de como elas serão alcançadas; o sistema cognitivo processa toda a informação necessária; e a área de conhecimento fornece o conteúdo necessário.

A nova Taxonomia de Marzano modifica alguns níveis cognitivos em relação a taxonomia de Bloom, muda de posição e adiciona o nível de metacognição que é orientado por hábitos, auto pensamentos críticos e criativos, os quais permitem a autoaprendizagem da pessoa, e, modifica o último nível de auto regulação (GRANADOS, 2015).

Ainda segundo Granados (2015), as mudanças classificaram os seguintes níveis:

- **Conhecimento:** Nomear; Executar;
- **Compreensão:** Síntese; Representação;
- **Análise:** Relação; Classificação; Análise de erros; Generalizações; Especificações;
- **Utilização:** Tomada de decisões; Resolução de problemas; Investigação experimental; Investigação;
- **Sistema de Metacognição:** Especificação das metas; Monitoramento de processos; Monitoramento da clareza; Monitoramento da precisão;
- **Sistemas de consciência do ser:** Avaliação da importância; Avaliação da eficácia; Avaliação de emoções; Avaliação da motivação;

3.2.3 TAXONOMIA DE VAN HIELE

Sustentado pelos resultados obtidos nos estudos em psicologia genética de Piaget, o professor holandês Van Hiele propõe um modelo para a aprendizagem da geometria em acordo com as ideias sobre o desenvolvimento da inteligência de Piaget. Van-Hiele parte de duas premissas básicas:

- O objetivo do ensino da geometria é de levar o aluno à aquisição de uma rede de relações servindo à expressão de raciocínios, rede na qual as relações são ligadas de forma lógica e dedutiva (SANTOS, 2002);
- Essa rede de relações deve ser construída pelo próprio aluno, recusando a ideia de receber do professor uma rede relacional completamente pronta (SANTOS, 2002).

Van Hiele propõe que “a aprendizagem é um processo recursivo que progride recursivamente por meio de níveis de pensamento descontínuos – saltos na curva de aprendizagem” (VAN HIELE; GELDOF, 1958, apud SANTOS, 2002), que pode ser melhorado por um procedimento didático adequado. Ele pressupõe que há diversos níveis de aprendizagem da Geometria e que a passagem de um nível para o próximo deve ocorrer por meio de uma sequência de fases de ensino.

Segue-se abaixo uma caracterização dos níveis de Van Hiele bem como suas propriedades segundo Pereira *et al*. 2005):

- **Nível 0 - Visualização:** os alunos veem o espaço apenas como algo que existe em torno deles. Reconhecem as figuras geométricas apenas pela sua forma (aparência física), não conseguindo identificar suas partes ou propriedades. São capazes de reproduzir figuras dadas e aprender um vocabulário geométrico básico.
- **Nível 1 - Análise**: é quando começa a análise dos conceitos geométricos. O aluno começa a discernir as características e propriedades das figuras, mas não consegue ainda estabelecer relações entre essas propriedades e nem entende as definições ou vê inter-relações entre figuras.
- **Nível 2 - Dedução Informal:** Neste nível o aluno começa a estabelecer inter-relações de propriedades dentro de figuras e entre figuras, deduzindo propriedades e reconhecendo classes de figuras. Agora, a definição já tem significado, todavia o aluno ainda não entende o significado da dedução como um todo ou o papel dos axiomas nas provas formais.
- **Nível 3 - Dedução:** Neste estágio o aluno analisa e compreende o processo dedutivo e as demonstrações com o processo axiomático associado, agora, ele já consegue construir demonstrações e desenvolvê-las de mais de uma maneira, também faz distinções entre uma afirmação e sua recíproca.
- **Nível 4 - Rigor**: Agora o aluno já é capaz de trabalhar em diferentes sistemas axiomáticos; analisa e compreende geometrias não euclidianas. A geometria é entendida sob um ponto de vista abstrato.

Além dos níveis apresentados, essa taxonomia possui algumas características importantes, apresentadas a seguir:

1. **Sequencial:** O aluno deve necessariamente passar por todos os níveis, uma vez que não é possível atingir um nível posterior sem dominar os anteriores.
2. **Avanço:** A progressão ou não de um nível para outro depende mais dos métodos de ensino e do conteúdo do que da idade ou maturação biológica. Nenhum método de ensino permite ao aluno pular um nível, alguns acentuam o progresso, mas há alguns que retardam (PEREIRA *et al.* 2005).

3. **Intrínseco e Extrínseco:** Os objetivos implícitos num nível tornam-se explícitos no nível seguinte.
4. **Linguística:** Cada nível tem sua própria linguagem e um conjunto de relações interligando-os. Assim, uma relação que é "correta" em certo nível, pode se modificar em outro nível.
5. **Combinação inadequada:** O professor e o aluno precisam estar raciocinando em um mesmo nível, caso contrário, o aprendizado não ocorre. Ou seja, professor, material didático, conteúdo e vocabulário devem estar compatíveis com o nível do aluno (PEREIRA *et al*. 2005).

Van Hiele propõe que "a transição de um nível para o seguinte não é um processo natural, ela acontece sob a influência de um programa de ensino-aprendizagem" (VAN HIELE, 1986, apud PEREIRA, 2005). Este programa de ensino-aprendizagem inclui uma sequência didática de cinco fases de aprendizado. O Quadro 2 resume as fases de aprendizagem de Van Hiele.

Fases de Aprendizagem	Características
Informação (Fase 1)	- Professor e aluno dialogam sobre o material de estudo; - O professor deve perceber quais os conhecimentos anteriores do aluno sobre o assunto a ser estudado.
Orientação Direta (Fase 2)	- Os alunos exploram o assunto de estudo por meio do material selecionado pelo professor; - As atividades deverão proporcionar respostas específicas e objetivas.
Explicação (Fase 3)	- O papel do professor é o de observador
Orientação Livre (Fase 4)	- Tarefas constituídas de várias etapas, possibilitando diversas respostas, a fim de que o aluno ganhe experiência e autonomia.
Integração (Fase 5)	- O professor auxilia no processo de síntese, fornecendo experiências e -Observações globais, sem apresentar novas e discordantes ideias.

Quadro 2 - Fases de aprendizagem segundo Van Hiele
(Extraído de: Klaus; Pazos, 2005).

Na fase de Informação o professor e aluno conversam e desenvolvem atividades sobre os objetos do estudo do respectivo nível. Aqui se introduz o vocabulário específico do nível, são feitas observações e várias perguntas. É uma fase preparatória para estudos posteriores.

Na fase de Orientação Direta as atividades são desenvolvidas para explorar as características de um nível e isto deve ser feito por meio do uso de material selecionado e preparado pelo professor.

A fase de explicação, considera que o papel do professor é de somente orientar o aluno no uso de uma linguagem precisa e adequada. Baseando-se em experiências anteriores os alunos revelam seus pensamentos e modificam seus pontos de vista sobre as estruturas trabalhadas e observadas.

Quanto à fase de Orientação livre, verifica-se que diante de tarefas mais complexas, os alunos procuram soluções próprias que podem ser concluídas de maneiras diferentes. Assim, eles ganham experiência ao descobrir sua própria maneira de resolver tarefas.

Por fim, na fase de integração, o aluno relê e resume o que foi aprendido, com o objetivo de formar uma visão geral da nova rede de objetos e relações, assim, o aluno alcança um novo nível de pensamento.

3.3 SIMULAÇÃO

Existem diversos modelos e categorias de *softwares* educacionais usados para auxiliar o ensino e aprendizagem de diversas disciplinas. Quanto a classificação pela forma em que o *software* educacional é utilizado eles podem ser de duas formas (FONTES, 2001):

- ***Software* Genérico**: utilizado em qualquer disciplina. Alguns desses são: processadores de texto, folhas de cálculos; e
- ***Software* Específico**: concebido com a finalidade de ser usado no ensino e nomeadamente na aprendizagem de temas concretos. São exemplos: os programas de simulação usados no ensino de temas específicos como ciências, idiomas, exercícios de matemática, dentre outros.

