
**UNIVERSIDADE ESTADUAL DA PARAÍBA**

**CENTRO DE CIÊNCIAS EXATAS E SOCIAIS APLICADAS**

**CAMPUS VII GOVERNADOR ANTÔNIO MARIZ**

**CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO**

**YASMIM COSTA FERREIRA**

# UMA ONTOLOGIA PARA REPRESENTAÇÃO DA TAXONOMIA REVISADA DE BLOOM EM QUESTÕES DE ALGORITMOS

**Patos - PB**
**2019**

YASMIM COSTA FERREIRA

# UMA ONTOLOGIA PARA REPRESENTAÇÃO DA TAXONOMIA REVISADA DE BLOOM EM QUESTÕES DE ALGORITMOS

Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Ciência da Computação da Universidade Estadual da Paraíba, em cumprimento à exigência para obtenção do grau de bacharel em Ciência da Computação.

**Área de concentração:** Web Semântica.

**Orientador:** Prof. MSc. Pablo Roberto Fernandes de Oliveira

**Patos - PB**
**2019**

F383o Ferreira, Yasmim Costa.

Uma ontologia para representação da taxonomia revisada de Bloom em questões de algoritmos [manuscrito] / Yasmim Costa Ferreira. - 2019.

65 p. : il. colorido.

Digitado.

Trabalho de Conclusão de Curso (Graduação em Computação) - Universidade Estadual da Paraíba, Centro de Ciências Exatas e Sociais Aplicadas , 2019.

"Orientação : Prof. Me. Pablo Roberto Fernandes de Oliveira , Coordenação do Curso de Computação - CCEA."

1. Taxonomia revisada de Bloom. 2. Ontologia. 3. Aprendizagem de algoritmos. I. Título

21. ed. CDD 004

Elaborada por José E. da S. Eugênio - CRB - 15/591 BSC7/UEPB

Yasmim Costa Ferreira

**UMA ONTOLOGIA PARA REPRESENTAÇÃO DA TAXONOMIA DE BLOOM EM QUESTÕES DE ALGORITMOS**

Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Ciências da Computação da Universidade Estadual da Paraíba, em cumprimento à exigência para obtenção do grau de Bacharel em Ciência da Computação.

Aprovado em 25/11/2019

BANCA EXAMINADORA

Pablo Roberto Fernandes de Oliveira

Prof. Me. Pablo Roberto F. de Oliveira
(Orientador)

Jannayna Domingues Barros Filgueira

Prof. Dra. Jannayna Domingues Barros Filgueira
(Examinadora)

Prof. Dr. Rodrigo Alves Costa
(Examinador)

Dedico este trabalho ao meu pai, José Ferreira Neto, e a minha mãe, Maria Bernadete de Meneses Costa, que sempre me incentivaram e nunca mediram esforços para me ajudar, obrigada por todo amor.

## AGRADECIMENTOS

Durante a minha caminhada acadêmica, muitas barreiras tive que ultrapassar, mas também vivi momentos que me fizeram crescer e me tornar a mulher que sou hoje, momentos estes pelos quais sou muito grata.

Sei que ninguém consegue chegar sozinho a lugar nenhum, precisamos uns dos outros. Para que eu pudesse chegar até aqui não faltaram pessoas para me ajudar e me apoiar. É por tanto, chegado o momento de agradecê-las.

Agradeço primeiramente a Deus, que em sua infinita bondade me deu a vida e conduziu meus passos até este momento, nunca me desamparando e me dando coragem para enfrentar todos os obstáculos.

Aos meus amados pais José Neto e Maria Bernadete por todo amor e dedicação, por terem me ensinado desde cedo que o melhor caminho é o da educação e por nunca terem me deixado desistir apesar de todas as dificuldades. Amo vocês mil milhões.

Aos meus irmãos Felicidade, Wesley, Wellington, Wendel e Wallisson pela força, carinho e companheirismo.

Aos meus avós e tios por todo o apoio e incentivo. Agradeço de forma muito especial a minha tia Olga e seu esposo Joselito por terem aberto as portas de sua casa para mim no momento que mais precisei. A vocês minha eterna gratidão.

Aos meus queridos primos e amigos por todo carinho e apoio, seja de perto ou de longe.

A toda turma 2015.1, em especial aos sobreviventes (Keila, Claudia, Lucas, Allan, Bruno, David, Jordão e Welton) pela amizade e força que me deram nessa caminhada.

Ao meu orientador Pablo Roberto Fernandes de Oliveira pela paciência e desprendimento em me ajudar na construção deste trabalho.

Aos professores que compõem a banca examinadora pela leitura minuciosa e pelas contribuições oferecidas a este trabalho.

A todos os professores da UEPB que foram responsáveis pela minha formação acadêmica.

A todos que embora não citados aqui contribuíram direta ou indiretamente para que eu conquistasse este objetivo.

Meu muito obrigado!

Seja Corajoso e faça épico!
(Kim Holden)

## RESUMO

O aprendizado de Algoritmos é considerado desafiador pelos estudantes de cursos tecnológicos. Os principais motivos para este fato são: o alto nível de abstração do conteúdo e a dificuldade dos docentes em detectar e atender as dificuldades de aprendizado dos alunos. A taxonomia de Bloom é uma prática pedagógica e metodológica de ensino que contribui para o desenvolvimento e avaliação de conteúdos e tem sido utilizada em diversas áreas do conhecimento como forma de avaliação da aprendizagem. Apesar disso, alguns autores têm relatado dificuldades em utilizá-la na elaboração de avaliações no contexto da programação introdutória. Com isso, modelos semânticos, como as ontologias, podem ser utilizados, pois, visam desenvolver um conjunto de regras que possibilitam a inferência de forma que a máquina possa, através do acesso a essas regras, abstrair um significado semântico das informações disponibilizadas. Dessa forma, o presente trabalho teve como objetivo desenvolver uma ontologia para a representação de questões de algoritmos, de acordo com o domínio Cognitivo da Taxonomia Revisada de Bloom. O procedimento metodológico adotado para construção da ontologia foi a metodologia proposta no guia 101 e a ferramenta utilizada para o desenvolvimento da mesma foi a *Protégé*. A validação foi feita de forma analítica por meio do motor de inferência *HermiT*, regras *SWRL*, consultas *SPARQL* e um protótipo no qual pôde-se demonstrar um cenário hipotético de uso. Como resultado obteve-se: automatizar o processo de inferência de conhecimento dentro do domínio da Taxonomia Revisada de Bloom; determinar o nível de aprendizagem de um estudante, considerando a taxonomia, e o nível de uma questão de algoritmos; trabalhar os conteúdos de Algoritmos de forma hierárquica. Considerou-se ao final da pesquisa que a ontologia cumpriu o propósito para o qual foi criada, além de oferecer um potencial a ser explorado para a melhoria do processo de ensino e aprendizagem em ambientes virtuais, utilizando a taxonomia de Bloom.



**Palavras-Chave:** Aprendizagem de Algoritmos. Taxonomia Revisada de Bloom. Ontologia.

## ABSTRACT

Algorithm learning is considered challenging by students of technology courses. The main reasons for this fact are: the high level of content abstraction and the difficulty of teachers to detect and meet the learning difficulties of students. Bloom's taxonomy is a pedagogical and methodological teaching practice that contributes to the development and assessment of content and has been used in various areas of knowledge as a means of learning assessment. Nevertheless, some authors have reported difficulties in using it in the elaboration of evaluations in the context of introductory programming. Thus, semantic models such as ontologies can be used because they aim to develop a set of rules that allow inference so that the machine can, through access to these rules, abstract a semantic meaning of the information provided. Thus, the present work aimed to develop an ontology for the representation of algorithm issues, according to the Cognitive domain of Bloom's Revised Taxonomy. The methodological procedure adopted to construct the ontology was the methodology proposed in guide 101 and the tool used for its development was Protégé. Validation was performed analytically using the *HermiT* inference engine, *SWRL* rules, *SPARQL* queries and a prototype in which a hypothetical usage scenario could be demonstrated. As a result we obtained: automate the knowledge inference process within the domain of Bloom's Revised Taxonomy; determine a student's level of learning, considering taxonomy, and the level of a matter of algorithms; hierarchically work the contents of Algorithms. It was considered at the end of the research that ontology fulfilled the purpose for which it was created, besides offering a potential to be explored for the improvement of the teaching and learning process in virtual environments, using Bloom's taxonomy.



**Keywords:** Algorithm Learning. Bloom's Revised Taxonomy. Ontology.

## LISTA DE ILUSTRAÇÕES

## LISTA DE QUADROS

**SUMÁRIO**

# 1 INTRODUÇÃO

As disciplinas dos cursos de graduação da área de computação que envolvem algoritmos e programação de computadores são essenciais para aqueles alunos que terão como formação a área de desenvolvimento de software, contudo, o ensino e a aprendizagem são consideradas tarefas complicadas para alguns professores e alunos (SOUTO; DUDUCHI, 2009). A começar pela complexidade dos conteúdos que acabam por dificultar o entendimento de alunos em determinados conceitos de programação, e ainda há a dificuldade em aplicá-los durante a construção de programas.

Além das muitas dificuldades de aprendizagem enfrentadas por alunos de programação nas disciplinas iniciais, os professores têm a tarefa de avaliar o conhecimento destes e muitas vezes, devido ao número de estudantes nessas turmas, o acompanhamento especializado torna-se difícil. O envolvimento ou não do professor é uma questão agravante no desempenho do aluno, pois o não envolvimento pode gerar desmotivação, desinteresse, e em alguns casos até a sua desistência, colaborando com o aumento no índice de evasão do curso (CHAVES, 2014).

Souza, Batista e Barbosa (2016) dizem que, referente às dificuldades enfrentadas pelos professores, destacam-se as seguintes: dificuldades dos professores em acompanhar a aprendizagem do aluno, desenvolver materiais e exercícios de apoio, bem como avaliar os trabalhos de programação.

Jesus e Raabe (2009) também se referem aos problemas que os professores de algoritmos enfrentam na avaliação de seus alunos. Eles levantam algumas questões referentes a essas dificuldades: Como um professor poderia mensurar objetivamente a aprendizagem de seus alunos? Supondo que os alunos obtenham ótimo desempenho, como saber, por exemplo, se o desempenho não foi causado por testes demasiadamente fáceis? O professor poderia comparar o desempenho de sua turma com uma turma de referência, mas nesse caso, como saber se a turma de referência é realmente referência?

Várias ferramentas desenvolvidas têm contribuído para auxiliar os docentes na avaliação dos seus alunos, dentre elas, pode-se citar algumas das estratégias mais promissoras: Os juízes *online*, que são sistemas de avaliação automática de códigos; O Moodle, um ambiente virtual de aprendizagem; Os Tutores Inteligentes, que atuam em ambientes afim de identificar dúvidas frequentes e assim fazer a recomendação de conteúdo; e ontologias, utilizadas em ambientes como os já mencionados para classificação da aprendizagem dos alunos e estratégias de indicação de conteúdo. Essas ferramentas apoiam tanto o ensino presencial, quanto o ensino a distância.

Acerca disso, o uso de ontologias já foi explorado em muitos trabalhos na área da Ciência da Computação, podendo ser citados os trabalhos de Carvalho (2017) e Nascimento (2018), que utilizaram ontologias integradas a Sistemas Tutores Inteligentes, com o intuito de contribuir com a aprendizagem de programação.

Mas além de ferramentas, algumas práticas pedagógicas e metodológicas de ensino contribuem para o desenvolvimento e avaliação de conteúdos. A taxonomia de Bloom (BLOOM, 1956), por exemplo, é muito utilizada em diversas áreas do conhecimento como forma de avaliação da aprendizagem. Trata-se de uma estrutura conceitual configurada para auxiliar a definição de objetivos de aprendizagem. Jesus e Raabe (2009), Santos (2016), Araújo et al. (2013) buscaram em seus trabalhos aplicá-la num contexto da programação introdutória para mapear a aprendizagem de algoritmos na elaboração de avaliações.

Visando contribuir com a melhoria do processo de ensino e aprendizagem de programação, no que diz respeito a interpretação de uma questão, considerando a Taxonomia Revisada de Bloom, o presente estudo apresenta a proposta de uma ontologia para representar a Taxonomia e classificar questões de programação em um dos níveis do domínio Cognitivo da mesma.

## 1.1 OBJETIVOS

O intuito desta pesquisa foi conceber um artefato de software que possibilite a interpretação das categorias da taxonomia de Bloom em questões de programação, a fim de facilitar o processo de avaliação de um determinado aluno. Para tanto, objetivou-se desenvolver uma ontologia para classificar questões de aprendizagem considerando a Taxonomia Revisada de Bloom[1].

Os objetivos específicos são:

- Identificar os requisitos para a construção da ontologia sob a perspectiva da Taxonomia Revisada de Bloom para conceitos introdutórios de programação;
- Desenvolver a ontologia notada semanticamente do domínio Cognitivo da Taxonomia Revisada de Bloom e conceitos de programação;
- Realizar Testes de aceitação da Ontologia por meio de um cenário hipotético;
- Verificar a corretude da ontologia por meio do motor de inferência.

[1] Proposta de revisão da Taxonomia original de Bloom feita por David Krathwohl e publicada em 2001. A estrutura da taxonomia original passou a ser bidimensional formada pelas dimensões Conhecimento e Processos Cognitivos.

## 1.2 JUSTIFICATIVA

O ensino e a aprendizagem de programação são considerados tarefas complexas e, como consequência, os cursos de programação frequentemente têm altas taxas de reprovação e desistência (PITEIRA e HADDAD, 2011). As ferramentas de auxílio existentes, somadas às pesquisas nesta área podem contribuir não apenas para minimizar os problemas de evasão e dificuldade do aprendizado de programação, como também para melhorar a qualidade do processo de ensino/aprendizagem.

Um dos principais pontos de se utilizar a Taxonomia Revisada de Bloom, como forma de avaliação de aprendizagem é que através dela é possível associar conteúdos em níveis de conhecimentos e a definição de objetivos educacionais desejados. Logo, tem-se uma metodologia que auxilia o educador no planejamento e na melhor elaboração de conteúdos, possibilitando também que eles tenham uma ampla visão do nível de entendimento de seus alunos (SANTOS, 2016). Podendo também ajudar os alunos a entenderem como percorrer o caminho em direção ao entendimento do assunto.

Araújo et al. (2013) afirmam que o ensino norteado pela taxonomia de Bloom foram fatores de sucesso no ensino de programação de maneira introdutória para alunos do ensino médio. Ela teve papel fundamental como instrumento para a análise e avaliação do ensino-aprendizagem dos alunos e também para orientação no planejamento e organização da didática utilizada, uma vez que a forma hierárquica dos domínios cognitivos apontou como construir um novo conhecimento e como avaliá-lo.

No que diz respeito à concepção de Ontologias, elas permitem o reuso do conhecimento de um domínio, ou seja, através da ontologia em que a taxonomia está modelada adequadamente, ela pode ser compartilhada e reutilizada.

Diante do exposto, este trabalho se justifica com o desenvolvimento de um artefato computacional baseado na Taxonomia Revisada de Bloom para auxiliar no processo de ensino-aprendizagem de programação.

## 1.3 METODOLOGIA

A pesquisa foi guiada pelos princípios do método Design Science Research (HEVNER et al. 2004), por meio de questões de pesquisa. De acordo com Hevner (2004) e colaboradores, os conhecimentos necessários para concretizar uma pesquisa envolvem dois paradigmas complementares: Ciência do Comportamento e Ciência do Design.

O paradigma da ciência do comportamento versa sobre a pesquisa pelo desenvolvimento de teorias que explicam ou predizem fenômenos relacionados à necessidade de negócio identificada. Já o paradigma da ciência do design, que será utilizado no presente trabalho, trata-se da pesquisa pelo desenvolvimento e avaliação de artefatos projetados para atender à necessidade de negócio identificada.

De acordo com Hevner et. al. (2004), pode-se apresentar uma questão geral de pesquisa, assim como decompor esta questão em questões secundárias. Primeiramente apresenta-se a questão geral de pesquisa (QGP):

**QGP -** Como uma ontologia pode ser estruturada para mapear questões de aprendizagem de algoritmos com a Taxonomia Revisada de Bloom?

