UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

# Apoio à identificação e recuperação de recursos educacionais abertos para o ensino de programação

**William Simão de Deus**

Tese de Doutorado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

ICMC USP
SÃO CARLOS

**William Simão de Deus**

# Apoio à identificação e recuperação de recursos educacionais abertos para o ensino de programação

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientadora: Profa. Dra. Ellen Francine Barbosa

**USP – São Carlos**
**Julho de 2024**

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

Simão de Deus, William
S588a Apoio à identificação e recuperação de recursos educacionais abertos para o ensino de programação / William Simão de Deus; orientadora Ellen Francine Barbosa. -- São Carlos, 2024.
155 p.

Tese (Doutorado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2024.

1. Recursos Educacionais Abertos. 2. Programação Introdutória. 3. Plataformas. 4. Metadados. I. Francine Barbosa, Ellen, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

**William Simão de Deus**

# Supporting the identification and retrieval of open educational resources for the programming education

Doctoral dissertation submitted to the Institute of Mathematics and Computer Sciences – ICMC-USP, in partial fulfillment of the requirements for the degree of the Doctorate Program in Computer Science and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Profa. Dra. Ellen Francine Barbosa

**USP – São Carlos**
**July 2024**

*Aos meus pais, Valmor e Edelzina.*

# AGRADECIMENTOS

Esta tese é o resultado da mobilização de muitos corações e mentes. Por isso mesmo, é essencial agradecer as pessoas que tornaram tudo isso possível, bem como as instituições que contribuíram para viabilizar esta pesquisa.

Agradeço imensamente a minha orientadora, professora Ellen, por guiar e resolver as dúvidas e os problemas mais diversos que surgiram durante todo esse período. Espero ter retribuído em partes todo seu empenho e dedicação.

Agradeço aos membros da banca de qualificação e de avaliação. Certamente, todos os comentários, críticas e considerações evidenciaram lacunas e pontos de melhorias fundamentais.

Esta pesquisa só foi possível graças ao apoio de diferentes pessoas e instituições. Por isso agradeço aos professores e profissionais mais diversos da Universidade de São Paulo e da Universidade da Virgínia. Em especial, agradeço a professora Briana pela sua receptividade, cooperação e colaboração.

Agradeço as instituições de fomento à pesquisa que apoiaram financeiramente o desenvolvimento deste projeto. O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001 e da Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), processo #2019/26871-4.

Agradeço aos membros dos grupos de pesquisa do Laboratório de Engenharia de Software (LABES) e do Laboratório de Computação Aplicada à Educação e Tecnologia Social Avançada (CAEd). Em especial, aos meus colegas de orientação que colaboraram com ideias, sugestões e críticas essenciais no desenvolvimento de toda a investigação.

Muitas pessoas colaboraram direta ou indiretamente para esta pesquisa. Por isso agradeço imensamente aos especialistas e participantes que fizeram parte de todo este projeto, bem como aos amigos que fiz antes ou durante esse período de formação.

Minha família é parte importante disso tudo. Desde que comecei com essa ideia de doutorado, meus pais, Valmor e Edelzina, e minha irmã, Adeline, me apoiaram incondicionalmente. Por isso sou grato por todo o amor, compreensão e suporte nos momentos mais variados.

Por fim, agradeço a Deus por me ajudar em todos os momentos dessa caminhada.

*“Ensinar não é transferir conhecimento,*
*mas criar as possibilidades para a sua própria produção ou a sua construção”*
*(Paulo Freire)*

# RESUMO

DEUS, W. S. **Apoio à identificação e recuperação de recursos educacionais abertos para o ensino de programação**. 2024. 155 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2024.

O crescimento de plataformas de Recursos Educacionais Abertos (REA) está ajudando a democratizar o ensino e o acesso à educação. Fatores como a redução de custos e a disponibilidade de recursos estão contribuindo para o desenvolvimento e disseminação de grandes coleções digitais. Em muitos desses acervos existem materiais que podem apoiar o ensino de programação introdutória, como exercícios, apresentações, materiais complementares e até mesmo softwares prontos para serem usados, personalizados e compartilhados sem nenhum custo. No entanto, ainda é muito difícil identificar e recuperar tais materiais devido à problemas variados, como a dimensão das coleções, ausência de suporte aos usuários, falta de padronização de termos e limitações dos motores de busca das plataformas abertas. Com esse cenário em vista, esta tese investigou a seguinte Questão de Pesquisa (QP): Como é possível auxiliar a identificação e a recuperação de REA para o ensino e aprendizado de programação introdutória? Para resolver essa QP, a *Design Science Research* (DSR) foi adotada como percurso metodológico. Inicialmente, as abordagens atuais foram investigadas por meio de um Mapeamento Sistemático (MS) e um Estudo Exploratório. Depois, foi realizado um grupo focal com professores de programação introdutória para verificar os principais desafios que ocorrem durante busca por REA. Diante disso, foram propostos dois mecanismos para reduzir os problemas: o primeiro é um vocabulário de termos que busca facilitar a identificação de recursos com conteúdo introdutório de programação. O segundo mecanismo utiliza *tags* e pesos para facilitar o processo de recuperação de REA introdutórios. Para avaliar a eficiência de ambos os mecanismos foi conduzido um Estudo de Caso múltiplo envolvendo três casos distintos. Os resultados demonstraram que os mecanismos propostos foram capazes de reduzir os problemas notados, apoiando na identificação e na recuperação de REA voltados ao ensino introdutório de programação por meio do conteúdo e da estrutura dos recursos ao invés da dependência de metadados, o que é a prática mais comuns das abordagens atuais. Além do mais, ficou evidente que os desafios referentes à identificação e à recuperação ocorrem antes, durante e depois de um usuário acessar uma iniciativa aberta. Para mitigá-los, é necessário estabelecer uma conexão entre três atores fundamentais: as plataformas que armazenam os REA, os responsáveis pela sua infraestrutura os usuários que buscam por REA.

**Palavras-chave:** Recursos Educacionais Abertos, Programação Introdutória, Plataformas, Metadados.

# ABSTRACT

DEUS, W. S. **Supporting the identification and retrieval of open educational resources for the programming education**. 2024. 155 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2024.

The growth of platforms for Open Educational Resources (OER) is helping to democratize teaching and access to education. Factors such as cost reduction and resource availability are contributing to the development and dissemination of large digital collections. Many of these collections contain resources that can support the teaching of introductory programming, including exercises, presentations, supplementary materials, and even software that can be used, customized, and shared at no cost. However, identifying and retrieving these materials remains challenging due to various issues such as the size of the collections, lack of user support, lack of standardization of terms, and limitations of search engines on open platforms. Following this scenario, this thesis investigated the following Research Question (RQ): How is it possible to assist in the identification and retrieval of OER for teaching and learning of introductory programming? To address this RQ, Design Science Research (DSR) was adopted as the methodological approach. Initially, current approaches were investigated by an Systematic Mapping (SM) and an Exploratory Study. Next, a focus group with professors of introductory programming was conducted to identify the main challenges in searching for OER. As a result, two mechanisms were proposed to reduce the identified problems: the first is a vocabulary of terms designed to facilitate the identification of resources with introductory programming content. The second mechanism uses tags and weights to simplify the OER retrieval process. To evaluate the efficiency of both mechanisms, a multiple Case Study was conducted involving three distinct scenarios. The results demonstrated that the proposed mechanisms effectively reduced the problems, supporting the identification and retrieval of OER using the content and structure of resources rather than relying on metadata, which is the most common practice of current approaches. Furthermore, it became evident that identification and retrieval challenges occur before, during, and after a user accesses an open initiative. To mitigate these challenges, it is necessary to establish a connection between three fundamental actors: the platforms for OER, those responsible for their infrastructure, and the users who search for OER.

**Keywords:** Open Educational Resources, Introductory Programming, Platforms, Metadata.

# LISTA DE ILUSTRAÇÕES

# LISTA DE QUADROS

# LISTA DE CÓDIGOS-FONTE

# LISTA DE TABELAS

# LISTA DE ABREVIATURAS E SIGLAS

| | |
|---|---|
| BNC | *British National Corpus* |
| CC | *Creative Commons* |
| DOM | *Document Object Model* |
| DSR | *Design Science Research* |
| DT | *Decision Tree* |
| HTML | *HyperText Markup Language* |
| IPAVL | *Introductory Programming Academic Vocabulary List* |
| MIT | *Massachusetts Institute of Technology* |
| MS | Mapeamento Sistemático |
| QP | Questão de Pesquisa |
| REA | Recursos Educacionais Abertos |
| SVM | *Support Vector Machine* |
| UNESCO | *United Nations Educational, Scientific and Cultural Organization* |

# SUMÁRIO

CAPÍTULO

1

# INTRODUÇÃO

Recursos Educacionais Abertos (REA) são materiais voltados ao ensino, ao aprendizado, ou à pesquisa, disponibilizados em qualquer formato e meio, desde que sob licenciamento aberto, e que permitem acesso gratuito, reuso, adaptação e redistribuição (UNESCO, 2019). O termo foi cunhado em 2002, em um evento convocado pela *United Nations Educational, Scientific and Cultural Organization* (UNESCO) em associação com a *William and Flora Hewlett Foundation* e a *Western Cooperative for Educational Telecommunications* (UNESCO, 2002).

A maioria desses recursos estão disponíveis em formatos digitais e são encontrados *online*, como textos, imagens, *websites*, vídeos, *podcasts*, entre outros (NAVARRETE; LUJÁN-MORA, 2018). Isso permite que diversas organizações, instituições e indivíduos contribuam com a sua produção, bem como a sua disseminação (PAWLOWSKI; BICK, 2012). Ademais, isso também potencializa o reuso, adaptação e redistribuição entre usuários (PULKER, 2019).

Os REA pertencem ao panteão da tecnologia possibilitada pelos movimentos "abertos", como o *open source* e o acesso aberto (HAVEMANN, 2017). É diante desse contexto que os recursos abertos reduzem as barreiras de acesso à educação e podem encorajar a inovação pedagógica, possibilitando oportunidades para criar, adaptar e compartilhar recursos com uma comunidade diversa e global de usuários (TODORINOVA; WILKINSON, 2020; SHU-HSIANG; JAITIP; ANA, 2015).

Assim, os REA fazem parte de um importante movimento sobre a educação. É nesse ponto que o ensino e o aprendizado de programação de computadores se une ao tema. Atualmente, as disciplinas introdutórias de programação compõem parte dos currículos acadêmicos (MEDEIROS; RAMALHO; FALCÃO, 2019). Em cursos da área da Computação, por exemplo, cerca de 40% da carga horária principal são dedicados para o ensino de programação e o desenvolvimento de software (ACM/IEEE-CS, 2020). Graduação de outras áreas, como os cursos de Ciência, Tecnologia, Engenharia e Matemática, têm dedicado uma porção de sua carga horária para ensinar conceitos introdutórios de programação (MEDEIROS; RAMALHO; FALCÃO,

2019).

Para além dessas graduações, aprender os fundamentos que envolvem a programação de computadores é uma agenda atual que vem ganhando atenção na última década (SCHERER; SIDDIQ; VIVEROS, 2020), sendo uma política de estado em diversos países. Reino Unido e Austrália, por exemplo, estão ensinando elementos de programação para alunos do ensino fundamental (DUNCAN; BELL; TANIMOTO, 2014). No Japão, o assunto tornou-se obrigatório em escolas primárias (OHASHI *et al.*, 2018). No Brasil, a lei Nº 14.533, instituída em 11 de janeiro de 2023, tornou a programação uma das estratégias prioritárias do eixo de Educação Digital Escolar (BRASIL, 2023). Movimento similar já foi adotado por outros países, como Estônia, Finlândia, França, Israel e Estados Unidos, que já incluíram o ensino de programação em unidades curriculares de diferentes níveis educacionais, devido à sua importância para futuros profissionais (LINDBERG; LAINE; HAARANEN, 2019; DUNCAN; BELL; TANIMOTO, 2014).

Hoje em dia, diversas iniciativas que conectam os REA para o ensino e o aprendizado de programação estão sendo promovidas. Bahamón (2022), por exemplo, relatou sua experiência em adotar REA em uma disciplina de programação. Baldwin (2015) mostrou que, apesar de algumas dificuldades, é possível adotar os REA para realizar uma espécie de sala de aula invertida para alunos iniciantes em programação. Tovar, Chan e Reisman (2017) evidenciaram o grande interesse de usuários em usar os recursos de diferentes disciplinas de Ciência da Computação, sendo o ensino de programação uma das principais.

Plataformas como o Merlot[1] e o EngageCSEdu[2] são exemplos do potencial do uso de REA para ensinar programação de computadores. Chan *et al.* (2020) analisou o acervo do Merlot e notou que entre os 100 recursos mais populares, a categoria de Programação/Linguagens de Programação foi a que recebeu maior destaque. Já o EngageCSEdu foi constituído como uma coleção digital de materiais instrucionais para aulas introdutórias de Ciência da Computação (MONGE; QUINN; FADJO, 2015). Em números atuais, cerca de 90% do acervo digital é voltado aos recursos com conteúdo de programação introdutória.

No entanto, existe um amplo caminho a ser percorrido na disseminação dos REA voltados ao ensino e aprendizado de programação introdutória. Esta tese investiga um aspecto em particular: o processo que envolve a identificação e a recuperação desses recursos.

## 1.1 Motivação

Identificar um REA voltado ao ensino ou aprendizado de programação introdutória não é uma tarefa simples. Há diversas barreiras que dificultam isso. Talvez, uma das mais conhecidas é

[1] https://www.merlot.org/merlot/
[2] https://engage-csedu.org/

"o problema da descoberta", definido por Wiley, Bliss e McEwen (2014) como uma dificuldade em encontrar um REA válido para ser reusado por outros usuários.

Em síntese, esse problema ocorre devido à dinâmica que as plataformas abertas são estruturadas e a natureza dos recursos ali armazenados. O aumento dessas coleções é desarmonizado, sem ter um padrão entre títulos ou resumos e o conteúdo propriamente dito do material (PIEDRA *et al.*, 2010). Ademais, muitas plataformas abertas são desenvolvidas para se tornarem armazéns digitais, sem estabelecer um diálogo entre as reais necessidades dos usuários e a coleção que está ali armazenada. Há anos, diversos trabalhos já apontavam problemas relacionados à omissão de informações e à falta de evidência empírica sobre a estrutura de tais plataformas (RODÉS-PARAGARINO; GEWERC-BARUJEL; LLAMAS-NISTAL, 2016; DICHEVA; DICHEV, 2014).

No ensino de programação esse desafio é ainda mais complexo. Por exemplo, a falta de uma terminologia padrão é comum entre professores que lecionam o tema (HERTZ, 2010). Como consequência, os REA incorporam a mesma característica. Muitas vezes são adotados termos distintos em materiais educacionais, como "laço de repetição" ou "iteração" para designar o mesmo conteúdo. O problema vai muito além da sala de aula ou da falta de padronização. Becker e Quille (2019) sintetizaram isso ao descrever que ainda falta uma compreensão adequada sobre o real significado de programação introdutória. Muitas *guidelines*/guias curriculares, como proposto pela ACM/IEEE-CS (2020) e por Araujo *et al.* (2017), seguem um modelo de competências expondo o que deve ser aprendido. Porém, em última instância, cabe ao professor decidir quais conteúdos serão ministrados. A tendência natural é ocorrer o que Hertz (2010) notou: professores de programação, em geral, possuem visões distintas sobre os assuntos que são introdutórios e avançados.

Há também o estrangeirismo, especificamente o uso do inglês, que é muito comum na área de programação, afetando alunos e até professores que não dominam o idioma (QIAN; LEHMAN, 2017). Na prática, muitas vezes termos em inglês, como *loops, arrays, functions*, entre outros, são adotados em materiais em português. Isso, além de confundir alunos e professores, associa os REA de programação aos materiais de outras áreas do conhecimento. O mesmo ocorre com o uso de termos reservados em códigos. Palavras como $for$ (para), *while* (enquanto) ou *return* (retorna) estão presentes em trechos dos códigos e também podem ser encontradas nos textos dos REA, no entanto, em cada caso há um significado.

Por fim, existe ainda o problema da heterogeneidade. Desde que o termo REA foi definido em 2002, a diversidade dos formatos sempre foi destaque. Até hoje, a recomendação mais recente da Unesco (2019) alerta sobre essa característica essencial. Ou seja, o armazenamento e a disponibilização dos REA devem considerar a natureza diversa dos materiais. Infelizmente, as soluções atuais não lidam com toda essa complexidade. Ao invés disso, usam estratégias convencionais, como listas de disciplinas (TOVAR; CHAN; REISMAN, 2017) ou classificações automáticas (MOURIÑO-GARCÍA *et al.*, 2018), o que não é intuitivo aos alunos e professores

de programação.

Por isso mesmo, os REA ainda não conseguem promover todo o seu potencial dentro da educação (CORTINOVIS *et al.*, 2019). Até os dias atuais existem poucas evidências sobre o real impacto do reuso de REA em práticas de ensino (PULKER, 2019). Diante desse cenário, por que pesquisar sobre a identificação e a recuperação REA para o ensino e o aprendizado de programação introdutória?

Em primeiro lugar, é necessário pensar no contexto político e educacional que os REA e o ensino de programação estão situados. Bahamón (2022) já destacou que os REA de programação promovem a diversidade, equidade e inclusão por irem diretamente ao encontro de uma barreira comum para os alunos: o alto custo dos livros didáticos. Assim, ao facilitar a descoberta de REA de programação, cria-se uma nova possibilidade para alunos de baixa renda, com limitações socioeconômicas.

Além disso, os REA reduzem a dependência de materiais e iniciativas proprietárias. Sobre isso, é importante destacar que muitos buscadores digitais possuem seus resultados principais impulsionados por propaganda, em detrimento de recursos e materiais gratuitos. Como já exposto por O'Neil (2021), esse comportamento focado na geração de cliques pode contribuir para aumentar a desigualdade e ofuscar a realidade. No entanto, mesmos os motores de buscas de plataformas abertas ainda possuem um desempenho considerado insatisfatório, como já foi alertado por Dichev e Dicheva (2012).

É necessário ainda considerar que os REA estão alinhados com a agenda para um mundo sustentável (UNESCO, 2019). Os REA podem proporcionar acesso à educação, fornecendo contribuições diretas para diferentes comunidades, promovendo uma educação de qualidade, inclusiva e equitativa para todos, como exposto pelo quarto objetivo sustentável da ONU (2023). Nessa perspectiva, o ensino e a aprendizagem da programação introdutória apoiada pelos REA estão diretamente relacionados com o desenvolvimento de competências e com a igualdade de acesso ao ensino e a formação profissional.

Em outras palavras, mais do que identificar e recuperar REA de programação introdutória, pesquisar sobre esse assunto é estar alinhado com uma agenda que busca democratizar a educação e facilitar seu acesso por um grupo maior de pessoas.

## 1.2 Questão de Pesquisa

Baseado no contexto em que esta pesquisa se insere, o principal objetivo é analisar a interseção entre REA e programação introdutória compreendendo e auxiliando o processo de identificação e de recuperação de REA para o ensino introdutório de programação. Como objetivos secundários pretende-se: (i) analisar o estado da arte e da prática sobre as abordagens atuais; (ii) determinar os principais desafios inerentes dessas atividades e (iii) propor e validar

intervenções que possam auxiliar os usuários a encontrarem REA de programação introdutória.

Baseado nos objetivos mencionados, a Questão de Pesquisa (QP) que norteia toda a investigação é a seguinte: **Como é possível auxiliar a identificação e a recuperação de REA para o ensino e aprendizado de programação introdutória?**

## 1.3 Percurso Metodológico

A *Design Science Research* (DSR) (DRESCH; LACERDA; ANTUNES, 2015; PIMENTEL; FILIPPO; SANTORO, 2019) foi adotada como o percurso metodológico dessa investigação. Ao todo, seis etapas distintas foram desenvolvidas:

1. *Identificação do Problema:* O foco da primeira etapa foi definir o escopo da investigação. Para isso, a literatura sobre REA e programação, bem como as plataformas abertas foram investigadas.

2. *Objetivos:* O propósito da segunda etapa foi refinar o escopo da investigação. Para isso, foi organizado um grupo focal com cinco professores de programação introdutória.

3. *Design e Desenvolvimento:* O principal objetivo da terceira etapa foi projetar e construir intervenções para auxiliar os usuários no processo de identificação e de recuperação de REA sobre programação introdutória. Como resultado, foram produzidos dois mecanismos distintos:

   - *Introductory Programming Academic Vocabulary List* (IPAVL): Um vocabulário acadêmico com 108 termos relacionados, contendo abreviaturas, palavras técnicas, nomes de linguagens de programação, entre outros.
   - TAGGER: Um protótipo de reconhecimento web para REA de programação introdutória que utiliza pesos e *tags* do *HyperText Markup Language* (HTML) para seleção de recursos.

4. *Demonstração:* A finalidade da quarta etapa foi realizar uma avaliação conceitual dos mecanismos propostos (IPAVL e TAGGER), verificando se ambos poderiam influenciar o processo de identificação e de recuperação de REA introdutórios.

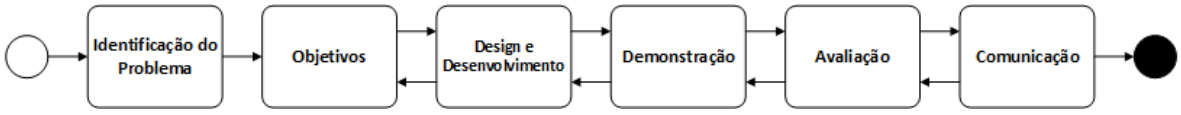
5. *Avaliação:* O objetivo central da quinta etapa foi avaliar os mecanismos desenvolvidos. Para isso, foi realizado um estudo de caso múltiplo em diferentes plataformas de REA.

6. *Comunicação:* O foco da última etapa de investigação foi compartilhar com a comunidade os produtos e os resultados alcançados.

Após a condução da primeira etapa, as demais etapas foram realizadas iterativamente, isto é, o resultado de uma etapa poderia influenciar no desenvolvimento da outra. Ademais,

uma etapa só era considerada concluída quando os resultados alcançados fossem considerados satisfatórios para o prosseguimento da tese. Esse percurso metodológico é ilustrado na Figura 1.

Figura 1 – O percurso metodológico de pesquisa




Fonte: Dados da Pesquisa.

## 1.4 Organização da Tese

O restante da tese encontra-se organizado da seguinte forma:

- O Capítulo 2 introduz os fundamentos em torno dos REA e do ensino e aprendizado de programação introdutória. O objetivo é estabelecer os conceitos essenciais aos leitores sobre os temas que fundamentam a pesquisa. Ademais, o capítulo também busca situar o atual estado da arte e da prática sobre REA e também definir com exatidão o escopo de atuação de investigação, apresentando uma lista de trabalhos relacionados.
- O Capítulo 3 detalha o desenvolvimento de um estudo exploratório em oito plataformas abertas. Entre os resultados apresentados, destaca-se o impacto que os metadados possuem no processo de identificação e de recuperação nos REA.
- O Capítulo 4 sintetiza os resultados alcançados com um grupo focal formado por cinco professores de programação introdutória. Em seu conteúdo é apresentado uma lista dos desafios que os usuários geralmente enfrentam quando estão tentando identificar e recuperar um REA de programação introdutória. O intuito do capítulo é estabelecer tudo isso em torno de um modelo de desafios.
- O Capítulo 5 expõe todo o *design* e o desenvolvimento dos mecanismos propostos nesta tese: a IPAVL e o TAGGER. Em síntese, esses dois mecanismos fazem uma interface entre os usuários que buscam por REA e as plataformas abertas que armazenam tais materiais. Em linhas gerais, o objetivo do capítulo é detalhar todo o processo de construção de cada mecanismo.
- O Capítulo 6 apresenta o processo de validação adotado. Em seu conteúdo é detalhado o estudo de caso conduzido em três plataformas abertas. A intenção do capítulo é apresentar uma visão geral sobre a estratégia de validação adotada bem como os resultados alcançados pela IPAVL e pelo TAGGER.
- O Capítulo 7 encerra a tese, expondo as considerações finais, limitações da pesquisa e perspectivas futuras de investigação. Sua finalidade é concentrar os principais aspectos da investigação, solucionando a QP e discutindo perspectivas futuras.

CAPÍTULO

# 2

# FUNDAMENTOS

Esta tese investiga a relação entre dois temas: REA e o ensino e aprendizado de programação introdutória. Por isso mesmo, o objetivo deste capítulo é apresentar uma síntese sobre o significado, a definição e o escopo de cada um desses assuntos, bem como a relação entre ambos. O conteúdo encontra-se estruturado em quatro seções distintas: A Seção 2.1 detalha o que são os REA. A Seção 2.2 examina o ensino e aprendizado de programação introdutória. A Seção 2.3 sumariza e aprofunda a relação entre esses dois temas analisando uma lista de trabalhos relacionados. Por fim, a Seção 2.4 apresenta as considerações finais do capítulo.

## 2.1 Recursos Educacionais Abertos

De acordo com a Unesco (2019), REA são materiais voltados ao ensino, aprendizado ou pesquisa que podem podem estar em qualquer formato ou meio, isto é, independentemente da sua extensão, formatação ou até mesmo forma de armazenamento, desde que estejam sob licenciamento que permita o seu acesso livre de custos, reuso, (re)propósito, adaptação e redistribuição por outros usuários.

A definição original sobre os REA foi proposta em 2002 durante o *Forum on the Impact of Open Courseware for Higher Education in Developing Countries*, realizado em Paris (UNESCO, 2002). Devido à sua própria natureza, os REA possibilitam o compartilhamento de materiais de ensino e aprendizado com liberdade de uso e reuso, alcançando assim cinco liberdades essenciais (MAZZARDO, 2018):

- **Reter:** o direito de fazer, possuir e controlar cópias do conteúdo.
- **Reutilizar:** o direito de usar o conteúdo de diversas maneiras.
- **Revisar:** o direito de adaptar, ajustar, modificar ou alterar o próprio conteúdo.

- **Remixar** o direito de combinar o conteúdo original ou revisado com outro conteúdo aberto para criar algo novo.
- **Redistribuir:** o direito de compartilhar cópias do conteúdo original, suas revisões ou seus remixes com outras pessoas.

Esse conjunto de liberdades foi originalmente proposto em 2007 e continha apenas quatro elementos (Reusar, Revisar, Remixar e Redistribuir). Em 2014, o próprio Wiley atualizou o *framework* propondo a adição de um novo R, o de Reter, criando o *framework* 5*R* (WILEY, 2014).

O *framework* foi profundamente inspirado pela cultura do movimento *Open Source*. De forma resumida, por volta dos anos 1970, um grupo de desenvolvedores trabalhavam em torno de uma dinâmica conhecida como compartilhamento de código (BRETTHAUER, 2001). Na prática, o código-fonte poderia ser lido, alterado ou até mesmo distribuído entre as pessoas, sem problemas de direitos autorais ou imposições financeiras. Isso era diametralmente oposto às práticas industriais de desenvolvimento de software, que apresentavam, entre outras características, a dependência de arquiteturas, tecnologias e empresas.

Os 5*R* partem da utilização do licenciamento aberto nos recursos educacionais. Segundo a Unesco (2019), um REA deve estar sob domínio público ou possuir o seu conteúdo sob licenças abertas. Isso significa que o conteúdo do material deve ter algum tipo de licenciamento que permita o seu reuso por outros usuários.

Em linhas gerais, o domínio público é uma condição jurídica especial para licenciamento de diferentes obras. No caso dos REA, um autor poder abrir mão dos seus direitos autorais ou pode ser que a obra entre em domínio público. Isso permite a exploração da obra de diferentes formas.

Já as licenças abertas são regramentos especiais que garantem diferentes níveis de abertura em um determinado material. No contexto dos REA, geralmente os materiais encontram-se sob alguma licença *Creative Commons* (CC). As licenças CC permitem que um autor distribua a sua obra por meio de diferentes perspectivas do reuso. O autor pode evitar, por exemplo, a exploração comercial de sua criação e ou restringir outras criações. Isso foi idealizado em 2002 em um grupo de pesquisa da Universidade de Stanford (KIM, 2007).

As licenças CC foram necessárias para elucidar as permissões dos recursos, fornecendo maior flexibilidade do que leis tradicionais de direito autoral. Também evitou a complexidade gerada por regramentos específicos, muito comum em obras que envolvem direito autoral. As licenças CC possibilitaram que os criadores ou detentores dos direitos autorais pudessem, de modo simples e padronizado, pudessem definir o que poderia ser feito com as suas produções (MAZZARDO, 2018).

Em união, o propósito educacional dos REA e a adoção de licenciamento aberto permite

manter a cultura do reuso, de modo que os recursos possam ser recombinados, usados em outros contextos e modificados por outros usuários (AMIEL; OREY; WEST, 2010). Para esse processo ocorrer, é necessário que o REA seja acessível para um grande público. Para isso, geralmente são adotados plataformas digitais que, na prática, representam grandes coleções de materiais educacionais.

### 2.1.1 Plataformas Abertas

Uma plataforma aberta[1] compreende um local no qual os REA estão disponíveis para serem acessados e utilizados. As plataformas são diversas, variando em tamanho, formato, gênero, entre outras características. Mas, em geral, podem ser classificadas conforme a sua natureza ou a sua estratégia de busca.

Do ponto de vista que considera a natureza de uma plataforma, Comas-Quinn e Borthwick (2015) classificaram os tipos mais comuns como oriundos de repositórios institucionais (de universidades/colégios); repositórios de comunidade (similares aos institucionais, mas com um foco em um determinado assunto/área/região); repositórios nacionais/internacionais (derivadas de organizações ou governos); cursos abertos ou *websites* de compartilhamento livres (concentram conteúdos digitais).

Já do ponto de vista da estratégia de busca, uma plataforma aberta geralmente é derivada segundo os modelos dos repositórios de objetos de aprendizagem, como proposto por McGreal (2008):

- *Centralizado:* tipo mais simples de armazenamento na qual existe uma coleção própria de REA disponíveis.
- *Portal:* tipo mais robusto de coleção que integra diferentes bases de plataformas abertas.
- *Híbrido:* emprega o tipo centralizado e portal em suas buscas.

Existem diversas plataformas abertas que promovem o acesso a recursos de programação. O MIT *OpenCourseWare*[2] é um dos exemplos mais bem conhecidos. A plataforma armazena e disponibiliza cursos e materiais educacionais do *Massachusetts Institute of Technology* (MIT) para o grande público. O Merlot é outra plataforma bem conhecida, que além de armazenar materiais em seu próprio acervo, permite a busca em outros acervos abertos.

No Brasil, o EduCapes[3] é um bom exemplo. A plataforma conta com um amplo acervo de materiais de todas as áreas do conhecimento e armazena diferentes tipos de recursos digitais.

[1] Nesta tese, o termo "plataforma aberta" abrange as diferentes iniciativas de REA, como "repositórios", "referatórios", entre outros.
[2] https://ocw.mit.edu/
[3] https://educapes.capes.gov.br/

Os projetos com perfil voltado ao ensino de programação são mais específicos. O EngageCSEdu, por exemplo, fornece REA para disciplinas de computação, sobretudo para disciplinas introdutórias. O projeto foi patrocinado pelo Google e atualmente é apoiado pela *Education Board* da ACM[4]. Com o foco nas práticas e na adoção dos REA, a plataforma também possui o envio de um documento complementar para facilitar a adoção dos recursos por outros profissionais.

Conforme o perfil da plataforma aberta, o acesso e o compartilhamento dos REA serão distintos. No geral, essas plataformas frequentemente são organizadas de modo manual, pelos próprios usuários (GARCÍA *et al.*, 2015). Em resumo, ao enviar um recurso, o próprio usuário preenche um conjunto de metadados, como título, descrição e palavras-chave (SUCUNUTA; RIOFRIO; TOVAR, 2019). É importante observar que cada plataforma possui suas próprias configurações e limitações, sendo que alguns possuem características de armazenamento de recursos enquanto outros atuam apenas buscadores de REA.

Geralmente, as estruturas de envio, busca e indexação de recursos são distintas, fazendo com que o processo de recuperação de REA seja totalmente diferente considerando cada tipo de plataforma. Ou seja, a busca por um REA é uma tarefa complicada (WILEY; BLISS; MCEWEN, 2014). Isso também se deve, em partes, ao processo de armazenados dos dados dos REA. No geral, são utilizados metadados, os quais apresentam informações relevantes do recurso, como título, descrição, assunto, etc.

### 2.1.2 Metadados

A solução mais comum para estruturar plataformas abertas é a adoção de metadados. Em sua definição mais conhecida, metadados são descritos como *dados sobre dados* (VALIENTE *et al.*, 2015). Seu funcionamento geralmente envolve um par formado por chave (identificador) e valor (conteúdo), como exposto na Tabela 1.

Tabela 1 – Exemplo de metadados

| **Chave** | **Valor** |
|---|---|
| *Title* | Curing the Web's Identity Crisis |
| *Creator* | Steve Pepper & Sylvia Schwab |
| *Subject* | RDF, topic maps, subject indicator |
| *Publisher* | IDEAlliance |
| *Date* | May 2003 |
| *Language* | English |
| *Format* | XML |

Fonte: Garshol (2004).

Muitas plataformas abertas baseiam suas buscas nessas chaves e valores para recuperar

[4] Informações extraídas da seção de *Frequently Asked Questions*, disponível em: https://engage-csedu.org/faqs

resultados relevantes. Isso gera três problemas específicos para o ensino de programação: o primeiro é a falta de padrão nos termos e conceitos introdutórios. O segundo problema é que muitos motores de busca foram projetados apenas para isso: ser um motor de busca, sem considerar as especificidades de identificação e de recuperação dos REA. Finalmente, o terceiro problema é relacionado à experiência do usuário, já que na maioria das vezes a interface de busca pode ser complexa ou confusa.

Além disso, não existe um único padrão de metadados para as plataformas abertas. Por exemplo, um padrão muito comum é o IEEE LOM, que é amplamente adotado em plataformas de REA para otimizar a descrição, armazenamento, busca e recuperação de recursos (SAMPSON; ZERVAS; CHLOROS, 2011). Em sua estrutura é adotado um conceito hierárquico de categorias e subcategorias. A versão original do IEEE LOM conta com nove categorias distintas, derivadas em subcategorias. Essas subcategorias, por sua vez, são derivadas em mais elementos quando necessário (BARKER, 2005). Assim, o padrão de metadados adota um esquema que tenta suprir a maioria das necessidades de identificação e classificação de recursos. Nesse sentido, cabe salientar que o IEEE LOM também apresenta os valores espaciais e o tipo de dado que cada elemento pode receber.

Outro exemplo adotado entre as plataformas abertas é o padrão o *Dublin Core*. Esse formato também favorece a busca e recuperação de recursos em coleções digitais. No entanto, o padrão *Dublin Core* possui uma estrutura mais simples. Segundo Garshol (2004), o padrão define o significado dos domínios de um recurso, mas não como representar suas propriedades. Ou seja, as chaves possuem definições fixas, mas os valores podem variar em valores espacias e tipos de dados.

Existem outros padrões de metadados que podem seguir ou não algumas das premissas do IEEE LOM e do *Dublin Core*. No entanto, cada padrão vai operar conforme o seu propósito de seu desenvolvimento. Sendo que nas plataformas abertas, é comum o uso do IEEE LOM e *Dublin Core* com pequenas especificações.

## 2.2 Programação Introdutória

O ensino de programação é uma atividade transversal e multidisciplinar que envolve diferentes habilidades e conhecimentos, relacionada a diversos campos da tecnologia (LAHTINEN; ALA-MUTKA; JARVINEN, 2005). No contexto que esta pesquisa se insere, o ensino introdutório de programação representa conceitos que os alunos que estão aprendendo a programar devem compreender. Tais conceitos foram sistematizados por diferentes autores. Tew e Guzdial (2010), por exemplo, elaboraram um conteúdo conceitual comum, como exposto na Tabela 2.

É necessário destacar que cada conceito pode variar de acordo com a linguagem de programação utilizada. Ou seja, a forma como são feitas as estruturas condicionais em *Python* ou *C* possuem diferenças. Mas, em síntese, o que é observado é a necessidade de ensinar o conceito,

Tabela 2 – Conceitos introdutórios de programação

| Constructo | Exemplos |
|---|---|
| *Fundamentos* | Conceitos elementares de programação |
| *Operadores Lógicos* | Operadores "E" e "OU" |
| *Estrutura Condicional* | Uso de condições no código |
| *Loops* | Uso de laços de repetição |
| *Arrays* | Armazenamento de dados em estruturas bidimensionais ou tridimensionais |
| *Funções* | Uso de funções e funções com parâmetros |
| *Básico de POO* | Conceitos sobre objetos, métodos e atributos |
| *Outros* | Recursão, depuração, etc... |

Fonte: Baseado em Tew e Guzdial (2010). Originalmente, os autores definiram 10 elementos, mas, para facilitar a compreensão, os elementos similares foram agrupados e foram adicionados exemplos do que cada um representa.

no caso a estrutura condicional, e não a tecnologia (linguagem de programação) aplicada.

Muitas vezes, o conjunto de conceitos é materializado em ementas de disciplinas introdutórias de programação. No Brasil, é comum que isso ocorra em cursos da área de computação, como Ciência da Computação, Engenharia de Computação, Engenharia de Software, Sistemas de Informação, Análise e Desenvolvimento de Sistemas, entre outros (ARAUJO *et al.*, 2017). Com a publicação da Lei 11.892, que estabeleceu a rede federal de educação profissional, científica e tecnológica, o ensino introdutório de programação também foi incorporado em diferentes cursos de formação técnica que estão integrados no nível médio.