Os *softwares* educativos podem ser classificados de acordo com a maneira que o conhecimento é manipulado e serão apresentadas a seguir:

- **Tutoriais:** São *software* que expõem ao aluno, através do computador, materiais e assuntos já existentes. A inserção desses *software* no processo educacional não causa muito impacto, pois seu uso não necessita de treinamento por ser muito didático, por utilizar mídia, além de ter um controle maior da performance do aluno. Porém, sua elaboração torna-se cara por exigir uma demanda grande de tempo;
- **Tutelado**: *software* que permitem ao aluno ensinar ao computador (resolução de problemas). A utilização das linguagens de programação para "ensinar" o computador a resolver problemas ou resolver ações desejadas permite que se faça uma adaptação ao processo educacional, além de trazer inovações significativas quanto aos aspectos pedagógicos da educação;
- **Exercício e Prática**: São *software* utilizados para revisar conteúdos já trabalhados por professores e alunos. São interativos, quase sempre aparecem na forma de jogos, permitindo a exploração, o exercício e a memorização, pelo aluno, do conteúdo ministrado. Esses *software* conseguem avaliar a assimilação do aluno diante de um determinado assunto, o que não eliminam o processo de avaliação através de outros meios, já que não há como detectar precisamente as deficiências apresentadas por cada um; e
- **Simulação**: São *software* que permitem a criação de modelos e hipóteses, retratando situações reais. Contudo, simulações boas são difíceis de serem desenvolvidas e exigem grande poder computacional, além disso, seu uso não é muito facilitado, pois por si só a simulação não cria o melhor modelo. Importante deixar claro que as simulações devem servir apenas de complemento das aulas, para que o aluno não seja levado a pensar que o mundo real é idêntico a simulação que o retrata.

Estes recursos, além de motivar os alunos, são possibilidades de instituir uma nova forma de aprendizagem, com uma linguagem muito próxima da dos alunos, com possibilidade de retorno imediato sobre a sua produção, além disso, se bem trabalhado, permite que cada aluno avance de acordo com os seus níveis, em ritmo próprio (LEITE *et. al*., 2009).

A utilização de simuladores é de grande utilidade no ensino e aprendizagem de conteúdos relacionados a disciplina de programação, por exemplo, a visualização de operações de fila, lista e pilha, além das estruturas de árvores de decisão que são conteúdos abordados nas disciplinas de estrutura de dados e algoritmos, podem se tornar mais compreensíveis a partir da simulação de suas operações.

Pazin Filho e Scarpelini (2007), definem a simulação como uma, “técnica em que se utiliza um simulador, considerando-se simulador como um objeto ou representação parcial ou total de uma tarefa a ser replicada”.

Os simuladores apresentam vantagens que auxiliam no processo de aprendizagem de estudantes. Pesquisas na área crescem cada vez mais, despertando interesse para propostas de diferentes aplicações pedagógicas (KNIPHOFF DA CRUZ *et al.* 2012).

É possível identificar como os pontos fortes da simulação a realização de experimentos que seriam impossíveis sem o seu uso, e proporcionar a verificação de hipóteses acerca dos fenômenos simulados PAZIN FILHO e SCARPELINI, 2007).

As animações e simulações são consideradas, por muitos, a solução dos vários problemas que os professores enfrentam ao tentar explicar para seus alunos fenômenos demasiadamente abstratos para serem “visualizados” através de uma descrição em palavras, e ao mesmo tempo complicados de mais para serem representados através de uma única figura (HECKLER *et al*. 2007).

Ainda segundo Heckler *et al*. (2007), a simulação possibilita observar em alguns minutos a evolução temporal de um fenômeno que levaria horas, dias ou anos em tempo real, além de permitir que o estudante repita a simulação sempre que desejar.

Simuladores envolvem a criação de modelos dinâmicos e simplificados do mundo real. O potencial educacional deste tipo de ferramenta é muito superior ao dos programas tradicionais. Na área da Ciência da Computação existem simuladores que auxiliam no ensino de várias disciplinas, como redes de computadores, técnicas de programação, arquitetura de computadores e sistemas operacionais (MACHADO e MAIA, 2001).

Segundo Neto e Schuvartz (2007), o uso de recursos computacionais para o ensino aumenta a produtividade e assimilação do conteúdo estudado. Desta forma, os simuladores podem ser vistos como ferramentas didáticas para incentivar e fomentar o

processo de aprendizagem, instigando crianças no contato com uma gama de informações de forma interativa, que permite testar hipóteses, avaliar conclusões e discutir com o grupo (KNIPHOFF DA CRUZ *et al.* 2012).

De acordo com Lévy (1999), o uso de simuladores não garante o desenvolvimento do raciocínio humano, mas contribuem para a capacidade de imaginação e pensamento.

As ferramentas computacionais para este fim de ensino, precisam ser imbuídas de propostas pedagógicas capazes de amenizar, ou até mesmo, eliminar tais problemas como: falta de motivação do aluno, o tradicional processo de avaliação, tornando o aluno preocupado; o relacionamento professor-aluno, e a falta da didática ou metodologia de ensino (JUNIOR DIAS e MERCADO, 2016).

3.4 TRABALHOS RELACIONADOS

Nas subseções a seguir serão apresentados os trabalhos que contribuíram para o desenvolvimento desse trabalho, tais como, as diretrizes e boas práticas de simulação nas disciplinas dos cuidados em saúde e na educação em ciências da saúde, e um estudo sobre os problemas e dificuldades relacionados ao ensino da programação, além do simulador VisuAlgo que apresenta vários aspectos importantes para o desenvolvimento das diretrizes.

3.4.1 INACSL *STANDARDS OF BEST PRACTICE: SIMULATION^{SM} SIMULATION DESIGN*.

Em 2013 a *International Nursing Association for Clinical Simulation and Learning* (INACSL) publicou pela Elsevier, os Padrões para as Melhores Práticas em simulação. Para a Associação, esses padrões refletem as melhores práticas nas disciplinas dos cuidados em saúde e na educação em ciências da saúde (INACSL, 2013). As normas para o desenvolvimento dos padrões de simulação de acordo com a INACSL incluem: fundamentação, resultados, critérios e diretrizes.

Para alcançar os resultados esperados, o design e o desenvolvimento de simulações devem considerar critérios que facilitem a eficácia de experiências baseadas em simulação. As possíveis consequências de não seguir esse padrão podem incluir a

avaliação ineficaz dos participantes e a incapacidade dos participantes de atingir os objetivos identificados ou alcançar os resultados esperados (INACSL - *Standards Committee*, 2016).

Para atender esse padrão a INACSL define como necessários os seguintes critérios:

1. Realize uma avaliação de necessidades para fornecer a evidência fundamental de uma experiência baseada em simulação bem projetada;
2. Construa objetivos mensuráveis;
3. Estruture o formato de uma simulação com base na finalidade, teoria e modalidade para a experiência baseada em simulação;
4. Crie um cenário ou caso para fornecer o contexto para a experiência baseada em simulação;
5. Use vários tipos de fidelidade para criar a percepção necessária de realismo;
6. Mantenha uma abordagem facilitadora que seja centrada no participante e conduzida pelos objetivos, conhecimento do participante ou nível de experiência e os resultados esperados;
7. Comece as experiências baseadas em simulação com um resumo;
8. Siga as experiências baseadas em simulação com uma sessão de *debriefing* e / ou *feedback*;
9. Inclua uma avaliação do (s) participante (s), facilitador (es), a experiência baseada em simulação, a instalação e a equipe de suporte;
10. Forneça materiais e recursos de preparação para promover a capacidade dos participantes de atingir os objetivos identificados e alcançar os resultados esperados da experiência baseada em simulação;
11. Utilize experiências baseadas em simulação de teste piloto antes da implementação completa.