Relacionadas à questão geral de pesquisa apresentada anteriormente, seguem as seguintes questões secundárias de pesquisa (QSP):

**QSP1** – Como cada uma das categorias da taxonomia podem ser interpretadas dentro do contexto da aprendizagem de programação?

**QSP2** – Como classificar questões de algoritmos com a Taxonomia Revisada de Bloom?

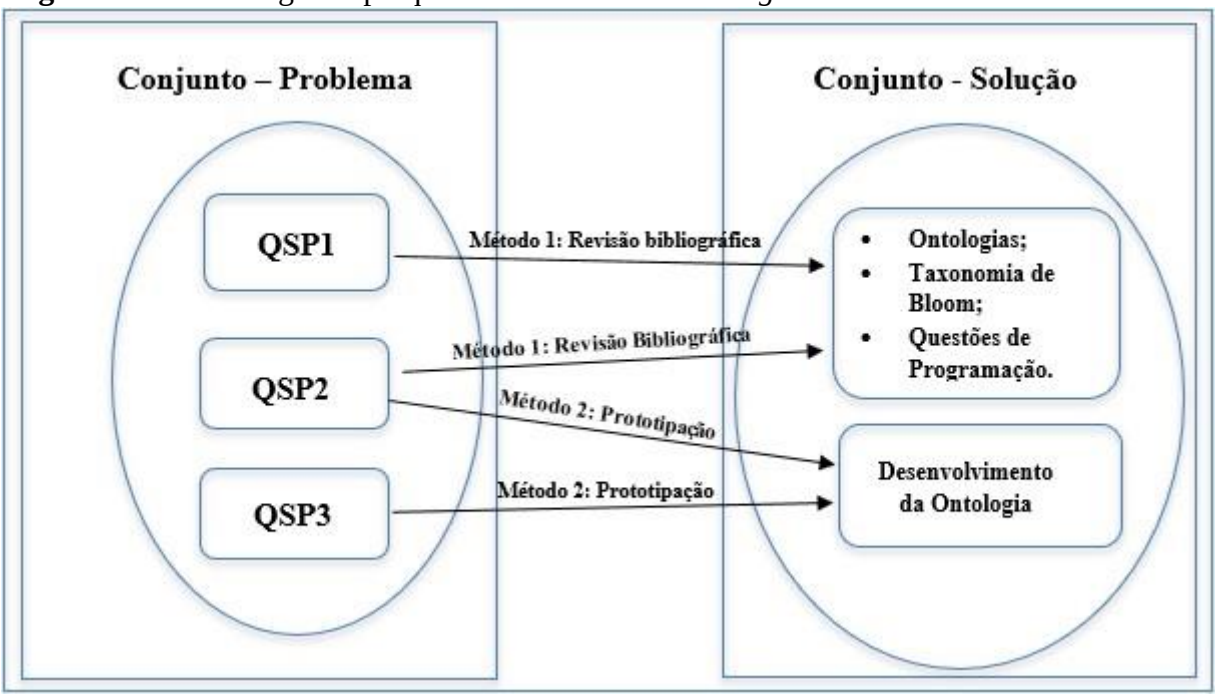
**QSP3** – Como permitir a classificação do aluno na ontologia?

Segundo Wieringa (2014), a pesquisa científica deve dispor de um conjunto de métodos específicos, por meio do qual o fluxo da investigação será norteado, partindo do espaço-problema ao espaço-solução, conforme ilustrado na Figura 1.

Dos métodos de pesquisa científica voltados para ciência da computação, foram utilizados dois, a saber:

- Revisão Bibliográfica da Literatura: tem como finalidade atualizar o pesquisador nas últimas discussões no campo de conhecimento em investigação, e identificar na literatura quais seriam os pilares teóricos que podem subsidiar a realização e finalização da pesquisa.
- Prototipação: em desenvolvimento de software, a prototipação pode ajudar a entender o negócio do cliente e consequentemente o propósito do software. Assim, obtém-se uma melhor definição dos requisitos do sistema, e caso necessário, pode-se propor melhorias minimizando riscos. A prototipação compreende as fases de análise e projeto do sistema.

**Figura 1 -** Metodologia da pesquisa de acordo com a *Design Science.*




**Fonte:** Elaborado pelo autor, 2019.

# 2 REFERENCIAL TEÓRICO

Este capítulo apresenta uma revisão bibliográfica com abordagem dos conteúdos que fundamentam o tema escolhido para o trabalho, e está organizado em três subseções. A primeira subseção descreve sobre trabalhos relacionadas a aprendizagem de programação, com ênfase no processo de ensino-aprendizagem, o qual inclui a avaliação dos alunos. A segunda subseção compreende uma pesquisa acerca da Taxonomia de Bloom, como foi criada, com qual objetivo, os níveis de conhecimento, tendo como foco a taxonomia revisada. A terceira subseção discute sobre ontologias, seus conceitos, componentes, tipos e metodologias de desenvolvimento. O capítulo é finalizado com uma sessão contendo uma explanação sobre trabalhos relacionados a presente pesquisa.

## 2.1. APRENDIZAGEM DE LINGUAGEM DE PROGRAMAÇÃO

A programação segundo Gomes, Henrique e Mendes (2008) é uma arte e uma ciência. Uma arte, pois envolve a criatividade para se pensar nas várias formas diferentes de codificar instruções. E também uma ciência, porque é constituída por um conjunto de regras orientadoras e é necessário o uso de lógica em métodos rigorosos de programação afim de assegurar a eficiência, economia e utilidade dos programas gerados.

Acerca disso, o ensino de algoritmos e programação é fundamental nos cursos da área tecnológica. Está presente nos primeiros semestres e é o passo inicial para o desenvolvimento do raciocínio lógico, e por consequência, para a introdução dos conceitos e práticas de programação. Basicamente, o propósito do ensino das linguagens de programação é conseguir que os alunos desenvolvam as suas capacidades, adquirindo os conhecimentos básicos necessários para conceber programas capazes de resolver problemas reais simples (GOMES, HENRIQUES E MENDES, 2008; RAABE E SILVA, 2005).

Bereiter e Ng (1991) concretizam que, a codificação implica na aprendizagem da sintaxe básica e gradualmente se aprenda a semântica, estrutura e finalmente o estilo, ou seja, trata-se de uma aprendizagem hierárquica de competências, em que geralmente começam pelas de mais baixo nível e vão progredindo gradativamente até às mais exigentes.

É importante ainda ressaltar que a apropriação ou não dos conceitos iniciais de programação tem relação direta com o desempenho do aluno no decorrer de todo o curso, já que disciplinas avançadas dependem fortemente desses conceitos (ROCHA et. al, 2010).

Apesar disso, normalmente, o ensino de programação é considerado desafiador para alunos e professores. Discute-se a seguir alguns trabalhos que apresentam as dificuldades no ensino/aprendizagem de programação.

### *2.1.1. Dificuldades no ensino/aprendizagem*

Ao realizar pesquisas na literatura observou-se que do ponto de vista dos alunos, como apontam os resultados observados no artigo de Queiroz et. al. (2018), os fatores motivacionais que estimulam os alunos a estudarem conteúdos de programação, segundo os próprios alunos da disciplina de algoritmos, incluem em sua maioria dificuldades na compreensão do enunciado da questão. Constatou-se, por exemplo, em relação às variáveis positivas, que os discentes envolvidos na pesquisa em questão, sentem-se mais motivados quando compreendem o enunciado da questão, absorvendo o conteúdo e conseguindo executar com êxito o código programado.

Moreira et. al. (2018), em sua pesquisa sobre os desafios na aprendizagem de programação, obtiveram as seguintes implicações: a maioria dos estudantes afirmaram possuir maior dificuldade no desenvolvimento da lógica de programação (42.72%), entendimento da sintaxe (34.54%), a falta de tempo para se dedicar a disciplina (26.36%) e dificuldade na interpretação das questões (15.45%).

Corroborando com isso, Gomes, Henriques e Mendes (2008), apontam os seguintes problemas na complexa tarefa de aprender a programar: **Métodos de estudo,** porque o ensino de muitos conceitos dinâmicos são, normalmente, realizados através de materiais de natureza estática como, apresentações projetadas, explicações verbais, diagramas, desenhos no quadro, textos, materiais que acabam por não promover uma plena compreensão da dinâmica envolvida; e a **Natureza específica do tipo de matéria** que diz respeito à rotina necessária para os estudos de programação que exigem prática intensiva, uma compreensão adequada dos assuntos e reflexão.

No que diz respeito aos desafios enfrentadas pelos docentes, um fator limitante apontado por Tobar et. al. (2001), Raabe e Silva (2005), Gomes, Henriques e Mendes (2008) e Rocha el. al (2010), é a complexa tarefa de identificar as dificuldades individuais de cada aluno, considerando uma turma com grande quantidade de alunos e o grande número de avaliações sugeridas pelas instituições de ensino, tornando o aprendizado passível de falhas. Por isso, é importante identificar os níveis de dificuldades dos educandos, pois há um significativo

aumento da dependência do entendimento dos conceitos básicos com as atividades mais complexas envolvendo programação, podendo acarretar assim, em dificuldades posteriores.

Na mesma linha, Souza, Batista, Barbosa (2016), apontam dificuldades dos professores em acompanhar a aprendizagem do aluno, desenvolver materiais e exercícios de apoio, bem como avaliar os trabalhos de programação. Esse fato acontece, pois, como enfatizou Pimentel et. al (2003), os critérios de avaliação utilizados pelas instituições não são satisfatórios, porque se um aluno adquirir média 5.0 numa disciplina não quer dizer que este aluno sabe 50% de todo o conteúdo que foi abordado.

Para dirimir essa situação, os professores devem procurar desenvolver um conjunto de aptidões para que seus alunos se mantenham motivados, dentre elas destacam-se a presteza quanto às atividades em geral e o *feedback* das atividades desenvolvidas (SALOMÃO e WATANABE, 2013). Diante disso, serão abordadas posteriormente algumas ferramentas desenvolvidas para auxiliar no processo de avaliação dos discentes.

### *2.1.2. Ferramentas desenvolvidas para auxiliar na aprendizagem de programação*

O desenvolvimento de esforços para identificação e atendimento adequado às dificuldades de aprendizagem comuns aos alunos desta disciplina é imprescindível e, com o passar dos anos foram desenvolvidos diversos tipos de sistemas computacionais para apoiarem o ensino e a aprendizagem de programação, quanto ao processo de avaliação dos discentes, pode-se citar: *Tutores Inteligentes*, *Juízes Online*, *Moodle* e *MOJO*, que serão melhor explanados a seguir:

- ***Tutores Inteligentes***

Um Sistema de Tutor Inteligente (STI) é segundo Silva, Fonseca e Silva (2015), "Um ambiente computacional de aprendizagem que possui modelos de conteúdos instrucionais que especificam o 'que' ensinar e estratégias de ensino que especificam 'como' ensinar".

Basicamente, os STIs procuram raciocinar sobre o domínio e as habilidades do estudante para então simular a atenção dada por um tutor humano (CARVALHO, 2017). O formato básico de um STI contém 5 módulos, que são: módulo do domínio, módulo do estudante, módulo de ensino, módulo do especialista e módulo de interface com o usuário.

Conforme Borges et. al. (2017) destaca, esses sistemas auxiliam no processo de identificação dos conteúdos programáticos, abordados no contexto da disciplina, e para a concepção do sistema que os autores desenvolveram, foi necessária a implementação de um

algoritmo que identifica as palavras chave utilizadas no ambiente da disciplina, em comparação com palavras previamente definidas. Este processo, segundo os autores, pode ser melhorado, através da utilização de ontologias.

- ***Juízes Online***

Os Juízes Online são ferramentas que fazem a avaliação automática de códigos. Basicamente, esses sistemas compilam, executam e testam códigos-fonte para julgar se estão corretos (ZHIGANG et al., 2012). Eles apresentam um banco de dados, com vários problemas a serem resolvidos. O usuário escolhe o problema que pretende solucionar, a linguagem de programação que quer utilizar e envia a solução para que possa ser avaliada. São facilmente encontrados online e são muito utilizados em competições de programação (CHAVES, 2014).

No que diz respeito à utilização dos juízes online no ensino da computação, eles oferecem aos alunos um feedback automático para as soluções dos problemas por eles propostos, assim como, acompanhar o desempenho de seus colegas, tornando assim, o trabalho do professor mais produtivo (FRANCISCO, JUNIOR e AMBRÓSIO, 2016).

Francisco, Junior e Ambrósio (2016) ainda classificaram em quatro grupos os requisitos funcionais que um juiz online precisa ter para ser usado como ferramenta de ensino: feedback, integração do sistema com os cursos, análise do desempenho geral dos alunos, e oferecer diferentes tipos de atividades. Quanto aos requisitos não-funcionais incluem: integração, usabilidade, segurança, escalabilidade, disponibilidade. Existem sistemas de Juízes Online que estão sendo produzidos e usados no Brasil com bastante popularidade, como o *BOCA* e o *URI*.

- ***Moodle***

O Moodle *(Modular Object-Oriented Dynamic Learning Environment)* é um Ambiente Virtual de Aprendizagem, que fornece a interface e um conjunto de funcionalidades necessárias à gestão e ao acompanhamento das atividades de programação. Essa plataforma possui uma comunidade formada por professores, pesquisadores, e, principalmente, programadores, que mantém um Portal na Web onde funciona como uma central de informações, discussões e colaborações e ainda oferece aos professores e alunos um ambiente capaz de reunir a maioria das informações e eventos relevantes, associados a uma disciplina de determinado curso (CHAVES, 2014).

- ***MOJO***

O MOJO *(Módulo de Integração com os Juízes Online)* é uma ferramenta proposta por Chaves et. al. (2013) que foi criada para integrar juízes online ao *Moodle*, a fim de auxiliar o professor no processo ESA (Elaboração, Submissão, Avaliação) de atividades de programação; acompanhar os resultados através de uma mesma interface disponível no ambiente virtual; proporcionar um feedback mais rápido ao aluno; além de oferecer suporte a diversas linguagens de programação.

A integração Moodle-MOJO, segundo Borges et. al. (2017), além de proporcionar a disponibilização de atividades através da plataforma Moodle e correção por juízes online, fornece base para um leque de funcionalidades que venham a automatizar o trabalho docente. De acordo com Chaves (2014), ao utilizar essa estratégia, os alunos podem ter melhor desempenho visto que recebem um retorno rápido com relação a avaliação das atividades.

Diante disso, percebe-se a importância de instrumentos que venham a auxiliar no processo de avaliação dos alunos de programação. E embora as ferramentas mencionadas ofereçam um ambiente motivador para a aprendizagem de programação, outros fatores são extremamente importantes, como a assistência especializada, metodologias empregadas no ensino e avaliação, entre outras. Conforme o trabalho de Jesus e Raabe (2009), a Taxonomia de Bloom é um conceito que pode ser aplicado na disciplina de Programação com bons resultados. Uma ontologia notada semanticamente deste domínio pode ser integrada a uma ferramenta, como as mencionadas até aqui.

## 2.2. TAXONOMIA DE BLOOM

Esta seção aborda conceitos importantes sobre a Taxonomia de Bloom em sua versão tradicional e os Objetivos Educacionais, além da importância do Planejamento no processo de ensino e aprendizagem.

Com o passar do tempo e em decorrência dos avanços tecnológicos, associado às dificuldades inerentes do processo de ensino e aprendizagem, se faz necessário retomar estudos da metade do século passado, a fim de evoluir e desenvolver este processo. Muitos são os instrumentos existentes para apoiar o planejamento didático-pedagógico, a estruturação, a organização, a definição de objetivos instrucionais e a escolha de instrumentos de avaliação. Um dos mais conhecidos e utilizados é a taxonomia de Bloom.

De acordo com Krathwohl (2002), Benjamin S. Bloom juntamente com uma comissão multidisciplinar de especialistas dos EUA criaram a Taxonomia com o propósito de buscar uma

forma de facilitar o intercâmbio de questões de testes entre professores de várias universidades, que em cada questão era realizada a avaliação do mesmo objetivo de aprendizagem e a partir disso buscavam promover formas mais elevadas de pensamento na educação.