Já no contexto internacional, o ensino introdutório de programação é geralmente atrelado às disciplinas de diferentes cursos, como *Computer Engineering, Computer Science, Information Systems, Information Technology, Software Engineering*, entre outros (ACM/IEEE-CS, 2020). Em diversos países o conteúdo já é disseminado em diferentes unidades curriculares e estágios do nível educacional (OHASHI *et al.*, 2018; LINDBERG; LAINE; HAARANEN, 2019; DUNCAN; BELL; TANIMOTO, 2014).

Por fim, há ainda outro recorte importante: disciplinas introdutórias em cursos de outras áreas, como Engenharia, Matemática, Tecnologia e Ciência (MEDEIROS; RAMALHO; FALCÃO, 2019). Nesse caso, o ensino de programação introdutória busca posicionar o estudante com as bases elementares sobre programação.

### 2.2.1 Ensino e Aprendizado

Em disciplinas introdutórias, o ensino de programação geralmente é estruturado em uma sequência de conteúdos, organizado por um professor (BEGEL; KO, 2019). Muitas disciplinas são baseadas em diretrizes técnicas de ensino, como a *guideline* proposta pela ACM/IEEE-CS (2020) e os referenciais de formação apresentados por Araujo *et al.* (2017). Na prática, tais diretrizes organizam os conceitos introdutórios de programação em alto nível. Mas, cabe ao educador organizar e definir a estratégia de aprendizagem e os critérios de avaliação (ROBINS, 2019).

Além das diretrizes técnicas, o ensino de programação também pode ocorrer por meio

da leitura de livros, participações de tutorias *online*, competição em *hackthons* ou até mesmo perguntando ou respondendo questões em sites de Q&A[5] (BEGEL; KO, 2019). Nesse caso, o processo de ensino é descentralizado, e ao invés de um educador organizar o conteúdo, o próprio estudante percorre o caminho de aprendizado.

Existem diversos modelos e técnicas capazes de apoiar o ensino de programação, mas a maioria dos professores opta pelo aprendizado por meio da exposição de problemas e supervisão da solução dos mesmos (LOKAR; PRETNAR, 2015). O mesmo ocorre com os alunos que, por vezes, estudam um determinado conceito e tentam aplicá-lo. Embora sejam práticas comuns, isso acaba sobrecarregando os dois atores fundamentais do ensino: professores e estudantes.

Sob o ponto de vista do professor, o ensino de programação é uma tarefa complexa e árdua por conta da carga de trabalho envolvida. Borges *et al.* (2018), por exemplo, citam que o trabalho é intensificado ao lecionar disciplinas de programação já que é necessário revisar o código desenvolvido pelos alunos, e como cada um possui sua própria lógica de programação, é essencial refletir e analisar as possibilidades. Na mesma linha, Oliveira, Stringhini e Correa (2018) também observaram que o elevado número de alunos por turma e os graus distintos da complexidade do código também acabam aumentando a complexidade do trabalho.

Em união, esses fatores influenciam a capacidade do professor fornecer *feedback* para seus alunos, como destacado por Falkner *et al.* (2014). Ao mesmo tempo, é complexo ilustrar cenários e aspectos relacionados à programação para que todos consigam compreender (SORVA; KARAVIRTA; MALMI, 2013).

Do ponto de vista dos estudantes, o ensino de programação também é muito desafiador. De acordo com Durak (2018), muitos estudantes encontram dificuldades em aprender o processo de codificação e acabam gerando códigos ruins. Isso ocorre pela complexidade de articular diferentes conhecimentos sintáticos, conceituais e estratégicos na elaboração de um código (QIAN; LEHMAN, 2017).

Tudo isso é ainda mais tensionado tendo em vista que muitas das atividades introdutórias de programação envolvem fundamentos sobre matemática, abstração, interpretação e articulam a capacidade do estudante em entender e resolver problemas (OLIVEIRA; STRINGHINI; CORREA, 2018). Tais fatores acabam contribuindo para a desmotivação, fazendo com que as disciplinas de programação alcancem as maiores taxas de reprovação na maioria dos cursos (BORGES *et al.*, 2018).

Por tudo isso, aprender a programar é uma atividade complexa de ensino e também de aprendizado. Mas, a programação introdutória pode oferecer muitos benefícios. Diversos pesquisadores concordam que os alunos que aprendem a programar melhoram suas habilidades cognitivas, incluindo criatividade, raciocínio, pensamento matemático, habilidades de trabalho

[5] Em inglês, *Questions and Answers*, sendo o Stack Overflow um dos principais sites para solução de dúvidas sobre programação de computadores.

em equipe, e podem produzir ideias mais originais/criativas (SCHERER; SIDDIQ; VIVEROS, 2018; MARCOLINO; BARBOSA, 2017). Ponto fundamentais para futuros profissionais de programação.

Cabe destacar ainda que aprender a programar também envolve a aquisição de competências, como técnicas fundamentais de *design* e pensamento crítico (MEHMOOD *et al.*, 2020). Em um sentido mais amplo, também está atrelado ao desenvolvimento econômico e social de diferentes países. Por isso mesmo, muitas nações buscam atualizar seu modelo de ensino, criando oportunidades de oferta de disciplinas programação para crianças e jovens (DUNCAN; BELL; TANIMOTO, 2014). Essa agenda é importante para o desenvolvimento dos estudantes por conta da espécie de trabalho do futuro, que poderá envolver diferentes tipos de tecnologias de programação, mas também pela capacidade de desenvolver habilidades que serão importantes o futuro de muitas profissões tecnológicas.

### 2.2.2 Uso de REA

Considerando todo o potencial que os REA possuem, o seu impacto na educação ainda é considerado baixo (CORTINOVIS *et al.*, 2019). Grande parte disso se deve ao desconhecimento dos educadores e dos estudantes sobre o uso de REA bem como as dificuldades de identificação e de recuperação desses recursos. O ensino de programação de computadores apoiado por REA pode romper essa bolha, pois concentra em si diversas características que contribuem para a disseminação do movimento.

Em primeiro lugar, historicamente, a programação de computadores evoluiu com a cultura aberta. O próprio movimento *Open Source*, a adoção de licenciamento aberto e grande disponibilidade de recursos digitais para auxiliar o aprendizado de temas em de tópicos da área de computação (COMBEFIS; MOFFARTS; JOVANOV, 2019). Em união, isso tudo permite aos alunos e professores da área a ideia de criação colaborativa, remixagem de materiais e cultura do reuso.

Em segundo lugar, muitos professores utilizam a estratégia de exposição de conceitos e de exercícios para ensinar programação. Portanto, as plataformas abertas com REA surgem como uma opção potencial de uso por parte desses profissionais, pois os mesmos concentram em suas coleções diversos materiais com essas características (HALSTEAD-NUSSLOCH; RUTHERFOORD, 2019). Os recursos são abertos e permitem o compartilhamento ou remixagem, o que implica na redução do tempo e do esforço. Isso também é importante aos estudantes que estão aprendendo a programar, buscando materiais com formatos mais adequados às suas inclinações ou interesses.

Em terceiro lugar, os professores de programação possuem a cultura de buscar e criar materiais educacionais em diferentes plataformas de ensino. Waite *et al.* (2020), por exemplo,

citaram que diferentes plataformas educativas, como o Barefoot[6] e o Code-it[7] são utilizadas por professores. Similarmente, Marcolino e Barbosa (2017) destacaram que ao ensinar programação, muitos professores adotam fontes alternativas de materiais, como cursos de programação *online* para apoiar seus alunos.

Por fim, a adoção de REA pode contribuir para o ensino introdutório de programação aliando o perfil do professor e inclinações naturais de um aluno. Por exemplo, caso um estudante prefira ler, podem ser usados os livros abertos de programação. Caso o aluno tenha preferência por mídias digitais, pode ser usados vídeos, cursos ou *podcasts*.

Algumas pesquisas já demonstraram impactos positivos ao se adotar REA no ensino de programação. Baldwin (2015), por exemplo, utilizou diferentes REA em um curso introdutório de programação e concluiu que os materiais - ainda de forma superficial - foram capazes de apoiar o curso e também implicaram em redução de custos. Além disso, Chan *et al.* (2020) identificaram que os REA mais populares em uma plataforma aberta eram relacionados ao ensino de programação ou ensino de linguagens de programação. Ou seja, há um interesse genuíno dos usuários em criar, disponibilizar e utilizar tais materiais.

## 2.3 Mapeamento Sistemático

Para compreender o atual estado da arte sobre a identificação e a recuperação de REA voltados ao ensino introdutório de programação, foi conduzido um Mapeamento Sistemático (MS). O estudo original é apresentado no artigo:

- Deus, W. S. de; Barbosa, E. F. *"A Systematic Mapping of the Classification of Open Educational Resources for Computer Science Education in Digital Sources"*. **IEEE Transactions on Education**, 2021. (IEEE ToE'2021). https://doi.org/10.1109/TE.20 21.3 128019

### 2.3.1 Design

O MS foi conduzido conforme uma personalização da *guideline* proposta por Kitchenham e Charters (2007). Ao todo, foram realizadas três estratégias de buscas: automáticas, manuais e *snowballing*.

Para a busca automática a seguinte *string* de pesquisa foi desenvolvida: *(categorization OR category OR classification OR schema OR scheme OR taxonomy) AND ("educational content" OR "educational resource" OR OER*). As buscas foram realizadas em cinco bibliotecas digitais: Scopus, IEEE Xplore, ACM DL, Science Direct e Web of Science. As buscas se concentraram no título, resumo e palavras-chave.

[6] https://www.barefootcomputing.org/
[7] http://code-it.co.uk/

Já para a busca manual foram adotados conferências e periódicos da área de Informática Aplicada à Educação. Para seleção das conferências, foi utilizado o índice h5 e as métricas de mediana h5 fornecidas pelo Google Scholar. Para seleção de periódicos, foi usado o *Journal Citations Report*. A lista final de conferências e periódicos é apresentada na Tabela 3.

Tabela 3 – Busca Manuais - Conferência e Periódicos

| Tipo | Nome | Abreviatura |
|---|---|---|
| Conferência | Technical Symposium on Computer Science Education | SIGCSE |
| Conferência | ACM Conference on Computer-Supported Cooperative Work & Social Computing | CSCW |
| Conferência | IEEE/ASE Frontiers in Education Conference | FIE |
| Conferência | International Conference on Learning Analytics & Knowledge | LAK |
| Periódico | Computers & Education | C&E |
| Periódico | IEEE Transactions on Education | IEEE ToE |
| Periódico | IEEE Transactions on Learning Technologies | IEEE TLT |
| Periódico | ACM Transactions on Computing Education | ACM TOCE |
| Periódico | Journal of Education and Information Technologies | EAIT |

Fonte: Dados da Pesquisa.

A técnica de *snowballing* foi aplicada para identificar novos estudos. As citações recebidas e a lista de referências de cada estudo selecionado foram verificadas, aplicando-se os critérios para a identificação de estudos potenciais.

Ao final das três buscas 22 estudos foram selecionados. Após uma filtragem específica para a síntese da tese, 6 trabalhos foram analisados. A seguir, esses estudos são detalhados.

### 2.3.2 Síntese

O primeiro estudo analisado (S01) foi proposto por Ruiz-Iniesta, Jiménez-Díaz e Gómez-Albarrán (2014). Nele, os autores utilizaram o enriquecimento semântico para apoiar a identificação de REA. A proposta dos autores possui duas grandes limitações: a primeira delas é que a solução foi projetada somente para REA construídos no software *Scratch*[8], desconsiderando assim materiais produzidos em outros formatos, como vídeos e imagens. A segunda limitação se deve ao fato da solução funcionar apenas em uma plataforma aberta, que atualmente possui uma coleção reduzida de REA.

Já o segundo estudo identificado (S02) foi apresentado por Gunarathne *et al.* (2018) e propõe um motor de busca de REA de computação que dispõe seus resultados em um grafo de links. A proposta dos autores, no entanto, possui limitações importantes. Em primeiro lugar, somente uma plataforma de REA foi utilizado em sua investigação. Em segundo lugar, a abordagem é limitada em apenas cinco tecnologias: CSS, JSP, Matlab, Python e SQL. Em terceiro lugar, o custo de processamento é muito alto e não pode ser escalada (por isso mesmo apenas cinco tecnologias foram utilizadas). Finalmente, é importante destacar que a abordagem proposta ainda não foi avaliada empiricamente por usuários reais, como professores de programação.

Em uma perspectiva similar, o terceiro estudo (S03) foi proposto por Mouriño-García *et al.* (2018) e utiliza aprendizado de máquina para identificar e recuperar REA em plataformas

[8] Software educativo para ensino de programação. Disponível em: https://scratch.mit.edu/

digitais. Embora a abordagem proposta pelos autores também seja capaz de identificar e recuperar recursos voltados ao ensino de programação, existem diversas limitações. Primeiro, a abordagem somente está disponível para recursos em inglês. Segundo, o sistema ignora metadados relevantes, como o tipo do recurso e abertura da licença. Terceiro, a solução também possui um alto custo de processamento. Isso, segundo os autores, pode gerar baixo desempenho em buscas complexas.

O quarto estudo (S04) analisado foi desenvolvido por Tovar, Chan e Reisman (2017). Nesse estudo, os autores propuseram uma listagem fixa para identificar REA da Ciência da Computação e de Sistemas de Informação. Em resumo, a proposta dos autores é usar uma lista de disciplinas para organizar o acervo de REA. As principais limitações da abordagem se devem ao fato dos autores validarem sua proposta em apenas uma plataforma digital. Ademais, a análise feita pelos autores focou em apenas uma fração da coleção de REA da plataforma, os 100 recursos mais acessados, descartando os demais recursos que possivelmente não estão sendo encontrados devido às dificuldades de recuperação.

O quinto estudo (S05) foi desenvolvido por Dessì *et al.* (2019). Nele, os autores apresentaram uma abordagem que extrai semanticamente o conteúdo de um REA e utiliza essa informação para identificar o recurso. O trabalho, no entanto, possui algumas limitações que impedem a generalização dos resultados. Primeiramente, a abordagem é direcionada apenas para vídeos, descartando os demais tipos que materiais abertos podem assumir, como textos, imagens e livros digitais, por exemplo. Além disso, a proposta dos autores funciona apenas em recursos do idioma inglês e utiliza recursos de uma única plataforma.

Por fim, o sexto estudo (S06) identificado considera a recuperação de REA em um contexto amplo. De modo geral, Kastrati, Imran e Kurti (2019) demonstraram o uso de uma rede neural convolucional para identificar e recuperar recursos da área de Ciência da Computação. A abordagem dos autores, porém, é focada apenas em um único tipo de recurso - vídeos - e desconsidera os demais formatos que um REA pode assumir. Cabe destacar ainda que os autores usaram apenas uma plataforma aberta e um número reduzido de recursos. Para facilitar a compreensão sobre esses desafios e os problemas identificados nos demais estudos, a Tabela 4 apresenta um resumo sobre todas as limitações identificadas.

Tabela 4 – Principais limitações dos estudos recuperados no MS

| **Estudo** | S01 | S02 | S03 | S04 | S05 | S06 |
|---|---|---|---|---|---|---|
| *1) Funciona em apenas uma plataforma aberta* | ✓ | ✓ | | ✓ | ✓ | ✓ |
| *2) Funciona com somente um tipo de REA* | ✓ | | | | ✓ | ✓ |
| *3) Funciona apenas com REA em inglês* | | | ✓ | | | ✓ |
| *4) Alto custo de processamento* | | ✓ | ✓ | | | |
| *5) Sem validação empírica com usuários* | ✓ | ✓ | | ✓ | ✓ | |
| *6) Considera aspectos de reuso* | | | | | | |

Fonte: Dados da Pesquisa.

Inicialmente, destaca-se que grande parte das abordagens para detecção e identificação

de REA para programação funcionam em apenas uma plataforma, no entanto, existem diversas fontes digitais em que tais materiais podem ser armazenados. Além do mais, grande parte das soluções atualmente disponíveis são direcionadas para somente um tipo de REA, o que não corresponde a heterogeneidade que esses materiais podem assumir.

Outro aspecto diz respeito à limitação das abordagens em relação ao cenário real de utilizam. Muitos estudos são apenas protótipos, sem a validação de usuários, voltados a um único idioma ou com alto custo de processamento. Grande parte dos estudos analisados ((RUIZ-INIESTA; JIMÉNEZ-DÍAZ; GÓMEZ-ALBARRÁN, 2014), (MOURIÑO-GARCÍA *et al.*, 2018), (TOVAR; CHAN; REISMAN, 2017), (KASTRATI; IMRAN; KURTI, 2019) e (DESSì *et al.*, 2019)) detectam e recuperam REA genéricos de Ciência da Computação usando categorias amplas como nome de disciplinas, deixando de lado os desafios que os alunos e professores encaram ao aprender ou ensinar programação.

É importante salientar que os estudos analisados não lidam com alguns problemas elementares sobre o processo de identificação e de recuperação de REA de ensino de programação. A seguir, essa discussão é detalhada.

### *Falta de padronização*

Conforme descrito por Becker e Quille (2019), existe uma grande dificuldade em separar o que é voltado ao ensino introdutório de programação e o que não é. Muitas vezes os REA possuem materiais introdutórios, mas são classificados como avançados. Ou ainda, possuem conteúdos avançados (como estrutura de dados), mas estão organizados como sendo introdutórios.

Essas características do ensino de programação são absorvidas nos REA. É comum identificar materiais que estão classificados em categorias aparentemente equivocadas ou com conteúdos que misturam elementos introdutórios com tópicos avançados. Por exemplo, o REA intitulado Aprenda programação gratuitamente pertence à área do conhecimento denominada *tecnologia*. Já o REA entitulado Estrutura Condicional if, pertence à área do conhecimento *Introdução à Programação*. Todavia, ambos os materiais possuem conteúdo voltado ao ensino de programação introdutória e estão disponiveis na mesma plataforma digital.

### *Linguagens e paradigmas de programação*

Algumas dimensões envolvidas no ensino e aprendizado de programação introdutória são interfaces do desenvolvimento de software. Em especial, duas delas se destacam: o uso de linguagens de programação e a aplicação de paradigmas de programação.

Em REA voltados aos estudantes que estão aprendendo a programar, é comum a aplicação de alguma linguagem de programação, como Java ou C, ou a escolha de algum paradigma de programação. Mas, esse comportamento é similar em recursos que produzem algum tipo de software, como projetos acadêmicos, pesquisas, entre outros.

Para simular como isso pode impactar a busca por materiais relevantes, é possível analisar o comportamento da busca da plataforma EduCapes[9]. Após realizar uma pesquisa pelo termo *Java*, foram apresentados 2.754 resultados Os seis primeiros resultados eram de recursos digitais que utilizam Java como uma tecnologia e apenas o sétimo apresentava uma aula digital que mostra como criar uma conta em uma plataforma de programação e acessar o curso de programação java em blocos.

### *Colisões com outras áreas do conhecimento*

Muitos tópicos introdutórios de programação também são comuns em outras áreas do conhecimento, gerando diversas colisões durante as buscas. O termo *variável*, por exemplo, é muito comum no ensino de programação, mas também possui uso em diversas áreas e/ou disciplinas, como em matemática e estatística.

A Figura 2 ilustra esse cenário. Foi realizada uma busca com o termo *loop* na plataforma Educapes e os três primeiros resultados estão relacionados a Computação, Engenharia e Física.

Figura 2 – Colisões de áreas e disciplinas distintas


Fonte: Extraído da plataforma EduCapes.

### *Estrangeirismo, Empréstimo e Neologismo*

O ensino de programação sofre grande influência do Inglês. Muitos termos e palavras reservadas são originais desse idioma. A consequência para os REA é o estrangeirismo (palavras

[9] Busca realizada no dia 03 de Julho de 2024.

que se originam em uma língua estrangeira sem correspondente na língua falada), Empréstimo (palavras que sofrem poucas modificações e passam a fazer parte do léxico) e Neologismo (surgimento de palavras novas) (GONÇALVES *et al.*, 2011).

Um exemplo comum que pode ocorrer é um(a) professor(a) empregar durante a aula o termo *laço de repetição*, mas adotar em seu material educacional o termo *loop*. Ou ainda, apresentar conceitos relacionados ao *debug*, mas referindo-se como sendo um ato de *debugar o código*. Um iniciante em programação ou até mesmo professores com conhecimento na área podem não identificar resultados relevantes devido à presença de tais estrangeirismos, empréstimos e neologismos nos materiais abertos.

Diante desses desafios, bem como da falta de soluções relevantes nos estudos selecionados, tais características foram incorporadas no desenvolvimento deste trabalho. Assim, os mecanismos propostos para identificarem e recuperarem REA de programação introdutória buscam contribuir para a literatura minimizando ou reduzindo as lacunas que os trabalhos atuais possuem.

Em conclusão ao MS, cabe ressaltar que o estudo original foi publicado por Deus e Barbosa (2022) na *IEEE Transactions on Education*. Para a escrita desta tese, o conteúdo original foi atualizado com a inserção de novas análises e a revisão do texto. Além do mais, este capítulo apresentou apenas uma síntese do trabalho original.

## 2.4 Considerações Finais

Este capítulo introduziu os fundamentos sobre REA e programação introdutória. Além do mais, também foi apresentada uma lista de trabalhos relacionados que buscam resolver partes do problema referente ao processo de identificar e recuperar um REA relevante.

Com base nos resultados obtidos e na discussão proposta, foi possível notar que muitas soluções atuais possuem limitações importantes. Em geral, tais limitações impedem ou viabilizam que o processo de identificação e de recuperação de um REA seja uma tarefa simples para estudantes e professores de programação introdutória. Muitas desses desafios são originadas na forma em que plataformas abertas indexam os REA. Por isso mesmo, essa discussão é aprofundada a seguir.

CAPÍTULO

3

# O USO DE METADADOS PARA IDENTIFICAÇÃO E RECUPERAÇÃO DE REA

O objetivo deste capítulo é apresentar como diferentes plataformas abertas estão identificando e recuperando REA. Em um sentido mais amplo, pretende-se compreender as barreiras e problemas comuns que existem em diferentes plataformas que armazenam os REA.

Para isso, foi realizado um estudo exploratório para analisar o impacto da identificação e da recuperação de REA em diferentes plataformas abertas. O estudo original é apresentado a seguir:

- Deus, W. S. de; Barbosa, E. F. *"The Use of Metadata in Open Educational Resources Repositories: An Exploratory Study"*. **44th Annual Computers, Software, and Applications Conference**, 2020, Espanha. (COMPSAC'2020). https://doi.org/10.1109/ COMPSAC48688.2020.00025

Este capítulo é uma síntese do trabalho original, com o conteúdo traduzido e atualizado. O restante do capítulo encontra-se organizado da seguinte forma: a Seção 3.1 detalha a preparação para o estudo exploratório. A Seção 3.2 apresenta os resultados alcançados. A Seção 3.3 discute as ameças à validade e a Seção 3.4 conclui o capítulo.

## 3.1 Design

O estudo exploratório foi conduzido em diferentes plataformas abertas. Para isso, foram investigadas quais informações estão disponíveis para identificar um REA, os padrões de preenchimento e a consistência dos dados. Logo a seguir, foi investigado o impacto dos metadados no processo de recuperação dos REA. O design da pesquisa envolveu duas grandes etapas: a seleção das plataformas abertas e a coleta dos dados.

### 3.1.1 Seleção de plataformas abertas

Para realizar a primeira etapa de investigação, foram estabelecidos os critérios para selecionar quais iniciativas seriam analisadas. Para isso, três critérios foram definidos, como mostra a Tabela 5.

Tabela 5 – Critérios de seleção de plataformas abertas

| Código | Descrição |
|---|---|
| $CS_1$ | Selecionar apenas plataformas com conteúdo interdisciplinar |
| $CS_2$ | Selecionar apenas plataformas ativas |
| $CS_3$ | Selecionar apenas plataformas com mecanismos de busca aberta |

Fonte: Dados da Pesquisa.

O primeiro critério de seleção ($CS_1$) foi estabelecido para evitar que plataformas com apenas uma área de conhecimento fossem analisadas, inserindo algum tipo de viés nos resultados. Já o segundo critério ($CS_2$) foi definido para evitar que coleções com dados corrompidos ou desatualizados fossem analisados. Por fim, o terceiro critério de seleção ($CS_3$) evitou que plataformas restritivas (como os que necessitam de *tokens*, *login* ou outros tipos de permissões) fossem selecionadas, tendo em vista que todo o processo de busca foi automatizado. Ou seja, também foram removidas as plataformas que ofereciam algum tipo de barreira para a busca, como carregamento assíncrono ou sob demanda.

Inicialmente, as plataformas identificadas no MS (Seção 2.3) foram selecionadas para análise. A seguir, foram feitas buscas na web para identificar novas plataformas abertas. Após finalizar as buscas, todas as plataformas identificadas foram filtradas de acordo com os critérios apresentados na Tabela 5. Ao final desse processo, oito plataformas foram selecionadas, como mostra a Tabela 6.

Tabela 6 – Lista das Plataformas Selecionadas

| Plataforma | Link |
|---|---|
| LivreSaber | http://livresaber.sead.ufscar.br:8080/jspui/ |
| RA | https://repositorioaberto.uab.pt/ |
| Metafinder | https://oer.deepwebaccess.com/oer/desktop/en/search.html |
| iOER | http://ioer.ilsharedlearning.org/ |
| Temoa | http://temoa.tec.mx/node |
| Merlot | https://www.merlot.org/merlot/index.htm |
| OER Commons | https://www.oercommons.org/ |
| Skills Commons | https://www.skillscommons.org/ |

Fonte: Elaborado pelo autor da pesquisa com os links de acesso. No momento da escrita da tese, o endereço da plataforma Temoa estava *offline*. Mesmo assim, optou-se por manter o link consultado, tendo em vista que a plataforma pode voltar a funcionar no mesmo endereço.

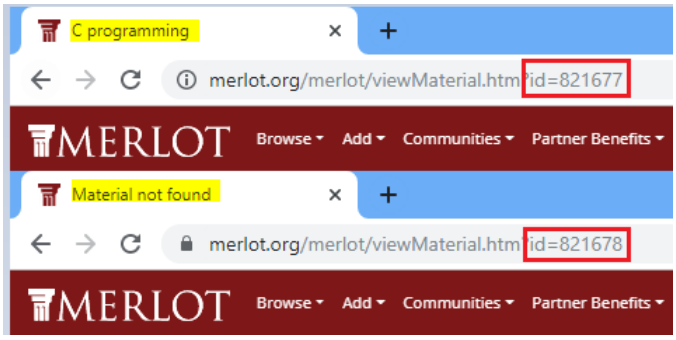
### 3.1.2 Coleta dos Dados

Todo o processo de busca foi automatizado. Para isso, foi criado um *web crawler*[1] capaz de navegar nas coleções e extrair os metadados dos recursos. O *web crawler* foi construído para realizar duas tarefas distintas: a primeira foi visitar a plataforma de REA e a segunda foi extrair os metadados de cada recurso encontrado.

Para a primeira tarefa, a ferramenta foi programada para realizar requisições síncronas em uma plataforma por vez, de acordo com um intervalo de tempo. A ferramenta fazia uma requisição para a plataforma, aguardava o intervalo de tempo definido para receber a resposta, e somente após esse intervalo realizava outra requisição. Esse comportamento foi programado para preservar o desempenho de cada plataforma. Caso a ferramenta realizasse diversas requisições assíncronas em intervalos curtos, isso poderia acarretar perda de desempenho.

Já para a realização da segunda tarefa, o *web crawler* fazia uma requisição para a plataforma usando como base um identificador aleatório[2]. Caso a página retornasse um resultado válido, o *web crawler* extraía os metadados da página. Caso contrário, um novo identificador sequencial era gerado. Essa dinâmica foi selecionada após notar que todas as plataformas utilizavam um identificador numérico na URL para identificar um recurso. Na Figura 3 é ilustrado como isso aconteceu no Merlot. No primeiro caso, o parâmetro $id = 821677$ retornou um REA nomeado *"C programming"* enquanto $id = 821678$ retornou um aviso de *"Material not found"* (Material não encontrado).

Figura 3 – Identificando possíveis REA para coleta

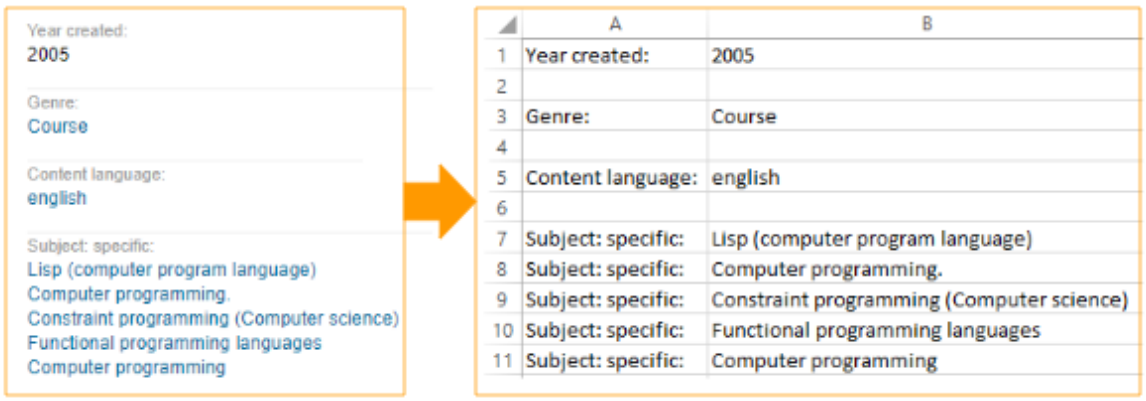



Fonte: Dados da Pesquisa.

Ao final da coleta, os dados foram exportados em uma planilha eletrônica com a extensão *.csv*. A planilha apresentava a chave e o valor do metadado, além da origem. Um exemplo desse procedimento é apresentado na Figura 4.

[1] Software que acessa e extrai dados de páginas web.
[2] O identificador era numérico, enviado por parâmetro.

Figura 4 – Exemplo de coleta dos dados




Fonte: Dados da Pesquisa.

## 3.2 Resultados

O processo de coleta de dados identificou 63.783 REA, o que gerou uma lista contendo 1.243.938 metadados, como mostra a Tabela 7.

Tabela 7 – Resumo da Extração

| **Plataforma** | **Metadados extraídos** | **REA extraídos** |
|---|---|---|
| LivreSaber | 37.781 | 3.538 |
| RA | 37.964 | 1.348 |
| Metafinder | 98.394 | 3.586 |
| Temoa | 146.908 | 9.948 |
| Skills Commons | 175.714 | 6.246 |
| Merlot | 208.852 | 12.634 |
| iOER | 238.263 | 10.182 |
| OER Commons | 300.062 | 16.301 |
| **TOTAL** | **1.243.938** | **63.783** |

Fonte: Dados da Pesquisa.

### *3.2.1 Como os metadados são utilizados para identificar os REA nas plataformas abertas?*

Inicialmente, observou-se que as plataformas adotavam diferentes organizações e estruturas em seus metadados. Frequentemente, a mesma plataforma apresentava conjuntos de metadados distintos. Na prática, isso representa que dois recursos oriundos de uma mesma coleção podem ter metadados diferentes. Esse comportamento é ilustrado na Tabela 8, na qual metadados de três recursos do iOER são comparados.

O que chama a atenção não é apenas o fato dos metadados não se repetirem. Mas, que muitos metadados não repetidos foram usados como base pelos trabalhos que procuram fornecer abordagens de identificação e recuperação de REA. Por exemplo, os metadados de *Subject* e

Tabela 8 – Comparação entre metadados

| **Recurso 01** | **Recurso 02** | **Recurso 03** |
|---|---|---|
| Access Rights | Access Rights | Access Rights |
| Created on | Created on | Created on |
| Creator | Creator | Creator |
| description | description | description |
| End User | keywords | Language |
| Language | Language | Media Type |
| Publisher | Media Type | Publisher |
| Resource Type | Publisher | Rights |
| Rights | Resource Type | Submitter |
| Subject | Rights | title |
| Submitter | Submitter | |
| title | title | |

Fonte: Dados da Pesquisa. Em vermelho, os metadados que não se repetem.

*Keywords* foram utilizados em diversos estudos analisados no MS, mas eles não estão presentes em todos os REA.

Além disso, cada plataforma possui uma estrutura de classificação própria e, por conta disso, não existe muita equivalência entre plataformas distintas. Ou seja, em muitas situações uma plataforma apresenta um metadado que outra não possui. É até mesmo uma tarefa complicada encontrar alguma equivalência entre eles. Um exemplo ocorre quando se observa o total de metadados por plataforma. O Temoa, por exemplo, nos dados analisados, utilizou no máximo 14 metadados para descrever um recurso enquanto o RA utilizou até 47. Ou seja, uma discrepância que coloca em evidência as diferenças estruturais e notacionais das plataformas identificadas a partir de seus metadados, como mostra a Tabela 24 exposta no Apêndice A.

Quais são os padrões utilizados nos metadados?

Após identificar os tipos de metadados disponíveis, todos os metadados foram agrupados conforme as suas principais características, emergindo assim dois grupos: Genéricos (metadados com informações comuns em todos ou na maioria dos REA, como título, descrição, etc...) e Específico (metadados que existem em REA específicos, como o número de páginas, ISBN, etc...).

**Metadados Genéricos:** Neste grupo foram classificados 112 metadados. Algumas plataformas, no entanto, usam chaves repetidas ou muito semelhantes. Por exemplo, o iOER usa a chave *rights* e *access rights* para descrever as permissões do recurso, sem deixar evidente qual é a diferença. Da mesma forma, o LivreSaber adota *citation_language* e *DC.language* para descrever o idioma do recurso e o idioma do recurso de citação. Ao total, dez tipos genéricos de metadados foram mapeados:

1. *Palavra-chave:* Metadados que identificam recursos de forma genérica, expondo caracte-

rísticas, finalidades ou contextos.

2. *Data:* Todas as plataformas forneceram metadados para a *data*. Mas os padrões eram diferentes, em alguns casos, era usado valor numérico ou textual, em outros apenas o ano era apresentado.

3. *Descrição:* A *descrição* também foi comum. Muitas vezes, a *descrição* tinha o mesmo valor que o *título* ou uma versão mais longa, sendo referido por vezes como *abstract*. Vale ressaltar também que foram encontradas descrições longas contendo mais de 32.767 caracteres.

4. *Tipo:* Este grupo organiza metadados sobre o *tipo de recurso* e foi encontrado em todas as plataformas. Notam-se algumas dificuldades de padronização em sua exibição, mesclando estruturas análogas, como *"web"* ou *"HTML"*.

5. *Título:* O *título* também foi uma ocorrência comum nos metadados. Provavelmente *title* é um metadado obrigatório e por isso teve uma alta incidência no conjunto de dados.

6. *Assunto:* Formado metadados muito semelhantes às *palavras-chave*. Algumas, plataformas usam *subject* para organizar termos predefinidos enquanto as *keywords* estão abertas para preenchimento pelo usuário.

7. *Autor:* Formado por metadados contendo os nomes dos autores dos recursos. Em alguns casos possuía o termo *"submitter"* ou *"contributor"*.

8. *Idioma:* O idioma esteve presente em muitos metadados, com diferentes padrões, como abreviaturas e nomes compostos.

9. *Licença:* Neste grupo, foram organizados os metadados sobre uso, acesso, direitos de compartilhamento e licenças/regulamentos gerais.

10. *Formato técnico:* Chaves de metadados que exibem o formato do recurso. A extensão do arquivo é geralmente exposta.

**Metadados Específicos:** Metadados presentes em recursos específicos, variando conforme cada plataforma e com muita informação segmentada. Diante disso, optou-se por classificá-los em apenas três grupos:

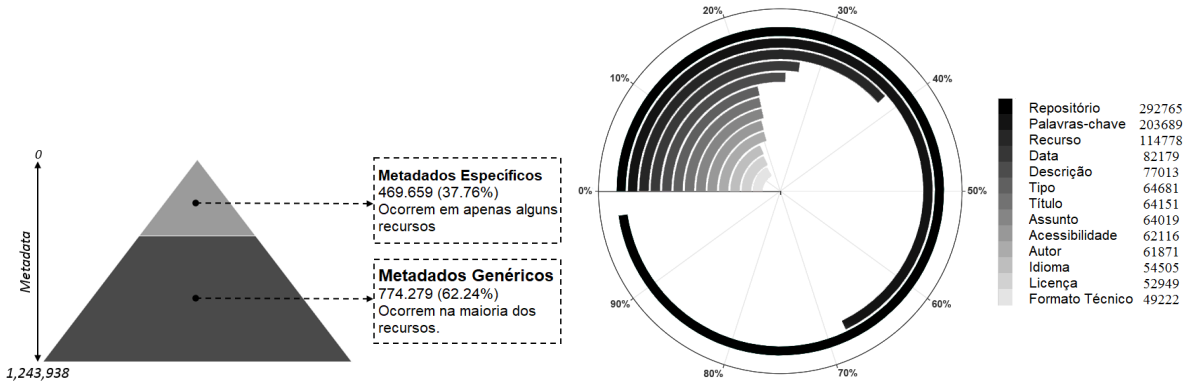
1. *Recurso:* Metadados que existem devido ao tipo de recurso. Exemplos desses metadados são: *DOI* (identificador de objeto digital) e *Grades* (nível educacional/série).