3.4.2 UM ESTUDO SOBRE OS PROBLEMAS E DIFICULDADES RELACIONADOS AO ENSINO E APRENDIZAGEM DE PROGRAMAÇÃO

No trabalho apresentado por Oliveira & Arimoto (2018), foi conduzido um survey aos estudantes e recém-formados nos cursos de Computação, abrangendo diferentes regiões do Brasil. Com o objetivo de investigar e analisar as principais dificuldades enfrentadas pelos estudantes nas disciplinas relacionadas à programação de computadores.

Segundo a pesquisa feita por Oliveira & Arimoto (2018), as principais dificuldades apresentadas em relação aos conceitos e práticas de programação são: Lógica de programação, Definição e uso de variáveis, Definição e uso das estruturas de repetição, Definição e uso de funções e métodos, Arrays, Ponteiros, Recursividade entre outros presentes na Figura 1.

Conceitos e Práticas de Programação	Nível de Dificuldade				
	1	2	3	4	5
Lógica de programação	20%	29%	29%	16%	6%
Definição e uso de variáveis	41%	28%	22%	6%	3%
Definição e uso de estrutura de decisão	34%	26%	22%	13%	5%
Definição e uso de estrutura de repetição	30%	30%	23%	13%	4%
Definição e uso de funções/métodos	23%	23%	25%	19%	10%
Passagem de parâmetro (valor/referência)	20%	16%	25%	22%	17%
Definição e uso de arrays	22%	20%	24%	18%	16%
Manipulação de strings	20%	21%	28%	19%	12%
Definição e uso de ponteiros	11%	14%	19%	22%	34%
Definição e uso de algoritmos recursivos	11%	14%	19%	22%	34%
Definição e uso de estrutura de dados	10%	17%	24%	22%	27%

Figura 1: Dificuldades em relação aos conceitos e práticas de programação.
Fonte: Oliveira & Arimoto (2018)

Além das dificuldades apresentadas pelos estudantes, também foram apresentadas algumas das possíveis soluções para os problemas encontrados. E segundo Oliveira & Arimoto (2018), as possíveis soluções propostas pelos alunos que participaram dessa pesquisa estão presentes na Figura 2.

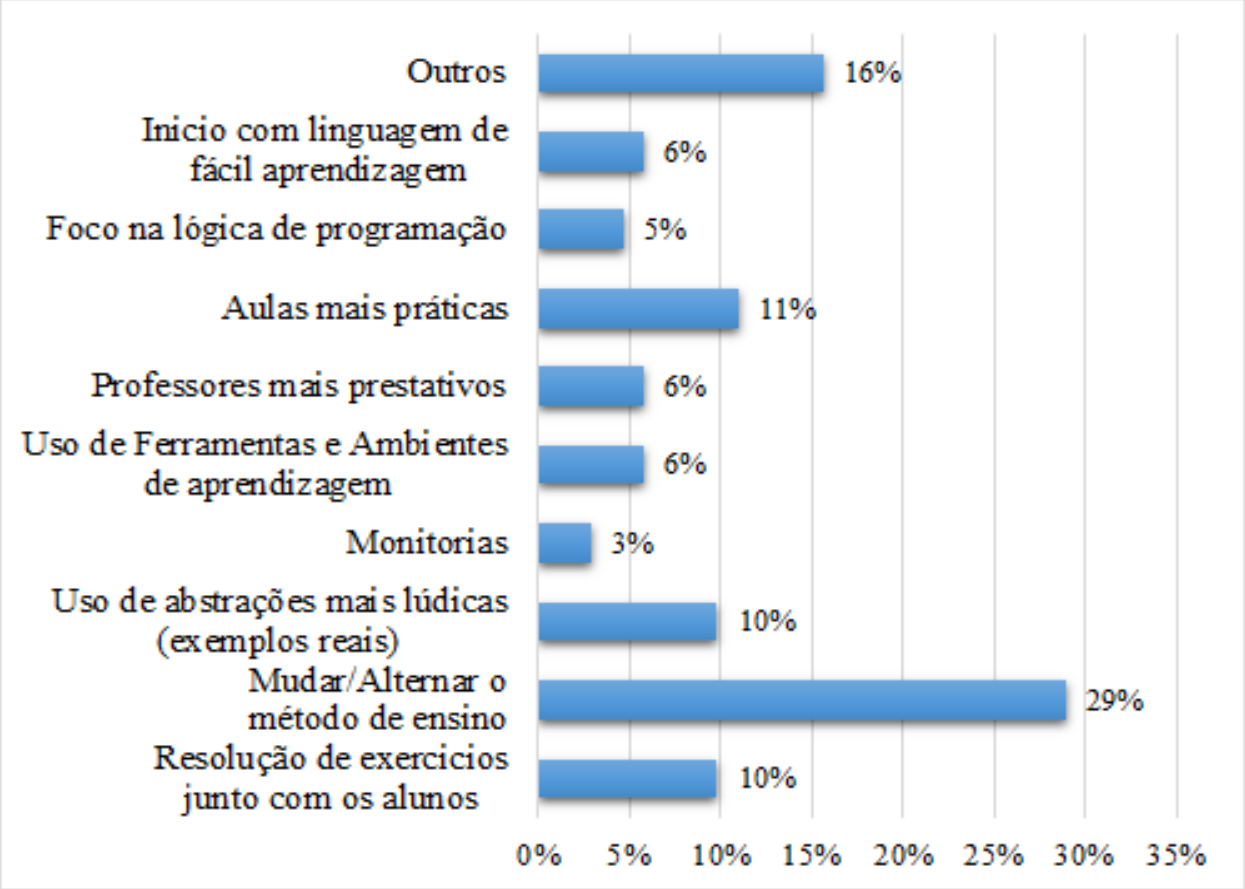

Figura 2: Soluções e melhorias no processo de ensino e aprendizagem de programação.
Fonte: Oliveira & Arimoto (2018)

Como é possível observar na Figura 2, dentre algumas soluções propostas o uso de ferramentas e ambientes de aprendizagem são citados como uma das soluções para os problemas apresentados pelos alunos.

3.4.3 VISUALGO.NET

O VisuAlgo foi conceitualizado em 2011 pelo Dr. Steven Halim como uma ferramenta para auxiliar seus estudantes a entenderem melhor estruturas de dados e algoritmos, permitindo que eles aprendessem o básico por conta e em seu próprio ritmo. VisuAlgo, é um site que fornece simulações animadas dos conteúdos de Estrutura de dados e Algoritmos, tornando mais fácil e intuitiva a maneira de se apresentar conteúdos como, métodos de ordenação, arvores de decisão entre outros.

A figura 3 apresenta alguns conteúdos presentes no simulador VisuAlgo, que possui vários tipos de simulações diferentes, referentes a estruturas de dados e algoritmos.

Figura 3 – Conteúdos disponíveis no simulador VISUALGO
Fonte: visualgo.net

O recurso é simples e fácil de usar, dando ao usuário diversas oportunidades de simulação, onde eles podem escolher qual conteúdo querem visualizar, além de poder alterar as estruturas fornecidas pelo sistema, como criar suas próprias estruturas.

Ao escolher um conteúdo, o sistema apresenta uma definição do que se trata, e explica como e quais ações podem ser executadas naquela simulação. Enquanto a simulação é feita, é possível visualizar como o código se comporta, o que traz ao usuário um *feedback* instantâneo das ações que estão sendo realizadas durante a execução.

O VisuAlgo foi de grande importância para a pesquisa, pois, trata-se de um simulador com diversos conteúdos de estrutura de dados e algoritmos, disciplinas correlatas a de programação, e que também apresentam grande dificuldade por partes dos estudantes, além de apresentar vários pontos que foram considerados importantes para o desenvolvimento das diretrizes.