A taxonomia de Bloom é uma estrutura conceitual concebida para auxiliar a definição de objetivos de aprendizagem. O objetivo era a criação de um modelo que fizesse todos os professores do país adotarem uma compreensão homogênea e prática, que fosse passível de generalização quanto ao planejamento e à avaliação (LIMA, 2009). Embora este seja um instrumento adequado para utilização no ensino superior, poucos educadores fazem uso dele por não conhecerem uma maneira adequada de utilizá-lo (NÄSSTRÖM, 2009; FERRAZ e BELHOT, 2010).

Apesar disso, a Taxonomia de Bloom é um referencial para classificar afirmações sob as quais se espera que os alunos aprendam como resultado da instrução (KRATHWOHL, 2002). A elaboração dos objetivos educacionais oferece o norteamento para o desenvolvimento de práticas e atividades que estimulem a aprendizagem cognitiva do aluno de maneira progressiva (OLIVEIRA, 2018).

Bloom et al. (1956) definem que os objetivos educacionais são divididos pela taxonomia de acordo com o domínio específico de desenvolvimento cognitivo, afetivo e psicomotor. O Quadro 1 apresenta um resumo das características de cada um dos domínios.

**Quadro 1 -** Características dos três domínios da taxonomia.

| **DOMÍNIO** | **CARACTERÍSTICAS** (Continua) |
|---|---|
| **Cognitivo** | ● Relacionado ao aprender, dominar um conhecimento.<br>● Envolve a aquisição de um novo conhecimento, do desenvolvimento intelectual, de habilidade e de atitudes.<br>● Inclui reconhecimento de fatos específicos, procedimentos padrões e conceitos que estimulam o desenvolvimento intelectual constantemente.<br>● Nesse domínio, os objetivos foram agrupados em seis categorias e são apresentados numa hierarquia de complexidade e dependência (categorias), do mais simples ao mais complexo.<br>● As categorias desse domínio são: Conhecimento; Compreensão; Aplicação; Análise; Síntese; e Avaliação; |

| | |
|---|---|
| **Afetivo** | (Conclusão)<br>● Relacionado a sentimentos e posturas.<br>● Envolve categorias ligadas ao desenvolvimento da área emocional e afetiva, que incluem comportamento, atitude, responsabilidade, respeito, emoção e valores.<br>● As categorias desse domínio são: Receptividade; Resposta; Valorização; Organização; e Caracterização; |
| **Psicomotor** | ● Relacionado a habilidades físicas específicas.<br>● Bloom e sua equipe não chegaram a definir uma taxonomia para a área psicomotora, mas outros o fizeram e chegaram a seis categorias que incluem ideias ligadas a reflexos, percepção, habilidades físicas, movimentos aperfeiçoados e comunicação não verbal.<br>● As categorias desse domínio são: Imitação; Manipulação; Articulação; e Naturalização. |

**Fonte:** Ferraz e Belhot (2010).

Para ascender a uma nova categoria é preciso ter obtido um desempenho adequado na anterior, pois cada uma utiliza capacidades adquiridas nos níveis anteriores para serem aprimoradas (BLOOM, 1956). Apesar dos três domínios (cognitivo, afetivo e psicomotor) terem sido amplamente discutidos e divulgados, em diferentes momentos e por diferentes pesquisadores, o domínio cognitivo é o mais conhecido e utilizado, portanto somente ele será abordado neste trabalho.

Desde a sua criação, a taxonomia passou por mudanças, a fim de sempre inovar e estar acompanhando a evolução do processo de ensino e aprendizagem. Após 45 anos de existência, surgiu a necessidade de atualizações em sua estrutura conceitual. Anderson e colaboradores revisaram a versão original da Taxonomia de Bloom para o domínio cognitivo. Depois da revisão a taxonomia passou de uma estrutura com apenas uma dimensão para uma estrutura com duas dimensões: Conhecimento e Processo Cognitivo (KRATHWOHL, 2002).

Assim sendo, a próxima seção apresenta a Taxonomia Revisada de Bloom com suas significativas mudanças. É importante destacar que o domínio desse trabalho será norteado por um artefato baseada nesse princípio.

### *2.2.1. Taxonomia Revisada de Bloom*

Apesar da taxonomia original ainda ser a mais utilizada, há vários autores que a criticaram e apontaram dificuldades em seu uso, relatando que as categorias nem sempre são fáceis de serem aplicadas, pois existe uma significativa sobreposição entre elas e que ocasiona debates sobre a ordem em que as categorias *análise, síntese e avaliação* aparecem na hierarquia. Ferraz e Belhot (2010) mencionam que muitos trabalhos foram originados a partir da primeira divulgação da Taxonomia de Bloom no domínio cognitivo, porém com as novas publicações e com tecnologias incorporadas ao sistema educacional foi necessário ser feita uma reavaliação e releitura dos pressupostos teóricos que sustentaram a pesquisa original para avaliação da necessidade de adaptações.

Dentre as muitas versões revisadas que foram feitas, a mais utilizada é a de David Krathwohl (2002), que inclusive participou do desenvolvimento da taxonomia original em 1956, foi ele que supervisionou o grupo de especialistas (psicólogos, educadores, especialistas em currículos, testes, avaliação etc.) e publicou o relatório da revisão da taxonomia em 2001. Esse grupo tentou buscar o equilíbrio entre o que existia, a estruturação da taxonomia original e os novos desenvolvimentos incorporados à educação nos quarenta e poucos anos de existência (FERRAZ e BELHOT, 2010).

Corroborando com isso, Ferraz e Belhot (2010), Nasström (2009), Santos (2016) citam as mudanças feitas na taxonomia revisada proposta por Krathwohl (2002). A taxonomia original passou de caráter unidimensional para bidimensional, onde houve a separação de substantivos e verbos, conhecimento e aspectos cognitivos. Cada uma das partes da estrutura bidimensional foi nominada como Dimensão Conhecimento e Dimensão dos Processos Cognitivos (KRATHWOHL, 2002).

A dimensão do conhecimento é formada por quatro tipos: Factual, relacionado aos elementos básicos que os educandos devem saber para se familiarizar com a disciplina para solucionar problemas nela; Conceitual, consiste em conhecer às inter-relações entre elementos básicos de uma estrutura maior que permite-os funcionar juntos; Procedural, é o conhecimento de como fazer algo, métodos de questionamentos; critérios para utilização de habilidades, algoritmos, técnicas e métodos; e Metacognitivo, relacionado ao reconhecimento da cognição em geral e da consciência da amplitude e profundidade de conhecimento adquirido de um determinado conteúdo.

Quanto aos objetivos cognitivos da aprendizagem, segundo Krathwohl (2002), ela abrange as cinco categorias da taxonomia original, porém renomeadas para suas formas verbais.

A categoria Conhecimento tornou-se Lembrar; Compreensão tornou-se Entender; Síntese tornou-se Criar (e foi promovida para a categoria mais alta da hierarquia); Aplicação, Análise e Avaliação tornaram-se respectivamente Aplicar, Analisar e Avaliar. Esses níveis serão melhor abordados na próxima sessão, num contexto da programação introdutória.

### *2.2.2. Taxonomia dos objetivos Cognitivos aplicada a programação introdutória*

O domínio Cognitivo envolve o conhecimento e o desenvolvimento de habilidades intelectuais (BLOOM, 1956). Muitos educadores se apoiam nos pressupostos teóricos desse domínio para definirem, em seus planejamentos educacionais, objetivos, estratégias e sistemas de avaliação.

Existem verbos associados a cada um dos níveis da taxonomia. Estes verbos auxiliam na classificação de uma questão de avaliação em um dos níveis da taxonomia. O Quadro 2 apresenta os níveis da taxonomia revisada e seus respectivos verbos:

**Quadro 2 -** Níveis da Taxonomia Revisada e seus respectivos verbos.

| 1 - Lembrar | 2- Entender | 3- Aplicar | 4- Analisar | 5- Avaliar | 6- Criar |
|---|---|---|---|---|---|
| Apontar | Classificar | Computar | Atribuir | Apreciar | Criar |
| Definir | Comparar | Construir | Categorizar | Criticar | Desenvolver |
| Escrever | Exemplificar | Demonstrar | Comparar | Julgar | Elaborar Hipóteses |
| Listar | Explicar | Executar | Contrastar | Justificar | Gerar |
| Nomear | Inferir | Implementar | Diferenciar | Ponderar | Inventar |
| Reconhecer | Interpretar | Resolver | Organizar | Recomendar | Planejar |
| Relembrar | Sumarizar | Utilizar | Separar | Verificar | Produzir |

**Fonte:** Galhardi e Azevedo (2013).

Jesus e Raabe (2009) em seu estudo apresentam estes verbos e como eles são especificamente interpretados no contexto da programação introdutória, classificando questões de acordo com os níveis da taxonomia. O quadro 3 apresenta uma síntese da interpretação da Taxonomia de Bloom em programação.

**Quadro 3** - Síntese da Interpretação da Taxonomia de Bloom em programação.

| **Categoria** | **Interpretação da Categoria em Programação**<br>(Continua) |
|---|---|
| Lembrar | *Recuperação de conhecimento relevante da memória de longo termo.*<br><br>Identificar elementos específicos em um trecho de código.<br>Reconhecer a implementação de um determinado conceito.<br>Reconhecer a descrição mais apropriada para um determinado conceito.<br>Lembrar de um conceito, processo, algoritmo, etc.<br>Listar operadores de acordo com a ordem de precedência.<br>Definir o propósito de um método construtor.<br>Descrever um determinado padrão de projeto.<br>Citar os nomes dos tipos de loops em uma linguagem de programação.<br>Listar N métodos que executem operações de entrada e saída de dados. |
| Entender | *Construção de significados através de diferentes tipos de linguagens.*<br><br>Escrever em pseudocódigo, fluxograma ou linguagem natural um programa que calcule uma fórmula bem conhecida.<br>Completar partes faltantes de um programa utilizando fragmentos de código.<br>Explicar com palavras o comportamento de um trecho de código.<br>Predizer valores de variáveis depois da execução de um trecho de código.<br>Traduzir um algoritmo de uma forma de representação para outra.<br>Explicar um conceito, algoritmo ou padrão de projeto.<br>Apresentar exemplos de um conceito, algoritmo ou padrão de projeto. |
| Aplicar | *Utilização de processos conhecidos para executar ou implementar.*<br><br>Implementar um programa utilizando como exemplo um código que resolva um problema semelhante.<br>Implementar ordenação de vetores não numéricos com alunos que já tenham ordenado vetores numéricos.<br>Executar mentalmente expressões seguindo as regras de precedência.<br>Resolver um problema familiar, mas com dados ou ferramentas não familiares.<br>Modificar o código de um loop do tipo for para um do tipo *while*. |
| Analisar | *Decomposição de um problema em suas partes constituintes e determinação das relações entre as partes e o todo.*<br><br>Dividir uma tarefa de programação em suas partes componentes.<br>Organizar as partes componentes para atingir um objetivo geral.<br>Identificar componentes críticos para o desenvolvimento.<br>Identificar componentes ou requisitos não importantes.<br>Diferenciar um método construtor dos demais métodos de uma classe. |
| Avaliar | *Realização de julgamentos baseados em critérios e padrões.*<br><br>Determinar se um código satisfaz os requisitos definindo uma estratégia de teste apropriada. |

| | (Conclusão) |
|---|---|
| Avaliar | Criticar a qualidade de um código baseando-se em boas práticas de programação ou critérios de eficiência do código.<br>Avaliar qual de dois algoritmos que resolvem a mesma tarefa é mais adequado.<br>Encontrar um erro de lógica em um trecho de código dado. |
| Criar | *Juntar elementos para formar um todo coerente e funcional.*<br><br>Propor algoritmo, processo ou estratégia alternativa para um problema.<br>Hipotetizar que uma nova combinação de algoritmos resolverá o problema<br>Construir um programa utilizando algoritmos inventados.<br>Aplicar algoritmos conhecidos em uma combinação não familiar para o aluno. |

**Fonte:** Jesus e Raabe (2009).

A Taxonomia de Bloom auxilia o professor na decisão sobre qual ação ou comportamento o aluno deverá desempenhar para realizar determinada atividade e alcançar o objetivo proposto, bem como a organizar um conteúdo de maneira coerente, seja no ensino presencial ou à distância. Mesmo mantendo parte da estrutura original, a taxonomia revisada é mais adequada para suportar as novas formas de aprendizagem e consequentemente tirar melhores proveitos de objetivos educacionais (SANTOS, 2016).

Diante disso, alguns autores têm relatado dificuldades em mapear a aprendizagem e a elaboração de avaliações num contexto de programação introdutória. Jesus e Raabe (2009) apresentaram um instrumento de avaliação onde questões foram classificadas segundo a Taxonomia Revisada de Bloom.

Já Oliveira (2018) afirma que ontologias são utilizadas para definir e descrever os domínios de objetos e conceitos, como é o caso dos objetos de aprendizagem e taxonomia de Bloom. E também cita a utilização de ontologias para modelar ou classificar o aprendizado de alunos em repositórios de objetos de aprendizagem, ambientes virtuais de aprendizagem e em jogos educacionais, por exemplo. A seção 2.3 delineia e conceitua ontologias.

## 2.3. ONTOLOGIA

O termo ontologia surgiu na Filosofia e, foi inicialmente utilizado pela área de Inteligência Artificial no início da década de 1990. Diversas pesquisas nessa área são direcionadas ao desenvolvimento de soluções computacionais capazes de incorporar conhecimentos sobre determinado domínio, permitindo inferências, raciocínios e tomada de decisões. A partir disso, passaram a ser utilizadas em diferentes áreas que buscam desenvolver

um vocabulário contendo os conceitos relativos ao domínio da aplicação (NEVES e COELLO, 2006).

Diversas definições podem ser encontradas na literatura para Ontologias em Ciência da Computação. Uma das mais conhecidas é dada por Gruber (1993), ele afirma que "*Uma ontologia é uma especificação explícita de uma conceitualização"*, ou seja, é uma representação de uma visão abstrata e simplificada do mundo para algum propósito e consiste de um conjunto de objetos, conceitos e outras entidades sobre as quais o conhecimento está sendo expresso, e dos relacionamentos entre eles.

Segundo Guarino (1998), ontologia refere-se a um artefato constituído por um vocabulário usado para descrever uma realidade específica, seguido de um conjunto de fatos explícitos e aceitos que dizem respeito ao sentido requerido para as palavras do vocabulário.

Desta forma, a taxonomia de Bloom pode ser representada por uma ontologia, principalmente por se tratar de uma estrutura hierárquica (OLIVEIRA, 2018). Assim como, questões de programação podem ser classificadas por meio de uma ontologia em um dos níveis da taxonomia.

Neves e Coello (2006) e Couto (2018) afirmam que a vantagem desses sistemas, além da representação simbólica do conhecimento, é a de poder estar separada dos aspectos relacionados à aplicação. Corroborando com isso, Guizzard (2000) atesta que Ontologias fornecem o conhecimento estruturado e uma infraestrutura para integrar bases de conhecimentos, independente da implementação e constituem uma ferramenta poderosa para suportar a especificação e a implementação de sistemas computacionais de qualquer complexidade.

Com base nessas definições, Noy e McGuinness (2001) apontam vantagens do uso de ontologias em sistemas de informação:

1. Permite o compartilhamento do entendimento comum da estrutura de informação de um domínio entre pessoas ou agentes de software;
2. Permite o reuso do conhecimento de domínio; caso exista uma ontologia que modele adequadamente certo conhecimento de um domínio, ela pode ser compartilhada e usada por pessoas que desenvolvam aplicações nesse e em outros domínios;
3. Tornar as suposições do domínio explícitas. As ontologias fornecem um vocabulário para representação do conhecimento. Esse vocabulário tem por trás uma conceitualização que o sustenta, evitando assim interpretações ambíguas;
4. Separar o conhecimento declarativo do domínio do conhecimento operacional/ procedural (utilizado para manipular o conhecimento declarativo); e

5. Fornecer uma descrição exata do conhecimento, possibilitando assim uma conceitualização comum das palavras e a análise do conhecimento de um domínio.

A próxima seção versa sobre os componentes básicos de uma ontologia.