2. *Repositório:* Metadados usados na plataforma. Por exemplo, o Metafinder adota o metadado *deepRank* para classificar seus recursos. O Temoa, por sua vez, possui uma chave de metadados *approval status* para mostrar o status de aprovação do material.

3. *Acessibilidade:* Metadados que abordagem questões de acessibilidade, como *available accessibility information* e *accessibility statement*.

A Figura 5 mostra a proporção e a ocorrência dos metadados com o agrupamento realizado. Como pode ser visto, a grande maioria das chaves de metadados são sobre *repositório, palavras-chave* e *recurso*. Isso evidencia que as plataformas apresentam informações e estruturas diferentes. Ademais, elementos importantes dos REA, como *idioma, formato técnico* e *licença* estão sendo ignoradas, pois esses metadados tiveram uma ocorrência menor do que o total de REA analisados (63.783).

Figura 5 – Frequência dos metadados




Fonte: Dados da Pesquisa.

### 3.2.2 Existe um padrão de metadados entre plataformas distintas?

Conforme os dados analisados, não foi possível estabelecer um padrão de preenchimento nos metadados. Por exemplo, os metadados *palavras-chave* e *assunto* apresentam dois padrões: simples (33%) ou agregado (67%). No primeiro caso, os valores são escritos empregando um termo simples como *"web"* ou *"software"*. No segundo caso, dois ou mais termos são usados, como *"engenharia de software"* ou *"programação java"*.

Em padrões agregados, existem muitos estilos para separação dos termos. Os mais comuns foram o uso de espaço simples (" "); vírgula (","); "E" comercial ("&"); e traço ("-"). No entanto, é muito comum que os metadados tenham dois ou mais padrões em seu valor. Como ilustração, o seguinte texto foi encontrado em um metadados: *"Design de Banco de Dados - Normalização, Administração de Banco de Dados, Abordagens de Gerenciamento de Banco de Dados & Data Warehousing"*. Considerando isso, verificou-se que 18% desses valores adotam espaço e vírgula; 15% usam espaço e "E" comercial; e 10% aplicam espaço e linha simples para agregar valores. Os valores restantes assumem diferentes formas, misturando esse formato com outros.

A implicação mais óbvia disso tudo é a operação dos mecanismos de pesquisa sobre estruturas tão heterogêneas de preenchimento. É muito comum, por exemplo, usar apenas o

espaço simples para a separação de dados. Mas, diante desse cenário, é difícil compreender como é o desempenho de busca.

A *data* também foi um tipo de metadado que sofreu grande variação. Foram identificados 82.179 metadados que apresentavam alguma informação sobre a data do recurso usando marcações diferentes, como data de submissão, data de revisão do recurso e data de disponibilização.

O preenchimento mais comum nas diferentes plataformas foi o ano. Paralelamente, também foi utilizado o mês, mas foram encontrados dois formatos: textual (janeiro, fevereiro, etc...) e numérico (01, 02, etc...). As frequências e as ocorrências são mostradas na Tabela 9.

Tabela 9 – Resumo dos padrões de data

| **Padrão (Exemplo)** | **Ocorrência** | **Frequência (%)** |
|---|---|---|
| 16/06/2011 | 29,950 | 36.445% |
| April 4 2017 | 25,279 | 30.761% |
| 1995 | 13,525 | 16.458% |
| Invalid | 6,142 | 7.474% |
| 2011-06-16T12:10:21Z | 4,468 | 5.437% |
| 2015-07 | 1,090 | 1.326% |
| Y:1875 | 919 | 1.118% |
| D:2013-01-01 | 317 | 0.386% |
| D:August 13, 2019 | 239 | 0.291% |
| 1875-01-01 | 213 | 0.259% |
| M:June 2010 | 25 | 0.030% |
| Fall 2016 | 7 | 0.009% |
| M:Autumn 2005 | 3 | 0.004% |
| Y:JANUARY - JUNE 2013 | 2 | 0.002% |
| **TOTAL** | **82,179** | **100%** |

Fonte: Dados da Pesquisa.

Similarmente ao que foi reportado em *palavra-chave* e *assunto*, o *tipo* também possui 81% das ocorrências um metadados simples, como *"Lesson"* ou *"Assessment"* enquanto 19% têm dois ou mais valores, como *"Activity/Lab"* ou *"FullCourse, LectureNotes"*.

O grupo de *autores* tem como padrão predominante o nome do autor ou instituição (97%) e uma pequena parcela (3%) apresentando o e-mail do autor.

Paralelamente, o *idioma* foi estruturado em três padrões principais: com o nome completo *("Português, Inglês...")* (88%); com abreviação sublinhada *("pt_BR, en_EN...")* (7%); e com abreviação simples *("por, eng...")* (3%). Em 2% dos *idiomas* foi encontrado um valor considerado inválido, como números.

Também foram identificados problemas relacionados aos metadados de *licença*. Geralmente, os metadados apresentavam licença *Creative Commons* ou licenças abertas (74%). Porém, 13% exibiram que a licença estava em arquivo externo; 7% estavam vazios ou desconhecidos; e 6% eram customizados.

Os grupos *repositório* e *recurso* apresentaram características semelhantes, variando seu

formato seguindo suas próprias estruturas. O Metafinder é um bom exemplo disso, usando o metadado *page* com o número entre um traço (“927-932”). Já nos dados analisados, a plataforma RA usava dois metadados distintos para mostrar a *primeira página* e a *última página* de um recurso.

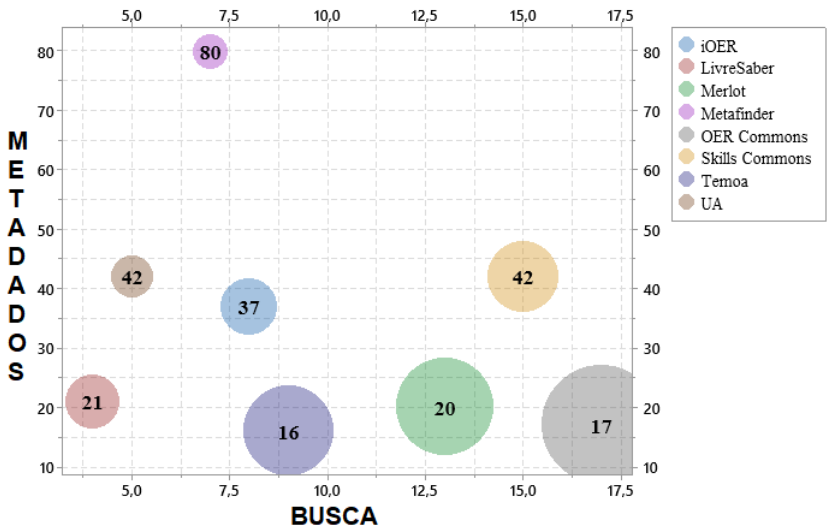
Finalmente, os outros grupos apresentaram formatos primitivos, como texto e números, derivados, em sua maioria, do idioma de origem do recurso. Assim, um desafio importante é traduzir o conteúdo para explicar os valores e padrões.

### 3.2.3 Quais são as principais implicações dos metadados para o processo de recuperação de REA?

A falta de padrão nos metadados influencia diretamente o processo de recuperação de REA. Para identificar alguns impactos, inicialmente, os metadados identificados nos REA foram comparados com os campos disponíveis para buscas nas plataformas. Ao todo, foram identificados 280 metadados. No entanto, os mecanismos de pesquisa das plataformas suportam pesquisas em apenas 78 opções. Isso representa que apenas uma porção (cerca de 28% dos metadados) pode estar sendo usada durante as pesquisas.

Constatou-se que apenas uma plataforma suporta pesquisas em todos os metadados recuperados nesta investigação, enquanto nos outros a cobertura de recuperação é muito pequena. A Figura 6 detalha os metadados por campos de buscas, exibindo o total de metadados identificados na plataforma (METADADOS) considerando o total de opções de busca em cada plataforma (BUSCA).

Figura 6 – Metadados e os campos de busca disponíveis



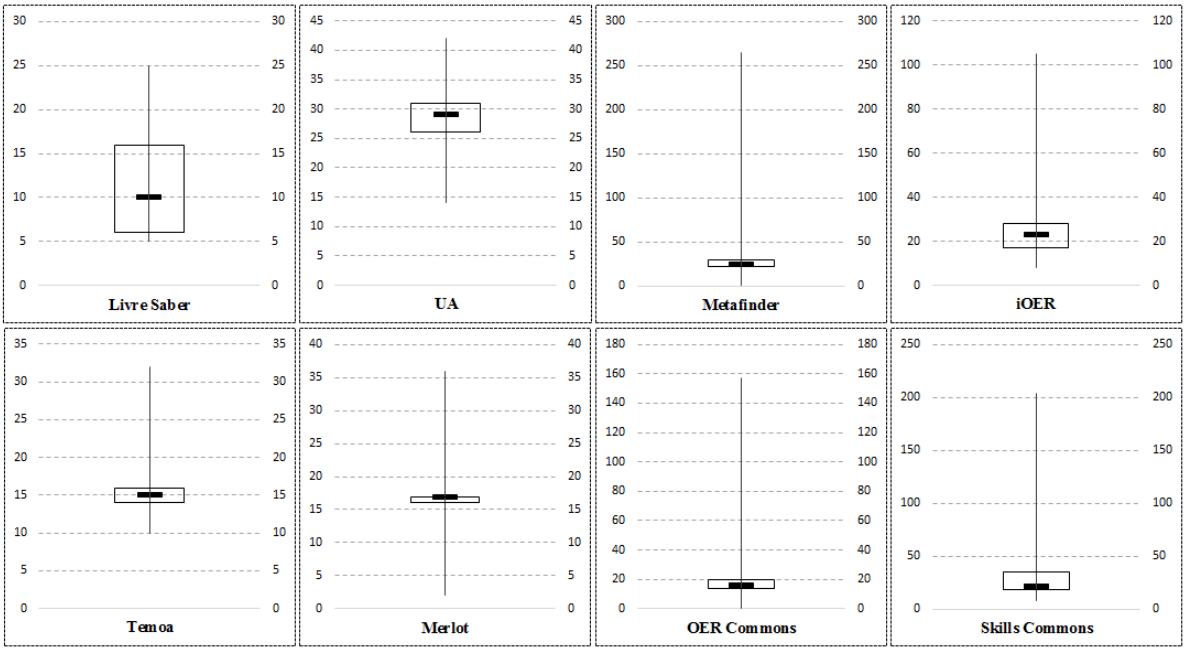

Fonte: Dados da Pesquisa.

Além disso, muitas vezes o mesmo recurso possui metadados repetidos. Esse comportamento aparece quando uma plataforma ao invés de usar um único metadado adota uma lista de

metadados para apresentar os valores. Isso acontece em diversas situações, como em recursos que apresentam a descrição no idioma de origem e outra descrição, em inglês. Ou ainda, quando ao invés de listar todas as palavras-chave em um único metadado, adota-se uma lista.

Todavia, o contrário também ocorre. Como as plataformas não adotam metadados mínimos por recurso, também foram encontrados REA com poucos metadados de informação. A Figura 7 mostra a concentração de metadados por recurso em cada plataforma analisada. Como pode ser visto, há muita heterogeneidade, como o Metafinder, que possui mais de 250 metadados em um único REA. Mas, também possui recurso com apenas 1 metadado.

Figura 7 – Visão geral dos metadados por recurso




Fonte: Dados da Pesquisa.

O comportamento anômalo descrito anteriormente gera três desafios para a recuperação de REA:

**1) Problema de amplitude:** Diferença entre o mínimo e o máximo de metadados por recurso. Em resumo, plataformas com problema de amplitude possuem materiais muito descritivos (acumulando vários metadados por recurso) e também materiais com poucas informações (apresentando poucos metadados por recurso). Como consequência, é comum que as buscas retornem resultados ineficazes, já que cada recurso pode ter um número diferente de metadados e os mecanismos de busca precisam se adaptar nesses diferentes contextos. Quatro plataformas apresentaram esse problema: *iOER, OER Commons, Skills Commons* e *Metafinder*. O *iOER* obteve uma amplitude de 97. *OER Commons* e *Skills Commons* atingiram a amplitude de 156 e 196, respectivamente. O pior caso detectado foi o *Metafinder*, com amplitude de 264.

**2) Problema de descrição com poucos metadados:** Ocorre quando o valor mínimo de metadados por recurso é muito baixo, impossibilitando a descrição do um recurso. Nesse

contexto, foram encontradas três plataformas: *OER Commons, Metafinder* e *Merlot*. O *OER Commons* e o *Metafinder* apresentaram o mesmo valor mínimo, apenas 1 metadado. Ou seja, existem recursos nessas plataformas que possuem apenas 1 metadado de informação. O *Merlot*, por sua vez, apresentou casos com no mínimo 2 metadados por recurso.

**3) Problema de descrição com muitos metadados:** Semelhante ao desafio anterior, esse problema também é derivado da amplitude. Porém, nesse caso, existem recursos com muitos metadados que retardam o processo de recuperação e se tornam ineficientes porque os mecanismos de busca não verificam todas as informações. As plataformas que apresentaram esse desafio foram: *iOER, OER Commons, Skills Commons* e *Metafinder*. A plataforma *iOER* obteve no máximo 105 metadados em um recurso simples. Em seguida, estava o *OER Commons* com 157 metadados. O *Skills Commons* e o *Metafinder* foram os piores casos, com até 204 e 265 metadados máximos por recurso, respectivamente.

Outro fator que merece atenção nas buscas de plataformas refere-se aos padrões adotados. Na seção anterior, foi apresentado como os metadados apresentam diversos padrões de preenchimento. Portanto, não é possível compreender como os mecanismos processam uma busca diante desses preenchimentos. Por exemplo, nos metadados *palavra-chave/assunto* há um problema devido aos termos complexos que são baseados em duas ou mais palavras, como *"engenharia de software"*. Como muitos metadados foram registrados sem aspas duplas, comumente as buscas de termos complexos trazem resultados não relevantes expondo resultados sobre *"software"* e *"engenharia"* que não estão necessariamente relacionados à busca realizada.

Também foram encontrados recursos com metadados incoerentes. Por exemplo, foi identificado um recurso com a *descrição* em alemão e o metadado de *título* em inglês. Essa situação também foi detectada para ocorrências de outros idiomas, como espanhol/inglês. Nesses casos, fica difícil entender qual será o comportamento do mecanismo de busca, pois o recurso possui metadados com idioma diferentes.

## 3.3 Ameaças à Validade

A principal ameaça nesta investigação é relacionada com o processo de extração e de conversão dos dados. Como a extração foi automática, pode ser que diferentes dados tenham sido corrompidos durante a busca do *web crawler* e a conversão dos dados para o formato *.csv*. Para mitigar esse risco, o *web crawler* apresentava um *log* com o identificador do recurso extraído. Caso houvesse alguma mensagem de erro ou problema durante a extração, os metadados referentes ao recurso eram removidos ou extraídos manualmente.

Conforme explorado na seção anterior, em muitos casos os valores dos metadados não seguiam um padrão. Isso gerou diversos desafios para a análise, pois os textos dos metadados continham marcadores, links ou textos extremamente longos. Como resultado, algumas células da planilha podem ter sido mal formatadas ou conter dados duplicados. Para reduzir esse problema,

as planilhas foram revisadas manualmente, selecionando células aleatórias para verificar se o conteúdo estava correto ou se era necessária alguma alteração.

Por fim, apesar do tamanho amostral representar uma base significativa de análise, é importante destacar que tudo isso é apenas uma fração das coleções de plataformas que armazenam REA. Pode ser que diferentes desafios e barreiras existam na aplicação dos metadados que não foram revelados nesta investigação. Para conter a dispersão desse risco no estudo exploratório, foram selecionadas plataformas abertas diferentes. Isso permitiu identificar que muitos desafios são comuns em duas ou mais plataformas, apoiando a generalização dos resultados alcançados.

## 3.4 Considerações Finais

Este capítulo forneceu uma visão geral sobre como os metadados são utilizados por diferentes plataformas abertas, bem como o impacto desses metadados para o processo de identificação e de recuperação de REA. Como exposto, existem diversos desafios sobre o padrão e o preenchimento dos metadados, o que exige uma solução integradora, capaz de funcionar em diferentes plataformas.

Para desenvolver tais soluções, também é essencial considerar a perspectiva dos usuários que buscam por REA. Por isso mesmo, a próxima etapa da pesquisa se concentrou em identificar os principais desafios e as possíveis soluções para usuários que pesquisam por REA de programação introdutória. Essa investigação é relatada a seguir.

CAPÍTULO

# 4

# IDENTIFICAÇÃO E RECUPERAÇÃO DE REA: DESAFIOS E SOLUÇÕES

Os objetivos deste capítulo articulam-se em duas linhas de ação. A primeira é apresentar uma lista dos principais desafios relacionados ao processo de identificação e de recuperação de REA voltados ao ensino e aprendizado de programação introdutória. A segunda linha de ação é evidenciar o que pode ser feito para minimizar, ou até mesmo resolver, os problemas mencionados. Esses resultados também estão disponíveis na seguinte publicação:

- Deus, W. S., Balbino, F. C., Barcelos, L. V., Barbosa, E. F. (2024). *"Fostering the Teaching and Learning of Computer Programming With Open Educational Resources: Challenges and Solutions"*. In C. Bosch, L. Goosen, & J. Chetty (Eds.), **Navigating Computer Science Education in the 21st Century (pp. 21-40)**. IGI Global. https://doi.org/10.4018/979-8-3693-1066-3.ch002

O conteúdo deste capítulo foi revisado, traduzido e atualizado. O texto encontra-se organizado da seguinte forma: a Seção 4.1 apresenta o percurso metodológico de pesquisa. A Seção 4.2 sumariza os desafios identificados bem como as possíveis soluções. A Seção 4.3 analisa as principais ameaças da pesquisa. Por fim, a Seção 4.4 apresenta as considerações finais do capítulo.

## 4.1 Design

A primeira etapa da investigação consistiu na elaboração da QP. Considerando o propósito desta tese e o desenvolvimento da pesquisa, a seguinte QP foi elaborada:

- **Quais desafios estão associados ao problema da identificação e recuperação de REA de programação introdutória?**

### 4.1.1 Coleta de dados

Para solucionar a QP previamente estabelecida, optou-se pela condução de um Grupo Focal com professores de programação introdutória. Para isso, foi estabelecida uma rede de colaboração em uma disciplina de pós-graduação do Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo. Os alunos da disciplina foram convidados a participarem e/ou convidarem colegas para um Grupo Focal sobre identificação e recuperação de REA de programação introdutória.

Para participar do Grupo Focal, o participante deveria ter experiência com o ensino e aprendizado de programação introdutória e disponibilidade de realizar as atividades de forma síncrona, com a supervisão de dois pesquisadores. Ao todo, cinco professores aceitaram participar do Grupo Focal. Uma síntese sobre a experiência e formação de cada um é apresentada na Tabela 10.

Tabela 10 – Experiência dos participantes

| Participante | Graduação | Mestrado | Doutorado |
|---|---|---|---|
| P1 | Ciência da Computação | Realidade Aumentada | Produção Virtual |
| P2 | Análise e Dev. de Sistemas | Engenharia de Software | Engenharia de Software |
| P3 | Sistemas de Informação | Processamento de Imagens/Sinais | Inteligência Artificial |
| P4 | Ciência da Computação | Inteligência Artificial | Engenharia de Produção |
| P5 | Sistemas de Informação | Inteligência Artificial | - |

Fonte: Dados da Pesquisa.

Um participante (P2) declarou ser aluno de doutorado e atuar como auxiliar de ensino de estudantes de graduação, tirando dúvidas e dando suporte em problemas de programação. Os demais participantes são professores de universidades públicas que ministraram disciplinas introdutórias de programação para a graduação.

Foram selecionadas duas plataformas de REA para a realização do Grupo Focal: o Merlot e o Mason OER Metafinder[1]. Essas plataformas foram selecionadas por possuírem um grande acervo de REA para programação introdutória e representarem duas tendências atuais em plataformas abertas: as buscas centralizadas e em portais. De forma resumida, os participantes do grupo focal realizaram cinco atividades:

1. Preparação: Nesta atividade, o participante foi apresentado ao contexto da pesquisa, ao conceito de REA e foram coletados dados sobre a sua experiência e formação.

2. Primeiro treinamento: A seguir, o Mason OER Metafinder foi apresentado a cada participante (interface, opções de pesquisa e filtros). Em seguida, o participante poderia acessar e navegar pelo Mason OER Metafinder e tirar dúvidas com os pesquisadores.

3. Primeira execução: Nesta atividade o participante deveria buscar um REA adequado para uma aula introdutória de programação. O REA deveria ter conteúdo focado no ensino

[1] https://oer.deepwebaccess.com/oer/desktop/en/search.htmlr

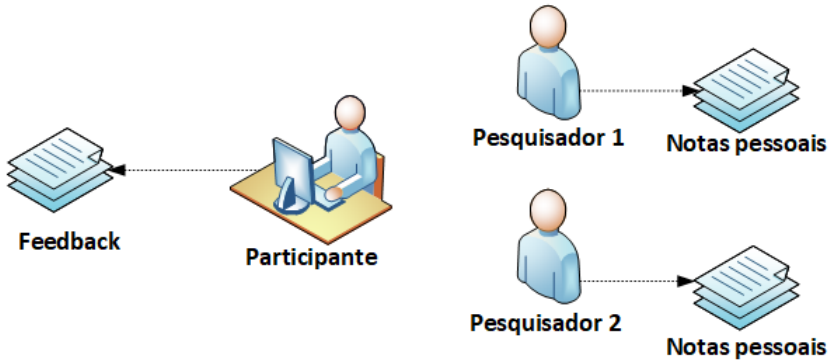
de *loops* (contendo elementos sobre *"while", "for"* ou *"do... while"*) na linguagem de programação Java. O participante então deveria realizar a busca por um recurso adequado, e depois comentar sobre a sua experiência.

4. Segundo treinamento: Esta atividade foi semelhante à atividade 2. Porém, o participante recebia o treinamento e realizava os testes no Merlot.

5. Segunda execução: Semelhante à atividade 3, em que o participante recebia a mesma tarefa de realizar a busca de um REA para o ensino de *loops* na linguagem de programação Java, mas a busca deveria ser no Merlot. Após finalizar a atividade, o participante deveria enviar o seu comentário também.

Os participantes realizam as atividades de forma individual, isto é, sem contato com outros participantes. Durante o Grupo Focal, dois pesquisadores observavam o comportamento das buscas de cada participante, realizando anotações sempre que necessário. Assim, os dados foram coletados por meio de duas fontes distintas: (1) as anotações pessoais feitas pelos pesquisadores durante as sessões, que capturavam expressões e observações sobre os participantes e (2) feedback gerado pelos próprios participantes após completarem as etapas 3 e 5. A Figura 8 resume esse processo.

Figura 8 – Estratégia para coleta de dados



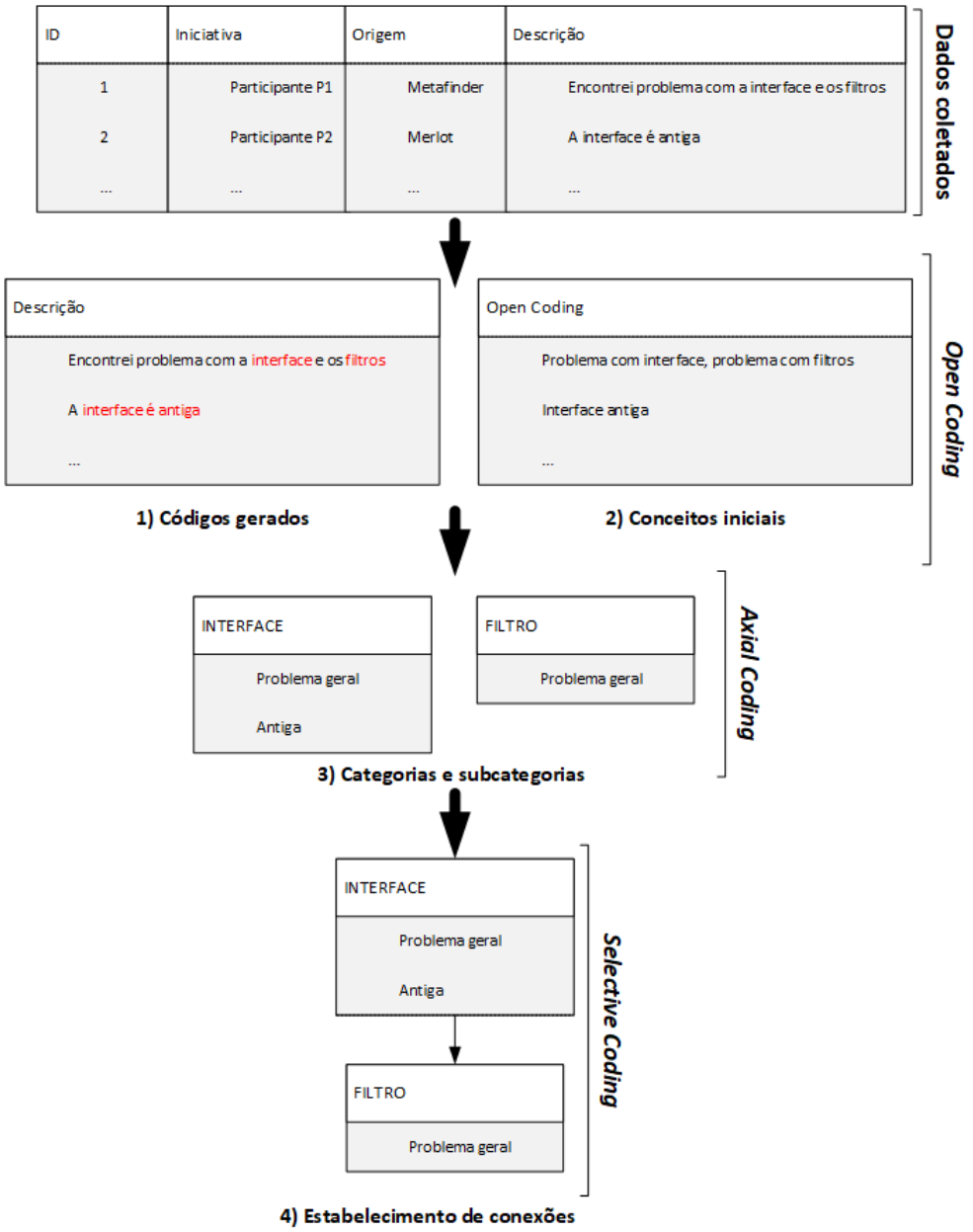

Fonte: Dados da Pesquisa.

A ideia de focar nas buscas em REA sobre *loops* em Java ocorreu tendo em vista que ambas as plataformas (Merlot e Mason OER Metafinder) possuíam resultados válidos na primeira página, bastando realizar uma busca simples sobre *programação, Java e Loops*. Além disso, deve-se destacar que o objetivo do grupo focal era analisar como os participantes realizavam suas buscas em plataformas distintas, para responder a QP previamente estabelecida. Por isso, o que estava sendo avaliado era todo o processo de busca, e não apenas se o participante conseguiria ou não identificar um material válido.

### 4.1.2 Análise de Dados

Para analisar os dados coletados, foram seguidos os pressupostos de Pandit (1996). Para isso, os dados foram organizados seguindo a ordem cronológica. Em seguida, os dados foram unificados em um único documento, acrescentando Identificador, Código do Participante (P1, P2, P3, P4 ou P5), Plataforma (Merlot ou Mason OER Metafinder), Origem (Pesquisador ou Participante) e Descrição. Com base nesse documento, foram utilizadas três estratégias de análise: *Open Coding, Axial Coding* e *Selective Coding*, como ilustrado na Figura 9.

Figura 9 – Exemplo do processo de análise de dados




Fonte: Dados da Pesquisa.

### Open Coding

O *open coding* (codificação aberta) foi a etapa inicial de análise dos dados. Essa atividade visa dividir os dados iniciais em partes menores, para examiná-las mais de perto (KHANDKAR, 2009). Os dados divididos então são observados detalhadamente, adicionando-se *códigos* para sintetizar o que aquele dado representa. Esse processo é responsável por gerar os primeiros conceitos, descritos por Pandit (1996) como sendo os blocos básicos na construção da Teoria Fundamentada.

Na prática, o *open coding* é o resultado da aplicação de um método comparativo de análise (KHANDKAR, 2009), produzindo como resultado uma lista de conceitos.

Nesse sentido, o *open coding* é um processo totalmente iterativo e exaustivo. Os dados iniciais, presentes no documento, são divididos e revisitados durante diversas iterações. A cada iteração, novos conceitos podem emergir, atualizando toda a lista. Esse processo só é encerrado após a exaustão da análise dos dados, ou seja, após o pesquisador constatar que nenhum novo conceito deverá emergir da análise.

### Axial Coding

O *axial coding* (codificação axial) é responsável por revisitar o resultado do *open coding*, para estabelecer conexões entre os conceitos inicialmente mapeados (PANDIT, 1996). Em resumo, os dados codificados são reorganizados agrupando-se conceitos semelhantes. Durante esse processo, os dados são refinados, alinhados e categorizados conforme propriedades gerais de um fenômeno e suas variações (CLIFF; MELISSA, 2017). Mais especificamente, a codificação axial resulta no desenvolvimento de um modelo mais detalhado e descritivo.

Pandit (1996) descreve esse modelo como um processo de desenvolvimento de categorias e subcategorias. Similar ao que ocorre com o *open coding*, o processo é totalmente iterativo e exaustivo, sendo encerrado apenas após a constatação de que nenhuma nova categoria ou subcategoria irá emergir.

### Selective Coding

O *selective coding* (codificação seletiva) é responsável por integrar as categorias em uma estrutura teórica (PANDIT, 1996). Em síntese, o produto final do *selective coding* é a criação de uma narrativa que descreve o fenômeno observado, identificando o ponto central de atenção, suas causas e o seu impacto.

Geralmente, uma categoria central é identificada e definida como o fenômeno principal. Com base nessa constatação, as categorias restantes são vinculadas ao fenômeno principal por meio de quatro conexões possíveis (PANDIT, 1996):

1. Condições causais: eventos que originaram o fenômeno;

2. Contexto: conjunto particular de condições;

3. Ação/Interação: respostas que ocorrem como resultado do fenômeno; e,

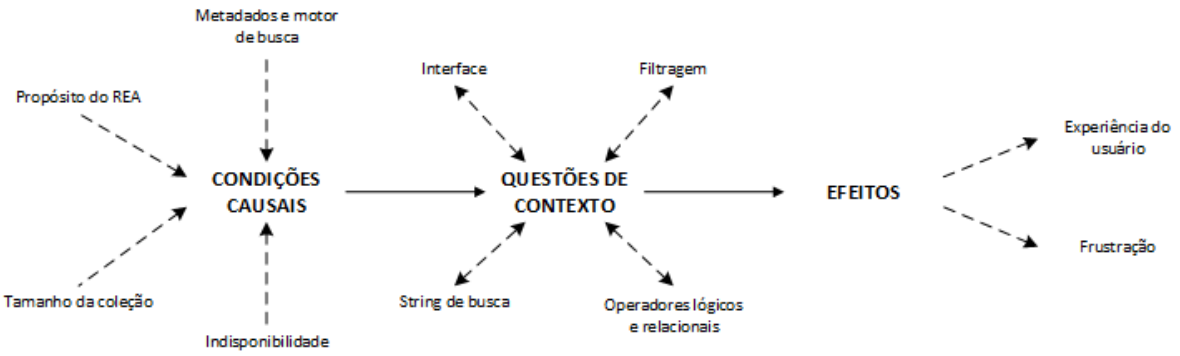
4. Resultados: consequências intencionais ou não intencionais.

Similarmente às atividades anteriores, o *selective coding* também é iterativo e exaustivo. Sua dinâmica exige a montagem e revisão de todo o conjunto de dados para propor uma teoria que seja aderente a realidade.

## 4.2 Resultados

Com base no procedimento de análise, foram identificados dez desafios que influenciam o processo de identificação e de recuperação de REA voltados ao ensino e aprendizado de programação introdutória. Os desafios foram agrupados em torno de três conexões: (1) condições causais, (2) questões de contexto e os (3) efeitos. A Figura 10 ilustra esse agrupamento.

Figura 10 – Síntese dos desafios mapeados




Fonte: Dados da Pesquisa.

### 4.2.1 Condições Causais

As condições causais ocorrem sem a interação do usuário com a plataforma aberta. Ou seja, sem o usuário sequer interagir (como fazer uma busca ou selecionar algum filtro) com a interface da plataforma. Esses desafios já estão presentes na própria coleção digital, causados, por vezes, durante o processo de elaboração do recurso, criação da plataforma ou armazenamento do recurso na coleção. São quatro os desafios relacionados às condições causais, detalhados a seguir.

#### *Propósito do REA*

Em tese, os REA são projetados e desenvolvidos para serem utilizados e adaptados por usuários com motivações distintas. Portanto, é fundamental identificar a finalidade de cada

REA, especificamente seu tema principal, os assuntos centrais cobertos pelo conteúdo e o seu público-alvo. Isso, no entanto, é uma tarefa muito onerosa de ser feita.

Um participante destacou o seguinte, evidenciando sucintamente o problema: *"[...] As palavras-chave podem ser utilizadas, porém, elas podem ser encontradas em qualquer lugar do REA, isso faz com que a busca encontre a palavra-chave em meio a sentenças que não refletem o objetivo real do REA"*.

A primeira interpretação pode parecer que o desafio relatado pelo usuário é relacionado ao uso de palavras-chave. Mas, o problema real está no trecho *"objetivo real do REA"*. Ou seja, é necessário considerar características elementares, como identificar os temas abordados por esse REA por meio de seus títulos, subtítulos e conteúdo. Isso foi notado com base em diversas observações, das quais as plataformas apresentaram resultados com finalidades diversas, incluindo recursos para outras áreas, como Física, Engenharia ou Matemática. Isso ocorria devido à frequente sobreposição de termos nesses diferentes campos e assuntos.

Esse problema é agravado por dois fatores. Em primeiro lugar, os usuários das plataformas abertas podem realizar um processo de classificação errôneo. Por exemplo, um recurso sobre "como declarar uma variável em Java" pode pertencer à programação de computadores, mas a sua classificação correta seria o ensino introdutório de programação. Em segundo lugar, muitos REA são interdisciplinares e cobrem dois ou mais temas ao mesmo tempo. O REA "como declarar uma variável em Java", como foi dito, poderia pertencer ao ensino introdutório de programação e também ao ensino de Java.

Identificar o real propósito de um REA não é uma tarefa fácil. Contudo, existem algumas inovações que podem ser adotadas por plataformas abertas, em especial uma delas se destaca, como exposto no Quadro 1.

Quadro 1 – Ação para facilitar a identificação do propósito do REA

| **Ação** | **Descrição** |
|---|---|
| #1 | Usar estruturas disponíveis nos REA para enriquecer e identificar qual é o real conteúdo do material, como o uso de termos em destaque, subtítulos e links externos. |

Fonte: Dados da Pesquisa.

### *Indisponibilidade*

REA podem ficar indisponíveis por uma série de fatores. Dois deles se tornaram muito presentes durante a investigação. O primeiro ocorria quando o link original do REA se tornava inacessível e o segundo quando alguma parte do recurso (como uma imagem ou vídeo) não era carregado/exibido.

Quase sempre, ambos os erros são causados por atualizações de hospedagem. Ou seja, o REA originalmente estava disponível em um local de acesso. Esse local foi modificado por

algum fator externo, como uma atualização do servidor ou do site. Como consequência, o recurso não é mais acessível.

O mesmo pode ocorrer com partes com compõem um REA. Um curso, por exemplo, pode conter uma série de vídeos hospedados em uma plataforma. Mas, a plataforma pode sofrer alguma atualização ou os autores dos vídeos podem mudar o nível de visibilidade, ocultando o conteúdo em outros sites.

Erros assim são bem comuns nos REA. Alguns feedbacks demonstram isso, como o participante que disse que o *"[Merlot] retornou resultados nulos por várias vezes"*. Outro participante também afirmou: *"Nenhum [REA] atendeu o requisito completamente, indisponível ou problemas assim"*.

O impacto disso é direto para os usuários. Ao realizar uma busca, espera-se que todos (ou pelo menos a maioria) dos resultados estejam disponíveis. Infelizmente, diversos REA retornados nas buscas estavam indisponíveis ou com partes inacessíveis. Isso desmotivava os participantes, expondo um grande falha na operação e manutenção do acervo.

Para reduzir problemas associados às propriedades de cada REA, diversas plataformas podem ser adotadas, entre elas destacam-se três, como mostra o Quadro 2.

Quadro 2 – Ações para reduzir o problema da indisponibilidade de recursos

| Ação | Descrição |
|---|---|
| #1 | Desenvolver um esquema específico de metadados, expandindo ou atualizando entradas específicas de acordo com a natureza de cada recurso. Assim, caso algum material possua dependências externas, pode ser sinalizado que partes daquele recurso podem ficar indisponíveis. |
| #2 | Usar links com identificadores persistentes. Dessa forma, os REA serão acessíveis, citáveis e facilmente verificáveis, reduzindo a ocorrência de recursos inválidos devido a alterações nos links originais. |
| #3 | Revisar as coleções adotando *web crawlers* para verificar quais materiais estão disponíveis e que devem ficar fora da listagem de resultados. |

Fonte: Dados da Pesquisa.

### *Tamanho da coleção*

Outro problema relatado pelos participantes foi relacionado ao tamanho dos acervos digitais. Tanto coleções grandes como pequenas apresentaram desafios para os participantes.