4. DIRETRIZES PARA O DESENVOLVIMENTO DE SIMULADORES DE APOIO AO ENSINO DE PROGRAMAÇÃO

Neste capítulo são apresentadas as análises, relações e constatações que permitiram a elaboração das diretrizes para o desenvolvimento de simuladores de apoio a área de programação.

4.1 RELACIONAMENTO DAS TAXONOMIAS DE BLOOM, MARZANO E VAN HIELE

Buscando atender todas as variáveis importantes para a construção do conhecimento, foram relacionadas as taxonomias de Bloom, Marzano e Van Hiele. Embora a taxonomia de Van Hiele tenha sido criada para o ensino de geometria, ela apresenta relação com o ensino de programação por apresentar as fases para a construção do raciocínio lógico, presentes na matemática e, de maneira semelhante, na programação.

No Quadro 3 é apresentada uma comparação entre as fases das taxonomias.

Fases de Bloom		**Fases de Marzano**		**Fases de Van Hiele**	
Conhecimento	- Acumular Informações	**Conhecimento**	-Nomear -Executar	**Visualização**	-Reconhecer
Compreensão	- Confirmar Aplicação	**Compreensão**	-Síntese -Representação		- Reproduzir
Aplicação	- Fazer uso do conhecimento	**Análise**	-Relação -Classificação -Análise de erros -Generalizações -Especificações	**Análise**	- Discernir características
Análise	- Identificar as partes e suas inter-relações	**Utilização**	-Tomada de Decisões -Resolução de Problemas -Investigação Experimental -Investigação	**Dedução Informal**	- Estabelecer inter-relações - Deduzir

Síntese	- Combinar, partes não organizadas para formar um todo	**Sistema de Metacogni-ção**	-Especificação de Metas -Monitorar Processos -Monitorar Clareza -Monitorar Precisão	**Dedução**	- Construir demonstrações - Desenvolver
Avaliação	- Julgar o Resultado	**Sistema Consciência do Ser**	-Avaliação da Importância -Avaliação da Eficácia -Avaliação Emocionais -Avaliação Motivacional	**Rigor**	- Trabalhar - Compreender - Abstrair

Quadro 3 – Fases das Taxonomias de Bloom, Marzano e Van Hiele
Elaborado pelo autor.

A relação entre as taxonomias de Bloom e Marzano foi realizada a partir da análise dos trabalhos de Granados (2015) e Brandão e Marques (2006), que já apresentavam esta relação. Já a relação com a taxonomia de Van Hiele foi incluída pelo autor desta pesquisa. Esta relação foi feita para facilitar a análise e a comparação entre as fases presentes em cada taxonomia, de modo que tornasse mais fácil sua utilização no desenvolvimento das diretrizes.

Todas as taxonomias apresentam semelhanças em suas fases, como visto no quadro 3 mesmo as fases que não possuem o mesmo nome e descrição acabam trazendo uma proposta similar as demais, como é possível ver na fase de Análise na taxonomia de Bloom, e a fase de Dedução Informal na taxonomia de Van Hiele, que buscam estabelecer e identificar os problemas e os relacionamentos entre os conteúdos.

4.2 AS TAXONOMIAS DE ENSINO-APRENDIZAGEM COMO FORMA DE MINIMIZAR AS DIFICULDADES NO ENSINO DE PROGRAMAÇÃO

Através do estudo realizado por Oliveira e Arimoto (2018) podemos ver que existem diversas dificuldades envolvendo o ensino e aprendizagem de programação,

além de que cada indivíduo aprende de diversas formas e em um determinado período de tempo (GOMES *et al*., 2008).

No quadro 4 é possível observar alguns dos conteúdos com um maior grau de dificuldade encontrados por Oliveira e Arimoto (2018) em seus estudos.

DIFICULDADES
Definição e uso de funções e métodos.
Passagem por parâmetro (Valor/Referência)
Definição e uso de Arrays
Manipulação de Strings
Definição e uso de ponteiros
Definição e uso de algoritmos recursivos
Definição e uso de estrutura de dados

Quadro 4 – Conteúdos de Programação com um maior grau de Dificuldade.
Fonte: Oliveira & Arimoto (2018)

Tendo em vista as dificuldades apresentadas por Oliveira e Arimoto (2018), é possível utilizar as taxonomias como uma estratégia para orientar o ensino a partir de suas fases, de maneira que a construção da aprendizagem possa ser gradual, e significativa, de forma a estimular os alunos.

Pretende-se com o uso de simuladores minimizar as dificuldades encontradas, por exemplo, no ensino das estruturas de dados como, os métodos de ordenação, arvores de decisão, entre outros, que é um dos conteúdos que foi apresentado com maior dificuldade pelos alunos, e através das fases das taxonomias, definir um método para a utilização da ferramenta de forma eficaz no processo de ensino e aprendizagem, aplicando cada uma das fases no processo de desenvolvimento, com o objetivo de desenvolver simuladores mais eficientes no processo de ensino.

Utilizando a taxonomia de Bloom, podemos por exemplo, propor o desenvolvimento da seguinte forma:

A fase de Conhecimento indica que lembrar consiste em reconhecer e recordar informações, fatos, já adquiridos e guardados na memória. Desta forma, é possível explicar conteúdos como fila, pilha e lista, dando o exemplo de pilhas de pratos, fila de pessoas, entre outros exemplos que são de simples abstração, e que fazem parte do cotidiano do aluno.

A Compreensão significa entender a informação ou o fato, captar seu significado e utilizá-lo em contextos diferentes dos apresentados inicialmente. Por exemplo, após a explicação dos conteúdos de fila, pilha e lista relacionados a coisas ou eventos do cotidiano do aluno, é possível apresentar situações que fogem de sua realidade para verificar se o conhecimento foi adquirido de forma necessária para resolver os problemas propostos.

Após a fase de Compreensão, a fase de Aplicação consiste em aplicar o conhecimento adquirido em situações concretas, visando analisar a capacidade dos alunos em abstrair os conteúdos apresentados. É a fase onde se inicia a resolução dos problemas, e onde é verificado se houve a compreensão do conteúdo.

A fase de Análise é onde os alunos começam a entender como as partes se relacionam, é onde eles começam a estabelecer hipóteses, e entender as características e propriedades do problema apresentado, é onde se analisa problema e quais as formas de resolvê-lo.

Síntese, é a fase na qual o aluno começa a construir, modificar e combinar as partes não organizadas para formar um todo, refere-se à capacidade de planejamento e criação a partir do conteúdo ensinado. Nessa fase é onde o aluno começa a elaborar formas diferentes para resolver os problemas, por exemplos, diferentes formas de ordenar uma lista, ou verificar se ela está ordenada.

Na fase de Avaliação, os alunos devem ser capazes de julgar, criticar, comparar, de forma que sejam capazes de encontrar erros e corrigi-los.

Desta forma, pretende-se que através de cada fase o aluno possa ser capaz de entender, reutilizar e se auto avaliar, tornando a aprendizagem significativa, dentro do tempo determinado para cada aluno.

4.3 COMPOSIÇÃO DAS DIRETRIZES PARA O DESENVOLVIMENTO DE SIMULADORES DE APOIO AO ENSINO DE PROGRAMAÇÃO

Verificando as dificuldades de aprendizagem em programação apresentadas por Oliveira e Arimoto (2018), observa-se que o maior grau de dificuldade dos alunos está relacionado a conteúdos que requerem alta capacidade de abstração, como é o caso da definição e uso de ponteiros, uso de ponteiros recursivos e uso de estrutura de dados. A

partir destas dificuldades o autor discute as sugestões de soluções apresentadas pelos próprios alunos que participaram da pesquisa. Dentre as sugestões destacam-se: mudar/alternar o método de ensino, aulas mais práticas, uso de abstrações mais lúdicas e resolução de exercícios junto com os alunos.