### *2.3.1. Componentes*

Segundo Gruber (1993), as ontologias não apresentam sempre a mesma estrutura, mas possuem algumas características e componentes comuns bem definidos. Como componentes básicos de uma ontologia tem-se: *classes*, *relações*, *funções*, *axiomas* e *indivíduos.*

*Classes ou conceitos* organizam os conceitos de um domínio em uma taxonomia. São, muitas vezes, o foco das ontologias. Dentro de uma classificação hierárquica as classes podem se subdividir em superclasses e subclasses, consecutivamente, conforme for a especificação. Uma subclasse herda as propriedades de sua superclasse.

*Relações ou propriedades* representam o tipo de interação entre os conceitos de um domínio. Existem dois tipos de propriedades, *Object Properties* (qualifica ou relaciona as classes) e *Datatype Property* (instancia uma classe).

*Funções* são um caso especial de relação. Nelas, temos que um conjunto de elementos tem uma relação única com um outro elemento.

*Axiomas* são usados para modelar frases que são sempre verdade. Restringem a interpretação e o uso dos conceitos envolvidos na ontologia.

*Indivíduos ou instâncias* são elementos específicos de uma ontologia, que representam objetos no domínio de um determinado problema, ou seja, são seus próprios dados;

### *2.3.2. Tipos de Ontologias*

As ontologias podem ser classificadas de acordo com o nível de generalização. Dessa forma, Guarino (1997) propõe as seguintes classificações:

- ***Ontologias Genéricas (Alto-Nível)***

São consideradas ontologias “gerais”. Descrevem conceitos mais amplos relacionados a elementos naturais tais como, espaço, tempo, casualidades, eventos, processos ou ações, independentemente de um problema específico ou domínio particular. Pesquisas com ontologias genéricas procuram construir teorias básicas do mundo, de caráter bastante abstrato, aplicáveis a qualquer domínio (conhecimento de senso comum).

- ***Ontologias de Domínio***

Descrevem conceitos e vocabulários sistematizados de termos relacionados a domínios particulares e específicos, tais como medicina ou computação, por exemplo. Este é o tipo de ontologia mais comum, geralmente construída para representar um “micromundo”.

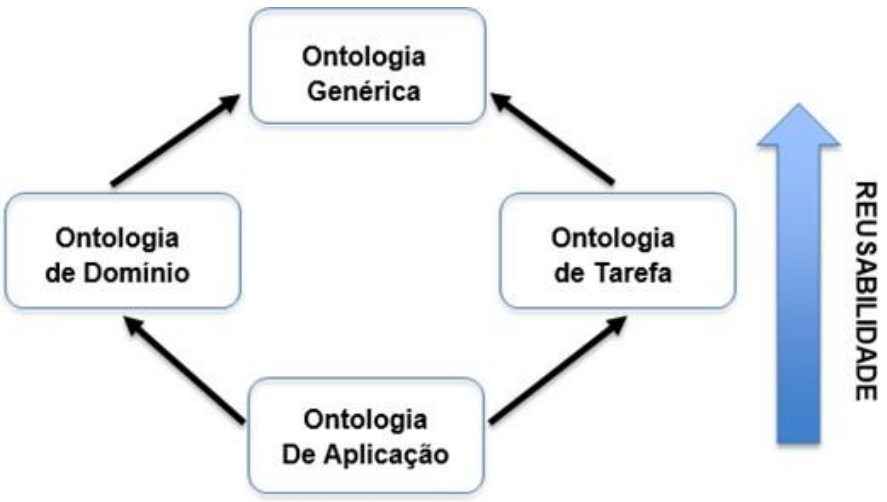
- ***Ontologias de Tarefas***

Descrevem tarefas ou atividades genéricas, que podem contribuir na resolução de problemas, independente do domínio que ocorrem, por exemplo, processos de vendas ou diagnóstico. Sua principal motivação é facilitar a integração dos conhecimentos de tarefa e domínio em uma abordagem mais uniforme e consistente.

- ***Ontologias de Aplicação***

Essas ontologias descrevem conceitos das ontologias de domínio e das ontologias de tarefas, que são em muitas vezes especializadas em ambas as ontologias. Essas ontologias contém as definições necessárias à aplicação para solucionar um problema específico de uma tarefa num dado domínio.

**Figura 2 -** Diferentes tipos de ontologias e suas relações.




**Fonte:** Adaptado Guarino (1998).

Na Figura 2 é possível observar o grau de reuso das ontologias. Por exemplo, ontologias de aplicação possuem menor capacidade de reuso, pois tratam-se de ontologias para aplicações específicas. Com relação a ontologias genéricas, elas são de alto nível, e apresentam maior

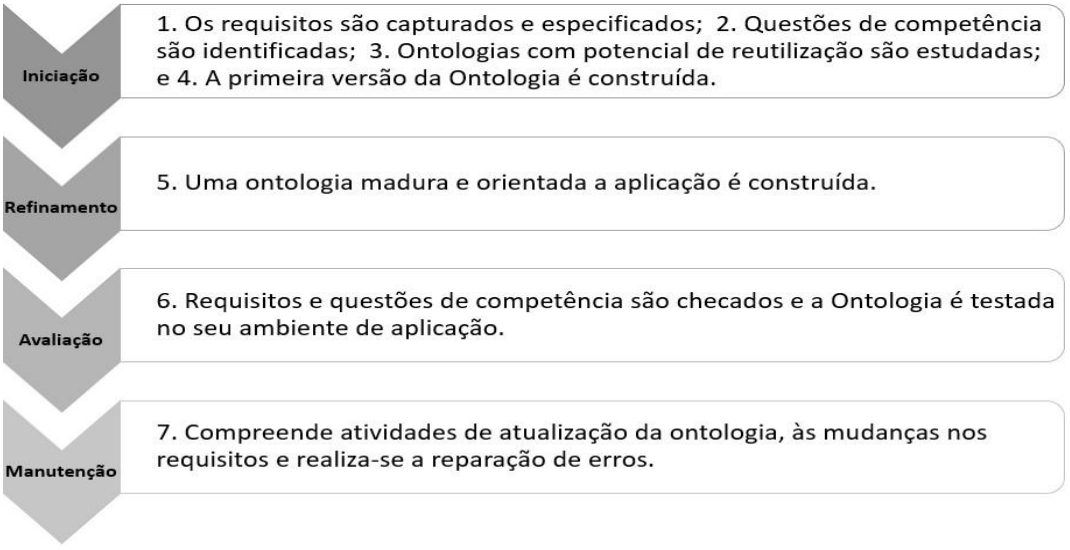
capacidade de reuso. Já as ontologias de domínio e de tarefa ficam no mesmo nível, pois especializam os conceitos introduzidos na ontologia de alto-nível.

### *2.3.3. Metodologias para o desenvolvimento de ontologias*

Conforme Noy e McGuinness (2001) afirmam, não há uma maneira correta de modelar um domínio, há sempre alternativas viáveis. A melhor solução sempre depende dos propósitos e objetivos para a construção de cada ontologia. Mas, apesar disso, pode- se encontrar ontologias desenvolvidas por vários grupos com diferentes abordagens e metodologias. No presente trabalho serão apresentadas apenas duas metodologias de construção de ontologias que se mostraram mais recorrentes na literatura.

A metodologia On-to-Knowledge, proposta por Staab et al. (2001), foi baseada em aplicações de gestão do conhecimento e mineração de dados. Faz o uso de questões de competência para determinar, de maneira simplificada, o escopo da ontologia e as suas principais características, bem como as propriedades, relações e instâncias. Esta metodologia possui quatro fases: *iniciação*, *refinamento*, *avaliação* e *manutenção*, como mostra a Figura 3.

**Figura 3 -** Fases da metodologia *On-to-Knowledge.*




**Fonte:** Couto (2018).

A segunda metodologia foi proposta por Noy e McGuinness (2001), é considerada um modelo de referência bastante utilizado em projetos de desenvolvimento de ontologias, por sua simplicidade, sendo composta por 7 passos, baseados no processo iterativo, em que se realiza várias iterações até alcançar o modelo pretendido. A descrição dos passos é feita a seguir:

1. **Determinar o domínio e o escopo da ontologia:** nesta etapa, os desenvolvedores deverão estabelecer algumas questões básicas como: – Qual o domínio que a ontologia vai cobrir? – Para que será usada a ontologia? – As informações contidas na ontologia deverão responder quais tipos de questões? – Quem irá usar e manter a ontologia? As respostas destas questões servirão para a compreensão do propósito da construção da ontologia.
2. **Considerar o reuso de ontologias existentes:** há muitas ontologias disponíveis em formato eletrônico, portanto é importante verificar se há outros artefatos referentes a área do domínio, afim de evitar desenvolver ontologias com propósitos redundantes.
3. **Enumerar os termos importantes da ontologia:** Nessa etapa, é construído uma lista com todos os termos relacionados ao domínio da ontologia com a finalidade de nomear os conceitos, identificar suas propriedades e as relações desejadas.
4. **Definir as classes e as hierarquias:** A partir dos termos definidos na etapa anterior, são definidas as classes juntamente com suas hierarquias. Existem algumas estratégias para o desenvolvimento da hierarquia: *top-down* (de cima para baixo) inicia-se com as definições mais gerais, passando-se em seguida para as mais específicas. A *bottom-up* começa com definições de classes mais específicas e vai agrupando em classes mais gerais. Já a estratégia *combinação*: é a junção dos dois métodos anteriores.
5. **Definir as propriedades das classes (*slots*)**: nessa fase são definidas as propriedades inerentes às classes e é necessário fazer a descrição da estrutura interna dos conceitos da ontologia.
6. **Definir as facetas dos slots:** definição das restrições das propriedades. Deve-se descrever os tipos de valores e a cardinalidade.
7. **Criar instâncias:** o último passo é criar as instâncias individuais das classes a partir das definições presentes na hierarquia.

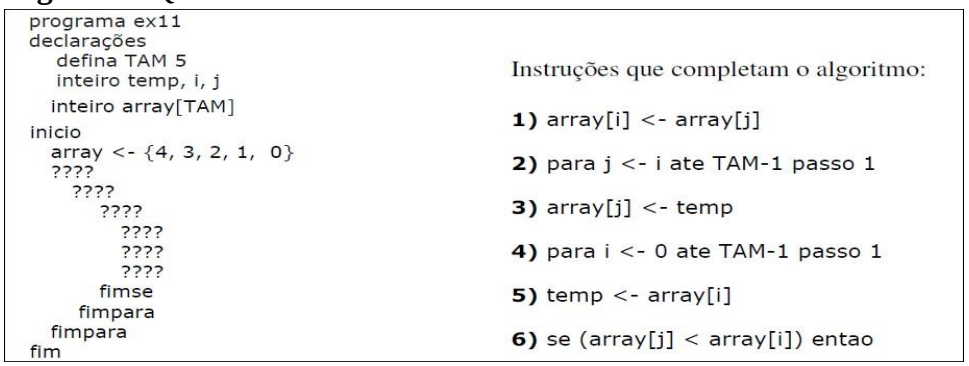
   A próxima seção, 2.4, aborda trabalhos relacionados.

## 2.4. TRABALHOS RELACIONADOS

Alguns autores realizaram estudos sobre a aplicabilidade da Taxonomia de Bloom no contexto de programação, assim como, o desenvolvimento de ontologias para contribuir na criação de ferramentas que venham a auxiliar o processo de aprendizagem de programação. Podem ser destacados três trabalhos, Jesus e Raabe (2009), Oliveira (2018) e Nascimento (2018).

O primeiro trabalho, com o seguinte título: *"Interpretações da Taxonomia de Bloom no Contexto da Programação Introdutória"*, trata-se de um artigo proposto por Jesus e Raabe (2009), ele discute sobre como cada uma das categorias da taxonomia vêm sendo interpretadas e utilizadas em avaliações de programação. O objetivo era apresentar um instrumento de avaliação em que questões de programação foram classificadas de acordo com a taxonomia revisada de Bloom, sendo assim, foram apresentados exemplos de como as categorias da taxonomia podem ser interpretadas e utilizadas em testes de programação, e com isso possibilitar que outros professores e pesquisadores tenham um instrumento padrão para utilizar em experimentos, a fim de comparar desempenhos entre turmas de alunos diferentes. Eles ainda apontam que o instrumento criado possivelmente não se adequará a todas as realidades educacionais, entretanto, ele servirá como base para a construção de outros instrumentos. A Figura 4 apresenta um exemplo de uma das questões que foram classificadas segundo a taxonomia. A questão pertence ao nível Analisar, e apresenta um trecho de código que ordena um *array* em ordem ascendente, porém o algoritmo está incompleto e o aluno deve indicar a sequência de instruções correta que o completa. Para isso, ele deve ser capaz de indicar dentre quatro opções qual é a sequência.

**Figura 4 -** Questão classificada no nível Analisar da taxonomia.



```
programa ex11
declarações
   defina TAM 5
   inteiro temp, i, j
   inteiro array[TAM]
inicio
   array <- {4, 3, 2, 1,  0}
   ????
      ????
         ????
            ????
            ????
            ????
         fimse
      fimpara
   fimpara
fim
```

Instruções que completam o algoritmo:

**1)** array[i] <- array[j]

**2)** para j <- i ate TAM-1 passo 1

**3)** array[j] <- temp

**4)** para i <- 0 ate TAM-1 passo 1

**5)** temp <- array[i]

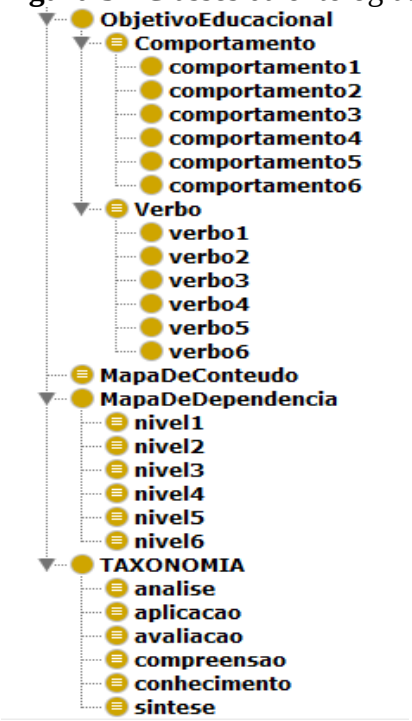
**6)** se (array[j] < array[i]) entao

**Fonte:** Jesus e Raabe (2009).

O segundo trabalho, uma dissertação de mestrado, intitulada "*Uma Ontologia para Classificação de Objetos de Aprendizagem Considerando o Domínio Cognitivo da Taxonomia de Bloom",* foi desenvolvida por Oliveira (2018) e apresentou uma ontologia para objetos de aprendizagem, considerando os níveis hierárquicos do domínio cognitivo da taxonomia de Bloom. O objetivo foi desenvolver uma ontologia para objetos de aprendizagem e taxonomia de Bloom, com o propósito de auxiliar na dinâmica da aprendizagem em ambientes virtuais a partir da classificação correta dos objetos de aprendizagem, considerando seus objetivos educacionais e possibilitando assim o acompanhamento do desenvolvimento cognitivo do aluno

a partir de sua classificação na taxonomia de Bloom. A avaliação da utilidade da ontologia foi aplicada em um contexto real de aplicação, através da associação da mesma ao ambiente virtual K-hunter (SILVA, 2018). Através de um estudo de caso verificou-se o potencial de aplicação da ontologia e sua contribuição para sistemas que utilizam objetos de aprendizagem na promoção da aprendizagem. A Figura 5 apresenta parte da ontologia que foi criada, contendo suas classes e subclasses.