Coleções grandes, muitas vezes, produziam resultados ruins. Isso foi mencionado por um participante na seguinte forma: *"Talvez o grande volume de links possa ter influenciado na localização do material buscado [...]"*. Por outro lado, coleções com poucos REA também reduziram a possibilidade de identificação de recursos. Um participante afirmou: *"Devido à quantidade menor de links disponibilizados, tornou-se mais difícil a localização de material [...]"*.

O problema pode ser compreendido como algo estrutural dos REA. Grandes coleções digitais tendem a ter problemas de localização devido à falta de padronização de termos e classificação de recursos. Por outro lado, coleções pequenas tendem a apresentar poucos resultados e/ou pouca variação nos resultados após diversas alterações de pesquisa.

Para resolver problemas relacionados ao tamanho da coleção de REA, uma única ação pontual pode desempenhar um papel fundamental em coleções pequenas ou grandes, como exposto no Quadro 3.

Quadro 3 – Ação para reduzir o impacto do tamanho da coleção

| **Ação** | **Descrição** |
|---|---|
| #1 | Adotar a vinculação de REA nos resultados retornados. Em acervos pequenos, essa vinculação pode ser feita por meio de pesquisas em outros acervos de REA, aumentando assim o total de REA listados após uma pesquisa. Em acervos grandes, ao invés de mostrar uma lista imensa (como 20 ou 30 mil resultados) mostrar apenas uma porcentagem, vinculando os principais resultados conforme a busca do usuário. |

Fonte: Dados da Pesquisa.

### *Metadados e Motor de Busca*

Outro desafio identificado foi a dependência de metadados por parte de muitos motores de busca. Embora as plataformas de REA venham adotando essa estratégia, em muitos casos, a lista de metadados pode não ser eficiente para a correta identificação e recuperação do recurso.

Metadados, muitas vezes, contém uma série de informação do REA, como título, autor e descrição. Mas, essa lista pode conter informações erradas ou até mesmo incompatíveis com o conteúdo do REA. Por exemplo, um participante mencionou: *"[...] penso que existem sim bons REAS, mas estão em meio a vários outros que não atendem o que eu esperava."*

Os REA têm uma distinção significativa entre os outros tipos de materiais educacionais: REA podem ser editados mediante revisões e remixagens, gerando variações do conteúdo. Como resultado, um REA pode ter diversas formas e funcionalidades que não podem ser capturadas adequadamente apenas pelos metadados e pelos mecanismos de busca. Ademais, muitos REA são construções complexas, como livros abertos, cursos e sites acadêmicos. Por isso mesmo, a dependência apenas de metadados pode gerar gargalos na busca por REA introdutórios de programação. Para resolver esse problema, o Quadro 4 lista duas ações pontuais.

Quadro 4 – Ações para reduzir os desafios em metadados e nos motores de buscas

| **Ação** | **Descrição** |
|---|---|
| #1 | Adotar dados ou informações disponíveis nos recursos, como total de visualizações, comentários e recomendações. |
| #2 | Evitar a dependência única de metadados. Embora os metadados possam fornecer um grande apoio no processo de identificação e recuperação dos REA, também é essencial pensar em soluções globais, que foquem também no conteúdo do material. |

Fonte: Dados da Pesquisa.

### 4.2.2 Questões de Contexto

As questões de contexto compõem a segunda categoria que envolvem os desafios de identificação e recuperação dos REA. Nesse caso, os problemas estão alinhados ao perfil do usuário e como o usuário interage com a interface da plataforma. Os desafios emergem quando o usuário está, de fato, utilizando uma plataforma aberta para realizar o processo de busca dos REA. Quatro desafios foram mapeados nessa categoria.

#### *Interface*

Os participantes relataram dificuldades com as interfaces das plataformas. Por exemplo, um participante afirmou: *"A ferramenta [plataforma] possui muitos filtros na interface de pesquisa inicial"*. Outro participante mencionou: *"Tempo foi curto para adaptação a interface e refinar a string de busca para alcançar o objetivo desejado"*. Um pesquisador também observou casos de "cliques perdidos". Isso indica que os participantes clicaram em algo da interface (por exemplo, um botão ou aba) e, após ver o resultado, clicaram novamente para desfazer a ação ou retornar para a página anterior.

Como consequência, nota-se um problema relacionado à disposição dos elementos que compõem tais interfaces. Os usuários perdem tempo clicando em botões ou guias que não suportam efetivamente o processo de pesquisa e/ou demoram muito tempo até compreender o funcionamento correto da interface. Dianto disso, três ações podem reduzir o impacto das interfaces nas plataformas abertas, como apontado no Quadro 5.

Quadro 5 – Ações para reduzir os problemas de interface

| Ação | Descrição |
|---|---|
| #1 | Criar sites simples para os usuários. Isso envolve remover campos ambíguos, utilizar rótulos de fácil compreensão, evitar cores fortes ou chamativas, reduzir a sobrecarga de informações e evitar o uso de estruturas complexas, como muitas abas ou botões. |
| #2 | Remover qualquer assimetria. Isso significa adotar um padrão consistente para rotular campos de pesquisa, botões e filtros de toda a plataforma aberta. Por exemplo, se a página inicial possui o filtro "Licenças", esse padrão deve ser seguido de forma consistente nas demais páginas e na apresentação dos resultados, evitando variações, como "Tipo de Licenças" ou "Licenciamento". |
| #3 | Minimizar cliques desnecessários. Por exemplo, em vez de apresentar uma lista de licenças abertas para o usuário selecionar manualmente cada uma, é recomendado que todas as licenças sejam preenchidas por padrão. |

Fonte: Dados da Pesquisa.

#### *Filtragem*

O segundo desafio dessa categoria está relacionado aos problemas de filtragem de REA. Os participantes destacaram que os filtros atuais não são eficazes. Às vezes, eles não são compreensíveis ou são complexos de serem utilizados. Por exemplo, um participante afirmou que *"os filtros gerados [pelo Metafinder] não são muito eficientes"*. Outro participante disse que

*"A busca não possui forma de você escolher o tipo de REA que se deseja buscar"*. Além disso, barreiras sobre o uso dos filtros também foram apresentadas nos desafios anteriores.

Embora pareça que seja um problema simples relacionado à falta de filtros, o desafio real é o processo de filtragem dos resultados. Por exemplo, alguns participantes usaram o termo *"loop"* em suas pesquisas, enquanto muitos REA disponíveis foram escritos usando o sinônimo *"iteration"*. Infelizmente, nenhum filtro foi fornecido para ajudar os usuários a identificar recursos semelhantes. Esse problema é complexo porque as opções de filtros eficientes podem variar conforme a área de cada recurso, exigindo considerável esforço para serem desenvolvidas.

Tal desafio é particularmente complexo no ensino de programação, área que não possui uma padronização de termos para representar diversos elementos encontrados em materiais educacionais. Por exemplo, termos como *"função", "método"* e *"procedimento"* são usados em vários REA, embora pareçam similares, podem retratar conceitos distintos.

Para reduzir os desafios relacionados ao uso dos filtros em REA é necessário criar novas soluções. Os filtros não podem depender apenas de dados preenchidos pelos usuários, pois muitos REA podem ser classificados incorretamente ou classificados de maneira imprecisa, com categorizações erradas ou palavras-chave irrelevantes. Além do mais, é necessário ir além da ideia tradicional de classificação, tendo em vista que muitos REA são multidisciplinares, podendo ser utilizados por diferentes contextos, ou ainda serem adaptados e atualizados para outros usos. Essencialmente, duas ações podem reduzir o impacto que os filtros atuais possuem nas plataformas abertas, como apontado no Quadro 6.

Quadro 6 – Ações para reduzir os desafios da filtragem de recursos

| Ação | Descrição |
|---|---|
| #1 | Ir além do uso tradicional de filtros, como "tipo de recurso" ou "licença". Uma contribuição sugerida é a construção de mecanismos mais sofisticados, como a busca por sinônimos ou busca por intenção de uso (remixar, compartilhar, revisar, etc...). |
| #2 | Evitar rótulos ou opções em filtros que fazem sentido em apenas contextos específicos ou regionais. Um exemplo é a utilização de padrões comuns, como "Assunto" ou "Área" ao invés de padrões que podem variar entre regiões ou países distintos, como os níveis educacionais (*"Grade School", "Middle School"* e *"High School"*). |

Fonte: Dados da Pesquisa.

### *String de Busca*

Em essência, as *strings* de busca são o conjunto de palavras usadas para realizar pesquisas. As coleções de REA muitas vezes não possuem elementos de identificação padronizados. Seus títulos, descrições e palavras-chave são organizados pela própria comunidade. Consequentemente, os usuários podem ter dificuldades para construir *strings* de pesquisa eficazes, usando termos imprecisos.

Como sintetizou um pesquisador durante a análise dos dados: *"Ao verificar os comentários dos participantes, percebi uma grande dificuldade no processo de construção de uma*

*string de busca. Os participantes muitas vezes ficavam perdidos porque nenhuma função ou recurso era apresentado para ajudá-los. Foi tentativa e erro, força bruta"*. Da mesma forma, um participante afirmou: *"Tempo foi curto para adaptação a interface e refinar a string de busca para alcançar o objetivo desejado"*.

Motores de busca modernos incluem sugestões, destaques e pesquisas relacionadas para facilitar a criação de uma *string* de busca. Infelizmente, na maioria das vezes, plataformas de REA não conseguem fornecer esse suporte aos usuários. Há, naturalmente, uma limitação de dados e também de recursos para isso. No entanto, soluções mais simples e práticas podem ser pensadas para esse desafio. Um exemplo é tornar o processo de classificação dos recursos mais transparente ao usuário. Ou seja, mostrar por qual motivo aquele recurso está em primeiro lugar na sua busca. Assim, no Quadro 7 duas propostas são apresentadas para reduzir os desafios associados ao processo de elaboração de uma *string* de busca:

Quadro 7 – Ações para reduzir os problemas das *strings* de buscas

| **Ação** | **Descrição** |
|---|---|
| #1 | Utilizar novas formas de construção de *strings*, como gráficos ou navegar por palavras-chave. |
| #2 | Desenvolver estratégias para criar e refinar *strings* de busca, como fornecer exemplos de sinônimos ou pesquisas semelhantes e destacar os principais resultados. |

Fonte: Dados da Pesquisa.

### *Operadores Lógicos e Relacionais*

Uma estratégia comum adotada pelos participantes foi adotar operadores lógicos (AND/ OR) para realizar buscas mais eficazes. Ou seja, ao invés de realizar três pesquisar *"loop", "for"* e *"while"*, o usuário poderia pesquisar *"loop OR for OR while"*. Porém, foi observado que os resultados permaneciam inalterados após a utilização desses operadores durante as buscas. Isso levantava incertezas sobre a eficiência das plataformas de REA na entrega de resultados relevantes com tais operadores.

O feedback dos participantes ilustra bem o problema. Por exemplo, um participante expressou o seguinte: *"[...] ao utilizar mais de um termo com operador lógico não encontrei os resultados esperados. Um termo sozinho retornou várias opções, mas ao colocar o segundo termo com AND não apareceu nenhum que atendesse ao critério solicitado"*. Outro participante acrescentou: *"Os operadores lógicos podem ajudar, porém a não utilização deles parece trazer os mesmos resultados"*.

Os operadores lógicos lógicos e relacionais são essenciais para listar os resultados. Eles permitem, entre outras funções, a criação de *strings* de pesquisa complexas utilizando diferentes sinônimos e comparações. Porém, apesar da documentação das plataformas apresentarem essa possibilidade, os participantes não perceberam eficiência em sua aplicação. Por isso, sugere-se que duas ações sejam adotadas em plataformas abertas, como mostra o Quadro 8.

Quadro 8 – Ações para minimizar o impacto sobre operadores lógicos e relacionais

| Ação | Descrição |
|---|---|
| #1 | Introduzir o uso de operadores lógicos e relacionais para busca de recursos. Isso pode ser inspirado no processo de condução de mapeamentos e revisões da literatura, na qual são adotados operadores como “AND” ou “OR”, parênteses “( )” e até mesmo operadores de exclusão, como “NOT IN”. |
| #2 | Fornecer novas interfaces para elaborar *strings* de busca complexas, como a sugestão de sinônimos, buscas semelhantes e possibilidade de refinar a busca realizada em resultados anteriores de forma simples. |

Fonte: Dados da Pesquisa.

### 4.2.3 Efeitos

A terceira categoria engloba os efeitos causados nos usuários. Apesar de parecer simples, tais consequências podem impedir um participante de adotar novamente os REA devido à sensação de frustração e à péssima experiência que o usuário pode ter ao usar uma plataforma aberta. Com base nos resultados alcançados, dois desafios dessa natureza emergiram.

#### *Experiência do Usuário*

A experiência do usuário é um conceito chave no design de softwares, mas o próprio termo ainda é compreendido de muitas formas distintas (BATTARBEE; KOSKINEN, 2005). Aqui, a experiência do usuário é definida como o conjunto de sensações que os usuários sentem ao utilizarem uma plataforma aberta. Softwares focados em experiência do usuário tendem a reduzir o esforço de utilização (como cliques desnecessários), adotar rótulos ou categorias facilmente compreensíveis e apresentar dados que são relevantes, omitindo informações desnecessárias.

Professores, de modo geral, estão acostumados a pesquisar por recursos educacionais em plataformas e buscadores digitais. Ao utilizar uma plataforma com REA, eles esperam uma experiência semelhante, na qual possam iniciar sua busca e refiná-la com base nos resultados apresentados.

No entanto, muitas plataformas abertas ainda são imaturas nesse aspecto. Muitas vezes usuários encontram vários campos para preencher na página de pesquisa (por exemplo, título, assunto e licença). Logo, o processo de refinamento é muito complexo, e por vezes exige que os usuários retornem à página inicial e selecionem novamente os filtros. Além disso, a falta de informações claras sobre o significado dos campos, filtros e menus nas plataformas de REA alimenta uma sensação de confusão mental.

Um participante, por exemplo, afirmou: *“Inicialmente tive dificuldades para fazer uma nova pesquisa. Tentei voltar o acesso para a página inicial para fazer a buscas com outros parâmetros não estabelecidos inicialmente e não consegui”*. Esse problema é corroborado pela nota de um pesquisador, que escreveu: *“Os participantes fazem caras e expressão de confusão ao verem os resultados listados. Muitos resultados não são aderentes à sua pesquisa e as opções*

*de filtragem são confusas. Por exemplo, a plataforma pode apresentar uma opção para filtrar materiais em '.pdf' e/ou 'livro aberto'. São tipos distintos, mas na prática, podem representar a mesma coisa. Isso gera confusão mental, típica de um usuário que não está entendendo o que o site está mostrando"*.

Diante dos dados analisados, duas ações podem contribuir para que o foco na experiência do usuário desempenhe um papel determinante no processo de identificação e recuperação de REA, como mostra o Quadro 9.

Quadro 9 – Ações para minimizar o impacto sobre a experiência do usuário

| Ação | Descrição |
|---|---|
| #1 | Reduzir a confusão mental durante o processo de pesquisa, simplificando o número de filtros ou entradas na página de pesquisa. Isso evitará que os usuários gastem muito tempo tentando entender os filtros/entradas ao invés de realizar sua pesquisa. |
| #2 | Fornecer informações relevantes para cada REA para apoiar a tomada de decisões. Isso pode incluir a exibição do número total de remixes ou visualizações de cada recurso. Atualmente, muitas plataformas de REA apresentam informações irrelevantes e que não contribuem diretamente para a busca do usuário. |

Fonte: Dados da Pesquisa.

### *Frustração*

O último desafio observado foi a identificação de más percepções dos usuários em relação às plataformas REA. Todos os participantes ficaram entusiasmados com o potencial do uso de REA. No entanto, eles relataram um sentimento de frustração após utilizarem plataformas abertas. Um participante afirmou que: *"Me senti motivado a princípio, vendo que existia a possibilidade de conseguir materiais para planejar aulas gratuitamente, porém ao verificar o tipo de material que o site indexa, penso que existem sim bons REAS, mas estão em meio a vários outros que não atendem o que eu esperava"*. Além do mais, muitos participantes afirmaram que não houve tempo suficiente para "entender" como funcionam as plataformas. Mesmo que eles tenham dedicado um tempo para testar a interface e fazer perguntas aos pesquisadores.

Esse problema é diferente da experiência do usuário, comentada anteriormente. A experiência do usuário surge da falta de atenção sobre como o usuário utiliza a plataforma aberta. Já a frustração aqui relatada ocorre após a finalização do uso da plataforma. É uma sensação ou bloqueio que provavelmente impedirá o usuário de usar plataformas abertas ao invés de buscadores tradicionais.

Os participantes do grupo focal, sendo professores experientes, demonstraram frustração ao perceber que as plataformas de REA estão longe de atingir seu objetivo de fornecer REA aos usuários. Provavelmente essa mesma sensação pode ocorrer com outros profissionais ou usuários de plataformas abertas. Portanto, três ações são propostas para evitar a propagação desse desafio, como mostra o Quadro 10.

Quadro 10 – Ações para minimizar a frustração dos usuários

| Ação | Descrição |
|---|---|
| #1 | Desenvolver estudos que explorem o nível de frustração dos usuários com as plataformas abertas. Apesar de diversas intervenções técnicas, como uso de algoritmos e especialistas, é necessário compreender como os usuários se sentem após utilizar uma plataforma aberta. E, sobretudo, como evitar sensações ruins ou que provavelmente irão inibir do usuário regressar a procura de REA. |
| #2 | Conceber plataformas abertas tendo em conta os desafios mencionados. A mera criação de novas plataformas não será suficiente como solução viável para partilhar REA. É necessário progredir no sentido de apoiar a utilização, reutilização e compartilhamento de recursos abertos. |
| #3 | Projetar soluções integradas às plataformas já existentes. A maioria dos desafios são comuns entre diversas plataformas, por isso criar um *plugin* que filtre REA de acordo com o uso pretendido, por exemplo, é uma alternativa viável que pode suportar diversas plataformas utilizando a mesma tecnologia. |

Fonte: Dados da Pesquisa.

## 4.3 Ameaças à Validade

A principal ameaça dessa investigação foi o processo de codificação, o qual pode ter descartado dados relevantes. Para reduzir esse risco, dois pesquisadores que possuem formação em REA e experiência em ensino de programação realizaram todo o processo de codificação. Ademais, as categorias e suas relações foram decididas com base nas opiniões de ambos os pesquisadores, que discutiram cada categoria, bem como suas relações.

Alguns pesquisadores podem considerar o número total de participantes (5) uma ameaça à validade. No entanto, foi seguido o paradigma qualitativo, que se centra na complexidade e no detalhe dos dados recolhidos (PANDIT, 1996). Por esse motivo, os participantes selecionados tinham um perfil aderente e relevante com base no fenômeno observado. Além disso, fontes distintas de coleta de dados foram adotadas na pesquisa. Por fim, dois pesquisadores (o autor da pesquisa e um segundo aluno de doutorado) realizaram a coleta e análise dos dados. Por conta dessas iniciativas, os riscos decorrentes do tamanho amostral foram consideravelmente reduzidos.

## 4.4 Considerações Finais

Este capítulo apresentou dez desafios relacionados ao processo de identificação e de recuperação de REA voltados ao ensino e aprendizado de programação introdutória bem como um conjunto de ações para direcionar a atenção de futuras pesquisas.

A seguir, é apresentado como o conjunto de desafios (bem como as ações propostas) foram articulados nesta pesquisa para gerar dois mecanismos capazes de resolver ou reduzir os desafios identificados.

CAPÍTULO

5

# PROPOSIÇÃO E REFINAMENTO DOS MECANISMOS DE IDENTIFICAÇÃO E DE RECUPERAÇÃO DE REA

A identificação e a recuperação dos REA para iniciantes em programação é um problema complexo. Para reduzir os desafios mapeados no Capítulo 4, são propostos dois mecanismos distintos: a IPAVL e o TAGGER.

Em linhas gerais, a IPAVL consiste em um vocabulário de termos comuns encontrados em REA para iniciantes em programação. Com base na frequência desses termos, é possível mapear em grandes coleções quais materiais são mais propensos a serem de programação introdutória. Já o TAGGER consiste em uma lista de *tags* HTML normalmente disponíveis em materiais introdutórios de programação. Esse mecanismo foca na recuperação de recursos segundo a organização de determinadas *tags* HTML. Tomando como base esse princípio, é proposta a recuperação dos recursos mais adequados ao usuário usando as informações que essas *tags* HTML possuem.

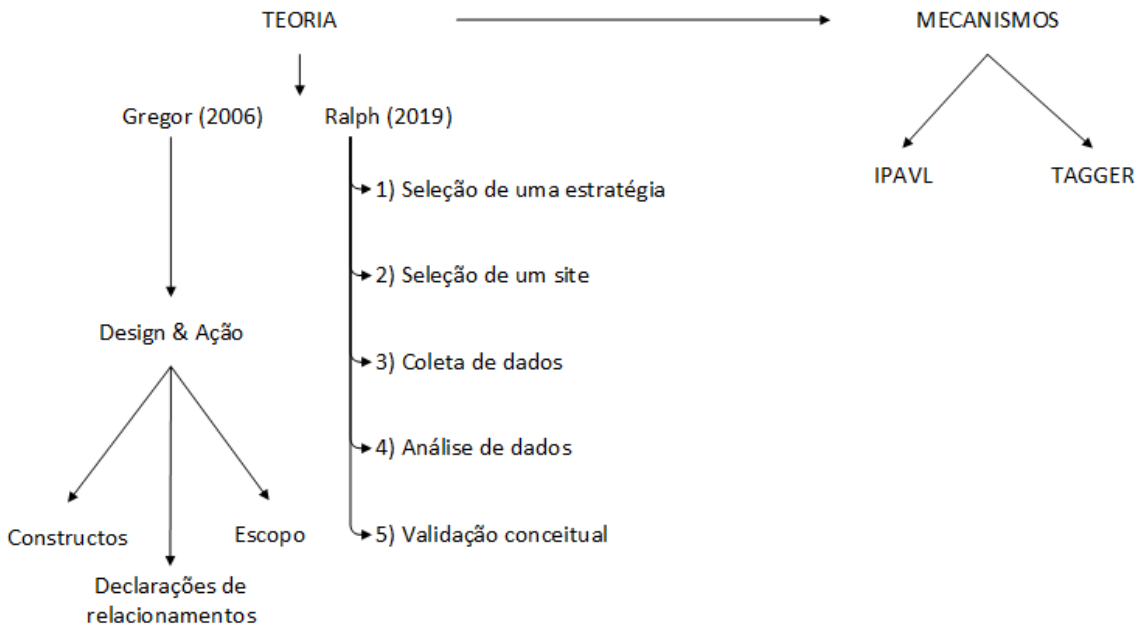
Ambos os mecanismos foram desenvolvidos após a construção de uma base teórica de investigação, seguindo uma teoria do tipo Design e Ação (GREGOR, 2006). Os mecanismos são úteis para diferentes perfis de usuários, como professores que estão buscando por materiais, profissionais que precisam organizar grandes coleções digitais ou outros usuários.

O objetivo deste capítulo é descrever os mecanismos propostos. A Seção 5.1 explica a abordagem metodológica adotada. A Seção 5.2 apresenta o processo de construção da base teórica utilizada para o desenvolvimento dos mecanismos. A Seção 5.3 introduz a IPAVL e o seu processo de desenvolvimento. A Seção 5.4 mostra a elaboração do TAGGER. As principais limitações são expostas na Seção 5.5. Por fim, a Seção 5.6 exibe as considerações finais sobre o conteúdo do capítulo.

## 5.1 Abordagem metodológica

Os mecanismos propostos neste capítulo representam os artefatos gerados pela aplicação da DSR, tendo sido concebidos por meio do desenvolvimento de uma teoria do tipo de Design e Ação. De forma resumida, esse tipo de teoria mostra como algo deve ser feito, similar a uma "receita", fornecendo prescrições explícitas para a construção de um artefato (GREGOR, 2006). A Figura 11 sintetiza o percurso metodológico adotado.

Figura 11 – Síntese da abordagem metodológica




Fonte: Dados da Pesquisa.

A ideia principal é estabelecer uma teoria que guiará o desenvolvimento dos mecanismos. Assim, inicialmente, foi necessário elaborar a concepção da teoria. Essa é uma tarefa abstrata, que pode variar conforme a natureza de cada teoria proposta. Resumidamente, nessa etapa três componentes estruturais propostos por Gregor (2006) foram definidos:

- Constructos: definição das bases que apoiam o desenvolvimento da teoria;
- Declarações de relacionamento: descrição das formas como os constructos se relacionam; e,
- Escopo: especificação do grau de generalidade das declarações de relacionamento.

Os constructos para o desenvolvimento da teoria foram REA coletados em diferentes plataformas abertas. A base racional foi adotar recursos voltados ao ensino e aprendizado de programação para identificar padrões nos materiais. Ou seja, a teoria por trás dos mecanismos poderá traduzir como os REA estão estruturados e como isso pode ser acionável por especialistas na hora de identificar e recuperar um recurso de programação introdutória.

As declarações de relacionamento dizem respeito à forma como esses REA se relacionam entre si. Aqui, o objetivo é buscar elementos repetidos que podem ser replicados em outros recursos.

O escopo diz repeito aos sistemas de informação (plataformas) que podem ser utilizados para identificar e recuperar REA de programação introdutória. Ou seja, o objetivo final é propor mecanismos que possam ser aplicados em tais sistemas. Complementarmente, outra forma de pensar esse escopo são os especialistas que interagem com os REA dessa natureza, sobretudo desenvolvedores que criam ou mantém plataformas abertas. Adicionalmente, os usuários finais, como professores e alunos, também podem utilizar os resultados dessa pesquisa.

Tendo definido esses três componentes da teoria, foi iniciado o processo de elaboração dos mecanismos. Esse processo foi inspirado no estudo de Ralph (2019) e consiste na transformação da teoria em algo tangível. Para isso, três estágios distintos de pesquisa foram desenvolvidos. No primeiro estágio foi elaborada a base teórica de construção dos mecanismos, gerando dois artigos distintos:

1. Deus, W. S. de; Barbosa, E. F. *"An Exploratory Study on the Availability of Open Educational Resources to Support the Teaching and Learning of Programming"*. In: **50th Frontiers in Education Conference**, 2020, Suécia. (FIE'2020). https://doi.org/10.1109/FIE44824.2020.9274202

2. Deus, W. S. de; Barbosa, E. F. *"Recursos Educacionais Abertos para o Ensino e Aprendizado de Programação no Brasil: Primeiros Referenciais"*. **XXXII Simpósio Brasileiro de Informática na Educação**, 2021, Brasil. (SBIE'2021). https://doi.org/ 10.57 53/sbie.2021.218 391

No segundo estágio foi feito o design e o refinamento da IPAVL (mecanismo 1) e no terceiro estágio foi desenvolvido e aperfeiçoado o TAGGER (mecanismo 2). Ambos os estágios estão descritos em dois artigos, que estão em desenvolvimento e serão submetidos para avaliação em periódicos da área:

1. Deus, W. S. de; Barbosa, E. F. *"An Academic Vocabulary List to Identify Open Educational Resources for Introductory Programming"*. In: **ACM Transactions on Computing Education**. *Em desenvolvimento*.

2. Deus, W. S. de; Morrison, B.; Barbosa, E. F. *"Identifying Open Educational Resources for Introductory Programming: A Case Study from TAGGER"*. In: **IEEE Transactions on Education**. *Em desenvolvimento*.

## 5.2 Base teórica dos mecanismos

A primeira etapa de pesquisa consistiu na elaboração de uma base teórica. Para isso, os três componentes estruturais apresentados (constructos, declaração de relacionamentos e escopo) serviram como um guia. Além deles, diferentes atividades propostas por Ralph (2019) foram desenvolvidas, sumarizadas a seguir.

### 5.2.1 Seleção de uma estratégia

A seleção de uma estratégia representa a concepção geral da pesquisa. Em linhas gerais, neste trabalho de doutorado optou-se pelo uso de uma base empírica de investigação, tomando como solução ideal a construção de um banco de dados contendo REA de programação introdutória, oriundos de uma grande diversidade de plataformas abertas. A ideia principal foi observar esses recursos, identificando padrões recorrentes em diferentes materiais a fim de conceber os mecanismos.

### 5.2.2 Seleção de um site

No contexto da Engenharia de Software, um *site* é uma organização, cultura, tribo ou grupo de algum tipo (RALPH, 2019). Por isso, não se deve confundir com *websites*, os quais os usuários acessam por meio de um navegador.

Com expresso na etapa anterior, o *site* relevante para esta pesquisa são sistemas de informação que possuem em seu acervo materiais voltados ao ensino de iniciantes em programação. Para mapear tais sistemas, foram utilizadas três estratégias distintas: (1) plataformas identificadas no MS (DEUS; BARBOSA, 2022), (2) plataformas abertas identificadas no estudo exploratório (DEUS; BARBOSA, 2020) e (3) plataformas identificadas no OER World Map[1].

Para uma plataforma ser considerada válida, ela deveria cumprir três requisitos básicos:

1. Apoiar o processo de extração automática dos dados;
2. Armazenar recursos voltados ao ensino e aprendizado introdutório de programação; e,
3. Estar disponível em inglês ou português.

O primeiro critério foi definido devido à estratégia de extração de dados. Em suma, foi adotado um processo de extração automático, no qual um *web crawler* visitava o site da plataforma aberta, simulava uma pesquisa de usuário e extraía os resultados em uma lista. Para

[1] O OER World Map foi um projeto dedicado a mapear plataformas de REA ao redor do mundo. O projeto foi encerrado em 2022. Mais informações estão disponíveis no seguinte link: https://oerworldmap.wordpress.com/2022/03/22/ adeus mundo/

isso funcionar, era necessário que a plataforma adotasse requisições GET[2] e que permitisse a extração de dados de forma síncrona[3]. O segundo critério foi definido para remover plataformas que não possuíam em seu acervo REA dedicados ao ensino ou aprendizado de programação. Para identificá-las, foi realizada uma busca manual no acervo de cada plataforma. Por fim, o terceiro critério foi selecionado para remover plataformas que estavam disponíveis em idiomas desconhecidos pelo autor da pesquisa. Ao todo, 11 plataformas foram selecionadas, como mostra a Tabela 11.

Tabela 11 – Lista das plataformas selecionadas

| Nome | Link | Origem |
|---|---|---|
| Curriki | https://www.curriki.org/ | OER World Map |
| EduCAPES | https://educapes.capes.gov.br/ | OER World Map |
| Merlot | https://www.merlot.org/merlot/ | SM/Estudo exploratório |
| OER Commons | https://www.oercommons.org/ | SM/Estudo exploratório |
| OpenStax CNX/Connexions | https://openstax.org/ | SM |
| Open UCT | https://open.uct.ac.za/ | OER World Map |
| Skills Commons | https://www.skillscommons.org/ | OER World Map/Estudo exploratório |
| Temoa | http://temoa.tec.mx/node | OER World Map/Estudo exploratório |
| TIB | https://www.tib.eu/en/ | OER World Map |
| Xpert | https://www.nottingham.ac.uk/xpert/ | OER World Map |
| Zenodo | https://zenodo.org/ | OER World Map |

Fonte: Dados da Pesquisa.

### 5.2.3 Coleta de dados

O processo de coleta de dados foi desenvolvido com o auxílio de *web crawlers*. Para isso, cada plataforma selecionada foi visitada e analisada pelo autor da pesquisa, buscando identificar o funcionamento das requisições e o padrão de resposta.

Depois dessa investigação, um *web crawler* foi desenvolvido. Em síntese, o *web crawler* visitava a página de busca da plataforma e extraía as informações de cada REA identificado. Para isso, o *web crawler* pesquisava pelo termo *programming* em plataformas do idioma inglês e buscava pelo termo *programação* para plataformas no idioma português. Logo a seguir, verificava o total de resultados e visitava cada resultado, gerando assim uma lista contendo todos os metadados disponíveis.

### 5.2.4 Análise de dados

O processo de coleta de dados gerou duas lista de REA. Uma menor, com 3.715 resultados contendo os recursos no idioma português; e outra maior, com mais de 50.000 resultados,

[2] Sites de busca quase sempre adotam o padrão GET ou POST em suas requisições. Em requisições GET, é possível identificar o funcionamento apenas observando o padrão da URL do site. Em requisições POST, esse processo é mais complexo aos usuários, sendo basicamente inviável (sem apoio de desenvolvedores ou da documentação) identificar quais são os parâmetros e qual é o padrão adotado.

[3] A requisição síncrona é o resultado do processamento de dados na qual é exibido, de forma ordenada e padronizada, a resposta da requisição.

contendo REA no idioma inglês. Nem todos esses REA eram relevantes. Por exemplo, havia recursos sem licenças abertas ou recursos que abordavam outros assuntos, como *programação linear* ou *programação genética*. Além do mais, muitos materiais tinham o seu título ou descrição em inglês, ou português, mas o conteúdo estava em outro idioma, como alemão ou italiano.

Como já apontado neste trabalho (ver Seção 2.3), não foi identificada nenhuma abordagem capaz de filtrar esse conjunto de dados, identificando apenas os REA de programação introdutória. Por isso mesmo, foi necessário desenvolver um procedimento próprio, considerando o objetivo do estudo e a natureza dos dados.

A primeira lista selecionada foi a de REA em português. Inicialmente, foi verificado que os metadados dessa lista mostravam quais REA possuíam licença aberta e quais estavam escritos em português. Por isso mesmo, essa filtragem foi aplicada, resultando na seleção de 1.669 recursos. Logo em seguida, foi criado um novo filtro para remover resultados não relevantes. Para isso, inicialmente foi usado a lista de conceitos básicos presentes na Tabela 2. Considerando o idioma Português, a lista de conceitos foi complementada com os termos do estudo de Araujo, Andrade e Guerrero (2016). De forma resumida, Araujo, Andrade e Guerrero (2016) apresentaram uma lista das habilidades, sobretudo conceitos e práticas inerentes de programação para o contexto brasileiro. A montagem final está exposta na Tabela 12

Tabela 12 – Conceitos básicos de programação

| **Constructo** | **Descrição** |
|---|---|
| *Fundamentos* | Algoritmo, Sequência, Sequenciamento, Sequencial, Arquivos, Depuração, Lógica |
| *Operadores Lógicos* | Operadores |
| *Estrutura Condicional* | Condicionais, Condição, Condicional, Controle, Fluxo |
| *Loops* | Loop |
| *Arrays* | Dados |
| *Métodos/Funções* | Funções, Parâmetros |
| *Básico de POO* | Abstração, Decomposição |
| *Outros* | Binária, Recursão, Evento, Paralelismo, Boolean |

Fonte: Elaborado pelo autor com os dados coletados em Tew e Guzdial (2010) e Araujo, Andrade e Guerrero (2016).

Desse modo, para um REA ser considerado válido, ele deveria ter alguma dessas palavras em seus metadados. Ao todo, 174 REA permaneceram na lista. Logo após isso, cada recurso foi analisado individualmente pelo autor da pesquisa, selecionado apenas os que se encaixam no escopo de programação introdutória. Ao final desse processo, 37 REA foram selecionados.

Após finalizar a filtragem dos REA em português, foi iniciado o processo de filtragem na lista mais extensa, de REA em inglês. Inicialmente, a primeira tentativa foi replicar o procedimento anterior: selecionar recursos que possuíam uma licença aberta e estivessem escritos em inglês. Isso, no entanto, não foi possível devido à natureza dos dados. Alguns REA, por exemplo, apresentavam a licença no metadado, outros não. O mesmo ocorria com o idioma, sendo que apenas uma fração dos REA possuíam a descrição de qual idioma era usado no REA. Com base nessas limitações, a primeira filtragem foi suprimida.

Na segunda tentativa, optou-se por realizar o segundo tipo filtro: utilizando uma lista de termos de programação introdutória. Essa estratégia, no entanto, demonstrou-se ineficaz. Muitos REA eram oriundos de diversos locais, como Estados Unidos, Austrália, Índia, além de países da Europa e América do Sul. Logo, não foi possível adotar um estudo que apresentasse um conjunto de termos consistentes para os diferentes idiomas.

Na terceira tentativa, foi aplicado o único padrão que demonstrou ser o mais eficaz: nomes de linguagens de programação. Essa tentativa demonstrou ser mais aderente, trazendo resultados relevantes e de uma ampla gama de locais. Por isso mesmo, foi montada uma lista contendo os nomes de todas as linguagens de programação conhecidas atualmente. Para isso, foi feita uma pesquisa na WikiData[4], uma fonte aberta de dados, para listar o nome de todas as linguagens de programação disponíveis.

Com isso, foram selecionados todos os REA que continham em seu conteúdo ao menos um nome de linguagem de programação conhecida, totalizando 3.990 resultados válidos. No final, cada resultado retornado foi manualmente analisado. Ao término do processo, 402 REA foram considerados válidos e integraram o banco de dados.

### 5.2.5 Avaliação conceitual

Esta foi a etapa final da elaboração da teoria. Nela ocorreu a transferência da teoria para a construção dos mecanismos. Para isso, a proposta de Ralph (2019) foi adotada da seguinte forma: para a construção de cada mecanismo foram definidos critérios de seleção. Esses critérios funcionam de modo similar ao processo de escolha de estudos em mapeamentos ou revisões sistemáticas da literatura, os quais selecionam ou removem estudos consoante ao que foi previamente definido.

Em suma, para cada mecanismo, foi definida uma lista de critérios detalhados, removendo ocorrências que não são relevantes para o andamento da pesquisa. Essa decisão foi baseada no argumento de Ralph (2019), o qual estabelece que em determinados cenários, como o explorado pelos mecanismos, não existe um conjunto universal de critérios. Por isso, optou-se por definir critérios que fossem aderentes à proposta de cada mecanismo.