Assim, analisando este cenário, observa-se o potencial dos simuladores no ensino de programação, uma vez que permitem a representação visual de um elemento ou de uma propriedade de um todo e a interpretação de um problema por meio de um exemplo real.

As diretrizes foram elencadas de acordo com o conhecimento adquirido nos principais trabalhos que fundamentam essa pesquisa. Os trabalhos foram escolhidos em congressos como CBIE - Congresso Brasileiro de Informática na Educação, ITiCSE - *Innovation and Technology in Computer Science Education*, SBIE - Simpósio Brasileiro de Informática na Educação e TISE - Conferência Internacional sobre Informática na Educação, além de trabalhos que discutem as taxonomias de Bloom, Marzano e Van Heile, dos Padrões de boas práticas em simulação da INACSL - *International Nursing Association for Clinical Simulation and Learning* e das dificuldades encontradas no ensino e aprendizagem de programação, apresentadas por Oliveira e Arimoto (2018).

No quadro 5 são apresentadas as diretrizes propostas, juntamente com sua origem, de onde ela foi fundamentada, e também é demonstrada a relação com as fases das taxonomias para reforçar seu uso e seu potencial.

Diretrizes	Origem	Taxonomias
1- Realize uma avalição das necessidades dentro da disciplina de programação, para fornecer uma experiência baseada em simulação bem projetada.	Baseada nos critérios de desenvolvimento de simuladores da INACSL - *Standards Committee*, 2016.	• Conhecimento - Bloom • Análise – Bloom, Marzano e Van Heile
2- Construa objetivos mensuráveis de aprendizagem.	Baseada nos critérios de desenvolvimento de simuladores da INACSL - *Standards Committee*, 2016.	• Análise - Marzano • Sistema de Metacognição - Marzano
3- Defina os conceitos e uma Abordagem Facilitadora.	Baseada nos critérios de desenvolvimento de simuladores da INACSL - *Standards Committee*, 2016.	• Dedução Informal – Van Heile • Dedução – Van Heile

4- Crie um cenário ou caso que forneça o contexto necessário para a experiência baseada em simulação, sempre promovendo a criatividade.	Baseada nos critérios de desenvolvimento de simuladores da INACSL - *Standards Committee*, 2016.	• Rigor – Van Heile • Visualização – Van Heile • Aplicação - Bloom
5- Projete uma Interface gráfica amigável.	Elaborada pelo autor: tendo em vista que as formas como as informações são passadas e estruturadas, influenciam a capacidade de aprendizado, uma interface bem desenvolvida colabora para o entendimento do conteúdo, facilitando a leitura de textos e visualização das imagens, fornecendo também um bem-estar ao usuário durante o uso da ferramenta.	• Sistema consciência de ser - Marzano
6- Comece as simulações com um resumo.	Baseada nos critérios de desenvolvimento de simuladores da INACSL - *Standards Committee*, 2016.	• Rigor – Van Heile • Conhecimento - Bloom
7- Forneça *Feedback.*	Elaborada pelo autor: como programação e suas correlatas envolvem conteúdos de difícil abstração é importante sempre fornecer ao usuário um *feedback*, que o ajude a lembrar e reforçar partes do conteúdo que podem ser esquecidas.	• Análise - Marzano • Dedução informal – Van Heile
8- Forneça os recursos necessários para que os participantes possam atingir seus objetivos, aumentando seu conhecimento e encontrando resultados diferentes das experiências baseada em simulação.	Elaborada pelo autor: é importante sempre fornecer ao usuário formas de verificar e sanar suas dúvidas caso a simulação não seja o suficiente. Desta forma ele pode também buscar diferentes formas para resolver o mesmo problema.	• Análise - Marzano • Síntese - Bloom • Utilização - Marzano
9 – Avalie os resultados obtidos	Elaborada pelo autor: ponto importante para avaliar se as diretrizes anteriores foram eficazes em sua proposta, e analisar a eficiência do simulador desenvolvido no processo de ensino e aprendizagem.	• Avalição - Bloom • Rigor – Van Heile • Sistema consciência de ser - Marzano

Quadro 5 – Diretrizes
Elaborado pelo autor.

Na sequência são apresentadas e detalhadas as diretrizes para a construção de simuladores de apoio ao ensino de programação.

1- Realize uma avalição das necessidades dentro da disciplina de programação, para fornecer uma experiência baseada em simulação bem projetada.

Ao início do desenvolvimento, deve ser avaliado quais as necessidades presentes dentro da disciplina, e definir o tema que será abordado no simulador, qual o público alvo e quais as dificuldades com relação à aprendizagem deste tema para este público alvo, visto que de acordo com o estudo de Oliveira e Arimoto (2018), os conteúdos com um maior grau de abstração são aqueles que geram maior dificuldade no processo de ensino e aprendizagem.

2- Construa objetivos mensuráveis de aprendizagem.

Nessa fase, deve-se avaliar quais as metas que se pretende alcançar. Uma vez definido o tema, deve-se delimitar quais os objetivos buscam-se atingir dentro do tema, estabelecendo quais conteúdos serão abordados e como irá tratá-los.

3- Defina os conceitos e uma abordagem facilitadora.

Um simulador deve disponibilizar ao usuário a definição dos principais conceitos do conteúdo abordado em uma linguagem clara e objetiva, evitando a utilização de termos técnicos desconhecidos aos usuários iniciantes. Os conceitos devem ser apresentados gradativamente, de acordo com o desenvolvimento do usuário e sua compreensão do conteúdo, ou conforme a necessidade do mesmo.

4- Crie um cenário ou caso que forneça o contexto necessário para a experiência baseada em simulação, sempre promovendo a criatividade.

Um simulador deve disponibilizar ao usuário a opção de aplicar o conhecimento em situações concretas, criando um cenário ou uma situação que leve o usuário a compreender o problema de forma eficaz, permitindo que o mesmo reflita a respeito do conteúdo abordado, para que através da reflexão possa reconstruir os conceitos, podendo aprimora-los e até mesmo se possível pensar em soluções diferentes para o mesmo problema.

5- Projete uma Interface gráfica amigável.

Um simulador deve possuir uma interface gráfica amigável com o usuário, de forma que sua utilização seja agradável e produtiva, mantendo uma abordagem facilitadora que seja centrada no participante e conduzida pelos seus objetivos, seu conhecimento ou nível de experiência e os resultados esperados.

6- Comece as simulações com um resumo.

Todo conteúdo deve ser explicado antes de se iniciar as simulações, por exemplo, na simulação de conteúdos como, algoritmos de ordenação, deve-se apresentar inicialmente de forma teórica, e com uma linguagem simples, o que é, quais são as ações possíveis de se realizar, para que serve, e como são usados, de forma que o usuário esclareça suas dúvidas antes que a simulação comece.

7- Forneça *Feedback.*

Um simulador deve fornecer aos usuários o *feedback* do conteúdo abordado durante a simulação para que dessa forma enriqueça o aprendizado e contribua para as experiências baseadas em simulação. Para Papert (1990), *feedback* é quando há reflexão sobre a ação, quando a ferramenta mostra o resultado daquilo que o sujeito propôs e através do resultado permita refletir e reconstruir. O *feedback* é sinalizado em vários materiais relacionados a simulação, e também na educação, fazendo com que os conteúdos abordados se fixem e melhorem a compreensão dos conceitos.

8- Forneça os recursos necessários para que os participantes possam atingir seus objetivos, aumentando seu conhecimento e encontrando resultados diferentes das experiências baseada em simulação.

Forneça matérias de apoio para os usuários, de forma com que eles possam de forma fácil, sanar suas dúvidas, fazer pesquisas e alcançar seus objetivos, dentro do seu próprio ritmo.