**Figura 5 -** Classes da ontologia.

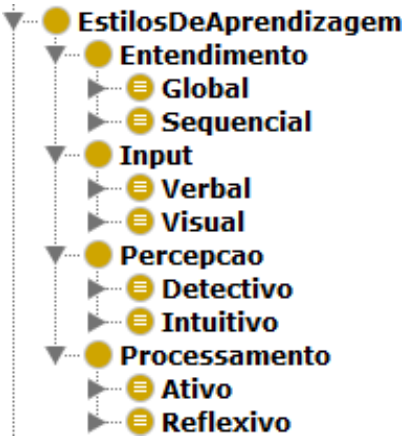



**Fonte:** Oliveira (2018).

O terceiro trabalho trata-se de uma monografia, cujo o título é "*Ensino De Programação Com O Auxílio De Sistemas De Tutores Inteligentes e Ontologias",* escrita por Nascimento (2018). A autora procurou formas de entender as características individuais dos alunos referentes ao aprendizado de conteúdos, estilos e formas de aprendizagem distintas. Para isso, se fez necessário realizar um mapeamento a fim de levantar as diferentes características e necessidades de cada aluno, assim como, os estilos de aprendizagem e abordagens de ensino que mais satisfizessem suas características. Então, o objetivo principal definido era a construção de um modelo computacional por meio de Ontologias que mapeiam os alunos e também a disciplina de Programação Orientada a Objetos (POO), do curso de Sistemas de Informação da Universidade Federal de Santa Catarina, possibilitando assim a recomendação de matérias, assuntos e exercícios específicos de acordo com cada perfil de aluno. Após o desenvolvimento

da ontologia, foram realizadas inferências dos estilos de aprendizagem dos alunos, com base nas informações colhidas por questionários, na própria ferramenta de desenvolvimento de ontologia, *Protégé*, e através de uma consulta *SPARQL*, retornou as recomendações de ferramentas e exercícios específicos que foram inferidos ao aluno, demonstrando assim a corretude do protótipo baseado em ontologia. A figura 6 ilustra uma das classes da ontologia, a classe "Estilos de Aprendizagem" que possui as subclasses "Entendimento", "Input", "Percepção" e "Processamento" os quais representam as dimensões do modelo Felder-Silvermann.

**Figura 6:** Classe Estilos de Aprendizagem.




**Fonte:** Nascimento (2018).

O próximo capítulo apresenta o processo de desenvolvimento da ontologia proposta pelo presente trabalho.

# 3 PROPOSTA DE UMA ONTOLOGIA PARA CLASSIFICAÇÃO DE QUESTÕES DE ALGORITMOS SEGUNDO A TAXONOMIA REVISADA DE BLOOM

Neste capítulo será apresentado o processo de desenvolvimento da ontologia para classificação de questões de algoritmos e alunos segundo a taxonomia revisada de Bloom, como a definição das classes, propriedades e indivíduos. Ao final, tem-se a visualização da ontologia por meio da *WebVOWL* e para a validação do artefato foram utilizadas consultas em *SPARQL* (Protocol and RDF Query Language). O processo metodológico para a concepção desta ontologia é apresentado na seção seguinte.

## 3.1. ESTRUTURA DA ONTOLOGIA

Neste capítulo serão abordadas as etapas do processo de desenvolvimento da ontologia que classifica questões de algoritmos segundo a Taxonomia Revisada de Bloom.

Como metodologia para construção da ontologia, foram seguidas as etapas definidas pelo guia 101, já apresentadas no referencial teórico, na seção 2.3.3. Esta metodologia segundo Noy e McGuinness (2001), utiliza passos iterativos para a construção de uma ontologia, mostrados a seguir:

- Determinar o domínio e o escopo da ontologia;
- Considerar o reuso de ontologias existentes;
- Enumerar os termos importantes na ontologia, definindo as terminologias iniciais;
- Definir as classes e suas hierarquias;
- Definir as propriedades das classes (slots), a estrutura interna dos conceitos explicitando suas extrínsecas propriedades (ex. nome, duração, uso), suas intrínsecas propriedades (ex. peso), partes, e relações com outras classes e indivíduos dessas classes;
- Definir características das propriedades (slots), como, por exemplo, tipo de valor, valores permitidos (domínio e faixa), cardinalidade, entre outras características que possam ter, e;
- Criação de instâncias incluindo a inclusão do valor da propriedade de cada instância criada.

A utilização dessa metodologia para o desenvolvimento da ontologia se deve pelo fato de ser um modelo considerado de referência em projetos de desenvolvimentos de ontologias, por sua simplicidade, além de utilizar restrições e a possibilidade de extensão da ontologia,

permitindo futuras definições. O escopo da ontologia proposta é restrito ao domínio da Taxonomia Revisada de Bloom (KRATHWOHL, 2001), baseado no trabalho de Oliveira (2018) e sobre questões de avaliação em disciplinas introdutórias de programação (JESUS e RAABE, 2009).

Os requisitos para a construção da ontologia são definidos na fase de especificação de requisitos. Nessa fase é estabelecido o porquê da ontologia estar sendo desenvolvida, quais serão os seus possíveis usuários e usos, e quais os requisitos que ela deve atender. Uma das técnicas comumente utilizada é a das Questões de Competência que, auxiliam na avaliação da ontologia depois de construída, além de justificar a existência da ontologia (GRUNNINGER e FOX, 1995).

Conforme as orientações metodológicas, produziu-se o Quadro 4 a seguir com a Especificação da Ontologia deste trabalho.

**Quadro 4 -** Documento de Especificação da Ontologia.

| **Documento de Especificação da Ontologia** |
|---|
| **Objetivo** |
| O objetivo da construção da ontologia é de auxiliar na classificação de questões de programação em um dos níveis do domínio Cognitivo da Taxonomia Revisada de Bloom. |
| **Escopo** |
| A ontologia concentra-se no domínio da Taxonomia Revisada de Bloom e às questões de avaliação em disciplinas introdutórias de programação. Considerando o domínio cognitivo da taxonomia de Bloom, a representação será por meio da enumeração dos principais termos (palavras-chaves e verbos). |
| **Linguagem de Implementação** |
| OWL - é uma linguagem para definição e instanciação de ontologias.<br>RDF - método geral para descrição ou modelagem conceitual de informações que são implementadas em recursos da Web. |
| **Usuários Finais** |
| Professores e Estudantes. |
| **Requisitos da Ontologia** |

(Continua)

| (Conclusão) |
| --- |
| 1- A Ontologia deve suportar idioma português;<br>2- A terminologia utilizada na ontologia deve ser retirada do domínio do modelo da Taxonomia Revisada de Bloom;<br>3- A ontologia deve representar o domínio Cognitivo da Taxonomia Revisada de Bloom;<br>4- Estabelecer uma dependência entre as instâncias que representam os níveis e os conteúdos;<br>5- A ontologia deve mapear as habilidades de alunos de programação de acordo com o nível cognitivo da taxonomia de Bloom. |
| **Questões de Competência** |
| 1. A partir dos verbos de uma determinada questão, como classificá-la em um dos níveis da Taxonomia Revisada de Bloom?<br>2. Como estabelecer a hierarquia dos conteúdos?<br>3. Como classificar a aprendizagem do aluno em um dos níveis da Taxonomia? |

**Fonte:** Elaborado pelo autor, 2019.

O vocabulário comum oferecido pela ontologia é formalizado e expressado semanticamente através do uso da linguagem OWL. Essa etapa foi executada através da ferramenta de edição e construção de ontologias Protégé (PROTEGE, 2019), onde é possível realizar a criação das classes, atributos e relações do modelo ontológico desenvolvido. A ferramenta ainda nos permite fazer verificações na ontologia através de consultas em *SPARQL*. Para a inferência e checagem lógica do modelo desenvolvido é utilizado o mecanismo de inferência *HermiT*, presente na ferramenta, que oferece um suporte completo para a linguagem OWL. A seção a seguir apresenta a descrição da ontologia desenvolvida.

### *3.1.1 Domínio e Escopo*

Como a metodologia *Ontology Development 101* propõe, o desenvolvimento de uma ontologia deve ser realizado em sete etapas, sendo a primeira responsável pela definição do domínio e do escopo. A ontologia proposta é classificada como ontologia de domínio e descreve os conceitos presentes no domínio da Taxonomia Revisada de Bloom. O objetivo desta é representar um modelo de classificação para questões de algoritmos e aprendizagem do aluno na Taxonomia Revisada de Bloom, tendo como base o trabalho de Jesus e Raabe (2009). O seu escopo é auxiliar na classificação de questões de algoritmos segundo a Taxonomia Revisada de Bloom.

A segunda etapa consiste em considerar o que outros pesquisadores já fizeram e verificar a possibilidade de refinar ou ampliar fontes existentes para o domínio. Considerou-se

assim a ontologia desenvolvida no trabalho de Oliveira (2018), como referência para o desenvolvimento da ontologia deste trabalho.

### *3.1.2 Enumeração de Termos*

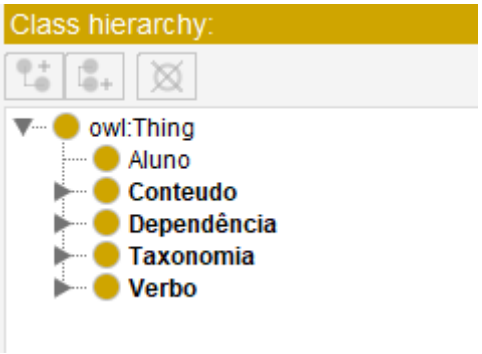
A terceira etapa do desenvolvimento consiste em listar os termos importantes da ontologia, ou seja, os termos relacionados ao domínio. Os termos foram enumerados por meio de investigações dos conhecimentos de acordo com o domínio da ontologia e a partir do trabalho de Jesus e Raabe (2009), associado a dissertação de Oliveira (2018). Os principais termos definidos foram: Taxonomia, Conteúdo, Verbo. Essas expressões foram utilizadas como base no desenvolvimento da ontologia.

### *3.1.3 Classes (Primitivas e Definidas)*

A quarta etapa consiste em definir as classes e suas hierarquias. As classes organizam conceitos de um domínio de forma hierárquica. Se uma classe tem apenas condições necessárias ela é conhecida como uma classe primitiva. Já classes que tem pelo menos um conjunto de condições necessárias e suficientes é conhecida como classe definida.

Fazendo uso da análise do domínio e executada na ferramenta *Protégé*, foram definidas as classes mais gerais, e em seguida, as classes mais específicas. Conforme ilustrado na Figura 7, as classes são criadas a partir de uma classe nativa que, está presente desde a criação de uma nova ontologia, a *owl:Thing*, é a classe padrão do *Protégé*, e ela se subdivide em 5 (cinco) classes primitivas que representam os conceitos gerais: Aluno, Conteúdo, Verbo, Taxonomia e Dependência.

**Figura 7 –** Conceitos Gerais em Classes.




**Fonte:** Elaborado pelo autor, 2019.

A ontologia desenvolvida possui 29 classes no total. A classe *Aluno* representa as instâncias do tipo aluno que serão classificadas e não possui subclasse. A classe *Conteúdo* possui 6 subclasses considerando os conteúdos ministrados na disciplina de algoritmos, e a hierarquia de conteúdos foi construída a partir de um questionário aplicado com três professores de uma instituição de ensino superior que ministram ou já ministraram esta disciplina. A classe *Verbo* possui 6 subclasses que contém verbos distintos correspondentes a cada um dos níveis da Taxonomia Revisada de Bloom: *Verbo1, ..., Verbo6*. A classe *Dependência* possui 6 subclasses considerando os seis níveis associados aos conteúdos: *Nível1, ..., Nível6*. Por fim, a classe *Taxonomia* possui 6 subclasses relacionadas a cada um dos níveis da taxonomia: *1Lembrar, 2Entender, 3Aplicar, 4Analisar, 5Avaliar* e 6*Criar*. Esta última classe é responsável por classificar um indivíduo que será uma questão ou um aluno em algum nível da Taxonomia Revisada de Bloom.

**Quadro 5 -** Identificação das Classes de domínio da ontologia.

| **Classes Primitivas** | **Descrição** |
|---|---|
| **Aluno** | Representa os indivíduos do tipo aluno que serão classificados na Taxonomia. |
| **Taxonomia** | Representa as dimensões do processo cognitivo da Taxonomia Revisada de Bloom. |
| **Verbo** | Representa a ação, definida pelo professor, que representa a pergunta. |
| **Conteúdo** | Representa os conteúdos da disciplina de Algoritmos e Programação I. |
| **Classe Definida** | **Descrição** |
| **Dependência** | Representa a dependência dos conteúdos. |

**Fonte:** Elaborado pelo autor, 2019.

A partir dos conceitos gerais foram criadas e agrupadas suas especificidades, ou seja, suas 24 (vinte e quatro) subclasses. As 6 (seis) subclasses de Taxonomia são chamadas classes

definidas, pois são classes que têm conjuntos de condições necessárias e suficientes, por exemplo, se algum indivíduo é membro da classe *1Lembrar*, então é obrigatório que satisfaça algumas condições. Se algum indivíduo satisfaz essas condições então pode ser inferido que ele seja membro da classe *1Lembrar*. O Quadro 6 ilustra as subclasses de "Conteúdo" "Taxonomia", "Verbo" e "Dependência", com suas descrições.

**Quadro 6 –** Identificação das subclasses definidas da Ontologia.

| **Subclasses Primitivas** | **Descrição** (Continua) |
|---|---|
| **Subclasses de Conteúdo** | |
| **Variáveis_e_Constantes** | Representa o conteúdo da disciplina de Algoritmos, Variáveis e Constantes. |
| **Atribuição_e_Tipos_de_Dados** | Representa o conteúdo da disciplina de Algoritmos, Atribuição e Tipos de Dados. |
| **Condicionais** | Representa o conteúdo da disciplina de Algoritmos, Condicionais. |
| **Laços** | Representa o conteúdo da disciplina de Algoritmos, Laços. |
| **Array_Vetores_e_Matrizes** | Representa o conteúdo da disciplina de Algoritmos, *Array*, Vetores e Matrizes. |
| **Funções_e_Procedimentos** | Representa o conteúdo da disciplina de Algoritmos, Funções e Procedimentos. |
| **Subclasses de Taxonomia** | |
| **1Lembrar** | Representa o nível Lembrar da Taxonomia Revisada de Bloom. |
| **2Entender** | Representa o nível Entender da Taxonomia Revisada de Bloom. |
| **3Aplicar** | Representa o nível Aplicar da Taxonomia Revisada de Bloom. |
| **4Analisar** | Representa o nível Analisar da Taxonomia Revisada de Bloom. |
| **5Avaliar** | Representa o nível Avaliar da Taxonomia Revisada de Bloom. |

| Subclasses Primitivas | Descrição<br>(Conclusão) |
|---|---|
| **6Criar** | Representa o nível Criar da Taxonomia Revisada de Bloom. |
| **Subclasses de Verbo** | |
| **Verbo1** | Representa um conjunto de verbos que melhor caracterizam o nível Lembrar. |
| **Verbo2** | Representa um conjunto de verbos que melhor caracterizam o nível Entender. |
| **Verbo3** | Representa um conjunto de verbos que melhor caracterizam o nível Aplicar. |
| **Verbo4** | Representa um conjunto de verbos que melhor caracterizam o nível Analisar. |
| **Verbo5** | Representa um conjunto de verbos que melhor caracterizam o nível Avaliar. |
| **Verbo6** | Representa um conjunto de verbos que melhor caracterizam o nível Criar. |
| **Subclasses de Dependência** | |
| **Nível_1** | Está relacionada à subclasse Variáveis_e_Constantes. |
| **Nível_2** | Está relacionada à subclasse “Atribuição_e_Tipos_de_Dados”. |
| **Nível_3** | Está relacionada à subclasse “Condicionais”. |
| **Nível_4** | Está relacionada à subclasse “Laços”. |
| **Nível_5** | Está relacionada à subclasse “Array_Vetores_e_Matrizes”. |
| **Nível_6** | Está relacionada à subclasse “Funções_e_Procedimentos”. |

**Fonte:** Elaborado pelo autor, 2019.

A Listagem 1 mostra a definição da subclasse *1Lembrar*. É estabelecido que para ser classificado como membro da subclasse *1Lembrar*, é necessário e suficiente que seja um indivíduo contido em Taxonomia, e cumpra a condição de possuir instância de *Verbo1*.