Nas próximas seções, o processo de criação dos mecanismos é detalhado, bem como o conjunto de critérios adotados para a sua elaboração.

## 5.3 IPAVL

Após finalizar a parte teórica, foi iniciado o segundo estágio da pesquisa, que consiste no design e refinamento de mecanismos. O primeiro mecanismo construído foi denominado IPAVL e a ideia surgiu durante o processo de seleção de REA relatado anteriormente. Como

[4] https://query.wikidata.org/

nenhuma abordagem para facilitar a identificação de REA foi encontrada na literatura, o processo de seleção foi realizado usando uma lista de termos e outra lista de nomes de linguagens de programação. É importante observar que tal problema relaciona-se aos diversos desafios mapeados no grupo focal (Capítulo 4), como construção da *string* de busca e o processo de filtragem de dados.

Tendo isso em vista, surgiu a ideia de sistematizar esse processo, criando um vocabulário acadêmico de termos referentes ao ensino e aprendizado de programação introdutória. Esse vocabulário, por sua vez, poderia ser utilizado para automatizar ou facilitar atividades alinhadas à identificação de REA para estudantes que estão aprendendo a programar. Além disso, o próprio mecanismo poderia ser usado pelos usuários ou pelos sistemas de informação que armazenam os REA.

### 5.3.1 Design e Refinamento

A construção do vocabulário seguiu uma estratégia empírica, concebida em diferentes pesquisas, sobretudo nos trabalhos de Roesler (2021), Lei e Liu (2016) e Wang, Liang e Ge (2008). Com isso em mente, o primeiro passo foi o estabelecimento de um corpus inicial.

Na prática, o corpus inicial é um conjunto de textos relevantes que serão processados para coletar apenas os termos mais importantes para o vocabulário. Nesse sentido, é uma compreensão comum que tal corpus seja estabelecido por meio de textos representativos para o campo de estudo (COXHEAD, 2000). Ao mesmo tempo, é essencial evitar vieses, como por exemplo, inserção de um conjunto restrito de autores ou textos de um único tipo.

Com base nesses aspectos, optou-se por priorizar duas fontes distintas para a criação do corpus inicial: REA e artigos científicos. Os REA foram selecionados por serem objeto de estudo desta pesquisa e serem uma fonte aberta e heterogênea de conteúdos. Já os artigos científicos foram selecionados pela ampla produção e disseminação de trabalhos na literatura recente. Cabe destacar ainda que os textos científicos são complementares à natureza aberta dos REA, gerando um corpus que mescla textos técnicos (artigos publicados em eventos e periódicos por pesquisadores da área) com produções educacionais (REA desenvolvidos por professores de programação).

#### *Criando o Corpus Inicial*

Os 402 REA utilizados foram apresentados na seção anterior e seu processo de seleção já foi descrito. Para a coleta de artigos científicos foi utilizado um mapeamento terciário da literatura. De forma resumida, o mapeamento terciário é uma estratégia utilizada quando existem muitas publicações secundárias (mapeamento e revisões sistemáticas) em um determinado campo de pesquisa.

O mapeamento terciário segue o mesmo padrão adotado nas revisões e mapeamentos da

literatura, mas remove estudos primários (como experimentos, estudos de casos, etc...). Assim, inicialmente, foram selecionados dois estudos para comporem o grupo de controle (estudos que deveriam ser retornados durantes as buscas):

- Andrew Luxton-Reilly, Simon, Ibrahim Albluwi, Brett A. Becker, Michail Giannakos, Amruth N. Kumar, Linda Ott, James Paterson, Michael James Scott, Judy Sheard, and Claudia Szabo. *"Introductory programming: a systematic literature review"*. In **23rd Annual ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE)**, 2018.

- Medeiros, R. P.; Ramalho, G. L.; e Falcão, T. P. *"A Systematic Literature Review on Teaching and Learning Introductory Programming in Higher Education"*. In **IEEE Transactions on Education**, 2019.

Os trabalhos foram escolhidos por serem publicações recentes e relevantes da área. Com base neles, os seguintes termos foram extraídos para compor a *string* de busca: *("introductory programming"OR "introduction to programming"OR "novice programming"OR "novice programmers"OR "CS1"OR "CS 1"OR "learn programming"OR "learning to program"OR "teach programming"OR "learning programming") AND ("systematic literature review' OR "systematic review" OR "systematic mapping" OR "mapping study"OR "systematic literature mapping").*

A *string* foi adaptada em cinco bibliotecas digitais: Scopus, IEEE Xplore, ACM DL, Science Direct, e Web Of Science.

As buscas foram performadas com base no título, resumo e palavras-chave, focando em trabalhos recentes. Baseado no critério de selecionar artigos voltados ao ensino e aprendizado de programação introdutória, 25 estudos foram selecionados, como apresentado na Tabela 25 (Apêndice A). É necessário destacar que o processo de busca foi similar ao realizado durante o MS. Ou seja, a mesma estratégia de seleção e os mesmos critérios foram usados. No entanto, não foi realizada uma busca manual e nem o *snowballing* tendo em vista que o objetivo do estudo foi a criação de um corpus e não a revisão da literatura em si. Ademais, como as tecnologias que envolvem o ensino de programação mudam constantemente, foram selecionados apenas trabalhos publicados nos últimos dez anos.

### *Processando o Corpus Inicial*

O corpus inicial possuía arquivos de duas estruturas distintas: os REA (da base teórica exposta na Seção 5.2) e os artigos científicos. Por isso, alguns arquivos tiveram de ser convertidos, pois estavam em formato proprietário (*.pdf*). Por conta disso, o seguinte procedimento foi adotado: cada arquivo *.pdf* foi baixado e armazenado em um diretório privado do Google Docs[5].

[5] https://www.google.com/intl/pt-BR/docs/about/

Logo em seguida, cada arquivo foi convertido para o formato aberto (*.txt*) usando o conversor nativo da ferramenta.

A seguir, foi efetuada uma análise da integridade textual das conversões. Nela, cada arquivo *.txt* gerado pelo Google Docs foi revisto manualmente. Durante esse procedimento, foram realizadas correções necessárias, como quebras de linhas e remoção de campos desnecessários para a criação do corpus inicial, como nome de autores, DOI, número de páginas, etc.

Depois, foi desenvolvido um software para processar todo o corpus inicial. O software realizou as seguintes ações:

1. *Limpeza de dados:* Remoção de caracteres e estruturas inválidas (como números, pontuações, e-mails, entre outros) e conversão do texto para um formato padrão (minúsculo);
2. *Tokenização:* Divisão do texto em palavras;
3. *Contabilização:* Contagem dos termos presentes.

Após esse procedimento, notou-se que havia muitas palavras flexionadas devido às regras gramaticais. Por exemplo, “desenvolvedores” e “desenvolvedor”, termos similares que por estarem no plural e no singular, respectivamente, eram contabilizadas como sendo duas ocorrências distintas. Em vista desse problema, optou-se pela técnica de *lematização*, removendo as flexões das palavras. Tal estratégia foi aplicada no conjunto de dados, contabilizando novamente a ocorrência de cada uma.

### *Critérios de seleção*

A última etapa para a construção do vocabulário foi o refinamento das palavras através critérios de seleção. Em última instância, o objetivo desses critérios é selecionar na base de dados apenas termos que são comuns e que podem auxiliar na identificação dos REA. Para isso, o seguinte conjunto de critérios foi definido:

- *Palavras populares:* São palavras comuns utilizadas nos textos, mas não se referem ao ensino de programação, como os conectivos *de, da, o, as, etc...* Essa estratégia foi baseada no trabalho de Wang, Liang e Ge (2008) e representa a primeira estratégia de limpeza e remoção de termos. Para identificar os termos que deveriam ser removidos, foi adotada a lista fornecida pela *General Service List*[6].
- *Propósito não acadêmico:* São termos que apesar de não se encaixarem no primeiro critério, não possuem relação direta com o ensino ou aprendizado. Por exemplo, os termos *parece, sendo, etc...* eram utilizados em muitos REA e artigos científicos. Mas, na prática,

[6] Lista que apresenta os termos mais populares em inglês. Disponível em: http://www.newgeneralservicelist.org/

representavam termos que são usados, em sua maioria, para dar maior sentido às frases. Para identificar esse termos, foi adotada a mesma estratégia de Lei e Liu (2016) e Roesler (2021): frequência da palavra no corpus inicial *versus* frequência da palavra em um corpus não acadêmico. Para isso, foi usado o corpus do *British National Corpus* (BNC)[7]. Para um termo ser considerado válido, ele deveria ter uma ocorrência igual ou acima de 150% no corpus inicial do que no corpus da BNC.

- *Frequência mínima:* São termos que embora não tenham sido removidos nos critérios anteriores, não são tão frequentes no corpus inicial. Para removê-los, foi seguido a métrica apresentada por Roesler (2021): 0,28 a cada 10.000 ocorrências. Ou seja, um termo válido deveria possuir esse índice para permanecer no vocabulário.

- *Significado técnico/educacional:* Para filtrar a lista final e remover termos inadequados, foi seguida a estratégia de Lei e Liu (2016) e Roesler (2021), que consiste no uso da revisão de cada termo por meio de um dicionário técnico/educacional. Para isso, foi adotado o *Oxford Dictionary of Computer Science* para comparar os termos restantes com a sua função e propósito conforme o dicionário.

- *Revisão manual:* Após consolidar a lista, foi efetuada uma revisão manual para remover qualquer termo remanescente não relacionado ao ensino e aprendizado introdutório de programação.

A aplicação dos critérios removeu palavras que não eram relevantes ao vocabulário; e cada etapa de filtragem dos dados é apresentada na Tabela 13. Devido à natureza distinta das fontes utilizadas (artigos científicos e REA), o procedimento foi realizado de forma separada. Ou seja, inicialmente, foram processados os artigos científicos (por ser um corpus de tamanho menor) e depois foi processado o corpus formado pelos REA. Ao final, as duas listas foram unificadas em uma, removendo-se as duplicações.

Tabela 13 – Procedimento de seleção de termos

| | **REA** | | **Artigos Científicos** | |
|---|---|---|---|---|
| **Etapa** | **Lemas** | **Ocorrência** | **Lemas** | **Ocorrência** |
| Corpus inicial | 18.584 | 1.633.867 | 7.582 | 176.467 |
| Palavras populares | 16.416 | 930.206 | 6.270 | 80.170 |
| Propósito não acadêmico | 11.621 | 864.023 | 5.354 | 75.088 |
| Frequência mínima | 1.750 | 827.970 | 2.344 | 71.727 |
| Significado técnico/educacional | 213 | 11.600 | 254 | 12.716 |
| Revisão Manual | 96 | 8.826 | 84 | 6.355 |

Fonte: Dados da Pesquisa.

Após a conclusão da seleção dos termos, foram identificados 101 lemas. Foram observadas algumas abreviações entre os lemas, como *OOP*. que significa *Programação Orientada*

[7] O BNC é um corpus que contém diversos termos em inglês para diferentes áreas. Disponível em: http://www.natcorp.ox.ac.uk/

*a Objetos*. Por esse motivo, também foram acrescentados os significados de cada abreviatura, totalizando 108 termos, conforme apresentado na Tabela 14.

Tabela 14 – Termos presentes no vocabulário

| **Termos** |
|---|
| abstraction, algorithm, api, app, application, archive, array, aspect, attribute, b, balanced, bug, c, coding, compiler, computer, conditional, cpu, debugging, declaration, decomposition, definition, desktop, developer, documentation, domain, dynamic, encapsulation, encoding, entity, exception, execute, expression, failure, flowchart, generic, hardware, haskell, heap, heapsort, hierarchy, ide, identifier, implementation, informatics, information, inheritance, instance, instruction, interface, is, iteration, java, javascript, language, learning, library, lisp, listing, location, logical, loop, matrix, memory, min, mouse, oop, operator, paradigm, parameter, pascal, php, pointer, polymorphism, precision, primitive, procedure, programmer, programming, pseudocode, python, quicksort, recursion, recursive, register, routine, ruby, script, scripting, sequential, sorting, stack, static, structure, subset, synchronous, syntax, terminal, testing, variable, vector, “application programming interface”, “byte”, “central processing unit”, “interactive development environment”, “information system”, “minimum”, “object oriented programming” |

Fonte: Dados da Pesquisa. Todos os termos estão em inglês. Para a versão em português da IPAVL, os termos foram traduzidos.

### 5.3.2 Funcionamento

Em resumo, a IPAVL foi projetada para funcionar como uma lista de termos sobre programação introdutória. Assim, o seu principal propósito é identificar REA voltados ao ensino de programação de acordo com a frequência e a presença de seus termos.

A aplicação da IPAVL pode ocorrer em outros cenários, como na padronização de termos usados durante a criação de materiais didáticos, identificação das abreviaturas mais comuns e até mesmo na elaboração de *strings* de buscas sobre programação introdutória.

Considerando a perspectiva sobre identificação de REA, a IPAVL pode apoiar duas estratégias comuns de classificação de recursos. A primeira delas é o uso de especialistas que revisam e organizam coleções manualmente e a segunda é a adoção de algoritmos para realizar a identificação automaticamente.

Ambas as soluções são complexas, necessitando de modelos para especialistas ou algoritmos de classificação. A IPAVL surge para apoiar ambas as práticas. Por um lado, a IPAVL baseia-se na ocorrência de termos introdutórios de programação, portanto, pode ser aplicada com algoritmos de aprendizado de máquina para identificar potenciais REA com base na frequência dos termos. Essa aplicação pode ser desenvolvida adotando classificação supervisionada e não supervisionada, gerando uma matriz de termos. Por outro lado, a IPAVL também pode apoiar os processos de tomada de decisão dos classificadores humanos. Nesse sentido, sua principal aplicação é apresentar um indicador baseado no número total de termos referentes ao ensino introdutório à programação. Outra possibilidade é destacar termos para chamar a atenção do classificador. A intenção não é eliminar o uso de algoritmos ou humanos, mas fornecer uma solução que pode reduzir o esforço.

De fato, nas validações desenvolvidas, a IPAVL foi capaz de auxiliar a identificação de recursos por meio do aprendizado de máquina, gerando modelos de frequência de termos e facilitando o processo de identificação de recursos introdutórios e não introdutórios. O mesmo serviu para apoiar a construção de *strings* de buscas para facilitar a identificação de recursos para

diferentes usuários. Tais resultados serão detalhados na próxima seção e também encontram-se relatados em um manuscrito que será submetido para avaliação no periódico *ACM Transactions on Computing Education*.

Com a conclusão da montagem da IPAVL, foi iniciado o desenvolvimento do segundo mecanismo da tese: TAGGER.

## 5.4 TAGGER

O TAGGER é o segundo mecanismo desenvolvido neste trabalho de doutorado. A ideia também surgiu durante a montagem do banco de dados apresentada na Seção 5.2. No entanto, tal mecanismo foi inspirado pela forma como os REA são construídos e armazenados em diferentes plataformas. Esse processo é descrito a seguir.

### 5.4.1 Design e Refinamento

Os REA investigados neste trabalho possuem duas características: i) são materiais para iniciantes em programação, e ii) são materiais digitais. Por ser digital, o REA é uma espécie de documento web que geralmente está disponível em alguma plataforma aberta, como um repositório ou *site* pessoal. Isso permite a adoção de regras do *Document Object Model* (DOM), o que pode apoiar a recuperação de recursos relevantes.

#### *Document Object Model*

O DOM é uma interface de programação utilizada em documentos web (WOOD *et al.*, 1998). Ele reflete a estrutura do documento usando uma árvore. Ou seja, toda a estrutura do documento é abstraída, bem como a hierarquia das informações. Para ilustrar esse processo, observe a Figura 12, a qual apresenta um exemplo de uma tabela que possui duas linhas e duas colunas.

Figura 12 – Exemplo de tabela HTML

| | |
|---|---|
| Shady Grove | Aeolian |
| Over the River, Charlie | Dorian |

Fonte: Wood *et al.* (1998).

A tabela também pode ser representada por meio de um código HTML, como mostrado no Código-fonte 1. Em linhas gerais, a tabela possui diversas informações, como a *tag* <*TABLE*> que marca o início da tabela. Já a *tag* <*TBODY*> representa o início do corpo da tabela. Enquanto isso, a *tag* <*TR*> exibe o início de uma linha e as células são representadas pela *tag* <*TD*>. Cada

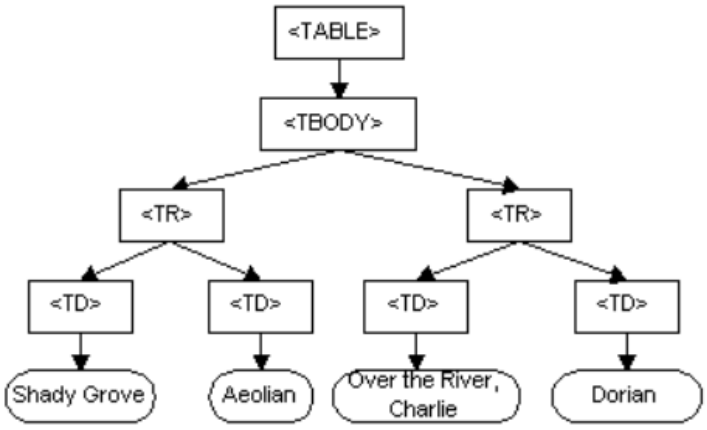
uma dessas *tags* possui um início e um fim, marcando pelo fechamento da *tag* correspondente. Nesse caso, as *tags* *</TABLE>*, *</TBODY>*, *</TR>* e *</TD>* encerram cada elemento.

**Código-fonte 1** – Exemplo de um documento HTML elaborado por Wood *et al.* (1998)

```
 1:  <TABLE>
 2:   <TBODY>
 3:     <TR>
 4:       <TD>Shady Grove</TD>
 5:       <TD>Aeolian</TD>
 6:     </TR>
 7:     <TR>
 8:       <TD>Over the River, Charlie</TD>
 9:       <TD>Dorian</TD>
10:     </TR>
11:   </TBODY>
12: </TABLE>
```

Além de marcar o início e o fim, as *tags* HTML também refletem a hierarquia de um documento. Isso permite verificar quais são as *tags* ancestrais e/ou filhas de cada elemento por meio do DOM. Usando o Código-fonte 1, por exemplo, obtém-se a estrutura DOM que está exposta na Figura 13.

Figura 13 – Estrutura DOM.




Fonte: Wood *et al.* (1998).

*Como o DOM pode ser utilizado para recuperar REA de programação introdutória?*

O DOM é uma interface entre três *stakeholders* importantes para o uso de REA de programação introdutória: autores, desenvolvedores e usuários.

Aqui, os autores são compreendidos como os responsáveis pela criação dos REA. Muitas vezes, um mesmo recurso possui a autoria de diversas pessoas devido ao processo de remix. Ao

mesmo tempo, são os autores quem geralmente disponibilizam esses materiais em plataformas abertas. Nesse processo, além do cadastro do material também são preenchidas informações relevantes, como título, descrição, palavras-chave, entre outras informações.

Já os desenvolvedores são responsáveis pela programação das plataformas abertas. Aqui, a programação refere-se ao processo de codificação, contendo as tarefas de criação e manutenção do código-fonte da plataforma.

Por fim, os usuários representam os alunos, professores ou qualquer tipo de usuário que possua interesse em utilizar um REA. Para isso, geralmente o REA será buscado pelo usuário em alguma plataforma aberta.

O DOM pode conectar esses *stakeholders* e facilitar o processo de recuperação dos REA ao representar uma interface comum entre eles. Em resumo, o DOM respeita a hierarquia das informações contidas no REA, adota a estrutura do código-fonte da plataforma aberta e abstrai o conteúdo e a origem desse conteúdo por meio das *tags*.

### *Tags*

As *tags* representam um elemento central da estruturação de um documento web. Elas são delimitadas por sinais de abertura e de fechamento e preenchidas por textos que podem descrever o documento de forma apropriada (PANT, 2003).

Por exemplo, considere que uma página web possui o seguinte título: *Aprendendo a programar: criando o seu primeiro* ***programa em linguagem C***. Uma forma comum de fazer tal descrição é adotando a *tag* <*title*> e a *tag* <*b*> ou <*strong*> para dar ênfase, como mostrado no Código-fonte 2.

**Código-fonte 2** – Exemplo de formatação em uma página web

```
1: <h1>Aprendendo a Programar: criando o seu primeiro <b>programa
     em linguagem C</b></h1>
```

A *tag* <*title*> informa que o conteúdo dentro dela é o título do documento. Já a *tag* <*b*> informa que dentro desse conteúdo existe ênfase nos termos “programa em linguagem C”. Embora essa informação pareça simples, ela estabelece uma conexão relevante: se a ênfase foi dada em relação à linguagem de programação C, provavelmente o público-alvo do recurso são usuários interessados nesse tipo de material.

Durante a construção da base de dados, foi observado que muitos recursos seguiam essas regras de estruturação, permitindo que a informação presente nas *tags* auxiliasse a recuperação daquele recurso. Assim, emergiu a ideia de criar o TAGGER, adotando esse conceito de funcionamento. Para isso, a primeira etapa foi selecionar quais *tags* deveriam estar presentes.

### *Critério de seleção*

Para realizar o processo de seleção das *tags*, foram estabelecidos três critérios de seleção:

1. As *tags* selecionadas precisam ser nativas do HTML;
2. As *tags* selecionadas devem ser comuns nos REA ou nas plataformas abertas;
3. As *tags* selecionadas devem prever blocos de informações úteis sobre a natureza do recurso;

O primeiro critério foi definido considerando a tendência moderna do uso de *frameworks* de programação, como *Java Servlets, Struts, Stencil, Rebel, Ruby on Rails*, entre outros (CURIE *et al.*, 2019). Tais *frameworks* podem gerar *tags* personalizadas, que não são suportadas por navegadores mais antigos. Para identificá-las, foram comparadas as *tags* identificadas nos REA com a lista de *tags* HTML[8]. Ao todo, foram removidas 241 *tags*.

O segundo critério evita a inserção de vieses na seleção. Por exemplo, uma única plataforma pode ter o uso massivo de uma determinada *tag* HTML. Mas, ela não é comum nas demais plataformas. O mesmo pode acontecer em um REA que, por ser extenso, pode fazer uso diversas vezes de uma mesma *tag*, mas isso não é reproduzido no restante dos recursos. Logo, para ser considerada válida, foi definida que a *tag* deveria estar presente em ao menos três plataformas ou na metade do conjunto de REA. Com isso, 100 *tags* foram removidas.

O terceiro critério seleciona as *tags* HTML capazes de trazer as principais informações sobre os REA. Desse modo, *tags* que representam outros conteúdos (como o rodapé da página, o qual geralmente ilustra alguma informação do site) são removidos. Para isso, o trabalho de Kabir, Kabir e Amin (2015), que ilustra as *tags* capazes de sintetizar as principais informações de uma página web, foi utilizado. Esse critério removeu 36 *tags* HTML.

Os critérios foram aplicados na base de dados, sendo que no final do processo, 12 *tags* foram selecionadas: <*a*>, <*em*>, <*h1*>, <*h2*>, <*h3*>, <*h4*>, <*h5*>, <*i*>, <*img*>, <*p*>, <*strong*> e <*title*>.

Porém, observou-se que as ações cobertas por essas *tags* ainda poderiam ser simuladas por duas *tags* distintas. A primeira é a *tag* <*i*>, que representa a formatação itálica. Essa formatação também é feita com a *tag* <*em*>, por isso, optou-se por ter as duas *tags* na listagem final.

Também foi notado que processo de títulos e subtítulos é feito pela presença das seis *tags* que variam da *tag* <*h1*> até a *tag* <*h6*>. Mas, essa última *tag* não estava presente na lista inicial. Por isso, optou-se por também adicioná-la.

[8] Disponível em: https://developer.mozilla.org/en-US/docs/Web/HTML/Element

Após as atualizações, a lista completa de *tags* ficou com 14 itens, sendo que cada um possui um significado na organização de um REA. A Tabela 15 lista as descrições e a utilização de cada *tag* selecionada.

Tabela 15 – *Tags* HTML selecionadas

| *tags* | **Descrição** | **Uso** |
|---|---|---|
| <*title*> | Define o título do documento | Título |
| <*h1*> | Define o título mais importante | Título |
| <*h2*> | Define o segundo título mais importante | Subtítulo |
| <*h3*> | Define o terceiro título mais importante | Subtítulo |
| <*h4*> | Define o quarto título mais importante | Subtítulo |
| <*h5*> | Define o quinto título mais importante | Subtítulo |
| <*h6*> | Define o título menos importante | Subtítulo |
| <*b*> | Define texto em negrito sem importância extra | Ênfase |
| <*strong*> | Define texto com importância forte | Ênfase |
| <*img*> | Usada para incorporar uma imagem | Imagem |
| <*i*>, <*em*> | Define uma parte do texto em um estilo alternativo | Itálico |
| <*a*> | Define um hyperlink para outra página | Link |
| <*p*> | Define um parágrafo | Parágrafo |

Fonte: Descrições baseadas em W3Schools.

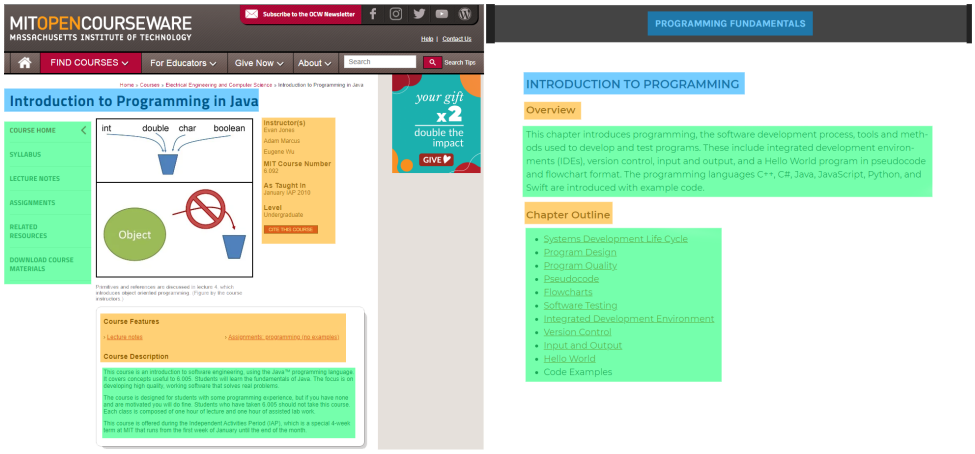
### 5.4.2 Funcionamento

O TAGGER investiga o problema de descoberta de REA usando um conjunto muito específico e reduzido de *tags* HTML. Isso é relevante por alguns fatores. Em primeiro lugar, as *tags* estão na base de páginas da web e estão presentes em todos os REA baseados na web (.html), sendo incorporadas de forma nativa, independente do formato do recurso digital. Isso permite que a abordagem seja usada em qualquer recurso disponível na Internet. Por exemplo, caso seja um vídeo disponível em uma plataforma *online*, o título da página estará dentro de uma *tag* <*title*>. Só isso já possibilita a recuperação do recurso, mesmo que nenhuma outra informação seja fornecida. Mas, para além disso, é comum que as páginas possuam diversas informações além do título, como descrições, imagens e outras informações.

O TAGGER foi desenvolvido considerando tudo isso, permitindo que os recursos (mesmo sem nenhum metadado) possam ser identificados. Além do mais, isso permite analisar os recursos de acordo com a sua heterogeneidade e a sua estrutura lógica. Como as *tags* HTML estão presentes em REA baseados na web, a heterogeneidade dos formatos é facilmente abordada. As diferentes estruturas que cada REA possui também são consideradas porque existem *tags* diferentes para suportar cada tipo de REA. Por exemplo, livros didáticos abertos geralmente têm vários capítulos e seções, então as *tags* de subtítulos são usadas <*h2*>, <*h3*>, <*h4*>, <*h5*> e <*h6*>. Normalmente os vídeos possuem uma descrição, por isso são adotadas as *tags* de parágrafo (<*p*>). A união de diferentes *tags* proporciona a construção dos mais variados REAs, como pode ser visto na Figura 14.

O TAGGER também foi concebido para permitir a utilização de pesos para cada *tag* HTML. Por exemplo, a *tag* <*title*> é única por arquivo .html. Essa é uma regra comum para

Figura 14 – Comparando dois REA de programação: um curso online e um livro digital




Fonte: Elaboração do autor a partir do MIT OpenCourseWare. Na figura, diferentes *tags* HTML são destacadas: H1 (azul), H2-H6 (laranja) e P (verde).

arquivos desse tipo. Assim sendo, ela deve ser usada apenas uma vez em todo o arquivo. Diferentemente, a *tag* <*p*>, dedicada ao processo de criação de parágrafos, sendo usada toda vez que um novo parágrafo é apresentado. Diante disso, caso o REA possua o título "Programação de Computadores: Usando o Java" na *tag* <*title*>, há altas chances daquele REA ser da linguagem de programação Java. Ou seja, a *tag* receberá um peso maior. Já, caso a mesma passagem ("Programação de Computadores: Usando o Java") esteja presente em algum parágrafo, as chances são menores. Logo, ele receberá um peso menor.

O TAGGER foi avaliado em diferentes oportunidades. Os resultados demonstraram aderência ao processo de recuperação dos REA, inclusive apoiando a recuperação de recursos relevantes em união com a IPAVL. Os resultados estão sendo relatados em um manuscrito que será submetido para o periódico *IEEE Transactions on Education*.

## 5.5 Ameaças à Validade

Uma das principais ameaças em torno dessa etapa da pesquisa refere-se ao desenvolvimento do banco de dados de REA. Caso esse banco fosse enviesado, poderia contaminar toda a criação dos mecanismos. Buscando reduzir a incidência desse risco, foram selecionadas plataformas abertas em três fontes distintas: plataformas identificadas no MS, no Estudo Exploratório e no OER World Map. Além do mais, todos os REA selecionados foram analisados manualmente pelo autor da pesquisa, garantindo que os materiais eram referentes ao ensino e aprendizado de programação introdutória.

No desenvolvimento da IPAVL, uma das principais ameaças é a presença de termos não relevantes. Para evitar a propagação desse risco, foi adotado um conjunto de critérios técnicos para filtrar e remover termos sem relevância. Ademais, os campos de identificação dos estudos (como DOI, nome do autor, afiliação e lista de referências) também foi removido. Por fim, foi

feita uma verificação manual para remover quebras de linha e organizar tabelas, evitando assim a inserção de termos inválidos.

Outra ameaça importante é referente ao desenvolvimento do TAGGER. Apesar das boas práticas serem recomendadas, muitos arquivos HTML podem apresentar problemas de formatação. Pode ser que uma *tag* de parágrafo (<*p*>) seja usada como lista ou como subtítulo. Um exemplo disso foi ilustrado na Figura 14. Outro problema é o "ruído" encontrado em páginas web. Nesse caso, o ruído representa outros tipos de conteúdo além do REA, como cabeçalhos e rodapés do site. Para reduzir tais desafios, o TAGGER utiliza o uso pesos conforme a importância de cada *tag* HTML. Dessa forma, as *tags* comuns terão uma influência menor na recuperação do recurso, já que elas podem ter múltiplas aplicações. As *tags* extraídas dos arquivos HTML estavam dentro da *tag* <*body*>, removendo automaticamente as *tags* que armazenam dados de cabeçalho (<*header*>) e rodapé (<*footer*>).

## 5.6 Considerações Finais

Este capítulo apresentou o processo de criação de dois mecanismos desenvolvidos na tese: a IPAVL e o TAGGER. Os mecanismos foram projetados para funcionarem de forma independente, atuando na identificação e na recuperação de REA, respectivamente. No entanto, ambos podem ser unificados em uma ferramenta para prover uma solução mais robusta. Por exemplo, os termos da IPAVL podem ser contabilizados de acordo com as *tags* e seus respectivos pesos usando o TAGGER.

Para avaliar cada mecanismo, foi realizado um estudo de caso em múltiplas plataformas de REA, utilizando diferentes usuários. O estudo de caso avaliou como a IPAVL e o TAGGER podem apoiar as atividades de identificação e de recuperação de recursos introdutórios de programação. Os resultados evidenciaram a capacidade de ambos os mecanismos em reduzirem os desafios inerentes dessas atividades bem como a sua adoção por diferentes perfis de usuários. Essa avaliação é descrita a seguir.

CAPÍTULO

6

# VALIDAÇÃO DOS MECANISMOS DE IDENTIFICAÇÃO E RECUPERAÇÃO DE REA

O objetivo deste capítulo é apresentar o processo da validação da IPAVL e do TAGGER. De forma resumida, optou-se pela execução de múltiplos estudos de casos inspirados pela abordagem de Yin (2010). Os casos foram selecionados visando três avaliações distintas, conforme os grupos de desafios relatados no Capítulo 4 (condições causais, questões de contexto e efeitos).

O conteúdo do capítulo encontra-se organizado da seguinte forma: a Seção 6.1 detalha a estratégia de validação adotada. A Seção 6.2 apresenta os principais resultados do primeiro estudo de caso, realizado no EngageCSEdu. A Seção 6.3 aborda os resultados alcançados com os especialistas no MIT OpenCourseWare. A Seção 6.4 expõe o terceiro estudo de caso, desenvolvido com o OER-Chat. Por fim, a Seção 6.5 sintetiza as considerações finais do capítulo.

## 6.1 Estudo de Caso

O estudo de caso foi selecionado para realizar a validação da IPAVL e do TAGGER. A opção pelo estudo de caso ao invés de outros métodos ocorreu em virtude de duas características sumarizadas por Yin (2010): (1) ser uma investigação empírica que analisa um fenômeno em profundidade e em seu contexto real; e (2) permitir a análise de diferentes variáveis, com múltiplas fontes de evidências e a adoção de proposições teóricas para orientar o processo de coleta e análise de dados.

Em união, essas duas características do estudo de caso permitiram avaliar em profundidade os mecanismos de identificação e de recuperação de REA de programação introdutória, evitando assim a adoção de ambientes simulados ou controlados que poderiam enviesar a análise por não serem aderentes à realidade educacional. Ao mesmo tempo, permitiu a coleta de dados de múltiplas fontes e favoreceu a análise de diferentes perspectivas sobre identificação e recuperação de REA.

O estudo de caso foi conduzido com base nas premissas propostas por Yin (2010). O restante da seção sumariza as principais etapas adotadas.

### 6.1.1 Questões de Pesquisa

Inicialmente, foi elaborada a QP responsável por guiar toda a investigação do estudo de caso. Considerando a questão principal, que norteia toda a investigação, a seguinte QP foi elaborada:

- **Como os mecanismos propostos (IPAVL e TAGGER) podem reduzir os desafios relacionados à identificação e à recuperação de REA voltados ao ensino e aprendizado introdutório de programação?**

### 6.1.2 Proposições

De acordo com Yin (2010), as proposições dirigem a atenção do pesquisador para algo que deve ser analisado no estudo de caso. Ou seja, as proposições direcionam a atenção para o que será investigado, gerando conjunturas teóricas a serem investigadas de modo a solucionar a questão investigada. Além disso, as proposições evitam que a atenção do pesquisador seja dispersa, coletando ou analisando dados que não contribuirão diretamente para a condução do estudo de caso.

As proposições do estudo de caso foram elaboradas mediante a conexão entre os mecanismos e o escopo de investigação. Como resultado, três proposições foram elaboradas:

$(P_1)$ *Os mecanismos são aderentes ao ensino introdutório de programação.*

$(P_2)$ *Os mecanismos podem ser adaptados para diferentes perfis de usuários e de recursos.*

$(P_3)$ *Os mecanismos podem facilitar o uso dos REA.*

A primeira proposição $(P_1)$ investiga se os mecanismos propostos são aderentes ao ensino introdutório de programação. De forma resumida, essa proposição analisa se os mecanismos são capazes de **identificar** corretamente os recursos válidos (com conteúdo de programação introdutória) e a remover os recursos inválidos (que não possuem conteúdo de programação introdutória).

Já a segunda proposição $(P_2)$ examina se os mecanismos podem ser adaptados para diferentes tipos de usuários. Um dos principais pontos de falha na disseminação de materiais abertos é a construção de plataformas que seguem uma natureza fechada, produzidas para padronizar a busca desses materiais, sem considerar os perfis dos diferentes usuários. Por isso, essa proposição está alinhada à ideia de **recuperação** de REA.

A terceira proposição ($P_3$) investiga se os mecanismos são capazes de facilitar o uso de REA por usuários interessados no ensino introdutório de programação. Nesse caso, se os mecanismos ajudam a **identificar** e a **recuperar** REA voltados ao enino introdutório de programação.

### 6.1.3 Unidade de Análise

Por definição, as unidades de análise estão intimamente relacionadas à(s) QP(s) que guia(m) o estudo de caso (YIN, 2010). Como pode ser visto, a QP deste estudo de caso examina a capacidade dos mecanismos em reduzir os desafios associados à identificação e recuperação dos REA. Os desafios que a QP analisa foram apresentados no Capítulo 4 e possuem três elementos distintos: condições causais, questões de contexto e efeitos.

Diante dessa estrutura, os três elementos foram adotados como sendo três unidades de análise distintas para o estudo de caso. Essa opção se deu em virtude da dinâmica dos desafios, tendo em vista que as condições causais ocorrem antes mesmo do usuário interagir com a plataforma aberta. Já as questões de contexto são relacionadas ao momento que o usuário usa tal plataforma. Por fim, os efeitos são sentidos pelos usuários após a adoção de tais plataformas.