9- Avalie os Resultados Obtidos

Avalie e analise quão significativa foi a aprendizagem com o uso do simulador desenvolvido, aplicando algum teste ou atividade relacionada ao tema abordado. Note se houve ou não um melhor rendimento por parte dos alunos na abstração do conteúdo simulado, e se há necessidade de reforçar ou modificar algum ponto do desenvolvimento.

4.4 EXEMPLO DE APLICAÇÃO DAS DIRETRIZES

A seguir será apresentado um exemplo de aplicação das diretrizes propostas, juntamente com a taxonomia que contribui para o fundamento da diretriz e se possível a imagem de um simulador que exemplifica o uso das diretrizes.

4.4.1 REALIZE UMA AVALIÇÃO DAS NECESSIDADES DENTRO DA DISCIPLINA DE PROGRAMAÇÃO, PARA FORNECER UMA EXPERIÊNCIA BASEADA EM SIMULAÇÃO BEM PROJETADA.

Exemplo: Definição e uso de estrutura de dados: os alunos apresentam dificuldades na compreensão de conteúdos como, métodos de ordenação, arvores de decisão, entre outros.

Taxonomias Relacionadas:

- Conhecimento: Bloom - Acumular Informações
- Análise: Bloom, Marzano e Van Heile – Especificações, discernir características, identificar as partes e suas inter-relações.

Esta fase se denomina como a fase de acúmulo de informações, onde se pode discernir as características e necessidades presentes na disciplina.

4.4.2 CONSTRUA OBJETIVOS MENSURÁVEIS DE APRENDIZAGEM.

Exemplo: Pretende-se que os alunos, consigam, entender e aplicar os conceitos de Algoritmos de Ordenação Baseados em Comparação, como, *Bubble Sort, Insertion Sort e Merge Sort.*

Taxonomias Relacionadas:

- Análise: Marzano - Classificação, Análise de erros, Especificações.
- Sistema de Metacognição: Marzano - Especificação de Metas, Monitoração dos Processos.

Esta diretriz busca os melhores resultados, analisando e identificando os problemas e as possíveis soluções, para propor objetivos e metas que possam ser cumpridas.

4.4.3 DEFINA OS CONCEITOS E UMA ABORDAGEM FACILITADORA.

Exemplo: Na computação existe uma série de algoritmos que utilizam diferentes técnicas de ordenação para organizar um conjunto de dados, eles são conhecidos como Métodos de Ordenação ou Algoritmos de Ordenação.

Taxonomias Relacionadas:

- Dedução Informal: Van Heile – Estabelecer inter-relações.
- Dedução: Van Heile - Construir demonstrações.

Na figura 4 é possível visualizar como o simulador VisuAlgo apresenta os conceitos do conteúdo de uma forma simples, levando o usuário a compreender o que se trata e o que será simulado.

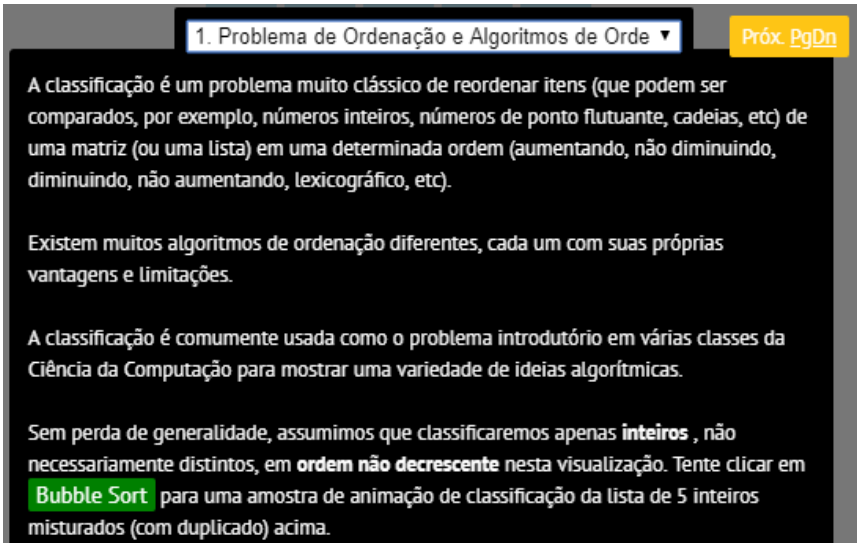

Figura 4 – Exemplo, Definição dos conceitos e Abordagem Facilitadora
Fonte: visualgo.net/pt

Esta diretriz busca apresentar os conceitos iniciais de forma clara, visando o entendimento do usuário, demonstrando e relacionando os conteúdos para que haja um melhor entendimento.

4.4.4 CRIE UM CENÁRIO OU CASO QUE FORNEÇA O CONTEXTO NECESSÁRIO PARA A EXPERIÊNCIA BASEADA EM SIMULAÇÃO, SEMPRE PROMOVENDO A CRIATIVIDADE.

Exemplo: Uma professora pediu para que seus alunos fossem a biblioteca e ordenassem todos os livros de programação presentes na prateleira, lembrando que os livros estavam todos desorganizados e eles precisariam ser postos de forma organizada por ano de publicação, a ordenação deve ser feita através do algoritmo *Bubble Sort.*

Taxonomias Relacionadas:

- Aplicação: Bloom - Fazer uso do conhecimento.
- Visualização: Van Heile – Reconhecer, Reproduzir.
- Rigor: Van Heile – Compreender, Abstrair.

Aqui as fases de aplicação, visualização e rigor, se aplicam de forma que, os usuários apliquem seu conhecimento já adquirido, e possa reproduzi-lo em situações similares.

4.4.5 PROJETE UMA INTERFACE GRÁFICA AMIGÁVEL.

Exemplo: Fonte legível, cores que não cansam a vista, botões bem posicionados, ou seja, um produto que seja atrativo ao usuário.

Taxonomia Relacionada:

- Sistema consciência de ser: Marzano – Avaliação da Eficácia, Avaliação Motivacional.

A figura 5 apresenta o simulador de estrutura de dados e algoritmos VisuAlgo, e mostra como uma interface simples e fácil de interagir é eficaz, e facilita para o usuário utilizar e encontrar os recursos disponíveis pelo simulador.

Figura 5 – Exemplo, Interface amigável
Fonte: visualgo.net/pt

Nesse ponto podemos destacar a fase Sistema consciência do ser presente na taxonomia de Marzano, que diz respeito a avaliações motivacionais e de eficácia, o que interferem no aprendizado, uma interface ruim prejudica e desmotiva o usuário, fazendo com que o aprendizado não seja eficaz.

4.4.6 COMECE AS SIMULAÇÕES COM UM RESUMO.

Exemplo: O *bubble sort*, ou literalmente “bolha", é um algoritmo de ordenação dos mais simples. A ideia é percorrer um vetor diversas vezes, e a cada passagem fazer uma troca se necessário entre os dois índices comparados, levando sempre para o fim o maior elemento da sequência.

Taxonomias Relacionadas:

- Rigor: Van Heile – Compreender, Abstrair.
- Conhecimento: Bloom – Acumular Informações.

É possível visualizar na figura 6, que apresentar um resumo do conteúdo simulado antes que a simulação comece, ajuda o usuário a entender o que irá acontecer durante a simulação.