**Listagem 1** – Representação da lógica de descrição da subclasse *1Lembrar.*

```
1 Class: 1Lembrar
2     EquivalentTo Taxonomia
3     and (tem_verbo some Verbo1)
4     or (responde some Verbo1)
```

A Listagem 2 mostra a definição da subclasse *2Entender. É* estabelecido que para ser classificado como membro da subclasse *2Entender,* é necessário e suficiente que seja um indivíduo contido em Taxonomia, e cumpra a condição de possuir uma instância de *Verbo2.*

**Listagem 2** – Representação da lógica de descrição da subclasse *2Entender.*

```
1 Class: 2Entender
2     EquivalentTo Taxonomia
3     and (tem_verbo some Verbo2)
4     or (responde some Verbo2
```

A Listagem 3 mostra a definição da subclasse *3Aplicar.* É estabelecido que para ser classificado como membro da subclasse *3Aplicar,* é necessário e suficiente que seja um indivíduo contido em Taxonomia, e cumpra a condição de possuir instância de *Verbo3.*

**Listagem 3** – Representação da lógica de descrição da subclasse *3Aplicar.*

```
1 Class: 3Aplicar
2     EquivalentTo Taxonomia
3     and (tem_verbo some Verbo3)
4     or (responde some Verbo3)
```

A Listagem 4 mostra a definição da subclasse *4Analisar.* É estabelecido que para ser classificado como membro da subclasse *4Analisar,* é necessário e suficiente que seja um indivíduo contido em Taxonomia, e cumpra a condição de possuir instância de *Verbo4.*

**Listagem 4** – Representação da lógica de descrição da subclasse *4Analisar.*

```
1 Class: 4Analisar
2     EquivalentTo Taxonomia
3     and (tem_verbo some Verbo4)
4     or (responde some Verbo4)
```

A Listagem 5 mostra a definição da subclasse *5Avaliar.* É estabelecido que para ser classificado como membro da subclasse *5Avaliar,* é necessário e suficiente que seja um indivíduo contido em Taxonomia, e cumpra a condição de possuir instância de *Verbo5.*

**Listagem 5** – Representação da lógica de descrição da subclasse *5Avaliar.*

```
1 Class: 5Avaliar
2     EquivalentTo Taxonomia
3     and (tem_verbo some Verbo5)
4     or (responde some Verbo5)
```

A Listagem 6 mostra a definição da subclasse *6Criar*. É estabelecido que para ser classificado como membro da subclasse *6Criar*, é necessário e suficiente que seja um indivíduo contido em Taxonomia, e cumpra a condição de possuir instância de *Verbo6.*

**Listagem 6** – Representação da lógica de descrição da subclasse *6Criar.*

```
1 Class: 6Criar
2     EquivalentTo Taxonomia
3     and (tem_verbo some Verbo6)
4     or (responde some Verbo6)
```

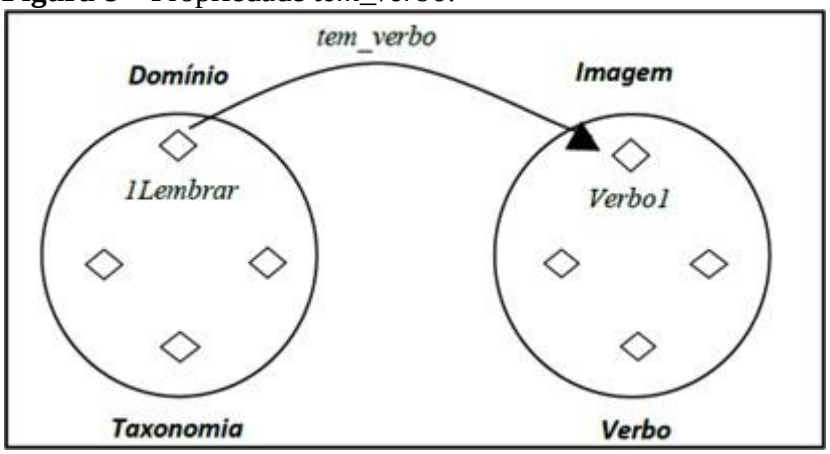
O relacionamento entre classes se dá por meio de propriedades. As propriedades de uma classe são herdadas por suas subclasses. A seção a seguir são apresentadas as propriedades da ontologia deste trabalho.

### *3.1.4 Propriedades das Classes*

A etapa cinco é o momento onde as propriedades pertencentes às classes são definidas. As propriedades definem as relações entre as classes e só precisam ser definidas em um dos níveis da hierarquia de uma classe pois elas são herdadas por suas subclasses (HINZ, 2008). Uma propriedade é declarada como Propriedade do Objeto (*Object Property*) quando tem o papel de relacionar uma classe à outra, unindo indivíduos ou classes de um domínio a indivíduos ou classes de uma imagem.

As propriedades de cada classe da ontologia foram especificadas à medida que as classes foram sendo definidas. Na ontologia resultante foram especificadas 5 (cinco) propriedades do tipo propriedade de objetos (*Object Properties*). Um exemplo é mostrado na Figura 8, com duas subclasses, Taxonomia e Verbo. A propriedade que os une é a *tem_verbo*. A classe *1Lembrar tem_verbo some Verbo1*, ou seja, as instâncias classificadas no nível Lembrar da Taxonomia têm verbos que são instâncias de *Verbo1*.

**Figura 8 –** Propriedade *tem_verbo*.




**Fonte:** Elaborado pelo autor, 2019.

O Quadro 7 sumariza as *Object Properties* da ontologia criada, com a indicação do domínio e da imagem de cada propriedade.

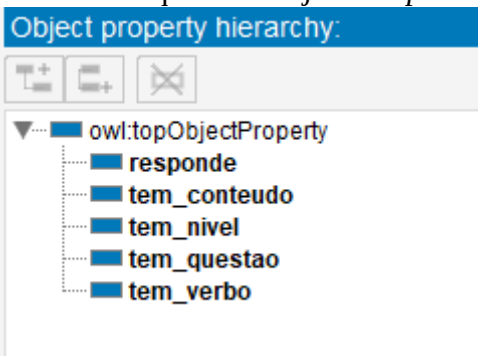
**Quadro 7 –** Propriedades dos Objetos.

| Propriedades (OBJECT PROPERTIES) | Descrição | Domínio | Imagem |
|---|---|---|---|
| ***tem_conteudo*** | Relação que indica a Dependência entre os Conteúdos. | *Dependência* | *Conteudo* |
| ***tem_nivel*** | Relação que indica que Conteúdo tem nível de Dependência. | *Conteudo* | *Dependência* |
| ***tem_questao*** | Relação que indica que Conteúdo tem questão classificada em Taxonomia. | *Conteudo* | *Taxonomia* |
| ***tem_verbo*** | Relação que indica que cada nível da Taxonomia possui um conjunto de Verbos distintos. | *Taxonomia* | *Verbo* |
| ***responde*** | Relação que indica que aluno responde questão. | *Taxonomia* | *Aluno* |

**Fonte:** Elaborado pelo autor, 2019.

A hierarquia dessas propriedades listadas na ferramenta *Protégé* é apresentada na Figura 9.

**Figura 9** – Hierarquia das *Object Properties.*




**Fonte:** Elaborado pelo autor, 2019.

A sexta etapa do desenvolvimento de ontologias de acordo com a metodologia 101 é a definição das restrições das propriedades. Cada propriedade é modelada por um conjunto de restrições que definem os valores que podem assumir, os tipos de valores e outras características dos valores.

Na ontologia deste trabalho foram utilizadas as características: funcional e transitiva. A característica funcional das propriedades é aquela que define que um indivíduo possui no máximo um outro indivíduo relacionado a si, esse tipo de propriedade pode ser observado entre as classes *Dependência* e *Conteudo.* A propriedade que relaciona essas classes é a *tem_conteudo*, e indica que cada um dos níveis de *Dependência* pertence a apenas um dos conteúdos da classe *Conteudo*.

A propriedade do tipo transitiva relaciona um indivíduo A com um indivíduo B, e se o indivíduo B se relacionar com um C, então os indivíduos A e C estarão relacionados. Essa propriedade pode ser observada na relação *tem_nivel* que relaciona as classes *Dependencia* e *Conteudo* e significa a dependência dos conteúdos, e a relação *tem_questao* relaciona as classes *Conteudo* e *Taxonomia*, ou seja, as classes *Dependencia* e *Taxonomia* se relacionam de forma transitiva. Essa propriedade ainda pode ser vista na relação *tem_verbo*, pois esta relaciona as classes *Taxonomia* e *Verbo* e significa que cada um dos níveis de *Taxonomia* contém um conjunto de verbos presentes na classe *Verbo*, ou seja, ao se adicionar uma instância que possui determinado conteúdo e verbo, ela será classificada em um dos níveis da hierarquia de conteúdos e em um nível da Taxonomia.

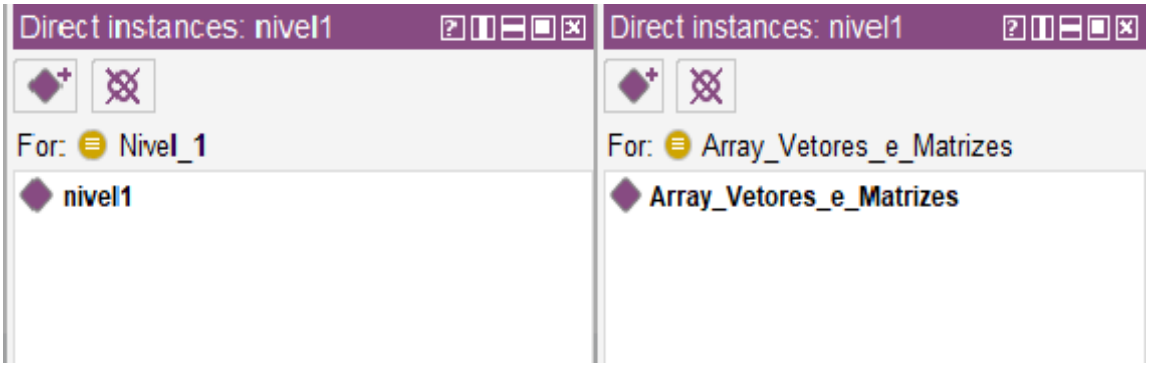
Por fim, foram criadas as instâncias para demonstração de expressividade da ontologia desenvolvida. A seguir serão descritas as instâncias utilizadas na ontologia.

### *3.1.5 Indivíduos*

Na sétima e última etapa, foram criados indivíduos para as classes, que também podem ser chamados de instâncias. Para a ontologia deste trabalho algumas instâncias foram inseridas a fim de auxiliar na classificação dos indivíduos que são do tipo questão e aluno.

Atualmente a ontologia conta com aproximadamente 70 instâncias que sempre permanecem na ontologia. Essas instâncias estão alocadas às subclasses das classes *Conteudo*, *Verbo* e *Dependencia*. A Figura 10 apresenta algumas instâncias da ontologia, exibidas através da ferramenta *Protégé*.

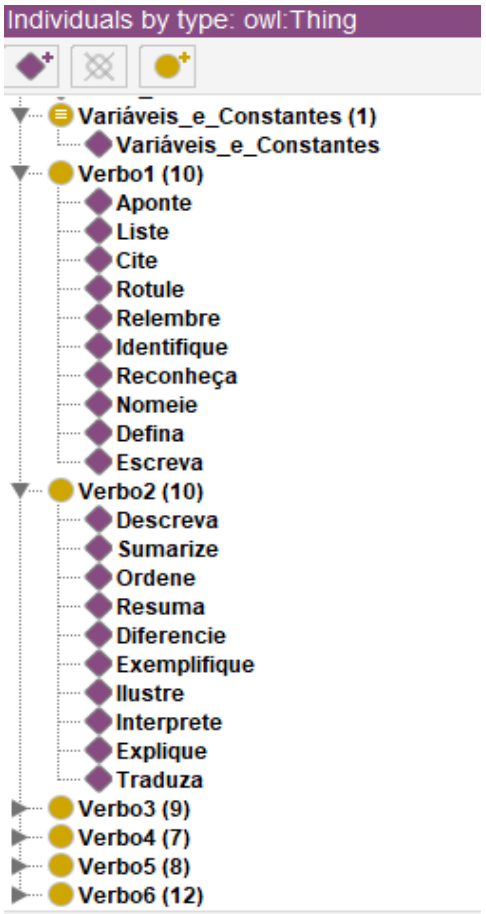
**Figura 10 –** Instâncias das classes *Nivel1* e *Array_Vetores_e_Matrizes*.




**Fonte:** Elaborado pelo autor, 2019.

Na Figura 11 são apresentadas as instâncias inseridas na ontologia. Para cada um dos níveis da Taxonomia são definidos um conjunto de verbos que o caracterizam. Os mesmos foram baseados, principalmente no trabalho de Galhardi e Azevedo (2013), que apresenta uma certa quantidade de verbos para auxiliar na identificação do nível a ser trabalhado. Cada verbo pertence a uma única classe – subclasses de verbo.

**Figura 11** – Instâncias da Ontologia.




**Fonte:** Elaborado pelo autor, 2019.

Algumas instâncias representando questões e alunos foram adicionadas para serem utilizadas em cenários hipotéticos para fins de demonstração da ontologia e testes.

### *3.1.6 Regras SWRL*

Após a criação dos indivíduos, foram criadas regras em SWRL (*Semantic Web Rule Language*). Segundo Horrocks et al. (2004), SWRL é uma expressiva linguagem de regras que combinam cláusulas com conceitos definidos em OWL, sendo usada para aumentar a capacidade de inferência sobre os indivíduos em uma base de conhecimento OWL. As regras em SWRL são compostas de duas partes: o antecedente (*body*) e o consequente (*head*). Cada regra é uma implicação lógica entre o antecedente e o consequente, e é entendida como: sendo as condições do antecedente verdadeiras, então as condições consequentes também serão verdadeiras.

Dessa forma, foram definidas características que classificam questões e alunos, através das propriedades. O Quadro 8 mostra as regras SWRL para a classificação de questões e estão divididas entre antecedentes e consequentes.

**Quadro 8 -** Regras SWRL de classificação de questões.

| Nível da Taxonomia | Antecedentes | (->) Consequentes |
|---|---|---|
| **Lembrar** | `Verbo1(?v) , tem_verbo(?q, ?v)` | `1Lembrar (?q)` |
| **Entender** | `Verbo2(?v), tem_verbo(?q, ?v)` | `2Entender (?q)` |
| **Aplicar** | `Verbo3(?v) , tem_verbo(?q, ?v)` | `3Aplicar (?q)` |
| **Analisar** | `Verbo4(?v) , tem_verbo(?q, ?v)` | `4Analisar(?q)` |
| **Avaliar** | `Verbo5(?v) , tem_verbo(?q, ?v)` | `5Avaliar (?q)` |
| **Criar** | `Verbo6(?v) , tem_verbo(?q, ?v)` | `6Criar (?q)` |

**Fonte:** Elaborado pelo autor, 2019.

O Quadro 9 mostra as regras SWRL para a classificação de Alunos e, assim como as questões, também estão divididas entre antecedentes e consequentes.

**Quadro 9 -** Regras SWRL de classificação do aluno.

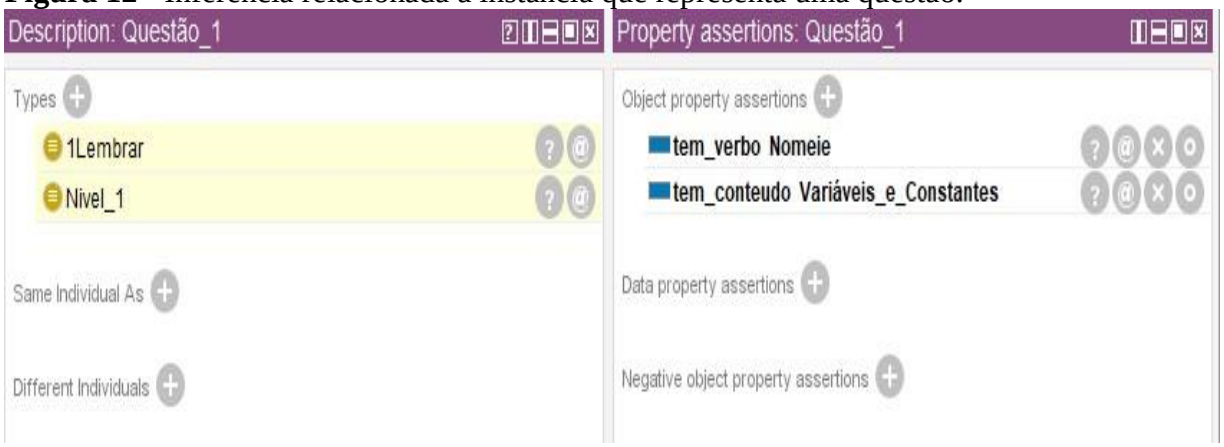
| Nível da Taxonomia | Antecedentes | (->) Consequentes (Continua) |
|---|---|---|
| **Lembrar** | `Aluno(?a), Verbo1(?v) , tem_verbo(?q, ?v) , responde(?a, ?q)` | `1Lembrar (?a)` |
| **Entender** | `Aluno(?a), Verbo2(?v) , tem_verbo(?q, ?v) , responde(?a, ?q)` | `2Entender (?a)` |
| **Aplicar** | `Aluno(?a), Verbo3(?v) , tem_verbo(?q, ?v) , responde(?a, ?q)` | `3Aplicar (?a)` |

| Nível da Taxonomia | Antecedentes | (->) Consequentes (Conclusão) |
|---|---|---|
| **Analisar** | `Aluno(?a), Verbo4(?v) , tem_verbo(?q, ?v) , responde(?a, ?q)` | `4Analisar(?a)` |
| **Avaliar** | `Aluno(?a), Verbo5(?v) , tem_verbo(?q, ?v) , responde(?a, ?q)` | `5Avaliar (?a)` |
| **Criar** | `Aluno(?a), Verbo6(?v) , tem_verbo(?q, ?v) , responde(?a, ?q)` | `6Criar (?a)` |

**Fonte:** Elaborado pelo autor, 2019.

Conforme as regras estabelecidas, pode-se observar na Figura 12, que a instância '*Questão_1'* se relaciona com as instâncias *'Variáveis_e_Constantes'*, e *'Aponte'*. Partindo disso, o motor de inferência deduziu que a instância deve ser classificada no nível Lembrar da Taxonomia e também no *Nivel_1*. Isso porque possui verbo da classe *Verbo1*, que contém verbos referentes ao nível Lembrar, e conteúdo de Nível 1, pois este, refere-se ao conteúdo Variáveis e Constantes.