### 6.1.4 Vinculação dos Dados às Proposições

Este componente é responsável por facilitar o processo de análise do estudo de caso. Para isso, deve-se selecionar uma estratégia analítica em conjunto com uma técnica analítica que permita a exploração e análise coerente dos dados coletados pela pesquisa.

A estratégia analítica apresenta uma visão geral sobre os procedimentos a serem desenvolvidos. Seguindo os critérios de Yin (2010), foi selecionado o uso de proposições. Para isso, foram utilizadas as proposições teóricas do estudo de caso para identificar quais os principais dados deveriam ser coletados e analisados. Essa técnica é detalhada em cada caso selecionado.

De acordo com Yin (2010), a técnica analítica tem a intenção especial de lidar com os problemas de validade interna e externa do estudo de caso. Para reduzir o risco de propagação de diferentes ameaças, optou-se pela adoção de duas técnicas: a combinação do padrão (comparar um padrão observado com um padrão previsto) e a síntese cruzada dos dados (investigar dados oriundos de diferentes casos). As estratégias adotadas são detalhadas nos caso analisados.

### 6.1.5 Critérios para Interpretação dos Achados

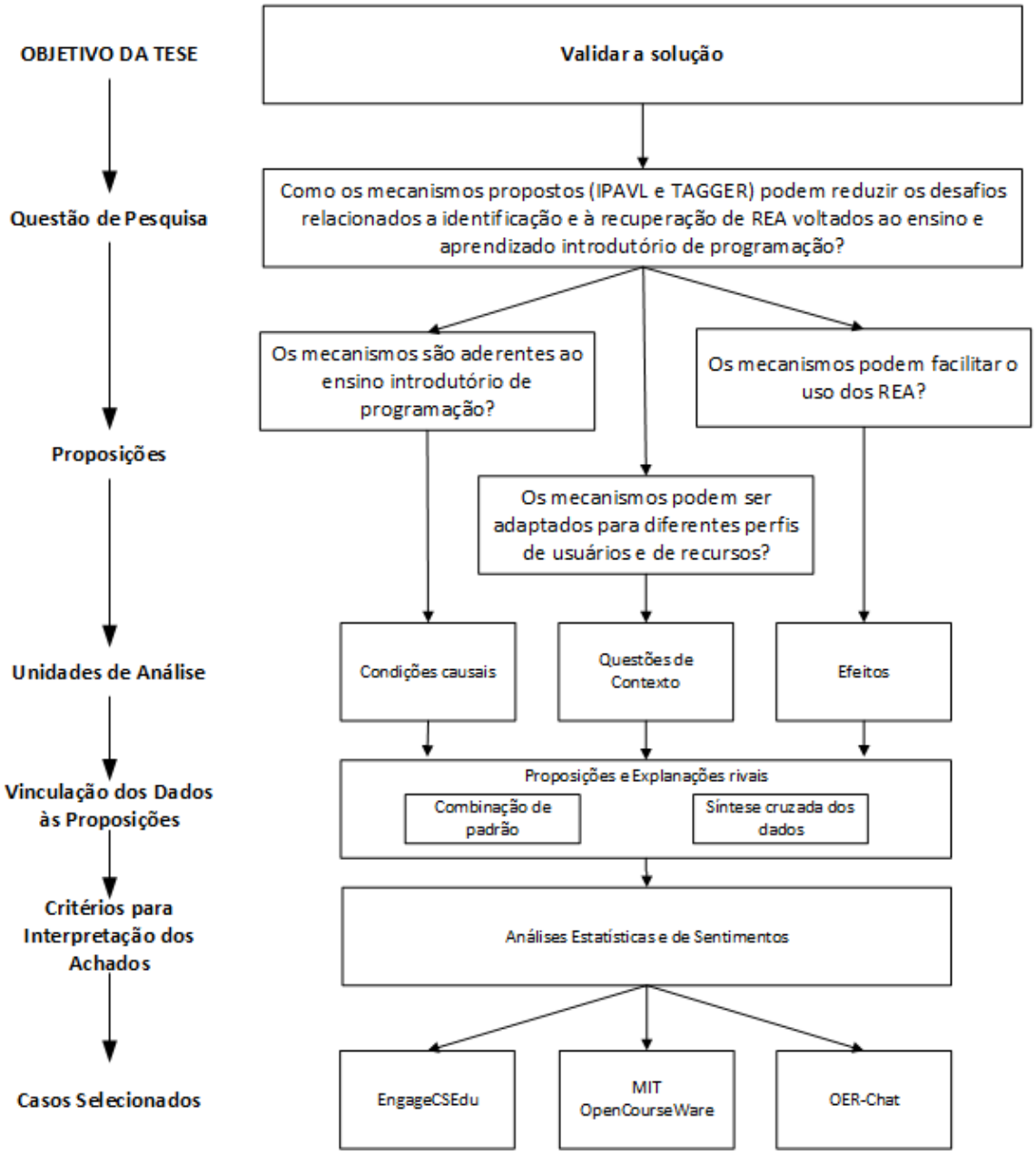
O último componente do estudo de caso é a definição de quais critérios serão adotados para interpretar os achados ou as descobertas da investigação. Yin (2010) explica que podem ser utilizadas diferentes critérios, conforme a estrutura de cada pesquisa.

Devido à natureza da investigação conduzida neste doutorado, optou-se pela adoção de análises estatísticas e também análises de sentimentos. O procedimento de cada análise será apresentado conforme a apresentação dos resultados de cada caso. Cabe ainda destacar que o estudo caso relatado foi instanciado em um teste piloto, analisando 100 REA de programação introdutória. A análise demonstrou que as premissas do estudo de caso estavam estáveis e por isso foi feita a seleção dos casos reais a serem analisados.

### 6.1.6 Síntese

A estrutura do estudo de caso é apresentada na Figura 15.

Figura 15 – Síntese do projeto do estudo de caso e dos casos selecionados




Fonte: Dados da Pesquisa.

As unidades de análise do estudo de caso são representadas pelos agrupamentos de desafios que os usuários encaram ao identificar e recuperar um REA de programação introdutória (Capítulo 4). Assim sendo, optou-se por selecionar três casos distintos de avaliação.

Também foi decidido que as plataformas usadas na elaboração dos mecanismos não seriam utilizadas para a validação. Essa escolha se deu em virtude do que foi chamado de dependência cíclica, na qual a mesma plataforma é usada para montar a solução e depois é usada para realizar a validação da sua eficiência. Apesar dessa prática ser comum em diversas investigações, o problema de identificar e de recuperar REA de programação introdutória é genérico e ocorre em diferentes tipos de plataformas abertas. Logo, se os mecanismos conseguissem auxiliar o processo de identificação e de recuperação, de fato, isso deveria ocorrer em outras plataformas também.

Além do mais, considerando a estrutura do projeto do estudo de caso, foi decidido que os casos a serem analisados deveriam fornecer dados consoantes às proposições e às unidades de análise. Ou seja, deveriam corresponder e apoiar essas premissas da investigação.

Após diferentes buscas, três casos distintos de validação foram selecionados. A plataforma EngageCSedu, a plataforma MIT OpenCourseWare e uma ferramenta denominada OER-Chat, que foi desenvolvida no âmbito desta pesquisa de doutorado. A seguir, cada caso selecionado é apresentado e detalhado.

## 6.2 Caso 1: EngageCSEdu

O primeiro caso selecionado foi o EngageCSedu. De forma resumida, essa plataforma busca disseminar o uso de REA testados em salas de aula para o ensino de Ciência da Computação, com foco central em materiais voltados aos cursos introdutórios.

O EngageCSedu é fruto de uma colaboração intensa entre diferentes pesquisadores e entidades. O projeto foi patrocinado pelo Google e conta com o apoio da *ACM Education Board*, tendo a professora Dr[a]. Briana Morrison (*University of Virginia*) como uma das responsáveis pelo seu desenvolvimento.

A análise do caso foi realizada por meio de uma visita técnica na *University of Virginia* (Processo: #2019/26871-4), contando com a colaboração da professora Dr[a]. Briana Morrison para supervisionar toda a investigação. Durante o período de análise, o acervo da plataforma contava com 220 REA, sendo que 191 recursos eram materiais voltados ao ensino de programação introdutória e o restante (29) eram de outras áreas da Ciência da Computação. Todo o acervo foi organizado manualmente pelos diferentes colaboradores do projeto, que envolvem, em sua maioria, professores e pesquisadores da área do ensino e aprendizado de Ciência da Computação.

### 6.2.1 Preparação e Coleta de Dados

A etapa de preparação e coleta de dados é crucial para a execução de um estudo de caso. De forma resumida, ela é responsável por identificar onde, quais e como os dados serão coletados. Inicialmente, foram adotadas duas fontes distintas de dados:

- **Reuniões:** Foram realizadas reuniões com a professora Dr[a]. Briana Morrison para compreender a estrutura da plataforma bem como elucidar potenciais dúvidas sobre REA.
- **Arquivos de dados:** Foram coletados todos os dados disponíveis na plataforma, que podem ser organizados em três grupos distintos:
    1. Arquivos *.html*: é uma página web que exibe os metadados do recurso bem como uma descrição do material e recomendações de uso.
    2. Arquivos incorporados: são diferentes tipos de arquivos utilizados para animar, formatar ou estruturar a página *web* anteriormente relatada, como *scripts*, imagens, folhas de estilos e outros tipos.
    3. Materiais complementares: são materiais que adicionam descrições sobre o recurso, fornecendo, por exemplo, instruções de uso, arquivos para *download*, entre outras opções. Esses materiais foram adicionados algum tempo depois do funcionamento da plataforma. A ideia original era evitar um problema que ocorria: a ausência de uma descrição completa sobre o REA. Por isso mesmo, nem todos os REA possuíam materiais dessa natureza.

Os dados coletados eram, essencialmente, em formato bruto e a sua análise não poderia fornecer uma contribuição direta à QP elaborada no início do capítulo. Por isso, foi necessário transformar os dados para realizar a análise.

Em primeiro lugar, o caso do EngageCSEdu envolve a investigação da primeira proposição do estudo de caso: $(P_1) \rightarrow$ *os mecanismos são aderentes ao ensino introdutório de programação*. Os dados coletados na plataforma permitiram estabelecer uma variável para observar e analisar tal proposição, que é a classificação dada pelos especialistas.

Em segundo lugar, a unidade de análise do caso são três condições causais que influenciam o processo de identificação e de recuperação de REA, que são: (1) metadados e motor de busca, (2) propósito do REA e (3) indisponibilidade. Para estudar cada uma, os dados forneciam uma série de informações, a saber:

- Metadados e motor de busca: foi possível extrair toda a lista de metadados utilizando os arquivos *.html* e compreender o processo de busca da plataforma por meio das reuniões com a professora Dr[a]. Briana Morrison.

- Propósito do REA: foi possível analisar o conteúdo dos arquivos .html e os materiais complementares para identificar o propósito de cada recurso.
- Indisponibilidade: foi possível verificar se o REA estava disponível analisando a URL do recurso, a URL dos materiais complementares e erros de códigos ou falhas nos arquivos embarcados.

Em terceiro lugar, o processo de vinculação dos dados às proposições foi desenvolvido com base no uso de padrões simples (YIN, 2010). Nesse caso, as variáveis dependentes ou independentes são selecionadas e verifica-se se a combinação do padrão é possível. Por exemplo, uma variável identificada foi a classificação dos especialistas. Com isso, foi possível estabelecer que o padrão esperado é que os mecanismos identifiquem e recuperem os REA que foram marcados como introdutórios por tais especialistas.

Todos os REA do EngageCSEdu foram baixados (220). Logo em seguida, foram gerados dois *datasets*: um com recursos inválidos (não eram voltados ao ensino de programação introdutória, conforme a classificação da plataforma, receberam o rótulo *Other*) e outro com recursos válidos (classificados como sendo introdutórios, receberam o rótulo *CS1*).

Os REA possuem diferentes estruturas mas, para realizar a validação, foram usados apenas os recursos disponíveis em cada mecanismo: as *tags* HTML do TAGGER e os termos presentes na IPAVL. Ao todo, foram gerados três modelos distintos:

1. O primeiro modelo (Tabela 16) foi gerado pelo TAGGER. Nesse modelo, apenas é contabilizado o total de termos que cada *tag* HTML possui.
2. O segundo modelo (Tabela 17) é apoiado no uso dos dois mecanismos. Nesse caso, é contabilizado apenas o total de termos da IPAVL que cada *tag* HTML possui.
3. O terceiro modelo (Tabela 18) é uma especificação dos mecanismos e consiste na adoção de pesos, que podem ser configurados conforme a importância de cada *tag* e/ou termo.

O terceiro modelo foi elaborado com base na capacidade de personalizar ambos os mecanismos. Para o seu desenvolvimento, foi adotada a estratégia de ajuste dos parâmetros, conforme sugerido por Mouriño-García *et al.* (2018): usar um intervalo de pesos para definir a relevância consoante a sequência Fibonacci (0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89). Para definir a relevância de cada item do modelo, foram usadas as métricas propostas por Kabir, Kabir e Amin (2015), que organizam a importância das *tags* HTML segundo uma hierarquia de prioridade. Os pesos finais estão disponibilizados na Tabela 19.

É necessário destacar que o primeiro valor da sequência Fibonacci (0) não foi usado, pois sua adoção iria neutralizar a *tag* P, tendo em vista que o modelo é gerado com base na multiplicação entre o modelo 01, modelo 02 e os ajustes expostos na Tabela 19.

Todos os modelos possuem como parâmetros o nome do REA, a contabilização da ocorrência por tag, o total de termos e a *label* (0 para recursos inválidos, não são voltados à programação introdutória; e 1 para recursos válidos, voltados ao ensino introdutório de programação).

Tabela 16 – Exemplo do Modelo 01 - TAGGER

| file | title | h1 | h2 | h3 | h4 | h5 | h6 | img | b | strong | i | em | p | a | total | label |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Loop Lab [...] | 3 | 0 | 0 | 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 43 | 19 | 71 | 1 |
| CS2 Graphical [...] | 3 | 3 | 7 | 0 | 0 | 0 | 0 | 0 | 0 | 10 | 0 | 0 | 171 | 19 | 210 | 0 |

Fonte: Dados da Pesquisa.

Tabela 17 – Exemplo do Modelo 02 - TAGGER e IPAVL

| file | title | h1 | h2 | h3 | h4 | h5 | h6 | img | b | strong | i | em | p | a | total | label |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Loop Lab [...] | 2 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 2 | 9 | 1 |
| CS2 Graphical [...] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 13 | 1 | 14 | 0 |

Fonte: Dados da Pesquisa.

Tabela 18 – Exemplo do Modelo 03 - TAGGER, IPAVL e pesos

| file | title | h1 | h2 | h3 | h4 | h5 | h6 | img | b | strong | i | em | p | a | total | label |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Loop Lab [...] | 534 | 0 | 0 | 612 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 215 | 76 | 1437 | 1 |
| CS2 Graphical [...] | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2223 | 38 | 2261 | 0 |

Fonte: Dados da Pesquisa. O valor é calculado aplicando os parâmetros da Tabela 19 * modelo 01 * modelo 02

Tabela 19 – Ajustes de parâmetros do modelo 03

| **Tag** | **Peso** |
|---|---|
| *<title>, <h1>* | 89 |
| *<h2>* | 55 |
| *<h3>* | 34 |
| *<h4>* | 21 |
| *<h5>* | 13 |
| *<h6>* | 8 |
| *<b>, * | 5 |
| *<img>* | 3 |
| *, <i>, <a>* | 2 |
| *<p>* | 1 |

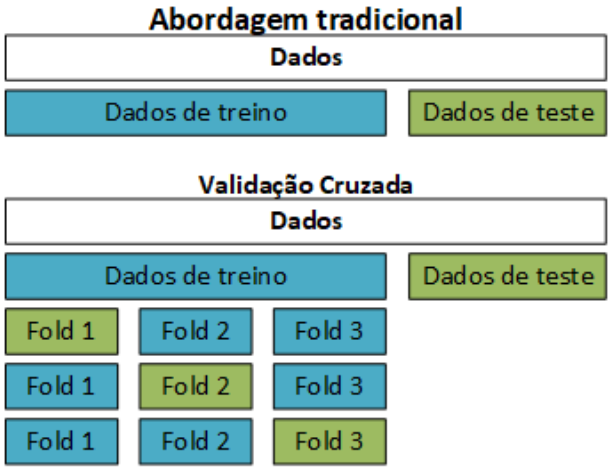
Fonte: Dados da Pesquisa.

Por vezes, abordagens que empregam aprendizado de máquina dividem os dados em dois grupos: os dados de treino e os dados de teste. O primeiro grupo é responsável por treinar o modelo de classificação. Depois, a avaliação final do modelo é feita usando apenas os dados do segundo grupo. Em tese, a abordagem permite avaliar o desempenho do classificador fazendo o treinamento e, depois, o seu teste.

Contudo, são necessárias diversas precauções conforme a natureza dos dados utilizados. Um dos maiores riscos é o problema do *overfitting*, definindo por Dietterich (1995) como um processo no qual o classificador memoriza as particularidades dos dados de treino ao invés de buscar uma regra preditiva para dados futuros. Por isso, o classificador é bom apenas nos dados que foram usados para treinamento e não é eficiente para generalizar resultados em novos dados. Esse risco é potencialmente grande na amostra usada nesta análise, tendo em vista que o tamanho amostral entre recursos introdutórios e não introdutórios estava desbalanceado.

Para evitar a propagação do *overfitting* na análise, foi adotada a técnica de validação cruzada (PEDREGOSA *et al.*, 2011). De forma resumida, essa técnica divide os dados de treinamento em diferentes partes (*folds*) e utiliza esse recurso para realizar diferentes treinamentos no classificador. Por exemplo, os dados de treinamento podem ser divididos em três *folds*, assim o *Fold 1* utiliza dados de treino dos *folds* 2 e 3 e realiza o teste com os dados do *Fold* 1. Já o *Fold 2* realiza o treinamento com os dados dos *folds* 1 e 3, mas o teste com os dados do *Fold 2*. Para facilitar a compreensão desse processo, a Figura 16 apresenta ilustra o funcionamento da validação cruzada.

Figura 16 – Abordagem tradicional e a validação cruzada.




Fonte: Imagem inspirada pelo guia do usuário do Scikit-learn.

A validação cruzada permite analisar o desempenho do classificador em diferentes cenários, sem aplicar os dados de teste. Para torná-la ainda mais robusta, foi garantido que todos os *folds* tivessem a mesma proporção de dados, isto é, cada *fold* teria proporcionalmente o mesmo total de REA introdutórios ou não introdutórios.

Dois classificadores foram utilizados: o *Support Vector Machine* (SVM) e o *Decision Tree* (DT) por serem adequados ao processo de identificação e recuperação de REA, conforme relatado pelos autores Mouriño-García *et al.* (2018) e Rathod e Cassel (2013). No estudo desenvolvido por Rathod e Cassel (2013), foi criado um motor de busca para materiais de Ciência da Computação, o que se assemelha com o desafio relacionado ao uso de metadados e motores de busca que este caso busca analisar. No estudo de Mouriño-García *et al.* (2018) foi adotado o uso de pesos nos

classificadores, como será realizado no terceiro modelo.

Para medir a eficiência de cada classificador, foram usadas as métricas tradicionalmente utilizadas para mensurar o desempenho de um algoritmo de classificação. Para calcular cada métrica é necessário analisar a matriz de confusão, como mostra a Figura 17.

Figura 17 – Matriz de Confusão

| | | EngageCSEdu | |
|---|---|---|---|
| | | CS1 | Other |
| Mecanismos | CS1 | Verdadeiro Positivo (VP) | Falso Positivo (FP) |
| | Other | Falso Negativo (FN) | Verdadeiro Negativo (VP) |

Fonte: Elaborado pelo autor com base nos dados coletados no EngageCSEdu.

Em tese, bons algoritmos de classificação conseguem prever corretamente os verdadeiros positivos e verdadeiros negativos. Ao mesmo tempo, os erros de previsão (falso positivo e falso negativo) devem ser reduzidos. Para verificar o desempenho, a Acurácia ($A$) foi usada para mensurar a efetividade geral do algoritmo classificador. Além disso, quatro indicadores inspirados em Mouriño-García *et al.* (2018) também foram adotados:

1. Precisão ($P$): avalia o poder preditivo do algoritmo classificador.
2. *Recall* ($R$): define a fração das classificações previstas corretamente em relação ao conjunto total de classificações corretas.
3. $F_1$ *Score* ($F_1$): combina a precisão e o *recall* para fornecer um indicador da performance geral.
4. *Receiver Operator Characteristic curve* (*ROC*): mostra como o número de classificações positivas classificadas corretamente varia conforme o número de classificações negativas classificados incorretamente.

### 6.2.2 Resultados

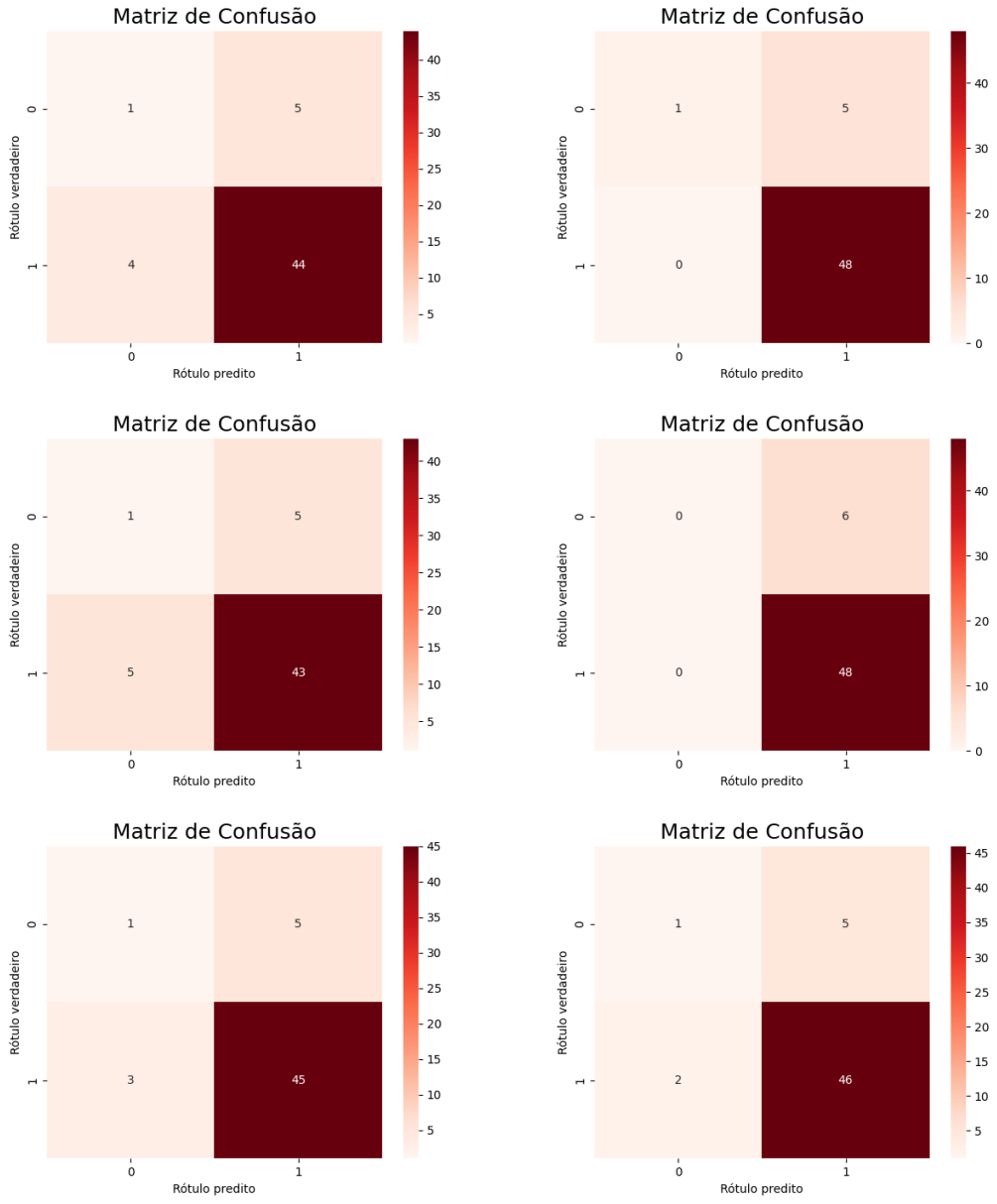
A matriz de confusão demonstra que o algoritmo SVM foi superior no processo de classificação correta dos REA. O primeiro modelo, por exemplo, previu corretamente a categoria de 49 REA e errou apenas 5. O modelo 2 previu corretamente 48 REA e errou 6 classificações. Já o modelo 3 alcançou 47 previsões corretas e errou 7 vezes.

Por sua vez, o algoritmo DT obteve um desempenho levemente reduzido, sendo que no primeiro modelo acertou 45 previsões e errou 9. O segundo modelo alcançou 44 previsões

corretas e errou 10. Por fim, o terceiro modelo conseguiu identificar corretamente 46 REA e errou 8.

Para ilustrar todas as matrizes geradas, a Figura 18 exibe os resultados de cada modelo e algoritmo adotados.

Figura 18 – Matriz de Confusão dos modelos 01, 02 e 03 (DT, esquerda; SVM, direta)




Fonte: Dados da Pesquisa.

O primeiro modelo foi inspirado pela TAGGER e utiliza apenas o total de termos que as *tags* HTML possuem. Esse modelo em particular obteve um bom desempenho, alcançando Acurácia, Precisão, *Recall* e $F_1$ Score acima 0.82 em ambos os algoritmos de classificação.

Já o segundo modelo foi inspirado pela unificação da TAGGER e do IPAVL. Nesse caso, é contabilizado apenas o total de termos da IPAVL que cada *tag* HTML possui. Esse modelo

também obteve um desempenho consistente em ambos os algoritmos. No entanto, a Acurácia, Precisão, *Recall* e $F_1$ *Score* foram acima de 0.80, sendo que em linhas gerais, o algoritmo SVM também obteve um desempenho geral melhor que o DT.

O terceiro modelo alcançou um desempenho superior, tendo a sua Acurácia, Precisão, *Recall* e $F_1$ *Score* superiores a 0.84. Apesar de ambos os algoritmos terem aumentando o seu desempenho, o algoritmo SVM ainda obteve superioridade de classificação.

Para facilitar a compreensão, bem como a sua posterior análise e discussão, a Tabela 20 apresenta uma síntese de cada métrica e índice alcançado segundo o modelo e o algoritmo classificador.

Tabela 20 – Indicadores gerais de desempenho

| **Métrica** | **Modelo 01** | | **Modelo 02** | | **Modelo 03** | |
|---|---|---|---|---|---|---|
| | *DT* | *SVM* | *DT* | *SVM* | *DT* | *SVM* |
| *A* | 0.8333 | 0.9074 | 0.8148 | 0.8889 | 0.8519 | 0.8704 |
| *P* | 0.8980 | 0.9057 | 0.8958 | 0.8889 | 0.9000 | 0.9020 |
| *Recall* | 0.9167 | 1.0000 | 0.8958 | 1.0000 | 0.9375 | 0.9583 |
| $F_1$ | 0.9072 | 0.9505 | 0.8958 | 0.9412 | 0.9184 | 0.9293 |
| *ROC* | 0.5417 | 0.6771 | 0.5156 | 0.6493 | 0.5521 | 0.5417 |

Fonte: Dados da Pesquisa.

Ao aprofundar a análise dos dados é possível verificar o desempenho dos algoritmos de forma mais nítida. O conjunto de teste possuía 54 REA, sendo que 6 eram materiais voltados ao ensino avançado e 48 eram REA introdutórios. O melhor cenário do algoritmo DT (modelo 03) foi capaz de prever 45 REA introdutórios de 48 possíveis, um índice de acerto superior a 93% considerando apenas os recursos voltados ao ensino introdutório. No entanto, o mesmo modelo conseguiu prever corretamente apenas 1 REA que era avançado, tendo um índice próximo a 16% ao se considerar apenas os recursos não introdutórios.

Um recorte mais detalhado demonstra que o algoritmo DT obteve um desempenho relevante quando é analisado apenas a sua capacidade de identificar os REA voltados ao ensino introdutório. Sua precisão, *recall* e $F_1$ *Score*, nos três modelos, foi precisão a 0.88, como exposto na Tabela 21.

Tabela 21 – Indicadores de desempenho por classe do algoritmo DT

| **Classe** | **Modelo** | *P* | *Recall* | $F_1$ |
|---|---|---|---|---|
| *Other* | Modelo 01 | 0.2000 | 0.1667 | 0.1818 |
| | Modelo 02 | 0.1667 | 0.1667 | 0.1667 |
| | Modelo 03 | 0.2500 | 0.1667 | 0.2000 |
| *CS1* | Modelo 01 | 0.8990 | 0.9167 | 0.9072 |
| | Modelo 02 | 0.8958 | 0.8958 | 0.8958 |
| | Modelo 03 | 0.9000 | 0.9375 | 0.9184 |

Fonte: Dados da Pesquisa.

Já o algoritmo SVM alcançou resultados ainda mais interessantes. Ao analisar e classificar os recursos não introdutórios, obteve um desempenho perfeito de precisão no modelo 01 e 0.33

no modelo 03. Todavia, esse algoritmo zerou todos os índices possíveis no modelo 02. Agora, considerando os REA voltados ao ensino introdutório, o algoritmo obteve um desempenho superior a 0.88. No caso da precisão, o modelo 02 performou com 0.87 e os modelos 01 e 03 atingiram ao menos a marca de 0.90. O *Recall* também foi perfeito nos modelos 01 e 02, tendo atingido 0.95 no terceiro modelo. Além de tudo isso, destaca-se ainda a capacidade geral de performance, que foi superior a 0.94 em todos os modelos analisados. Todos os índices do algoritmo SVM estão disponíveis na Tabela 22.

Tabela 22 – Indicadores de desempenho por classe do algoritmo SVM

| **Classe** | **Modelo** | *P* | *Recall* | $F_1$ |
|---|---|---|---|---|
| *Other* | Modelo 01 | 1.0000 | 0.1667 | 0.2857 |
| | Modelo 02 | 0.0000 | 0.0000 | 0.0000 |
| | Modelo 03 | 0.3333 | 0.1667 | 0.2222 |
| *CS1* | Modelo 01 | 0.9057 | 1.0000 | 0.9505 |
| | Modelo 02 | 0.8889 | 1.0000 | 0.9412 |
| | Modelo 03 | 0.9000 | 0.9583 | 0.9583 |

Fonte: Dados da Pesquisa.

### 6.2.3 Discussões

Diante dos índices alcançados ficou evidente que todos os modelos possuem uma consistência interna. Isso quer dizer que independente do mecanismo adotado, os resultados foram estáveis, sem apresentar grandes perturbações na disposição dos resultados. Esse é um elemento que fortalece sua aplicação dentro dos REA, considerando a necessidade de adaptação dos recursos.

Evidentemente, é muito difícil comparar os resultados aqui apresentados com outros trabalhos. Esse problema ocorre pela falta de dados que permitem analisar o desempenho de outros estudos. Mas, o problema principal que impede tal comparação é pela natureza dos REA. Os mecanismos de identificação e de recuperação de REA desenvolvidos durante esta pesquisa são dedicados ao ensino e aprendizado de programação introdutória. Mas, a maior parte dos trabalhos que realizam uma tarefa similar focam em organizar materiais genéricos usando como grandes áreas, como Matemática, História, Computação, etc. Essa atividade, apesar de complexa, é mais simples do que separar recursos que são de programação, mas podem ter um conteúdo avançado ou introdutório.

No entanto, a IPAVL e o TAGGER foram desenvolvidos para reduzir esse problema. Diante dos resultados alcançados, evidencia-se que os mecanismos atingiram esse objetivo, obtendo resultados relevantes sobre as métricas de precisão, *Recall* e $F_1$ *Score*.

Todavia, os resultados aqui expostos demonstram que o uso dos mecanismos foi eficiente para atingir um desempenho relevante na importante tarefa de identificar REA para o ensino introdutório de programação. Sua aplicação em dois algoritmos distintos, SVM e DT, demonstram

que a replicação e a generalização dos modelos pode encontrar desempenho ainda melhor com outras técnicas.

#### 6.2.3.1 Conectando os achados com o projeto do estudo de caso

Este caso específico lida com a seguinte proposição: *os mecanismos são aderentes ao ensino introdutório de programação?* Como exposto, a proposição busca analisar se os mecanismos seriam de fato capazes de identificar corretamente os recursos válidos (com conteúdo de programação introdutória) e a remover os recursos inválidos (que não possuem conteúdo de programação introdutória).

Diante dos resultados expostos, é possível identificar o potencial dos mecanismos em identificarem os REA introdutórios. Essa afirmação se dá em decorrência de duas percepções. A primeira delas se refere a consistência interna dos mecanismos para identificar os REA introdutórios em todos os modelos e em todos os algoritmos. Como foi exposto, todos os índices de Precisão, *Recall* e $F_1$ *Score* foram superiores a 0.87 para os recursos introdutórios. Isso quer dizer que independentemente do mecanismo, modelo ou algoritmo adotado, os resultados foram estáveis, sem apresentarem grandes perturbações na disposição dos resultados. A segunda percepção é baseada nos índices alcançados pelo algoritmo SVM, que conseguiu alcançar uma Precisão e $F_1$ *Score* superiores ao algoritmo DT. Isso representa que há espaço para melhorias, como seleção de outros algoritmos ou até mesmo um novo ajuste nos parâmetros adotados. Ademais, como será exposto nas limitações, a amostra reduzida de REA não introdutório pode ter limitado o alcance dos resultados obtidos. Naturalmente, existem pontos de melhoria que demonstram possíveis otimizações. Para aprofundar essa discussão, dois novos casos foram executados e são apresentados nas próximas seções.

Este caso, em particular, lida com as condições causais que envolvem a identificação e a recuperação de REA. A saber, são três os problemas relatados: metadados e motor de busca, propósito do REA e indisponibilidade. A indisponibilidade não pôde ser mensurada diretamente devido à estratégia utilizada. Mesmo assim, foi realizada uma verificação na integridade da estrutura dos REA. Em suma, foram observadas duas regras: a primeira delas é a presença de uma *tag* <*title*> no recurso. A segunda é a abertura e fechamento de todas as tags.

Todos os REA e materiais analisados contavam com uma *tag* <*title*>, favorecendo o uso dos mecanismos. Seis REA estavam formatados incorretamente, isto é, as *tags* não eram abertas e fechadas. No entanto, esse problema não ocasionou nenhum erro ou imprevisão no processo de análise dos dados. Mesmo com recursos formatados de modo incorreto, o TAGGER alcançou êxito.

Em relação aos metadados e motor de busca, esta pesquisa demonstrou que muito pode ser feito e otimizado por meio do uso dos mecanismos. É necessário reafirmar que o TAGGER lida apenas com um conjunto de *tags* HTML, focando na forma como o conteúdo foi estruturado, adotando convenções e padrões de programação ao invés de usar apenas os metadados. Já a

IPAVL apresenta uma lista de termos que pode ser identificado nos metadados ou no conteúdo dos recursos. Os resultados aqui apresentados demonstraram que ambos os mecanismos podem fornecer soluções às plataformas abertas, otimizando as suas buscas e reduzindo a dependência dos metadados e do motor de busca.

Já o propósito do REA é um assunto delicado, tendo em vista que ele também está relacionado com a dificuldade de definir o que é programação introdutória. Mesmo com essa dificuldade em vista, ambos os mecanismos foram aderentes dentro do desafio analisado. Mesmo não sendo possível alcançar uma identificação perfeita, ou seja, acertar 100% dos REA que são introdutórios e 100% dos REA que possuem conteúdo avançado, foi possível observar que muitas tarefas que hoje demandam a classificação de especialistas podem ser aperfeiçoadas. A adoção de um modelo simples, contabilizando termos nas tags, pode resultar em amplas revisões de coleções, reduzido o tempo e o esforço gasto nessa tarefa.

#### 6.2.3.2 Ameaças à validade

A principal limitação deste caso foi o tamanho da amostra, sobretudo o total de recursos não introdutórios. Isso pode ter limitado o alcance dos resultados bem como a generalização dos mesmos. Para evitar a propagação desse viés na análise, foi adotada a estratégia de validação cruzada, reduzindo possíveis contaminações.

Outro ponto de atenção é a dificuldade em definir o que é ensino introdutório de programação. Para evitar o viés pessoal de classificação, os REA foram coletados da plataforma EngageCSEdu e as classificações usadas pela plataforma serviram de guia para a análise efetuada.

## 6.3 Caso 2: MIT OpenCourseWare

O segundo caso selecionado foi o MIT OpenCourseWare. A plataforma possui um conjunto amplo de REA voltados em diferentes áreas da educação, como Engenharia, Computação, Literatura, entre outras. Apesar de amplo, o acervo da plataforma unifica os recursos em níveis educacionais. Ou seja, recursos voltados ao ensino de programação encontram-se misturados entre materiais com conteúdo introdutório e avançado.

### 6.3.1 Preparação e coleta de dados

Este caso foi investigado com o auxílio de dois especialistas da área de programação. Seu objetivo foi alinhar o processo de recuperação de REA e a capacidade dos mecanismos em apoiarem essa característica. De forma resumida, os especialistas classificaram todos os recursos relacionados aos tópicos de *"Algoritmos & Estrutura de Dados"* e *"Linguagem de Programação"* do acervo do MIT OpenCourseWare de acordo com três critérios:

1. **SIM**: O conteúdo do REA é voltado ao ensino e aprendizado de programação introdutória;

2. **TALVEZ**: O conteúdo do REA possui alguns trechos, como códigos ou exemplificações, que podem ser usadas no ensino e aprendizado de programação introdutória; e,
3. **NÃO**: O conteúdo do REA não é voltado ao ensino e aprendizado de programação introdutória.