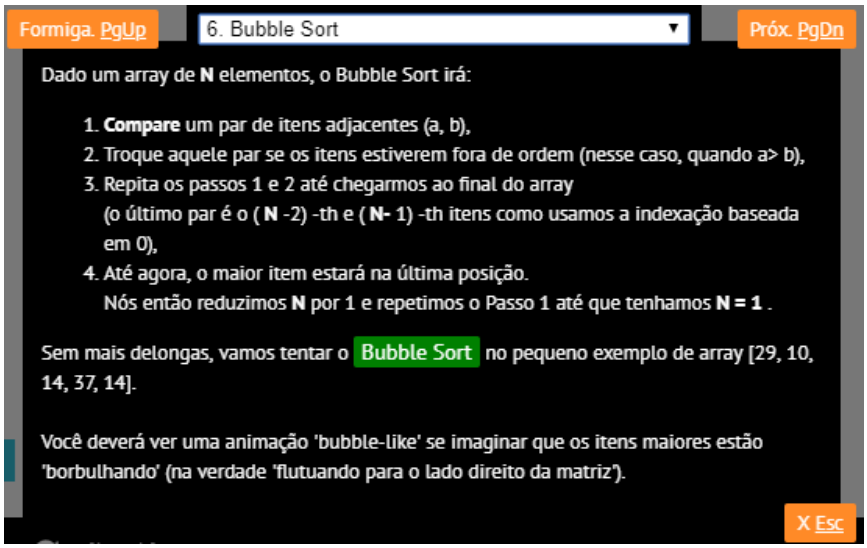

Figura 6 – Exemplo, Comece as simulações com um resumo

Fonte: visualgo.net/pt

As fases de Rigor e conhecimento se aplicam de forma que, o conteúdo seja apresentado ao usuário com o objetivo de supri-lo de informações que sejam necessárias para o seu entendimento, fazendo com que ele possa compreender e acumular as informações que serão precisas para resolver os problemas apresentados.

4.4.7 FORNEÇA *FEEDBACK*

Exemplo: Enquanto as trocas são simuladas, é importante que o simulador apresente detalhes das ações que estão sendo executadas, isso dá ao usuário um *feedback* instantâneo do passo a passo da execução.

Taxonomias Relacionadas:

- Análise: Marzano – Relação, Especificação, Classificação.
- Dedução Informal: Van Heile – Estabelecer inter-relações.

Na figura 7 é possível visualizar que é fornecido durante a simulação um *feedback* da execução do código, o que ajuda o usuário a entender quais ações estão sendo executadas em cada momento.

Figura 7 – Exemplo, Fornecer *Feedback*

Fonte: visualgo.net/pt

Nessa diretriz pode-se aplicar as fases de análise e dedução informal, que dizem respeito a identificação e relação dos problemas e soluções, reforçando aquilo que pode ter sido esquecido.

4.4.8 FORNEÇA OS RECURSOS NECESSÁRIOS PARA QUE OS PARTICIPANTES POSSAM ATINGIR SEUS OBJETIVOS, AUMENTANDO SEU CONHECIMENTO E ENCONTRANDO RESULTADOS DIFERENTES DAS EXPERIÊNCIAS BASEADA EM SIMULAÇÃO.

Exemplo: Nesta etapa pode ser fornecido links de artigos, vídeos e livros, que complemente de forma teórica, as ações realizadas pelo simulador.

Taxonomias Relacionadas:

- Análise: Marzano – Relação, Especificação, Classificação.
- Síntese: Bloom – Combinar, partes não organizadas para formar um todo.
- Utilização: Marzano - Tomada de Decisões, Resolução de Problemas, Investigação Experimental, Investigação.

Aqui podem ser destacadas as fases de análise, síntese e utilização, que provê usuário formas diferentes de julgar os resultados apresentados, fazendo com que eles busquem soluções diferentes para o mesmo problema, aumentando dessa forma seu conhecimento e sua capacidade de tomada de decisões e resolução de problemas.

4.4.9 AVALIE OS RESULTADOS OBTIDOS.

Exemplo: No final das simulações deve ser aplicado um questionário ou algum tipo de atividade sobre os principais conceitos abordados no simulador, com o intuito de identificar se o aluno entendeu ou não conteúdo, o que traz também para o professor um *feedback* de qual conteúdo necessita de mais ênfase durante o processo de ensino.

Taxonomias Relacionadas:

- Avaliação: Bloom – Julgar o Resultado.
- Rigor: Van Hiele – Compreender, Abstrair.
- Sistema consciência de ser: Marzano – Avaliação da Eficácia.

Nessa diretriz devem ser propostas maneiras de avaliar a eficácia do uso de simuladores, analisando se o seu uso é eficiente para o processo de ensino, auxiliando o professor na demonstração de conteúdos de difícil abstração.

5. CONSIDERAÇÕES FINAIS

Para o desenvolvimento desse trabalho, foi necessário entender quais eram os problemas e dificuldades enfrentados pelos alunos nas disciplinas de programação e suas correlatas, para isso foi utilizado o trabalho de Oliveira e Arimoto (2018), onde são apontadas as principais dificuldades apresentadas pelos alunos nessas disciplinas.

Também foram analisadas as taxonomias de ensino, que possuem diversas fases e são instrumentos que apoiam a análise do desenvolvimento cognitivo do aluno. As taxonomias foram combinadas e utilizadas para fundamentar e validar as diretrizes propostas.

Para a elaboração e desenvolvimento das diretrizes foram analisados diversos trabalhos relacionados ao ensino e aprendizagem de programação além de trabalhos sobre simuladores nas mais diversas áreas de conhecimento, também foram analisados os critérios de desenvolvimento de simuladores da área da saúde, propostos pela INACSL - *Standards Committee* (2016), que foi de suma importância para a elaboração das diretrizes. As diretrizes propostas, inicialmente, pretendiam contribuir para o desenvolvimento de simuladores de apoio ao ensino e aprendizagem na área de programação e disciplinas correlatas, por se tratarem de conteúdos de difícil abstração e possuir uma alta taxa de reprova e desistência por conta das dificuldades apresentadas pelos alunos. Porém, levando em consideração que as taxonomias de ensino utilizadas no desenvolvimento das diretrizes são instrumentos também aplicáveis ao processo de aprendizagem de outras áreas de conhecimento, sugere-se que as diretrizes possam ser aplicadas a outras áreas de ensino.

Visto que os simuladores tem-se apresentado como uma ferramenta de auxílio ao ensino nas mais diversas áreas de conhecimento, como trabalho futuro, buscando também verificar a aplicabilidade das diretrizes propostas, pretende-se utilizá-las no desenvolvimento de um simulador relacionado ao ensino de programação ou suas áreas correlatas, como algoritmos e estrutura de dados.

Por fim, espera-se com esta pesquisa contribuir, por meio de suas diretrizes, para o desenvolvimento de simuladores que busquem proporcionar o desenvolvimento cognitivo, englobando a aquisição do conhecimento, competência e atitudes, visando facilitar o planejamento do processo de ensino e aprendizagem.

REFERÊNCIAS

ARAÚJO de O. S. L. A. *et al.* (2013) **"Aplicação da Taxonomia de Bloom no ensino de programação com Scratch"**, Em: II Congresso Brasileiro de Informática na Educação (CBIE 2013), e XIX Workshop de Informática na Escola (WIE 2013).

BORGES, M. A. F. (2000) **"Avaliação de uma Metodologia Alternativa para a Aprendizagem de Programação"**, Em: Congresso da Sociedade Brasileira de Computação. Florianópolis, SC, Brasil.

BRANDÃO, Maria de Fátima Ramos; MARQUES, Jaqueline. **Competências para cursos de Licenciatura em Computação segundo um modelo de avaliação formativa**. In: Anais do XVII Simpósio Brasileiro de Informática na Educação SBIE. Brasília: I Workshop sobre Licenciatura em Computação WLC, 2006.

CÓRDOVA, K. E. G. (setembro de 2009). **La Nueva Taxonomía de Marzano y Kendall: una alternativa para enriquecer el trabajo educativo desde su planeación.** Manual Nueva Taxonomía Marzano y Kendall. 66 p.

DA CRUZ KNIPHOFF, E. J. M.; FROZZA, R.; MARQUES, G. S.; VIANA, R. R.; MOLZ, F. R.; REBELATTO, T. **Simulador para Introdução da Programação para Crianças e Análise da Aprendizagem com apoio da Neurociência.** CINTED-UFRGS, Novas Tecnologias na Educação. V. 10, Nº 3, Dez. 2012.