**Figura 12 -** Inferência relacionada à instância que representa uma questão.




**Fonte:** Elaborado pelo autor, 2019.

O motor de inferência também classificou a instância *Aluno_1*, conforme mostra a Figura 13, apresentando o resultado da inferência para a instância que é classificada como sendo do tipo *Aluno.* A instância representa um Aluno e se relaciona com uma outra instância: *'Questão_1'*, através da propriedade *responde*. Dessa forma ela foi classificada tomando a interpretação de que ao responder uma questão corretamente o aluno é classificado no nível da questão respondida.

**Figura 13** - Inferência relacionada à instância de um aluno.

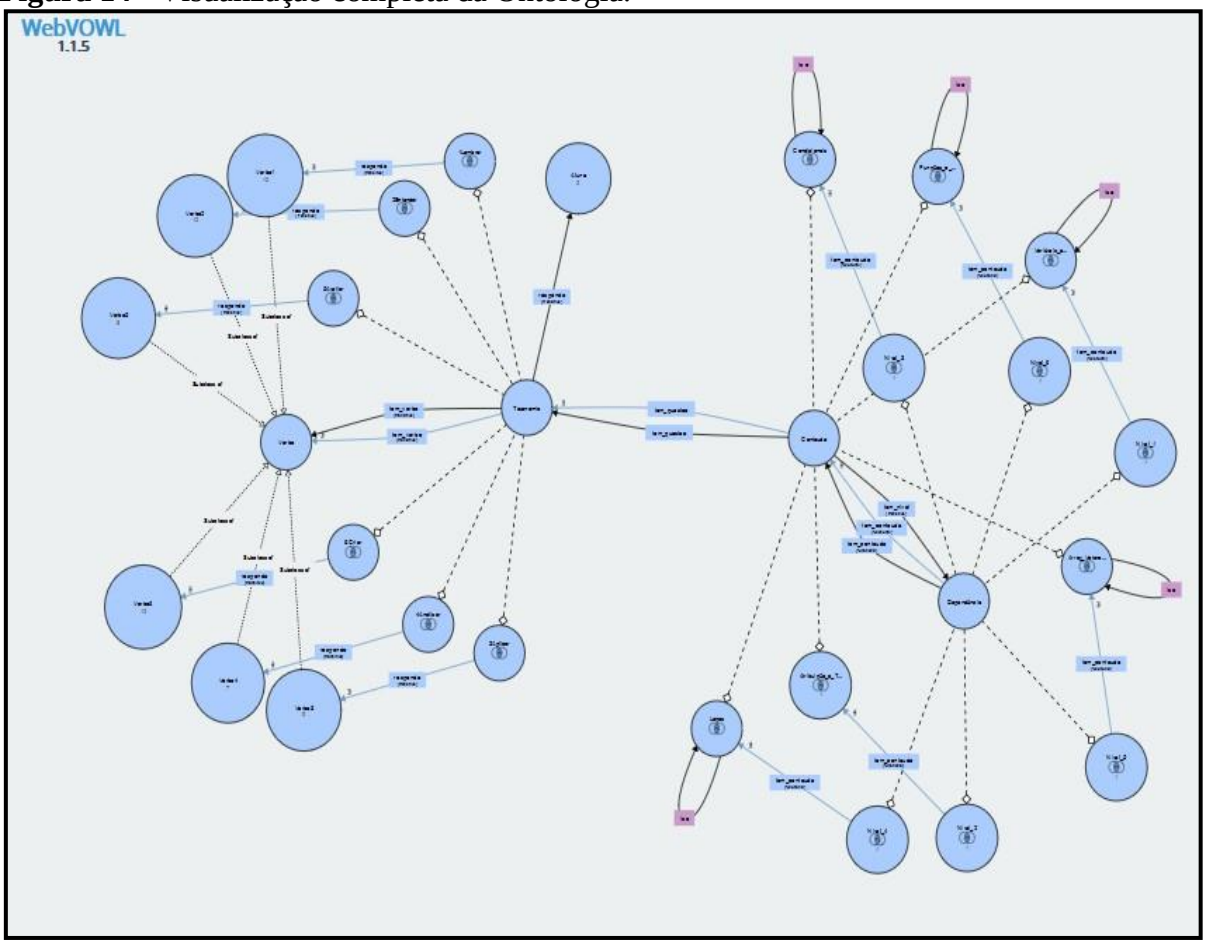

**Fonte:** Elaborado pelo autor, 2019.

### *3.1.7 Visualização da Ontologia*

A ferramenta WebVOWL[2] (*Web-Based Visualization of Ontologies*) foi desenvolvida baseada em padrões abertos da Web, fornecendo representações em grafo para a linguagem OWL. Através desta aplicação web, a ontologia pode ser visualizada graficamente estando mais clara a conexão das classes e propriedades da mesma (Figura 14).

[2] http://www.visualdataweb.de/webvowl/

**Figura 14 -** Visualização completa da Ontologia.




**Fonte:** Elaborado pelo autor, 2019.

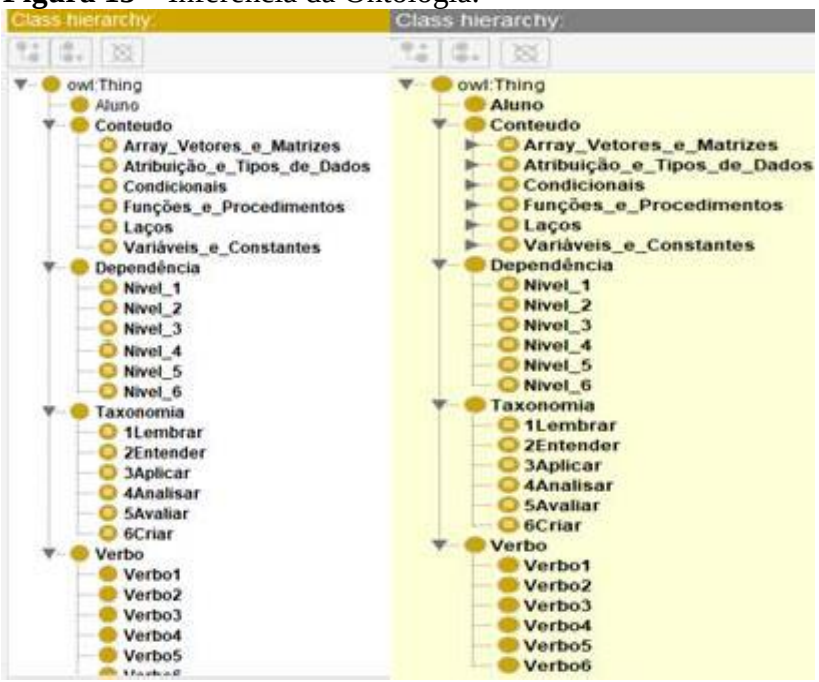
É possível observar que todas as classes estão ligadas e não existe nenhum nó/classe desconectada. Isso nos permite inferir que a ontologia deste trabalho está compacta e consistente.

# 4 VALIDAÇÃO DA ONTOLOGIA

A validação da ontologia acontece a partir da verificação da sua consistência, corretude e completude. A verificação foi feita através da inferência que tem o objetivo de averiguar se a ontologia é ou não consistente, como também identificar relações não explícitas entre classes. A ontologia proposta neste trabalho foi verificada pelo mecanismo de inferência *HermiT*, versão 1.3.8.

A consistência de uma ontologia é verificada quando conclusões contraditórias não são encontradas nas definições dos conceitos. A Figura 15 apresenta o resultado da execução do *HermiT* na ontologia, permitindo visualizar que nenhuma inconsistência foi verificada. Do lado esquerdo da figura é possível observar as classes definidas, e do lado direito da figura encontra-se o modelo inferido da ontologia.

**Figura 15 –** Inferência da Ontologia.




**Fonte:** Elaborado pelo autor, 2019.

Os motivos de não existirem inconsistências na ontologia advém do fato das classes estarem corretamente ligadas dentro de uma hierarquia pré-determinada e as relações entre as classes, subclasses e indivíduos não possuírem inconsistências.

No que diz respeito a corretude, uma ontologia deve atender basicamente os seguintes aspectos: não armazenar definições desnecessárias ou inúteis; não conter redundâncias explícitas entre definições de termos; e redundâncias não podem ser inferidas de outras

definições e axiomas. Por exemplo, os conceitos definidos para as classificações *1Lembrar* e *4Analisar* não podem conter ambiguidade:

**Taxonomia *1Lembrar*:** Todos os indivíduos sejam do tipo questão ou aluno, serão classificados neste nível se estiverem relacionadas com instâncias da classe 'V*erbo1*', que caracteriza o nível Lembrar da Taxonomia (Figura 16).

**Figura 16 -** Descrição da classe *1Lembrar* no *Protégé*.


**Fonte:** Elaborado pelo autor, 2019.

**Taxonomia *4Analisar:*** Assim como o nível lembrar, só será classificado no nível analisar as instâncias que se relacionarem com instâncias da classe *'Verbo4'*, pois esta caracteriza o nível Analisar da Taxonomia (Figura 17).

**Figura 17 -** Descrição da classe *4Analisar* no Protégé.


**Fonte:** Elaborado pelo autor, 2019.

Da mesma forma foram modelados os demais níveis da Taxonomia.

A fase de verificação da ontologia é concluída com a análise de completude. A completude da ontologia é verificada se o artefato estiver fornecendo as respostas das questões de Competência estabelecidas no capítulo 3. E para respondê-las utilizou-se consultas em *SPARQL*, inserindo dados hipotéticos de instância que são apresentados no Quadro 10.

**Quadro 10 -** Dados hipotéticos de instâncias.

| Instâncias | Informações | Classificação |
|---|---|---|
| **Questão_1** | Tem conteúdo Variáveis e Constantes e tem Verbo Aponte | *1Lembrar e Nivel_1* |
| **Questão_2** | Tem conteúdo Condicionais e tem Verbo descreva | *2Entender e Nivel_3* |
| **Questão_3** | Tem conteúdo Funções e Procedimentos e tem Verbo Use | *3Aplicar e Nivel_6* |
| **Questão_4** | Tem conteúdo Laços e tem Verbo Analise | *4Analisar e Nivel_4* |
| **Questão_5** | Tem conteúdo Array, Vetores e Matrizes e tem Verbo Verifique | *5Avaliar e Nivel_5* |
| **Questão_6** | Tem conteúdo Atribuição e Tipos de Dados e tem Verbo Desenvolva | *6Criar e Nivel_2* |
| **Aluno_1** | Responde Questão_1 | *1Lembrar* |
| **Aluno_2** | Responde Questão_2 | *2Entender* |
| **Aluno_3** | Responde Questão_3 | *3Aplicar* |
| **Aluno_4** | Responde Questão_4 | *4Analisar* |
| **Aluno_5** | Responde Questão_5 | *5Avaliar* |
| **Aluno_6** | Responde Questão_6 | *6Criar* |

**Fonte:** Elaborado pelo autor, 2019.

A primeira consulta SPARQL (QC1) recupera instâncias do tipo questão, que foram classificadas na Taxonomia. A Listagem 7 apresenta a consulta SPARQL para resolução desta questão de competência. Como resultado, são exibidas as instâncias, os conteúdos e a classificação delas.

**Listagem 7** – Constatação da completude, resolução da QC1.

**QC1: A partir dos verbos de uma determinada questão, como classificá-la em um dos níveis da Taxonomia Revisada de Bloom?**

Consulta SPARQL

```
1 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2 PREFIX owl: <http://www.w3.org/2002/07/owl#>
3 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
4 PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
```

```
5 PREFIX ont:
<http://www.semanticweb.org/usuário/ontologies/2019/7/untitled-ontology-
17#>
6
7 SELECT  ?questao ?conteudo ?typequestao
8 WHERE
9    {
10     ?questao ont:tem_verbo ?verbo .
11     ?questao ont:tem_conteudo ?conteudo .
12     ?questão rdf:type ?typequestao
13   }
```

**Resultado**

| Questão (questao) | Conteudo (conteúdo) | Taxonomia(typequestao) |
|---|---|---|
| Questão_1 | Variáveis_e_Constantes | 1Lembrar |
| Questão_2 | Condicionais | 2Entender |
| Questão_3 | Funcoes_e_Procedimentos | 3Aplicar |
| Questão_4 | Laços | 4Analisar |
| Questão_5 | Array_Vetores_e_Matrizes | 5Avaliar |
| Questão_6 | Atribuição_e_Tipos_de_Dados | 6Criar |

Assim, a ontologia responde à questão de competência QC1, pois foi possível classificar questões nos níveis da taxonomia a partir de suas propriedades.

A Listagem 8 apresenta a consulta SPARQL para resolução da segunda questão de competência. Como resultado, são exibidos as instâncias e os níveis de dependência de cada uma delas.

**Listagem 8** – Constatação da completude, resolução da QC2.

**QC2: Como estabelecer a hierarquia dos conteúdos?**

Consulta SPARQL

```
1 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2 PREFIX owl: <http://www.w3.org/2002/07/owl#>
3 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
4 PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
5 PREFIX ont:
<http://www.semanticweb.org/usuário/ontologies/2019/7/untitled-ontology-
17#>
6
7 SELECT ?questao ?dependencia
8 WHERE
9 {
10   ?conteudo rdf:type ?typeConteudo.
11   ?questao ont:tem_conteudo ?conteudo.
12   ?conteudo ont:tem_nivel ?dependencia
13   ?questão rdf:type ?typeDependencia
14   }
```

**Resultado**

| Questão (questao) | Dependência (dependencia) |
|---|---|
| Questão_1 | Nivel_1 |
| Questão_6 | Nivel_2 |
| Questão_2 | Nivel_3 |
| Questão_4 | Nivel_4 |
| Questão_5 | Nivel_5 |
| Questão_3 | Nivel_6 |

Para a questão de competência QC2, foi possível estabelecer a hierarquia dos conteúdos contidos considerando a dependência entre eles. Na ontologia, isso é demonstrado através da propriedade tem_nivel que relaciona as classes Conteudo e Dependencia.

Relacionada a resposta da questão de competência QC3, a Listagem 9 demonstra a consulta SPARQL que retorna os alunos e suas referidas classificações.