Cabe destacar que tal classificação foi desenvolvida de modo subjetivo e baseado na avaliação de cada especialista. Ou seja, cada especialista classificou os REA conforme a sua percepção e desempenhou a tarefa individualmente, sem supervisão. O objetivo foi testar o desempenho dos mecanismos sob a visão de diferentes tipos de usuários.

A seguir, a classificação dos especialistas foi comparada com o processamento de dados da IPAVL e do TAGGER. Mais especificamente, foram observados o total de termos presentes nos REA, os seus sinônimos e suas *tags* HTML.

Este caso, em particular, analisa a segunda proposição do estudo de caso: $(P_2) \rightarrow$ *os mecanismos podem ser adaptados para diferentes perfis de usuários e de recursos.* Tendo isso em vista, o MIT OpenCourseWare foi selecionado por conter em seu acervo diferentes tipos de recursos, como tarefas, exames, leituras, aulas digitais, entre outros.

Além do mais, considerando a segundo proposição do estudo de caso, foram selecionados especialistas que possuem visões diferentes sobre o que é programação introdutória. O primeiro especialista não considera REA voltados ao ensino de orientação a objetos como introdutório. Já o segundo especialista considera que os fundamentos da orientação a objetos, como *objetos* e *classes*, são temas introdutórios.

A unidade de análise foi justamente as questões de contexto. Ou seja, está alinhado ao perfil do usuário e como o usuário interage com a interface da plataforma. Nesse caso, três desafios foram analisados: o de filtragem, o da elaboração de *string* de busca e o uso de operadores lógicos e relacionais. O desafio de interface foi avaliado no terceiro caso.

Por fim, o processo de vinculação dos dados às proposições foi similar ao relatado no primeiro caso (Seção 6.2).

### 6.3.2 Resultados

Para realizar o processo de avaliação, os estudos de Gardner e Davies (2013) e Coxhead (2000) foram utilizados de modo a estabelecer um conjunto inicial de observações. Em síntese, foram definidas três métricas de avaliação:

1. *Repetição:* Termos que estão presentes na IPAVL e são repetidos nos REA. Por exemplo, se o termo *loop* for utilizado 3 vezes, o valor dessa métrica 3.
2. *Termo únicos:* Representa o total de termos da IPAVL sem considerar as suas repetições. Por exemplo, caso o termo *loop* ocorra 3 vezes no recurso, é contabilizado apenas 1.

3. *TF-IDF:* Representa a importância de um determinado termo em um documento.

Com base nessas métricas, é possível estabelecer uma relação direta entre os rótulos utilizados pelos especialistas e o nível de repetição e de termos únicos utilizados. Ao todo, os especialistas classificaram 96 REA como sendo introdutórios ou não.

O primeiro especialista é aluno de doutorado e possui como uma de suas áreas de *expertise* o ensino de programação. Sua experiência com pesquisa é articulada em diferentes instituições de ensino do Brasil. O especialista atuou em diferentes projetos de pesquisa sobre programação e computadores. Em relação ao perfil profissional, o especialista realizou diferentes estágios de docência e atualmente é vinculado a um centro universitário de ensino particular.

O segundo especialista também é aluno de doutorado e possui interesse de pesquisa no ensino de programação para jovens e adolescentes. Sua experiência com pesquisa é relacionada ao envolvimento de tecnologias para auxiliar o processo de ensino e aprendizado. O especialista atuou como programador em diferentes organizações e também atuou como monitor em disciplinas da graduação. Em relação ao perfil profissional, o especialista declarou trabalhar como professor de programação introdutória em nível de graduação.

Ambos os especialistas possuem graduação e mestrado na área de computação. Ademais, ambos possuem experiência profissional e acadêmica com programação. Mas, divergem da percepção sobre o ensino introdutório.

O segundo especialista considera que o ensino introdutório de programação deve cobrir os tópicos elementares e envolver até o ensino sobre os fundamentos da programação orientada a objetos. Desse modo, o especialista compartilha uma percepção de que o ensino introdutório é similar ao apresentado na Tabela 2.

Enquanto isso, o primeiro especialista considera que o ensino de programação para iniciantes cobre os conteúdos fundamentais (variáveis, tipos de dados, etc...) e vai até a criação de funções/reorganização do código. Ou seja, o especialista acredita que os conceitos sobre a orientação a objetos não pertencem ao ensino de programação introdutória.

Como já observado por Hertz (2010), essa divisão sobre o que é programação introdutória ou não é um assunto comum dentro da literatura. Por isso mesmo, os especialistas foram convidados a participarem da validação, tendo em vista que o perfil distinto de cada um contribuiria para uma avaliação mais robusta dos mecanismos de identificação e de recuperação dos REA.

É importante destacar que o objetivo deste caso, em particular, é analisar a segunda proposição do estudo de caso ($P_2$): *Os mecanismos podem ser adaptados para diferentes perfis de usuários e de recursos*. O que está sendo observado são as questões de contexto, mais especificamente a filtragem, *string* de busca e os operadores lógicos e relacionais.

Inicialmente, foi adotado o coeficiente Kappa de Cohen *(Cohen's kappa coefficient)* para verificar o índice de concordância entre os especialistas. Para isso, foram considerados os

recursos rotulados como válidos (SIM ou TALVEZ) e os recursos considerados inválidos (NÃO). O valor alcançado foi igual a 0.48, indicando apenas concordância moderada entre os pares (LANDIS; KOCH, 1977). Com isso, foi dado início ao processo de avaliação dos mecanismos.

#### *6.3.2.1 Análise dos dados*

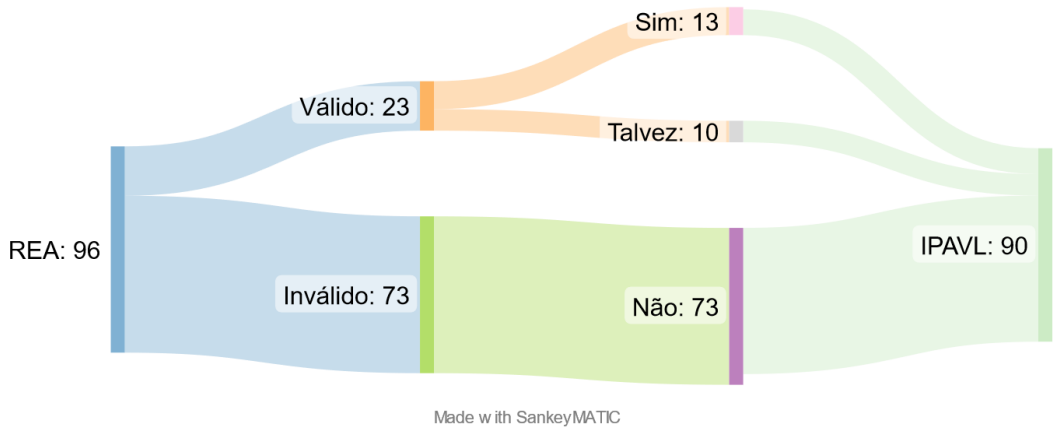
O primeiro especialista classificou 23 REA como voltados ao ensino ou aprendizado de programação, sendo que desse conjunto, 10 foram rotulados com TALVEZ e 13 foram rotulados com SIM. Para o primeiro especialista, 73 não são introdutórios e por isso foram rotulados com NÃO.

Já o segundo especialista também classificou 23 REA como sendo introdutórios (12 SIM e 11 NÃO). O restante dos recursos (73) foram classificados como não sendo introdutório. É importante destacar que, como já foi exposto pelo coeficiente, não são os mesmos recursos que receberam a mesma classificação.

A IPAVL possui termos em 22 REA que haviam sido classificados como sendo válidos pelo primeiro especialista. Em outras palavras, apenas 1 REA foi classificado como válido mesmo sem ter nenhum termo da IPAVL.

Em relação ao segundo especialista, o desempenho da IPAVL foi menor. Dos 23 REA válidos, 4 deles não tinham nenhum termo da IPAVL. A seguir, as Figuras 19 e 20 apresentam uma síntese desses relacionamentos.

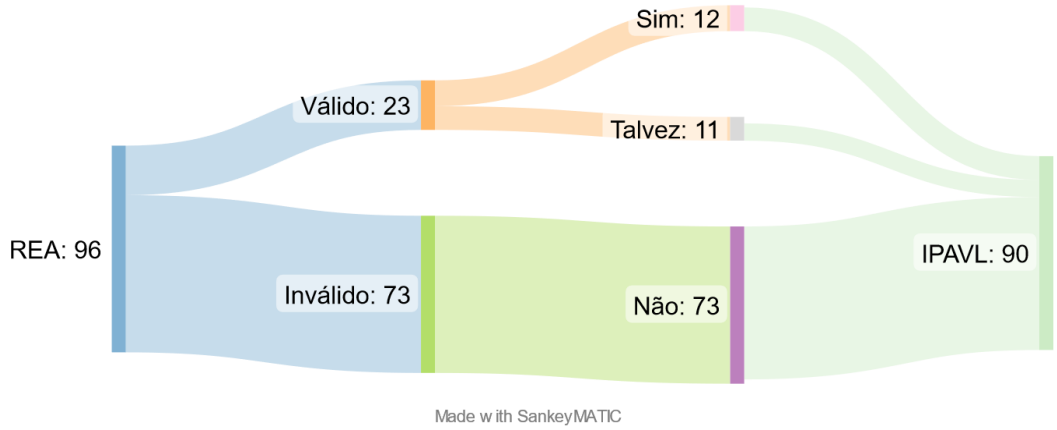
Figura 19 – Comparação entre os rótulos adotados pelo Especialista 01 e a IPAVL




Fonte: Elaborado pelo autor da pesquisa com o SankeyMATIC.

Considerando apenas os REA com termos da IPAVL, as métricas de avaliação evidenciam como a IPAVL foi aderente a seleção dos especialistas. No caso do primeiro especialista, 103 termos foram repetidos nos 12 recursos rotulados com SIM. Já nos recursos rotulados com TALVEZ, o índice de repetição foi 72. Por fim, 452 repetições ocorreram nos REA rotulados com NÃO. Ao se colocar em perspectiva a média, nota-se que os recursos válidos obtiveram um desempenho superior, como exposto a seguir.

Figura 20 – Comparação entre os rótulos adotados pelo Especialista 02 e a IPAVL




Fonte: Elaborado pelo autor da pesquisa com o SankeyMATIC.

Considerando apenas os dados do primeiro especialista, em média, cada REA rotulado com SIM possui cerca de 8,58 termos repetidos da IPAVL, já os recursos rotulados com TALVEZ possuem uma média de 7,20 e os recursos rotulados com NÃO possuem apenas 6,64 termos repetidos. Esse desempenho é similar à classificação realizada pelo segundo especialista. Mas, nesse caso, os REA classificados com TALVEZ obtiveram 10,63 termos repetidos da IPAVL e os REA classificados como SIM obtiveram uma média de 8,83 termos repetidos.

Em relação aos termos únicos, também se nota um desempenho similar. O primeiro e o segundo especialista atingiram em média de 2 termos únicos para os recursos válidos. Em outras palavras, caso um REA fosse classificado como válido por alguns dos especialistas, esse recurso teria, em média, ao menos 2 termos distintos da IPAVL.

Já os recursos inválidos alcançaram uma média de 0,76 e 0,74 para o primeiro e segundo especialista, respectivamente, isto é, os REA inválidos possuem, em média, cerca de apenas 1 termo da IPAVL. Para facilitar a compreensão desses dados, a Tabela 23 apresenta uma síntese das métricas de avaliação que foram adotadas.

Tabela 23 – Resumo das métricas de avaliação

| Especialista | Aderência | Repetição | TU | IPAVL | Repetição/IPAVL | TU/IPAVL |
|---|---|---|---|---|---|---|
| 01 | SIM | 103 | 27 | 12 | 8,58 | 2,25 |
| | TALVEZ | 72 | 26 | 10 | 7,20 | 2,60 |
| | NÃO | 452 | 52 | 68 | 6,64 | 0,76 |
| 02 | SIM | 97 | 29 | 11 | 8,81 | 2,63 |
| | TALVEZ | 85 | 22 | 8 | 10,62 | 2,75 |
| | NÃO | 445 | 53 | 71 | 6,26 | 0,74 |

Fonte: Dados da Pesquisa. TU = Termos Únicos.

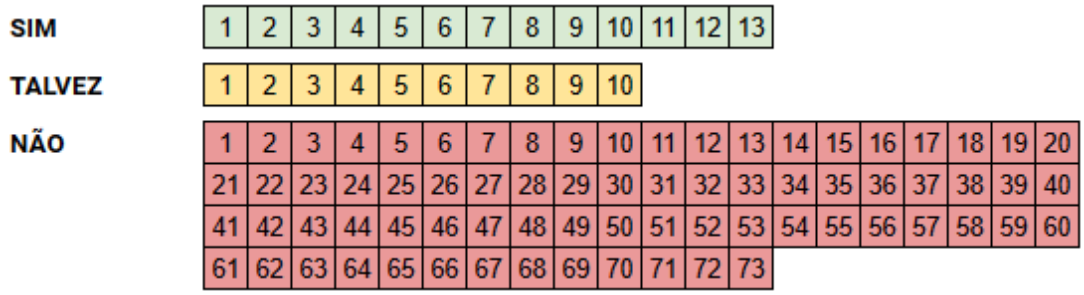
#### 6.3.2.2 *Filtragem dos recursos conforme o perfil do especialista*

Após verificar a presença dos termos da IPAVL, o foco voltou-se em analisar a falta de filtros aderentes às práticas dos usuários, complicações na montagem da *string* de busca e a falta

do uso de operadores lógicos e relacionais. Para testar tudo isso, foram utilizadas apenas duas *tags* do TAGGER: <*title*> e <*p*>, que apresentavam o título e a descrição do REA. Essas *tags* foram selecionadas pois o MIT OpenCourseWare apresenta tais informações para cada REA usando apenas tais elementos em sua página de busca. Além disso, foi adotada apenas a lista de termos da IPAVL.

Para facilitar o entendimento do processo de validação, foi utilizada uma representação gráfica que simula o total de recursos retornados conforme cada especialista. A Figura 21 mostra o padrão usado: verde para REA rotulados como SIM, amarelo para REA marcados com TALVEZ e a vermelho para recursos classificados com NÃO. É necessário destacar que esse é apenas um exemplo para facilitar a compreensão.

Figura 21 – Representação gráfica dos rótulos dos REA




Fonte: Dados da Pesquisa.

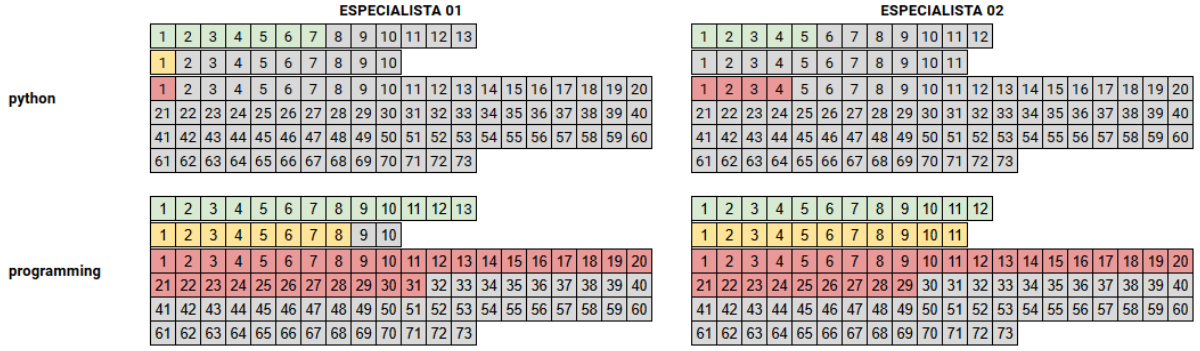
Inicialmente, foram usadas as métricas referentes ao TF-IDF e frequência dos termos da IPAVL nos REA. Dois termos ganharam destaque: *Python*, nome de uma linguagem de programação; e *Programming*, termo genérico para representar programação. Assim, a primeira *string* de busca foi baseada no ocorrência desses dois termos. Em síntese, foi observado que o uso desses dois termos atendia aos perfis dos dois especialistas.

O termo *Python* quando utilizado na *string* de busca retornou 9 REA, sendo que 7 haviam sido classificados com SIM, 1 com TALVEZ e 1 com NÃO pelo primeiro especialista. A mesma *string* para o segundo especialista retornou o mesmo total de REA, mas 5 foram classificados com SIM e 4 com NÃO, como mostra a Figura 22. Em outras palavras, isso quer dizer que se apenas o termo *Python* fosse usado, para ambos os especialistas, já seria suficiente para identificar REA válidos.

O termo *Programming*, por sua vez, teve um desempenho ainda maior. Esse termo sozinho retornou todos os REA classificados com SIM pelos dois especialistas. Trouxe ainda 8 REA assinalados com TALVEZ pelo primeiro especialista e os 11 REA assinalados com esse rótulo pelo segundo especialista. Mas, recuperou, respectivamente, 31 e 29 REA inválidos assinalados pelo primeiro e pelo segundo especialista, como pode ser visto na Figura 22. Isso quer dizer que se essa *string* de busca fosse usada, dos 96 REA que foram classificados, seriam removidos ao menos 41 recursos inválidos para ambos os especialistas. Isso poderia economizar quase metade do tempo usado na classificação.

Um dos problemas apontados pelos participantes dessa pesquisa foi a falta de aderência ao uso de operadores lógicos e relacionais (ver Seção 4.2). Por isso, os termos da IPAVL também foram mesclados com os operadores AND, OR e NOT IN. De forma resumida, o operador OR é usado para expandir o escopo da busca, enquanto os operadores AND e NOT IN são usados para reduzi-la.

Figura 22 – Representação gráfica dos rótulos dos REA com *string* básica. Em cinza, REA não retornados.




Fonte: Dados da Pesquisa.

Como o termo *programming* trouxe quase todos os REA válidos, foi testado se era possível reduzir o total de recursos inválidos. Na primeira *string* foram usados dois novos termos que eram frequentes em todos os REA: *language* e *computer*. Como mostra a Figura 23, adicionar apenas o operador OR aumentou o total de resultados inválidos tendo em vista que a maioria dos recursos válidos ja eram identificados pelo termo *programming*.

No entanto, ao elaborar *strings* mais complexas, os resultados também mudaram. Ao verificar que apenas dois dos três termos ocorriam nos REA, foi possível manter grande parte dos recursos válidos e reduzir os REA inválidos. No caso do primeiro especialista, foram retornados 5 recursos rotulados com TALVEZ e apenas 14 rotulados com NÃO. Já para o segundo especialista, foram retornados 10 REA anotados com SIM, 6 rotulados com TALVEZ e 15 classificados com NÃO.

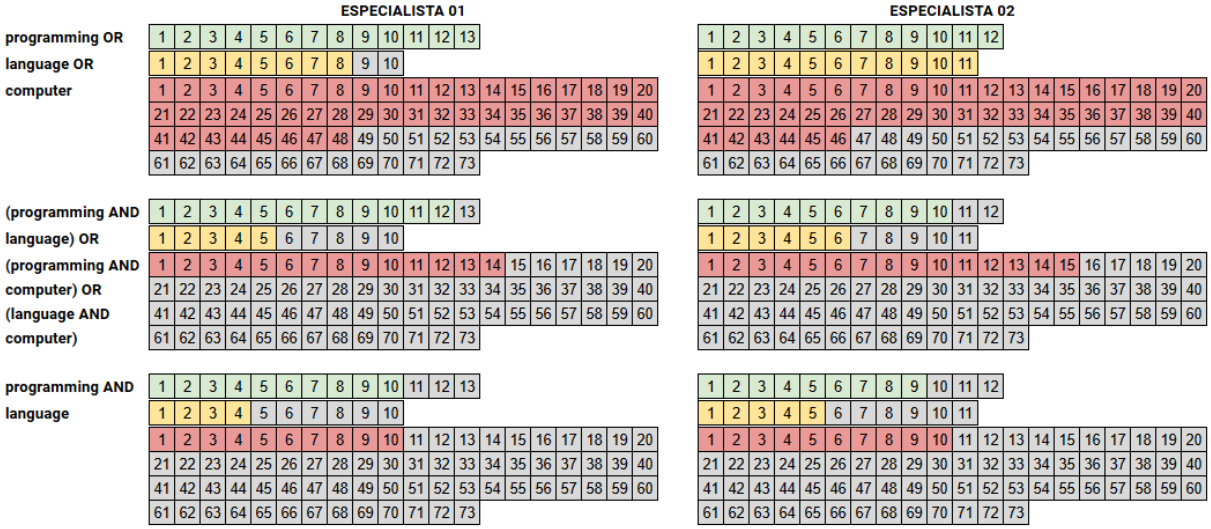
Ao adicionar os operadores AND e OR foi possível remover cerca de 15 REA inválidos. Mas, como consequência, houve uma perda de recursos assinalados com TALVEZ ou SIM. Ao restringir a busca com a ocorrência obrigatória de dois termos *programming* e *language*, os resultados foram ainda mais direcionados: apenas 10 REA inválidos foram retornados (Figura 23).

Diante disso, considerando a *string* de busca *programming* AND *language*, seriam removidos 63 REA inválidos para ambos os especialistas. A consequência é que alguns recursos válidos, sobretudo aqueles assinalados com TALVEZ também seriam removidos.

Logo a seguir, foi analisado o comportamento dos termos considerando o operador NOT IN. De forma resumida, esse operador é utilizado quando o usuário deseja remover recursos que possuem um determinado termo. Por exemplo, pode ser que uma busca por variável retorne

resultados de computação e da matemática. Com isso em mente, poderia ser adotado o operador NOT IN para remover os resultados irrelevantes, algo como *variável AND NOT-IN(matemática)*.

Figura 23 – Representação gráfica dos rótulos dos REA com operadores lógicos e relacionais. Em cinza, REA não retornados.




Fonte: Dados da Pesquisa.

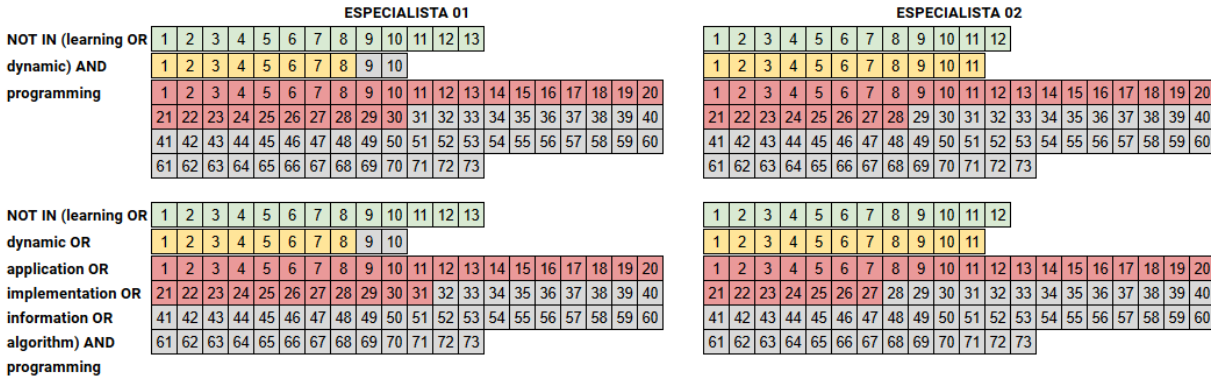
Para o teste, foram considerados os termos mais comuns nos REA rotulados como NÃO por ambos os especialistas. Os termos que se destacaram foram *learning* e *dynamic*. Logo em seguida vieram os termos *application, implementation, information* e *algorithm*.

A primeira *string* de busca foi uma junção entre os dois termos principais (*learning* e *dynamic*) e o termo mais comum nos recursos válidos (*programming*). Para isso, foi gerada uma *string* que, em tese, deveria trazer os recursos válidos e remover os recursos inválidos.

Como pode ser visto na Figura 24, esse objetivo foi alcançado para ambos os especialistas. No caso do segundo especialista, todos os recursos rotulados como SIM ou TALVEZ foram recuperados e no caso do primeiro especialista, apenas dois recursos rotulados como TALVEZ não foram retornados. Por outro lado, 30 recursos inválidos (especialista 01) e 28 recursos inválidos (especialista 02) foram retornados. Ou seja, se o primeiro especialista usasse essa *string* não precisaria classificar 43 REA inválidos. Já o segundo especialista evitaria a análise de 45 recursos inválidos. Para ambos os casos, seria poupado quase metade do total de classificação.

Para testar a variação da *string*, os termos secundários (*application, implementation, information* e *algorithm*) também foram adicionados. O propósito do teste foi verificar se a elaboração de uma *string* mais complexa poderia retornar melhores resultados. Mas, como pode ser visto na Figura 24, o resultado foi basicamente o mesmo da *string* anterior. Isso significa que os termos secundários ocorreram nos REA inválidos, mas não nos recursos válidos.

Figura 24 – Representação gráfica dos rótulos dos REA com o uso do operador NOT IN. Em cinza, REA não retornados.




Fonte: Dados da Pesquisa.

### 6.3.3 Discussões

Este caso analisou a eficiência da IPAVL e do TAGGER para recuperar REA de programação introdutória. O recorte feito refere-se aos três desafios que envolvem as questões de contexto: filtragem, *string* de busca e uso dos operadores lógicos e relacionais. Com isso em mente, três elementos devem ser destacados.

Em primeiro lugar, grande parte das abordagens atuais precisam lidar com um desafio comum da seleção de REA: retornar resultados que estão desalinhados com a busca feita pelo usuário. Os testes realizados neste caso demonstram que a IPAVL e o TAGGER foram capazes de ser adaptados por usuários que possuem um perfil distinto sobre o real significado de programação introdutória. Os mecanismos podem ser utilizados com as abordagens já existentes. Como exposto, criações simples de *string* de busca removeram diversos recursos inválidos. Se isso fosse aplicado em uma grande coleção, diversos recursos provavelmente seriam removidos com pouco esforço, evitando, por exemplo, que um especialista ou algoritmo revisasse toda a coleção. Além do mais, o uso dos mecanismos poderia apoiar o processo de identificação e de recuperação de recursos válidos ou inválidos para testes e treinamentos.

Além disso, é essencial dizer como os mecanismos são facilmente adaptados. Com o uso de três métricas distintas (repetição, termos únicos e TF-IDF), foi possível mapear rapidamente a presença dos termos e a identificação das palavras com maior e menor relevância.

Como exposto, o uso de uma *string* de busca simples poderia reduzir consideravelmente a coleção de recursos a ser revisada. Dois motivos destacam essa relevância: (i) professores de programação, em geral, são sobrecarregados de tarefas didáticas, como a correção de exercícios que envolvem lógica de programação. Desse modo, os mecanismos podem automatizar etapas da recuperação de recursos, como a remoção de resultados inválidos ou o maior destaque aos recursos válidos; (ii) muitos usuários sentem uma sensação de desânimo ou de frustração com o processo de busca dos REA devido ao número reduzido ou muito amplo de resultados retornados em plataformas abertas. Os mecanismos aqui expostos podem colaborar com a redução desse e

de outros problemas.

#### 6.3.3.1 Conectando os achados com o estudo de caso

A proposição que rege toda esta avaliação é a adaptação dos mecanismos para usuários e materiais distintos. Diante disso, foi possível verificar que os mecanismos podem colaborar com os usuários, demonstrando a presença de termos em recursos válidos e a ausência de termos nos materiais inválidos. Ademais, os mecanismos podem ser usados para compor estratégias de buscas mais sofisticadas, como *strings* que mesclam operadores lógicos e relacionais. Isso pode ser aplicado em qualquer recurso que possua algum tipo de conteúdo. Se o material for estruturado em HTML, o TAGGER pode refinar ainda mais o processo.

As possibilidades para as plataformas abertas que armazenam REA são diversas. A primeira delas é o desenvolvimento de um *plugin* que pode ser instalado nos navegadores. Tal *plugin* poderia auxiliar a construção de *string* de busca ou até mesmo sugerir termos para identificar ou remover recursos. Outra possibilidade é mesclar os modelos (Caso 1) para recuperar os REA conforme o perfil de cada usuário. Ou seja, se um usuário deseja encontrar REA para o ensino de Java, tal termo pode receber um peso maior na criação dos modelos.

Todo o processo de filtragem pode ser repensando com os mecanismos, como a criação de novos filtros, destaque dos termos, ou adição dos recursos gráficos, como negrito ou itálico, para ressaltar a presença de termos específicos. Isso tudo, em última instância, poderia facilitar a escolha dos recursos.

Já todo o processo que envolve a construção de uma *string* de busca pode ser otimizado. Ainda hoje, ao acessar uma plataforma aberta, quase sempre o usuário se depara com uma interface que possui um campo de busca, um botão de pesquisa e diversos filtros sobre licença, nível educacional, e outros tipos de campos de preenchimento. Tudo isso pode ser revisitado, adicionando elementos e recursos para facilitar a elaboração das *strings*, como sugestão de termos ou uso dos operadores lógicos e relacionais.

#### 6.3.3.2 Ameaças à validade

Uma das principais limitações deste caso é o desalinhamento entre o que foi feito pelos especialistas e os métodos e procedimentos usados para reproduzir tais ações. Para evitar a propagação desse risco, uma série de iniciativas foi tomada. A primeira delas foi a escolha de especialistas com perfis diferentes para tornar a análise mais robusta. A segunda iniciativa foi a verificação dessa diferença por meio de uma medida estatística, segundo as premissas de Landis e Koch (1977). Ademais, outra iniciativa realizada foi o uso de métricas gerais sobre a frequência e a relevância dos termos.

## 6.4 Caso 3: OER-Chat

O terceiro caso selecionado é o OER-Chat, uma ferramenta desenvolvida no âmbito desta pesquisa de doutorado. O OER-Chat é um *chatbot* aberto que fornece REA de programação introdutória de acordo com a solicitação do usuário.

O terceiro caso foi selecionado considerando que ambos os mecanismos produzidos, IPAVL e o TAGGER, foram projetados para serem instanciados em plataformas abertas que armazenam REA. Considerando o interesse dos usuários com o uso do *chatbots*[1], emergiu uma oportunidade de conectar educação aberta, REA e ensino de programação. Isso poderia entusiasmar novos usuários ao mesmo tempo que reduzia o desenvolvimento de plataformas abertas que relembram repositórios digitais.

Com isso em mente surgiu o OER-Chat, um *chatbot* aberto que sugere REA de programação ao invés de respostas textuais. Resumindamente, o OER-Chat disponibiliza diferentes tipos de REA, como slides, cursos, livros digitais, entre outros materiais abertos, para sanar dúvidas dos usuários sobre programação. Dessa forma, o OER-Chat busca promover a reutilização de materiais livres e fomentar a educação aberta. Cabe destacar que as bases do OER-Chat foram concebidas utilizando a IPAVL e o TAGGER, como exposto a seguir.

### 6.4.1 Design

O OER-Chat foi desenvolvido, essencialmente, usando os mecanismos anteriormente descritos para identificar e recuperar REA de programação introdutória. Para isso, o *chatbot* possui seis módulos:

1. **Módulo de pré-processamento:** Inspirado pela IPAVL, o módulo é responsável por utilizar o Processamento de Linguagem Natural para processar os termos presentes nos REA, removendo termos desnecessários.

2. **Módulo de vocabulário:** Inspirado pela IPAVL, o módulo apresenta uma lista de termos presentes no respectivo mecanismo. Ele funciona como um arcabouço para: (1) identificar o total de termos do REA; (2) identificar quais termos estão presentes no comando do usuário; (3) verificar potenciais REA; e (4) identificar sinônimos.

3. **Módulo de programação introdutória:** o módulo também foi inspirado pela IPAVL e é responsável por adicionar uma nova funcionalidade no *chatbot*: prever qual é a área de concentração do REA e do comando do usuário. Para isso, os termos da IPAVL foram traduzidos para o português e cada termo foi classificado em quatro áreas distintas:

[1] O uso de *chatbots* se tornou uma "febre" no final de 2022 impulsionado pelo ChatGPT.

- *Fundamentos:* Conceitos básicos de programação, como variável, bug, compilador, etc). A ideia principal é introduzir aspectos e conceitos essenciais de programação, e não codificar.
- *Técnicas:* Temas avançados de programação introdutória. Geralmente, utilizam dois ou mais termos da área de Fundamentos para criar um novo conceito, como matriz, uma array ou vetor.
- *Linguagens:* Termos relacionados a alguma linguagem de programação, como Java, C ou Python.
- *Paradigmas:* Palavras relacionadas ao ensino de Programação Orientada a Objetos.

4. **Módulo de tags:** o módulo foi inspirado pelo TAGGER. Ele é baseado no uso das 14 *tags* HTML comumente encontradas em REA.
5. **Módulo de pesos:** o módulo foi inspirado pelo TAGGER também, e atua adicionando pesos de acordo com as *tags* HTML.
6. **Módulo de probabilidade:** o módulo unifica a IPAVL e o TAGGER, considerando o total de termos presentes, a área prevalente dos termos, as *tags* e seus respectivos pesos.

Os módulos atuam em conjunto. Inicialmente, os REA são processados por cada módulo do OER-Chat. Como resultado é gerada uma matriz que lista os seus termos, *tags* de origem, frequência, entre outros dados. Logo em seguida, é analisado o comando do usuário, o qual passa pelo mesmo processamento. Ao final, o OER-Chat oferece uma lista de REA que deram *match*, isto é, os recursos mais parecidos com os comandos enviados pelo usuário.

#### 6.4.1.1 Participantes

Nesta validação, foi adotado um paradigma qualitativo de pesquisa. A intenção foi observar detalhadamente como usuários de diferentes perfis iriam interagir com os mecanismos. Em maior medida, analisar se os mecanismos iriam produzir um efeito positivo no processo de identificação e de recuperação de REA de programação introdutória.

Para isso, cinco participantes foram convidados a realizarem a avaliação do OER-Chat. Todos foram selecionados por serem professores de programação e/ou terem tido experiência como Professor Assistente. Os participantes estão envolvidos em atividades pedagógicas e didáticas, tais como planejamento de aulas, construção de materiais educacionais, suporte a alunos, entre outros. A experiência de cada participante é sumarizada no Quadro 11.

#### 6.4.1.2 Preparação e Coleta de Dados

Inicialmente, foi gerado um questionário digital para: (1) apresentar o processo de validação; (2) apresentar o termo de consentimento; e (3) coletar dados de cada participante.

Quadro 11 – Perfis dos participantes

| Participante | Experiência |
|---|---|
| **P1** | Usuário interessado no desenvolvimento de aplicações educacionais de programação. O participante declarou que utiliza REA de forma inconstante |
| **P2** | Professor de programação introdutória. Possui mais de 10 anos com o ensino e aprendizado de programação e utiliza REA de forma inconstante nos últimos 3 anos |
| **P3** | Programador interessado em REA e educação aberta. Possui experiência como professor de programação introdutória e atualmente trabalha no setor industrial de programação. O participante declarou ter mais de cinco anos de experiência com programação e com REA. Também declarou que utiliza REA constantemente |
| **P4** | Analista de Sistemas com mais de 10 anos de experiência em programação introdutória e mais de cinco anos com REA. O participante declarou que conhece os REA, mas os utiliza de forma inconstante |
| **P5** | Professor de programação introdutória e usuário interessado no desenvolvimento de aplicações educacionais de programação. Possui mais de 10 anos de experiência com programação e utiliza REA de forma inconstante. |

Fonte: Dados da Pesquisa.

Consequentemente, essa etapa incluiu perguntas sobre o perfil dos participantes, sua experiência com programação introdutória e REA. Esses dados foram coletados para avaliar o perfil de cada participante, resumindo suas principais características e percepções.

A seguir, os participantes acessaram e utilizaram o OER-Chat voluntariamente. O objetivo foi simular a busca por REA para fins educacionais e pessoais. Portanto, o tópico de pesquisa não se limitou em uma linguagem ou tecnologia de programação específica. Seguindo as suposições de Selltiz *et al.* (1991), os participantes usaram a ferramenta livremente para simular um cenário de pesquisa do mundo real. Além disso, também não foi imposto nenhum limite de tempo. Cada usuário utilizou o *chatbot* com base em seu interesse. Após alcançar uma opinião concreta sobre o OER-Chat, o participante enviava seus comentários.

Os dados foram coletados por meio de perguntas abertas, que exploraram a percepção do usuário sobre o OER-Chat. Para isso, foram adotadas premissas de Selltiz *et al.* (1991). Em resumo, as perguntas foram elaboradas para evitar a introdução de vieses pessoais nos participantes. Por exemplo, ao invés de perguntar *"Quais desafios você enfrentou ao usar o chatbot?"*, foi perguntado *"Por favor, descreva sua experiência usando o chatbot"*. No primeiro caso, os participantes podem sentir-se obrigados a descrever algum desafio, mesmo que nenhum problema tivesse ocorrido. No segundo caso, se os participantes encontrassem algum desafio, provavelmente seria observado em seus comentários por meio de críticas ou comentários.