DOURADO, S. S A.; GIANNELLA, R. T. **Ensino Baseado em Simulação na Formação Continuada de médicos: análise das Percepções de alunos e Professores de um Hospital do rio de Janeiro.** Revista Brasileira de educação Médica. Universidade Federal do Rio de Janeiro, Rio de Janeiro, RJ, Brasil. 2014.

FERRAZ, A. P. C. M.; BELHOT, R. V. **Taxonomia de Bloom: revisão teórica e apresentação das adequações do instrumento para definição de objetivos instrucionais.** Gest. Prod., São Carlos, v. 17, n. 2, p. 421-431, 2010.

FONTES, Carlos. **Tipos de *Software* Educativo, Universidade Federal do Mato Grosso**, 2001. Instituto de Ciências Exatas e Naturais.

GIL, A. C. **Como elaborar projetos de pesquisa**. 4. ed. - São Paulo: Atlas, 2002. 173 p.

GOMES, A.; AREIAS, C.; HENRIQUES, J.; MENDES. A. J.(2008). **Aprendizagem de programação de computadores:** dificuldades e ferramentas de suporte. *Revista Portuguesa de Pedagogia, 19.*

GRANADOS, O. J. (novembro de 2015). **SISTEMA DE IDENTIFICACIÓN DE CONOCIMIENTO TÁCITO EN TEXTOS UTILIZANDO TÉCNICAS DE ANÁLISIS DEL DISCURSO**.

HECKLER, V.; SARAIVA M. F. O.; FILHO, K. S. O. **Uso de simuladores, imagens e animações como ferramentas auxiliares no ensino/aprendizagem de ótica**. Revista Brasileira de Ensino de Física, v. 29, n. 2, p. 267-273, 2007.

HOED, R. M. (2016). **Análise da evasão em cursos superiores: o caso da evasão em cursos superiores da área de Computação.** p. 61.

INACSL Standards Committee (2016, December). **INACSL standards of best practice: SimulationSM Simulation design.** Clinical Simulation in Nursing, 12(S), S5-S12.

INACSL.International Nursing Association for Clinical Simulation and Learning (2013).ClinicalSimulation in Nursing -Volume 9 - Issue 6S. p.9-32.

IXQUIER, T. E. P. (2017). **Cuadro comparativo sobre la taxonomía de bloom y marzano.** Disponível em: <http://taxonomiadebloomymarzo.blogspot.com/2017/05/cuadro-comparativo-sobre-la-taxonomia.html> Acesso em: 19 de setembro de 2018.

JUNIOR DIAS, M. V.; MERCADO, L. L. P. **A importância da estratégia de ensino por simulação para a disciplina de algoritmos.** CIAIQ, v. 4. 2016.

KLAUS**,** Tiago Stolben; PAZOS, Rubém Panta. **Os Níveis de Van Hiele com o Auxílio de Ferramentas Computacionais,** 2005. Disponível em< http://miltonborba.org/CD/Interdisciplinaridade/Encontro_Gaucho_Ed_Matem/cientificos/CC69.pdf>. Acesso em: 12 de julho de 2018.

LEITE, M. D.; PESSOA, C. A. S.; FERRAZ, M. C.; SOUZA R. B. R. **Softwares educativos e objetos de aprendizagem: um olhar sobre a análise combinatória.** X Encontro Gaúcho de Educação Matemática 02 a 05 de junho de 2009, Ijuí/RS.

LÉVY, Pierre**. Simulações.** In: LÉVY, Pierre. **Cibercultura**. Edição brasileira. Rio de Janeiro, EDITORA 34, 1999. p. 68 – 71.

MACHADO F. B.; MAIA L. P. **Um framework construtivista no aprendizado de Sistemas Operacionais** – uma proposta pedagógica com o uso do simulador SOsim.

MERCADO, L. P. L. **Novas Tecnologias na Educação: Reflexões sobre a Prática.** Maceió Al: EDUFAL, 2002. 210 p.

NETO, B.C. W.; SCHUVARTZ A. A. **Ferramenta Computacional de Apoio ao Processo de Ensino Aprendizagem dos Fundamentos de Programação de Computadores**. XVIII Simpósio Brasileiro de Informática na Educação - SBIE - Mackenzie – 2007.

OLIVERIA, W. T., & ARIMOTO, M. M. (2018). **Um Estudo sobre os Problemas e Dificuldades relacionados ao Ensino e Aprendizagem de Programação.** *SITE 2018*.

PAPERT, Seymour. **A Critique of Technocentrism in Thinking About the School of the Future.** M.I.T. Media Lab Epistemology and Learning Memo. M.I.T., 1990.

PAZIN FILHO, A. SCARPELINI, S**. Simulação:** definição. Medicina (Ribeirão Preto) 2007; 40 (2): 162-6.

PEARS, A., SEIDMAN, S., MALMI, L., MANNILA, L. ADAMS, E. BENNEDSEN, J., DEVLIN, M. e PATERSON, J. (2007) **"A survey of literature on the teaching of introductory programming",** Em: Working group reports on ITiCSE on Innovation and

technology in computer science education, ITiCSE-WGR '07, ACM , New York, NY, USA , 204-223.

PELLAS, N. & VOSINAKIS, S. (2018). **The effect of simulation games on learning computer programming:** A comparative study on high school students' learning performance by assessing computational problemsolving strategies. Education & Information Technologies (Springer).

PEREIRA, Gislaine A.; SILVA, Sandreane P.; MOTTA, Walter Santos. **O Modelo Van Hiele de Ensino de Geometria aplicado à 5 e 6 séries do Ensino Fundamental,** Uberlandia-MG, 2005.

PEREIRA, J. J. C. R. e RAPKIEWICZ, C. E. (2004), **"O processo de Ensino e Aprendizagem de Algoritmos e Programação**: Uma Visão Crítica da Literatura**"**, Em: III WEIMIG - Workshop de Educação em Computação e Informática, Belo Horizonte, MG, Brasil.

PRIETCH, S. S., PAZETO, T. A. (2010), **"Mapeamento de Cursos de Licenciatura em Computação seguido de Proposta de Padronização de Matriz Curricular"**, Em: WEI – XVIII Workshop sobre Educação em Computação, Belo Horizonte, MG, Brasil.

RAABE, A. L. A. e SILVA, J. M. C. (2005), "**Um Ambiente para Atendimento as Dificuldades de Aprendizagem de Algoritmos"**, Em: XXV CSBC - Congresso da Sociedade Brasileira de Computação. São Leopoldo, RS, Brasil.

ROBINS, A. (2010), **"Learning edge momentum:** A new account of outcomes in CS1", Computer Science Education, v. 20, n. 1, p. 37–71.

ROBINS, A.; ROUNTREE, J.; ROUNTREE, N. **Learning and Teaching Programming:** A Review and Discussion. Computer Science Education, 2003. v. 13, No. 2, p. 137-172.

SANTOS, R. P., & Costa, H. A. (2006). **Análise de metodologias e ambientes de ensino para Algoritmos, Estruturas de Dados e Programação aos iniciantes em Computação e Informática.**

SANTOS, W. **Uso de simuladores como ferramenta no ensino-aprendizagem de redes de computadores**. Belo Horizonte, 2015.

SCHOENFELD, A. **Learning to think mathematically**: Problem Solving, Metacognition, and sense Making in Mathematics, New York, 1992.

VIEGAS, T. R.; OKUYAMA, F. Y.; PARAVISI, M.; BERTAGNOLLI, S. C. **Uso das TICs no processo de ensino-aprendizagem de programação.** Nuevas Ideas en Informática Educativa TISE. 2015.

VISUALGO. **Visualizando Estrutura de Dados e Algoritmos através de Animação.** Disponível em: <https://visualgo.net/pt>. Acesso em: 12 de fevereiro de 2019.