**Listagem 9** – Constatação da completude, resolução da QC3.

**QC3: Como classificar a aprendizagem do aluno em um dos níveis da Taxonomia?**

Consulta SPARQL

```
1 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2 PREFIX owl: <http://www.w3.org/2002/07/owl#>
3 PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
4 PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
5 PREFIX ont:
<http://www.semanticweb.org/usuário/ontologies/2019/7/untitled-ontology-
17#>
6
7 SELECT  ?aluno ?typealuno
8 WHERE
9 {
10    ?questao ont:tem_verbo ?verbo.
11    ?verbo rdf:type ?typeVerbo.
12    ?aluno ont:responde ?questao.
13    ?aluno rdf:type ?typealuno.
14   }
```

**Resultado**

| Aluno (aluno) | Taxonomia (typealuno) |
|---|---|
| Aluno_1 | 1Lembrar |
| Aluno_2 | 2Entender |
| Aluno_3 | 3Aplicar |
| Aluno_4 | 4Analisar |
| Aluno_5 | 5Avaliar |
| Aluno_6 | 6Criar |

Conforme mostrado, quando uma instância de aluno se relaciona com outra instância do tipo questão, a ontologia verifica o nível a que pertence a questão e em seguida, classifica o aluno em um dos níveis da taxonomia. Dessa forma, respondendo a QC3.

Ainda como forma de validar esta ontologia, foi desenvolvido um protótipo de uma ferramenta, que possuísse o contexto de aplicação da ontologia deste trabalho. Nesta validação foram utilizados dados fictícios. A próxima seção apresenta o contexto de aplicação da ontologia desta pesquisa.

# 5 CONSIDERAÇÕES FINAIS

Este capítulo apresenta as conclusões do trabalho realizado, retomando as questões de pesquisa, mostrando suas contribuições e limitações encontradas no seu desenvolvimento. Descreve, ainda, possíveis trabalhos futuros que podem surgir baseados neste.

## 5.1 SUMÁRIO DE PESQUISA

Tendo em vista alcançar os objetivos desta pesquisa algumas questões foram levantadas para serem respondidas no decorrer do trabalho. Os estudos primários da revisão bibliográfica auxiliaram a responder algumas das questões de pesquisa.

Com relação à questão de pesquisa **QSP1** – Como cada uma das categorias da taxonomia podem ser interpretadas dentro do contexto da aprendizagem de programação? A resposta para essa pergunta foi respondida através da pesquisa bibliográfica, e pode ser observada no tópico 2.2.2 do referencial teórico, em que se apresentou o quadro 3 com uma Síntese da Interpretação da Taxonomia de Bloom em programação.

A respeito da questão de pesquisa **QSP2** – Como classificar questões de algoritmos com a taxonomia de Bloom? Conforme a própria teoria da taxonomia de Bloom e do trabalho de Oliveira (2018), bem como mencionado no trabalho de Jesus e Raabe (2009), existem verbos associados a cada um dos níveis da taxonomia e estes verbos auxiliam na classificação de uma questão de avaliação. Dessa forma, ao associar um verbo a uma questão, é possível indicar a qual nível ela pertence.

A respeito da questão de pesquisa **QSP3** – Como permitir a classificação da aprendizagem do aluno na ontologia? Por meio da atribuição do nível da questão, através da definição do verbo, do conteúdo e da relação entre o aluno e a instância de questão, foi possível atribuir o nível do domínio cognitivo a qual o aluno pode ser classificado.

**Com relação à questão geral de pesquisa QGP -** Como uma ontologia pode ser estruturada para mapear questões de aprendizagem de algoritmos com a Taxonomia de Bloom? Além dos resultados e respostas às questões secundárias de pesquisa já apresentadas utilizou-se a hierarquia de conteúdos definida por professores que lecionam/lecionaram a disciplina de algoritmos, e, uma lista de verbos que caracterizam cada um dos níveis da Taxonomia. Dessa forma, pôde-se classificar questões em um nível da taxonomia revisada de Bloom e no nível correspondente a cada conteúdo, através da definição da dependência entre os conteúdos.

Também foi possível a classificação do aluno através da relação dele com as instâncias já classificadas na taxonomia.

Nestas perspectivas a ontologia de domínio deste trabalho pôde ser concebida atendendo também aos objetivos específicos desta pesquisa.

## 5.2 CONTRIBUIÇÕES

A taxonomia de Bloom, embora formulada na década de 50, tem sido revisitada por pesquisadores que reconhecem nela mais do que uma ferramenta para a avaliação do processo ensino-aprendizagem, mas uma ferramenta útil e eficaz no planejamento e implementação de aulas; na organização e criação de estratégias de ensino.

A ontologia deste trabalho é um artefato de software que visa classificar questões de algoritmos, visto que, interpretar cada um dos níveis da taxonomia é considerada por muitos, uma tarefa complicada.

Além disso, a ontologia permite a possibilidade de extensão pois tem a capacidade de reuso, onde poderão ser adicionadas outras classes, propriedades e regras de inferência.

## 5.3 LIMITAÇÕES

A ontologia descrita neste trabalho apresenta restrições, tais como:

- A ontologia ainda não faz parte de um sistema de informação consolidado;
- A ontologia limita-se apenas ao domínio cognitivo da Taxonomia, tendo em vista que existem os domínios afetivo e psicomotor;
- A integração da ontologia com o protótipo criado não foi concluída para este trabalho, tendo ficado para trabalhos futuros.

## 5.4 TRABALHOS FUTUROS

A ontologia proposta para a representação da Taxonomia Revisada de Bloom em questões de algoritmos foi validada neste trabalho de forma analítica e demonstrou que a ontologia responde às questões de competência. Como trabalhos futuros pretende-se:

- Introduzir o modelo ontológico proposto em um sistema de aprendizagem;
- Validar a utilidade da solução em um ambiente virtual de aprendizagem;

- Extensão da ontologia para o aspecto psicomotor e afetivo da Taxonomia de Bloom.

# REFERÊNCIAS


ARAÚJO, A. L. S. O. et al. Aplicação da Taxonomia de Bloom no ensino de programação com Scratch. **II Congresso Brasileiro de Informática na Educação (CBIE 2013)**. Rio Tinto – PB. p. 33.

BEREITER, C. and NG. E. **Three Levels of Goal Orientation in Learning**. *In Journal of the Learning Sciences*, nº 3, (vol. 1), 243-271. 1991.

BLOOM, B. S. et al. **Taxonomy of educational objectives**. New York: David Mckay, 1956. v. 1. p.262.

BORGES, R. P. et. al. Tutor Inteligente para Recomendação de Atividades de Programação em um Ambiente Virtual de Aprendizagem. **VI Congresso Brasileiro de Informática na Educação (CBIE 2017).** 2017.

CARVALHO, V. C. **OntAES : Uma ontologia para sistemas adaptativos educacionais baseada em objetos de aprendizagem e estilos de aprendizagem.** 2017. Dissertação (Mestrado em Ciência da Computação). Universidade Federal de Uberlândia. Uberlândia - MG. 2017.

CHAVES, J. O. et al. **Uma ferramenta baseada em juízes online para apoio às atividades de programação de computadores no moodle.** *Novas Tecnologias na Educação*, v. 11, n. 3, dezembro 2013.

CHAVES, J. O. M. **Uma ferramenta de apoio ao processo de ensino-aprendizagem em disciplinas de programação de computadores por meio da integração dos Juízes Online ao Moodle.** 2014. Dissertação (Mestrado em Ciência da Computação). Universidade do Estado do Rio Grande do Norte. Universidade Federal Rural do Semi-Árido. Mossoró - RN, 2014.

COUTO, D. C. **Classificação semiautomática de prioridade no atendimento a emergências médicas.** 2018. Dissertação (Mestrado em Ciência da Computação). Universidade do Estado do Rio Grande do Norte. Universidade Federal Rural do Semi-Árido. Mossoró - RN, 2018.

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.

FRANCISCO, R. E.; JÚNIOR, C. X. P.; AMBRÓSIO, A. P. Juiz Online no ensino de Programação Introdutória – Uma Revisão Sistemática da Literatura. **V Congresso Brasileiro de Informática na Educação (CBIE 2016).** Goiás. 2016.

GALHARDI, A. C.; AZEVEDO, M. M. Avaliações de Aprendizagem: O uso da Taxonomia de Bloom. **VII Workshop de Pós-Graduação e Pesquisa do Centro Paula Souza**, São Paulo. Out. 2013.

GOMES, A.; HENRIQUES, J.; MENDES, A. J. (2008). **Uma proposta para ajudar alunos com dificuldades na aprendizagem inicial de programação de computadores**. In *Educação, Formação & Tecnoloigias;* vol.1(1), pp. 93-103. Disponível em: <http://eft.educom.pt>

GUARINO, N. **Understanding, building and using ontologies. International Journal of Human and Computer Studies**, v. 45(2/3), 1997.

GUARINO, N. (Ed.). **Formal ontology in information systems: Proceedings of the first international conference** (FOIS'98), Trento, Itália. 1998.

GUIZZARDI, G. **Uma Abordagem Metodológica de Desenvolvimento para e com Reuso, Baseada em Ontologias Formais de Domínio.** 2000. Dissertação de Mestrado. Universidade Federal do Espírito Santo. Espírito Santo.2000.

GRUBER, T. R. A **Translation Approach to Portable Ontology Specifications. Knowledge Acquisition.** [S.l.], p. v. 5, n.2, p. 199-221. 1993

GRUBER, T. R. A. **What is an ontology?** [S. l. : s. n.], 1996. Disponível em: <http://www.ksl.stanford.edu/ kst/ what-is-a-ontology.html>. Acesso em: 20 mai. 2019.

GRÜNINGER, M.; FOX, M. S. **Methodology for the design and evaluation of ontologies.** 1995. Disponível em: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1. 1.44.8723. Acesso em: 31 de agosto de 2019.

HEVNER, A. R., MARCH, S. T., PARK, J., RAM, S. **Design Science in Information Systems Research**. MIS Quarterly, vol. 28, n. 1, pp. 75-105. 2004.

HINZ, V. T. **Algoritmos para Interoperabilidade entre Ontologias.** Universidade Católica de Pelotas. Programa de Pós-Graduação em Informática. Pelotas, p. 90. 2008.

HORROCKS, I; PATEL-SCHNEIDER, P. F.; BOLEY, H.; TABET, S.; GROSOF, B.; DEAN, M. SWRL: **A semantic web rule language combining OWL and ruleml.** W3C. 2004. Disponível em: <http://www.w3.org/Submission/SWRL/>. Acesso em: 21 de setembro de 2019.

JESUS, E. A.; RAABE, A. L. A. Interpretações da Taxonomia de Bloom no Contexto da Programação Introdutória. **XX Simpósio Brasileiro de Informática na Educação.** Itajaí. 2009.

KRATHWOHL, D. R. **A revision of Bloom's taxonomy: an overview.** Theory in Practice, v. 41, n. 4, p. 212-218, 2002.

LIMA, R. W. **Mapa de Conteúdos e Mapa de Dependências: ferramentas pedagógicas para uma metodologia de planejamento baseada em objetivos educacionais e sua implementação em um ambiente virtual de aprendizagem.** 2009. Tese (Doutorado em Engenharia Elétrica) – Universidade Federal do Rio Grande do Norte, Natal, 2009.

MORESI, E. **Metodologia de Pesquisa**. Universidade Católica de Brasília, 2003.

NASCIMENTO, G. S. **Ensino De Programação Com O Auxílio De Sistemas De Tutores Inteligentes e Ontologias.** 2018. Monografia (Bacharel em Sistemas de Informação) - Universidade Federal De Santa Catarina - Departamento De Informática E Estatística, Florianópolis, 2018.

NÄSSTRÖM, G. Interpretation of standards with Bloom's revised taxonomy: a comparison of teachers and assessment experts. **International Journal of Research & Method in Education.** Vol. 32, N. 1, 39–51. April 2009.

NEVES, M. F.; COELLO, J. M. A. OntoRevPro: Uma Ontologia sobre Revisão de Programas para o Aprendizado Colaborativo de Programação em Java. **XVII Brasileiro de Informática na Educação – SBIE - UNB/UCB**, pp. 359, 2006.

NOY, N. F.; MCGUINNESS, D. L. et al. **Ontology development 101: A guide to creating your first ontology.** [S.l.]: Stanford knowledge systems laboratory technical report KSL-01-05 and Stanford medical informatics technical report SMI-2001-0880, Stanford, CA, 2001.

OLIVEIRA, P. R. F. **Uma ontologia para classificação de objetos de aprendizagem considerando o domínio cognitivo da taxonomia de Bloom.** Dissertação (Mestrado em Ciência da Computação). Universidade do Estado do Rio Grande do Norte. Universidade Federal Rural do Semi-Árido. Mossoró - RN, 2018.

PIMENTEL, E. P. et. al. **Avaliação Contínua da Aprendizagem, das Competências e Habilidades em Programação de Computadores.** IX Workshop de Informática na escola – WIE – São Paulo. 2003.

PITEIRA M.; HADDAD S. R. Innovate in your program computer class: An approach based on a serious game. **In ACM International Conference Proceeding Series, pages** 49–54, 2011.

PROTEGE. Stanford University School of Medicine, 2017. Disponível em: http://protege.stanford.edu/.

RAABE, A. L. A.; SILVA, J. M. C. Um Ambiente para Atendimento as Dificuldades de Aprendizagem de Algoritmos. **XXV Congresso da Sociedade Brasileira de Computação.** São Leopoldo/RS. 2005.

ROCHA, P. S. et. al. **Ensino e Aprendizagem de Programação: Análise da Aplicação de Proposta Metodológica Baseada no Sistema Personalizado de Ensino.** *Novas Tecnologias na Educação - CINTED-UFRGS.* v. 8. Nº 3, Rio Grande do Sul. Dez., 2010.

SALOMÃO, L. F. S.; WATANABE, R. H. **Evasion in distance education courses offered by an organization of Brazilian Army: Actions to reduce**. In: International Conference on Information Systems and Technology Management (CONTECSI), 2013, São Paulo. Disponível em: <http://www.contecsi.fea.usp.br/envio/index.php/contecsi/10contecsi/paper/download/3539/2061 > Acesso em: 20 de mai. de 2019.

SANTOS, R. S. F. **Inserindo a taxonomia revisada de Bloom em um MOOC.** 2016. Dissertação (Mestrado em Ciência da Computação). Universidade do Estado do Rio Grande do Norte. Universidade Federal Rural do Semiárido. Mossoró – RN, 2016.

SILVA, I. C. da; FONSECA, L. C. C.; SILVA, R. d. J. da. Um sistema tutor inteligente para o ensino no domínio de lógica de programação. **Nuevas Ideas en Informática Educativa TISE.** 2015.

SILVA, S. D. **Knowledgemon Hunters: Um Jogo Sério com Geolocalização para Apoiar a Aprendizagem de Crianças com Autismo e Dificuldades de Aprendizado.** 143 p. Dissertação (Programa de Pós-Graduação em Ciência da Computação) Universidade do Estado do Rio Grande do Norte, Universidade Federal Rural do Semiárido, 2018.

SOUTO, A. V. M.; DUDUCHI, M. **Um processo de avaliação baseado em ferramenta computadorizada para o apoio ao ensino de programação de computadores.** Centro Estadual de Educação Tecnológica Paula Souza (CEETEPS). São Paulo, 2009.

SOUZA, D. M.; BATISTA, M. H. S., BARBOSA, E. F. Problemas e Dificuldades no Ensino e na Aprendizagem de Programação: Um Mapeamento Sistemático. **Revista Brasileira de Informática na Educação**, vol 24, n. 1, pp 42. 2016.

STAAB, S. et al. **Knowledge processes and ontologies. Intelligent systems.** IEEE, v. 16, n. 1, p. 26-34, 2001.

TOBAR, C. M.; ROSA, J. L. G.; ADÁN COELLO, J. M.; PANNAIN, R. Uma Arquitetura de Ambiente Colaborativo para o Aprendizado de Programação, **Anais do Simpósio Brasileiro de Informática na Educação, SBIE - XII**, Vitória, 21-23 nov. 2001.

WIERINGA, R. **Design Science as nested problem solving**, Proceedings of the 4th int. conf. on design science research in information systems and technology, ACM, p. 8, 2009.

ZHIGANG, S et. al. Moodle Plugins for Highly Efficient Programmin Courses. In: **MOODLE RESEARCH CONFERENCE, 1,** Heraklion, Crete-Greece, 2012.

YIN, Robert (1994). **Case Study Research: Design and Methods**. (2ª Ed) Thousand Oaks, CA: SAGE Publications. p. 13.