Cabe destacar ainda que caso alguma questão apresentasse viés, uma pergunta contrária era feita imediatamente em seguida. Por exemplo, se fosse questionado *"Quais sentimentos NEGATIVOS descrevem sua experiência ao usar o Chatbot?"*, logo em seguida era perguntando *"Quais sentimentos POSITIVOS descrevem sua experiência ao usar o Chatbot?"*. Ao fazer isso, a intenção era neutralizar a introdução de vieses pessoais.

O OER-Chat foi avaliado usando o foco nos efeitos que os usuários sentem após utilizar plataformas abertas. Por isso, sua avaliação englobou a análise da interface, a experiência do usuário e a frustração que os mesmos poderiam sentir após usar a ferramenta.

Como exposto nos casos anteriores, o processo de vinculação dos dados às proposições foi similar aos casos anteriores. Para a análise dos dados, foi utilizado o software QDA Miner[2]. O software permite a classificação dos comentários em diferentes categorias, facilitando a visualização dos dados. Para isso, os elogios recebidos pelo OER-Chat foram anotados em tom verde, já as críticas em tonalidade vermelha e os pontos de melhoria em amarelo.

### 6.4.2 Resultados

O terceiro caso é baseado no OER-Chat, um *chatbot* aberto para facilitar o reuso de REA de programação introdutória. Esse caso empregou a análise de sentimentos e por isso é complementar aos casos anteriores. Para analisar os dados, foi adotado o framework proposto por Pandit (1996), usando as atividades de codificação (*open coding* e *axial coding*).

Ambas as atividades foram desenvolvidas segundo a estratégia relatada na Seção 4.1, envolvendo a divisão dos dados em partes menores e a conexão dos conceitos iniciais em categorias. Por exemplo, notou-se que alguns *feedbacks* e comentários recebidos tratavam da usabilidade do *chatbot*. Então, foi gerada uma categoria denominada *Usabilidade*. Em seguida, essa categoria recebeu uma lista de itens relacionados, como design e simplicidade.

Após concluir todo o processo analítico, a compreensão dos resultados foi desenvolvida. Com esse processo, foi possível estabelecer a percepção final da eficiência do OER-Chat para apoiar o uso de REA para programação introdutória, bem como pontos para melhorias futuras.

#### 6.4.2.1 Análise de Sentimentos

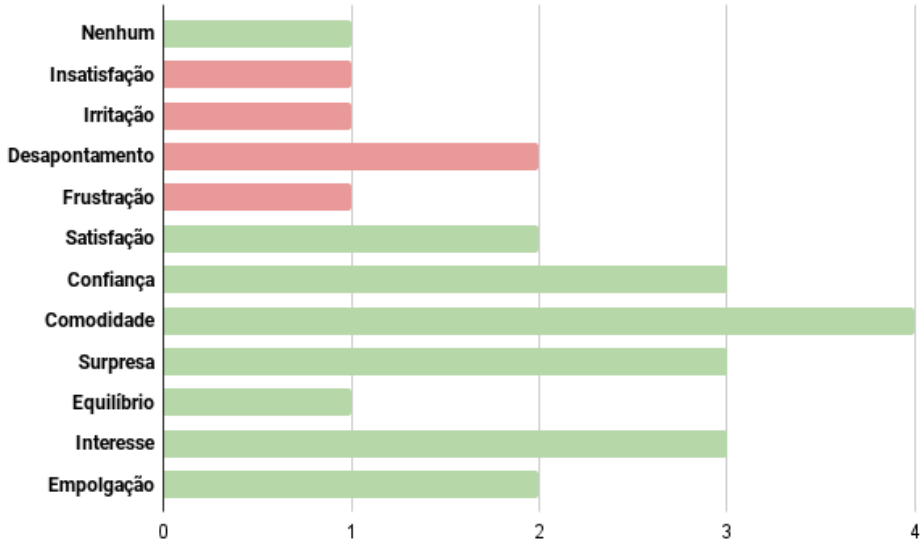
A primeira etapa da análise foi a verificação as sensações dos usuários durante a utilização do OER-Chat. Essa análise foi adotada para evidenciar se emoções negativas, como raiva, frustração ou irritação, emergiram durante o uso. Ou ainda, se os usuários experimentaram emoções positivas, como empolgação e confiança.

Para isso, as orientações de Selltiz *et al.* (1991) foram adotadas. Em suma, duas listas foram criadas: uma com sentimentos positivos e outras com sentimentos negativos. As listas eram complementares entre si, apresentando itens como “Satisfação x Insatisfação”. A Figura 25 ilustra os resultados alcançados.

No total, quatro sentimentos negativos emergiram (Insatisfação, Irritação, Desapontamento (2x) e Frustração). Porém, oito emoções positivas surgiram (Satisfação (2x), Confiança (3x), Comodidade (4x), Surpresa (3x), Equilíbrio (1x), Interesse (3x), Empolgação (2x) e Nenhum sentimento negativo). Com exceção do desapontamento, nenhum sentimento negativo se

[2] https://provalisresearch.com/products/qualitative-data-analysis-software/

Figura 25 – Sentimentos positivos (verde) e sentimentos negativos (vermelho)

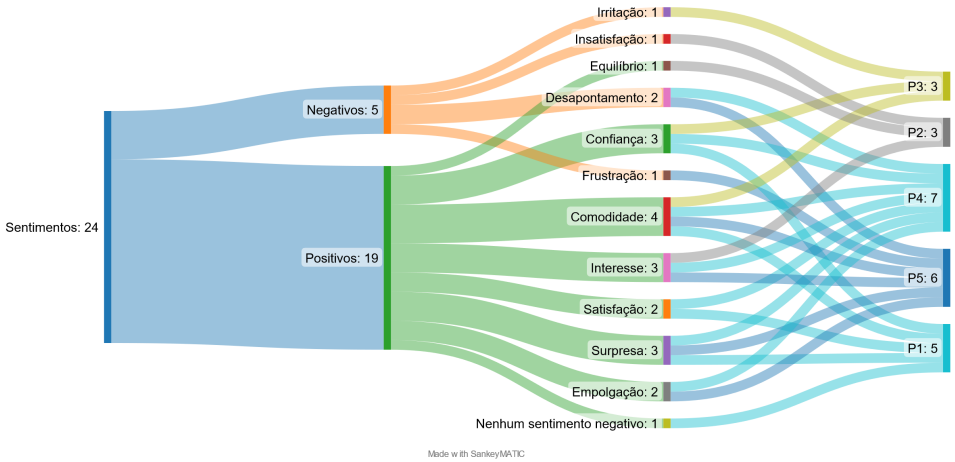



Fonte: Dados da Pesquisa.

repetiu. Enquanto isso, os sentimentos positivos foram repetidos em diversas ocasiões vezes, chegando a 3 ou 4 participantes.

Além disso, também foi analisado como essas emoções surgiram em cada participante. Na Figura 26 é apresentada a lista completa. Conforme observado, com exceção do participante P1, que apresentou apenas sentimentos positivos, os demais participantes elencaram 1 ou 2 sentimentos positivos para cada sentimento negativo. Diante disso, os resultados demonstram que os participantes, em sua maioria, experimentaram mais sensações positivas do que negativas.

Figura 26 – Sentimentos de cada participante




Fonte: Elaborado pelo autor da pesquisa com o SankeyMATIC.

### *Experiências do Usuário*

Como apresentado no Capítulo 4, a experiência do usuário é um conjunto de sensações dos usuários ao interagirem com uma plataforma aberta. No geral, essa experiência vem sendo descartada por muitas pesquisas sobre plataformas abertas (MOURIÑO-GARCÍA *et al.*, 2018; KASTRATI; IMRAN; KURTI, 2019; DESSì *et al.*, 2019).

O OER-Chat foi projetado pensando nisso, sendo um software desenvolvido para ser simples de ser utilizado, tendo poucos campos e um design minimalista. Para mapear a experiência dos usuários durante o uso do OER-Chat, os comentários recebidos foram classificados conforme a presença de adjetivos e/ou substantivos que demonstravam alguma sensação do usuário. O resultado alcançado é disposto na Figura 27, que apresenta o conjunto de elogios (em verde) e o conjunto de críticas (em vermelho). Os termos definidores da categoria estão sublinhados.

Figura 27 – A experiência de cada usuário

| **De um modo geral, como você descreve a sua experiência após utilizar o Chatbot?** |
|---|
| Muito boa, retornou os resultados em uma velocidade impressionante e de forma muito precisa e relevante para o que eu busquei |
| Em um primeiro momento, houve o interesse e a expectativa de interagir de uma forma mais natural por meio do Chatbot. Mas a repetição dos resultados, apesar da mudança nos comandos de interação, reduziram os sentimentos iniciais. |
| Achei prático. O formato de chat é intuitivo e rápido. Me irritei um pouco com o fato de ao clicar nos links eu sou redirecionado ao invés de abrir uma nova aba. Faz com que a inspeção de vários recursos simultaneamente seja laboriosa de maneira desnecessária. |
| De um modo geral, após utilizar o Chatbot, tive uma experiência positiva. A interface é simples e intuitiva, o que tornou o uso muito mais agradável e eficiente. Além disso, a mensagem inicial foi muito útil para entender o propósito da ferramenta. Isso parece ser óbvio, mas pela minha experiência em ensino sei que "o óbvio não existe" :)) |
| Os resultados aparecem muito rápido, mas há resultados repetidos e muito parecidos com perguntas que são semelhantes mas que apresentam grandes diferenças. Eu perguntei "Como manter os alunos motivados ao ensinar programação introdutória?" e também "Qual linguagem de programação ensinar primeiro?" e os resultados foram duas listas muito parecidas, com poucas menções a motivação diretamente. |

Fonte: Dados da Pesquisa.

Como pode ser observado, a maioria dos comentários com elogios referem-se aos aspectos relacionados à própria jornada do usuário, mostrando satisfação durante a utilização do OER-Chat. Por outro lado, sentimentos negativos também emergiram, em sua maioria, oriundos de desafios e problemas de natureza mais técnica.

### *Usabilidade*

A usabilidade foi o segundo elemento analisado. A interface de uma plataforma aberta é central e pode apresentar pontos de dificuldades para os usuários se mal projetada. No geral, as plataformas abertas possuem interfaces complexas, difíceis de serem usadas. O OER-Chat, no entanto, foi projetado para reduzir a complexidade de sua interface, contendo apenas elementos essenciais. O resultado final desse esforço é apresentado na Figura 28.

De forma geral, grande parte dos participantes elogiaram o design e a aparência do OER-Chat. Esses elogios estão destacados em verde. Também foram notados diferentes sugestões

Figura 28 – A usabilidade do OER-Chat

| Como você descreveria o design e a aparência do Chatbot? |
| --- |
| Elegante, simples e de fácil utilização |
| Simples, "leve", agradável e fácil de usar. |
| Achei simples e prático. |
| O design do Chatbot é limpo e direto ao ponto. Ele é intuitivo e me fez sentir mais confortável na hora de usar. Embora não seja totalmente responsivo, o uso em dispositivos móveis ainda é viável, o que é um plus. |
| Muito bom. Simples. Objetivo. Seria legal ter um histórico de conversas. Também seria legal ter um guia de como a entrada de busca é esperada. O que eu quero dizer é, qual é o melhor "prompt" para esse chat bot? Essa questão é pertinente em todos chatbots e buscadores, temos que entender como o algoritmo espera a entrada para podermos chegar aos resultados desejados. |

Fonte: Dados da Pesquisa.

para melhorar a interface do *chatbot* (destacados em amarelo), como a inclusão de um *layout* responsivo, a manutenção do histórico de conversa e um guia sobre entradas esperadas. Em preto ficaram os comentários considerados neutros. Ou seja, que aparentam opiniões que não estão relacionadas ao OER-Chat, diretamente.

### *Aderência dos Resultados*

A aderência dos resultados representa um conjunto de elementos desafiadores da identificação e da recuperação por REA. O OER-Chat realiza o processo usando a IPAVL e o TAGGER para remover recursos inválidos e manter apenas os recursos que são considerados aderentes ao ensino de programação e ordená-los de acordo com o interesse do usuário. Com base nos problemas mais comuns, o OER-Chat foi programado com a seguinte lógica:

1. Para facilitar a elaboração das *strings* de buscas, o OER-Chat adota um índice de classificação baseado na ocorrência de termos e das *tags* HTML. O objetivo desse índice é demonstrar quais recursos são mais aderentes à busca do usuário.

2. Para identificar o real propósito do REA (tema principal, público-alvo e assuntos cobertos por aquele material), o OER-Chat realiza esse processo usando a ocorrência e a frequência dos termos da IPAVL nas *tags* do TAGGER.

3. Para evitar a exibição de recursos indisponíveis, o OER-Chat testa todos os links disponíveis no material.

4. Para evitar os riscos do uso apenas dos metadados, o motor de busca adota os termos disponíveis nas *tags* HTML, conforme as regras do TAGGER.

Para analisar esses desafios, foram mapeadas todas as respostas dos usuários sobre questões que dizem respeito à aderência dos resultados que o OER-Chat proporcionou. Por isso, foi observado nos relatos dos participantes se algum desses desafios era citado direta ou indiretamente. Após essa citação, o comentário era classificado como sendo um elogio, crítica

ou uma sugestão de melhoria. O resultado é observado na Figura 29 e utiliza as cores verde para elogios, vermelho para críticas e amarelo para sugestões.

Figura 29 – A aderência dos resultados do OER-Chat

| **Como você descreveria a aderência dos resultados apresentados pelo Chatbot?** |
|---|
| Os resultados foram muito precisos sobre que eu busquei |
| Bem menos aderência que nas buscas realizadas no EduCapes. Apenas dois resultados pareciam mais aderentes, mas ao acessar os REA, percebi que o conteúdo de apenas um deles atendia parcialmente. |
| Em comparação com os resultados do Educapes, eu achei que esses foram mais relevantes, embora ainda tenha aparecido REAs desnecessários. |
| Os resultados apresentados pelo Chatbot se mostraram bastante relevantes. Além disso, deixar claro o critério de ordenação das respostas (REAs) pelo índice de semelhança, isso me ajudou muito a entender os resultados das minhas buscas. Por outro lado, senti um comportamento similar ao eduCAPES quanto à interpretação das buscas usando linguagem natural, o que trouxe resultados fora de contexto com alguma frequencia, mas acredito que isso deve melhorar com o aumento dos REAs abstraídos pelo Chatbot. |
| É aderente, mas em relação a motivação, achei que os resultados estavam menos aderentes. |

| **Você encontrou erros ou resultados irrelevantes ao usar o Chatbot?** |
|---|
| Não |
| Erros não, mas os resultados irrelevantes prevaleceram. |
| Sim, alguns retornos apontavam índices de semelhança maiores do que o esperado, na minha opinião. |
| Em relação a erros ou resultados irrelevantes, não encontrei nada muito significativo. A principal área de melhoria parece estar na eficácia da busca por linguagem natural, mas acredito que isso irá melhorar conforme mais REAs forem indexados. |
| Acredito que não. Alguns materiais têm nomes esquisitos, tipo "Viagem de ônibus", mas é para ensinar programação. Acho que aí o problema é com o material e o mérito é do chatbot que incluiu ele mesmo tendo um título genérico. |

Fonte: Dados da Pesquisa.

A *string* de busca emergiu naturalmente no comentário de um participante, que disse que a ordenação ficou mais efetiva por demonstrar o índice de classificação. Mas, o mesmo participante sugeriu uma melhoria futura para tornar esse índice mais eficiente.

A indisponibilidade não foi identificada nos comentários analisados, sugerindo que o *chatbot* foi capaz de superar esse desafio utilizando uma estratégia simples: verificar se o link encontra-se disponível.

Outro aspecto relevante que emergiu foi o tamanho da coleção. Não foi notado nenhum elogio ou crítica que influenciasse diretamente esse problema. Apenas um participante destacou que o acervo pode ser aumentado, trazendo resultados mais interessantes. Tal comentário não foi classificado como uma crítica, pois o próprio participante classificou isso como sendo uma área de melhoria futura.

Dois desafios, no entanto, foram alvo de ambiguidade: propósito dos REA e uso de metadados. Essa ambiguidade será discutida em detalhes na Seção 6.4.3. Mas, antes disso, deve-se analisar a contradição desses desafios.

Sobre o propósito do REA, alguns participantes relataram que o *chatbot* conseguiu trazer resultados aderentes ao propósito. Por exemplo, um participante destacou a identificação e recuperação do REA *Viagem de ônibus*, que mesmo tendo um título incomum, serve para ensinar

programação. Assim sendo, o *chatbot* conseguiu buscar e oferecer esse resultado ao participante.

Por outro lado, críticas também emergiram. Foi destacado que o OER-Chat não conseguiu exibir resultados aderentes aos propósitos das buscas, citando, por exemplo, a exibição de apenas dois resultados no qual apenas um atendia parcialmente a busca realizada. O mesmo se pode dizer sobre o comportamento de metadados. Em suma, o que se nota é, novamente, uma espécie de ambiguidade na visão de diferentes participantes.

#### *Funcionalidades*

Por fim, os participantes foram convidados a enviarem sugestões de melhorias ao OER-Chat. O objetivo era mapear novas funcionalidades e/ou pontos de atenção ao projeto. Como já foi apresentado nos comentários anteriores, muitos participantes introduziram sugestões durante suas próprias respostas. Por isso, após analisá-las, duas grandes áreas de melhoria foram identificadas:

1. Melhorar a abertura de links: abrir o link do recurso em uma nova página (evitando apagar o histórico de conversa) e adicionar uma opção de *ver mais* ao visualizar os resultados; e,

2. Otimizar o cálculo do índice de classificação: fazer correções e demonstrações de como o índice é calculado.

### 6.4.3 Discussões

O terceiro caso analisou a eficiência da IPAVL e do TAGGER para identificar e recuperar REA voltados ao ensino introdutório de programação. Para isso foi construído um *chatbot* aberto, que apresenta REA ao invés de respostas textuais. O caso, em especial, lida com a terceira proposição do estudo de caso: $(P_3) \rightarrow$ *os mecanismos podem facilitar o uso de REA* e adota como unidade de análise os efeitos mapeados no Capítulo 4. Diante dos resultados alcançados, quatro pontos emergiram na análise:

- **Preferências pessoais:** As preferências pessoais podem ter interferido no processo de validação. Por exemplo, um participante afirmou: *"Acho que a única coisa que me irritou no chat foi o comportamento ao clicar nos links. Ele os mantém na mesma guia em vez de abrir outra. Se retornar 5 resultados que eu gosto, tenho que clicar, visualizar, voltar, clicar, visualizar, voltar em vez de abrir todas as 5 guias de uma vez"*. Esse comentário é de um participante que trabalha na indústria de software (P3). Ele tem experiência com programação e por isso a usabilidade do sistema tem um peso maior, levando a um sentimento de "Irritação". Porém, essa irritação não se deve a falhas do OER-Chat, como a ausência de resultados relevantes, mas sim a um aspecto técnico que poderá ser abordado futuramente com uma nova versão.

- **Problemas semânticos:** Os participantes usaram mais palavras para descrever sentimentos negativos do que sentimentos positivos. Porém, após as entrevistas, percebeu-se que os participantes elaboraram mais sobre os problemas identificados para os mesmos serem corrigidos em uma versão futura. Essa percepção surgiu no relato do participante P2: *"Eu tive alguns sentimentos de frustração ao usar o chatbot. Detalhei isso em meu feedback para apontar essas falhas e sugerir melhorias futuras. Bom, pessoalmente tive imenso interesse em utilizar o chatbot para [o ensino] e para as minhas aulas. Por isso mesmo, as minhas críticas visam torná-lo melhor em versões futuras"*.

- **O OER-Chat é uma opção ao ChatGPT?** Não. Essa questão emergiu na visão de alguns participantes, sendo necessário elucidá-la. O objetivo do OER-Chat é fornecer REA aos usuários conforme a sua demanda. Para isso, diversos recursos são armazenados e processados pelo *chatbot* antes de fornecer uma resposta. Além disso, o foco do OER-Chat é disponibilizar REA que possam ser baixados, reutilizados e editados pelos usuários. Enquanto isso, o Chat-GPT e outros softwares semelhantes concentram-se em fornecer uma resposta a cada entrada do usuário. Ademais, tais softwares possuem uma arquitetura fechada, no qual não é evidente ao usuário como o processamento de dados ocorre. A proposta do OER-Chat é promover uma arquitetura distinta, permitindo: (i) extensões e melhorias dos usuários, (ii) disponibilidade dos dados utilizados; e (iii) aperfeiçoamento por outros profissionais.

- **Questões Complexas:** Alguns usuários fizeram perguntas complexas durante o uso do OER-Chat, por exemplo *"Como manter os alunos motivados ao ensinar programação introdutória?"* Embora o *chatbot* possa oferecer REA para ajudar com essas perguntas, é essencial reconhecer que abordá-las com o uso de REA é uma tarefa muito difícil.

Cabe ainda destacar que a pesquisa que descreve o design e a validação do OER-Chat foi publicado na 57ª *Hawaii International Conference on System Sciences* (HICSS'57), como apresentado a seguir:

1. Deus, W. S. de; Barbosa. E. F. OER-Chat: An Open Chatbot to Support the Reuse of Open Educational Resources of Introductory Programming. **57 Hawaii International Conference on System Sciences**, 2024, Estados Unidos (HICSS'57). https://hdl.handle.net/10125/107008

#### 6.4.3.1 Conectando os achados com o estudo de caso

O OER-Chat teve sua interface desenvolvida com base nas críticas e desafios notados pelos usuários do grupo focal (Seção 4.2). Por isso sua interface era minimalista, sem adicionar rótulos ou botões que complicassem o entendimento do usuário. Como exposto nos resultados, isso surtiu efeito, já que o design do OER-Chat foi elogiado por grande parte dos participantes.

Outro ponto que merece atenção são os sentimentos positivos que o OER-Chat provocou nos participantes. Apesar de quatro sentimentos negativos terem sido notados, a maioria foram dos sentimentos relatados foram positivos. Ou seja, é provável que os participantes retornem ao uso do OER-Chat no futuro. Inclusive, alguns participantes perguntaram se novas versões do *chatbot* seriam lançadas para serem utilizadas, o que não é comum em plataformas que possuem deficiências em seu design.

Ademais, ao fazer a análise do grupo focal (Capítulo 4), ficou evidente a sensação de frustração nos participantes após usarem as plataformas abertas. Essa sensação provocou grande impacto, pois a sua identificação demonstrou que muitas plataformas, infelizmente, não dialogavam com a essência do que é esperado pelos seus usuários. Isso gera, na maior parte do tempo, uma barreira quase intransponível para que novos usuários possam usar recursos abertos.

O OER-Chat dialogou com os seus usuários, promovendo uma interface simples, funcional e que facilitasse a disseminação dos REA. Há, naturalmente, um longo caminho a ser trilhado para que o *chatbot* se torne uma realidade na rotina de alunos, professores ou qualquer usuário que deseje usar REA introdutórios de programação. Mas, o fato da sensação de frustração não ter ficado presente na maioria dos participantes demonstrou que o projeto pode gerar contribuições relevantes para o futuro, evitando erros do passado.

#### *Ameaças à validade*

Embora o número de participantes seja uma limitação potencial, vale a pena notar que todos os participantes possuem origens diversas e carreiras que lidam com a programação na indústria e na academia. Apesar dessa diferença, todos reconheceram a eficiência do *chatbot* para o uso dos REA. Na prática, foi observado uma convergência de opiniões entre os participantes, o que é significativo visto que eles avaliaram o OER-Chat individualmente, sem qualquer interação entre si.

## 6.5 Considerações Finais

Este capítulo apresentou a estratégia de validação adotada nesta pesquisa. De forma resumida, foi estruturado um projeto de estudo de caso que organiza todos os elementos da pesquisa conduzida. O projeto foi instanciado em três casos distintos referentes aos desafios que os usuários encaram ao pesquisar por um REA de programação introdutória (Capítulo 4). O propósito final foi avaliar a eficiência dos mecanismos de identificação e de recuperação de REA, detalhados no Capítulo 5.

O primeiro caso focou nas condições causais, analisando REA coletados no EngageCSEdu. O segundo caso analisou as questões de contexto, investigando a opinião de dois especialistas usando o MIT *Open CourseWare*. O terceiro caso explorou os efeitos dos usuários, adotando o OER-Chat, um *chatbot* aberto e desenvolvido no âmbito desta pesquisa de doutorado.

No geral, os resultados demonstraram a eficiência de ambos os mecanismos, IPAVL e TAGGER, para identificarem e recuperarem REA de programação introdutória. Também mostraram em diversos ângulos como é possível adotar os mecanismos em diferentes cenários, desde a adoção de algoritmos, classificação de especialistas e desenvolvimento de novas plataformas abertas. A seguir, as conclusões e as perspectivas futuras para a continuidade da pesquisa são discutidas.

CAPÍTULO

7

# CONCLUSÃO

A QP que direcionou toda a investigação desta tese é a seguinte: **Como é possível auxiliar a identificação e a recuperação de REA para o ensino e aprendizado de programação introdutória?** Os capítulos anteriores discutiram essa resposta indiretamente, agora, diante de tudo o que foi exposto e escrito, é possível delinear uma resposta unificadora.

A identificação em si é um processo que se torna complexo de acordo com as particularidades de busca. Identificar um REA de programação entre recursos de Português ou História, por exemplo, pode ser considerado uma tarefa mais simples que pode ser solucionada com aprendizado de máquina. No entanto, identificar se um recurso de programação é introdutório ou avançado envolve camadas mais complexas sobre definir, com exatidão, o significado de *introdutório*.

A recuperação, por sua vez, possui nuances mais delicadas. Aqui o desafio principal emerge da heterogeneidade dos REA. As soluções atuais, na maioria das vezes, usam como base os metadados. Essa estratégia pode ser simples e funcional em muitos casos. Todavia, em recursos mais complexos, como cursos digitais, livros abertos ou vídeos, os metadados cobrem, no melhor dos cenários, apenas uma fração do real conteúdo daquele material. Por isso, muita informação útil do recurso acaba por ser desconsiderada nas formas em que a recuperação é feita atualmente.

Com base nesse cenário, a investigação aqui relatada fornece três elementos de resposta à QP postulada. A primeira delas é a análise sobre quais são os desafios relacionados à identificação e à recuperação de REA voltados ao ensino introdutório de programação. Esta pesquisa demonstrou que os desafios emergem em três momentos distintos: antes de um usuário acessar uma plataforma (condições causais); no momento que o usuário utiliza a interface da plataforma (questões de contexto); e na sensação que o usuário possui ao finalizar sua busca (efeitos).

Esses desafios são amplos e complexos e grande parte do que tem sido proposto é centrado apenas no momento que um usuário usa a plataforma aberta. Isso, no entanto, não

é suficiente. É preciso pensar no momento anterior, sobretudo aos usuários que não possuem contato algum com REA e não sabem, por exemplo, o que representa uma licença aberta. Além disso, é também preciso pensar no momento após a busca. Ou seja, como tornar as sensações do usuários positivas ao invés de negativas. Essa é a primeira parte da resposta da QP.

A segunda parte refere-se em como de resolver esses conflitos e auxiliar, de fato, os usuários a encontrarem REA que sejam úteis para o seu propósito. Diante disso, foi necessário estabelecer uma conexão entre três atores fundamentais, criando uma espécie de tripé tecnológico: (i) as plataformas que armazenam os REA, (ii) os responsáveis pela infraestrutura de tais projetos e (iii) os usuários que acessam essas plataformas em busca de REA. Criar uma plataforma de recursos com conteúdo de programação provavelmente não irá resolver o problema. Ao contrário, será adicionado mais um sistema que o professor ou aluno terá que aprender a usar. É preciso criar ferramentas simples, que conectem coleções e que sejam facilmente utilizadas, tanto pelos usuários quanto pelos responsáveis. Por tudo isso, a segunda parte da responde para a QP desta tese é a proposição de mecanismos, que, na prática, representam interfaces entre esses atores. As regras usadas para os mecanismos são simples e podem ser adaptadas em diferentes plataformas, de diferentes formas. O presente trabalho de doutorado mostrou alguns cenários, mas a aplicação não se limita apenas ao que foi exposto e apresentado. Diversas soluções adicionais, como *plugins*, podem ser criadas.

A terceira parte da resposta diz respeito ao processo de validação que foi conduzido. Esta tese investiga a interseção de duas áreas de pesquisa, a Computação e a Educação. Por um lado, muitas pesquisas que envolvem Computação adotam avaliações por meio um paradigma quantitativo, adotando, por exemplo, o teste de hipótese em um experimento controlado. A Educação, por sua vez, muitas adota um paradigma qualitativo, focando na subjetividade dos dados e das observações conduzidas. Pesquisas que combinam tais abordagens também são uma solução, mesclando elementos quantitativos e qualitativos.

A pesquisa aqui relatada apoiou-se nessa possibilidade, apresentando um percurso metodológico que evoluiu conforme os cenários e casos explorados. Há, naturalmente, desafios nessa abordagem, como a complexidade de coletar e gerenciar os dados bem como estabelecer uma racionalização para unir teoria e prática. Mas, por tudo que foi exposto, esse caminho pareceu ser o mais adequado para gerar resultados e impactos.

Os REA permitem o acesso à educação facilitando sua democratização. Quando se coloca em perspectiva o ensino de programação introdutória, é criada uma ampla janela de oportunidades. Todavia, alguns elementos chamam a atenção sobre pesquisas que envolvem recursos educacionais e programação. O primeiro deles é que muitos trabalhos dificultam ou inviabilizam o acesso aos dados gerados pela pesquisa[1]. Isso gera uma barreira de uso e reuso dos

[1] Isso não é um problema apenas das áreas de Computação e Educação. No entanto, para esta tese, foram encontradas diversas barreiras e ausências de dados sobre pesquisas que analisavam REA e ensino de computação, por isso é necessário sublinhar esse problema.

dados. O segundo é a disseminação de muitos materiais educacionais em formatos proprietários. Livros *abertos* que estão no formato *.pdf* são um bom exemplo disso. É muito difícil, talvez até inviável, editar ou adicionar um novo conteúdo em um livro que possui um formato tão restritivo. Por isso, a atenção da comunidade também deve estar voltada às práticas abertas.

Também é essencial que as ferramentas ou técnicas geradas para facilitar o reuso de REA sejam aliadas às práticas docentes, ou discentes, quando for o caso. Ferramentas complexas, que exigem um alto grau de compreensão, provavelmente terão o seu uso e impacto muito restrito. É preciso dialogar com a prática educacional, oferecendo soluções que sejam aderentes aos desafios encontrados no cotidiano educativo.

Outro ponto que merece atenção é a possibilidade de estabelecer construções concretas usando como base em assuntos que despertam atenção do público. O avanço da Inteligência Artificial é um excelente exemplo disso. Apesar da prática educacional ser transformada com o avanço dessa e de outras tecnologias, ao mesmo tempo, são expostas oportunidades para todos que trabalham com o tema.

Por fim, ao término desta pesquisa, a Inteligência Artificial vem se tornado cada vez mais constante no dia a dia de muitas pessoas. Em particular, na rotina de alunos e professores de programação, o assunto despertou grande atenção pelo lançamento de ferramentas que são capazes de gerar código-fonte para diversas linguagens de programação. Não é possível ignorar essa tendência quando se coloca em perspectiva os REA voltados ao ensino de programação. Assim, é necessário pensar em como conectar a Inteligência Artificial com o uso de REA e o ensino de programação. O OER-Chat, em parte, é um exemplo de contribuição que pode ser aperfeiçoada no futuro.

## 7.1 Contribuições

Em termos científicos, as contribuições da tese se aglutinam em quatro categorias:

1. Situar o atual estado da arte e da prática sobre REA e o ensino e o aprendizado de programação introdutória, mapeando abordagens, técnicas e soluções, bem como a estrutura e a dinâmica de plataformas abertas.

2. Propor um modelo teórico sobre quais são os desafios que impedem a identificação e a recuperação de REA de programação introdutória e o que deve ser feito para reduzir seu impacto no dia a dia.

3. Desenvolver dois mecanismos (IPAVL e TAGGER) que podem ser adotados tanto em plataformas abertas como podem ser operados diretamente pelos usuários que desejam identificar e recuperar REA de programação introdutória.

4. Apresentar um chatbot aberto, denominado OER-Chat, que fornece REA ao invés de respostas textuais. Dessa forma, caso um usuário pergunte ao chatbot *"como criar meu primeiro programa em Java?"* o chatbot lista uma série de materiais educacionais livres e abertos, como slides, videoaulas e livros abertos, ao invés de apresentar um trecho de código.

## Publicações

Durante o percurso de investigação, diferentes artigos científicos foram publicados. A lista dos estudos produzidos pelo autor da pesquisa e/ou em colaboração com outros pesquisadores é apresentada a seguir:

**Periódicos**

1. Deus, W. S. de; Barbosa, E. F. *"A Systematic Mapping of the Classification of Open Educational Resources for Computer Science Education in Digital Sources"*. **IEEE Transactions on Education**, 2021. (IEEE ToE'2021). https://doi.org/10.1109/TE.2021.3 128019
2. Deus, W. S. de; Fioravanti, M. L.; Oliveira, C. D. de; Barbosa, E. F. *"Emergency Remote Computer Science Education in Brazil during the COVID-19 pandemic: Impacts and Strategies"*. **Revista Brasileira de Informática na Educação**, 2020. (RBIE'2020). Available at: https://doi.org/10.5753/RBIE.2020.28.0.1032
3. Avellar, G.; Fioravanti, M.; Deus, W.; Branco, K.; Barbosa, E.; *"SSPOT-VR: mobile and low-cost immersive virtual reality for supporting K-12 students on learning programming"*. **Education and Information Technologies Journal**, 2024. (EIT'2024) https://doi.org/10. 1007/s10639-024-12499-0
4. Oliveira, K. K.; Marcolino, A.; Deus, W. S. de; Falcão, T. P.; Barbosa, E. F.; *"Pensamento Computacional na Programação Introdutória e Habilidades do Século XXI: Um Mapeamento Sistemático da Literatura"*. **Revista Novas Tecnologias na Educação**. (RENOTE'2023). https://doi.org/10.22456/1679-1916.137787
5. Fioravanti, M. L.; Oliveira, C. D. de; Deus, W. S. de; Scatalon, L. P.; Barbosa, E. F. *"Role-Playing Games for Fostering Communication and Negotiation Skills"*. **IEEE Transactions on Education**, 2021. (IEEE ToE'2021). https://doi.org/10.1109/TE.2021.3117 898

**Conferências**

6. Deus, W. S. de; Barbosa. E. F. *"OER-Chat: An Open Chatbot to Support the Reuse of Open Educational Resources of Introductory Programming"*. **57 Hawaii International Conference on System Sciences**, 2024, Estados Unidos (HICSS'57). https://hdl.handle.net/10125/107008

7. Deus, W. S. de; Barbosa, E. F. *"Recursos Educacionais Abertos para o Ensino e Aprendizado de Programação no Brasil: Primeiros Referenciais"*. **XXXII Simpósio Brasileiro de Informática na Educação**, 2021, Brasil. (SBIE'2021). https://doi.org/10.57 53/sbie.2021.218 391

8. Deus, W. S. de; Barbosa, E. F. *"The Use of Metadata in Open Educational Resources Repositories: An Exploratory Study"*. **44th Annual Computers, Software, and Applications Conference**, 2020, Espanha. (COMPSAC'2020). https://doi.org/10.1109/COMPSAC48688.2020.00025

9. Deus, W. S. de; Barbosa, E. F. *"An Exploratory Study on the Availability of Open Educational Resources to Support the Teaching and Learning of Programming"*. In: **50th Frontiers in Education Conference**, 2020, Suécia. (FIE'2020). https://doi.org/10.1109/FIE44824.2020.9274202

10. Deus, W. S. de; Barbosa, E. F. *"Supporting the Retrieval of Open Educational Resources for Programming Education"*. In: **Postgraduate Students Experience**, 2022, Brasil. (StudX'2022). https://doi.org/10.5753/cbie_estendido.2022.226720

11. Balbino, F. C.; Deus, W. S.; Barbosa, E. F. *"A Dynamic Open Educational Resources Repository to Enhance Primary and Secondary Education"*. **IEEE International Conference on Advanced Learning Technologies**, 2023, Estados Unidos. (ICALT'2023). https://doi.org/10.1109/ICALT58122.2023.00008.

12. Oliveira, K. K.; Deus, W. S. de; Avellar, G. M. N.; Falcão, T. P.; Barbosa, E. F.; *"Análise dos Cursos de Computação no Brasil: Ensino de Programação sob a perspectiva das Necessidades do Século XXI"*. **In XXXIV Simpósio Brasileiro de Informática na Educação**. Passo Fundo, RS. https://doi.org/10.5753/sbie.2023.234970

13. Balbino, F. C.; Deus, W. S.; Barbosa, E. F. *"Recursos Educacionais Abertos para Apoio ao Ensino de Computação na Educação Básica"*. **Simpósio Brasileiro de Educação em Computação**, 2023, Brasil (EduComp'2023). DOI: https://doi.org/10.5753/edu comp_estendido.2023.228421.

14. Paula, L. B. de; Deus, W. S. de; Barbosa, E. F. *"Análise de Repositórios de REAs em Relação ao Uso dos Padrões de Linked Open Data"*. **XXXI Simpósio Brasileiro de Informática na Educação**, 2020, Brasil. (SBIE'2021). https://doi.org/10.5753/cbie.sbie.2020.312

**Capítulo de livro**

15. Deus, W. S., Balbino, F. C., Barcelos, L. V., Barbosa, E. F. (2024). *"Fostering the Teaching and Learning of Computer Programming With Open Educational Resources: Challenges and Solutions"*. In C. Bosch, L. Goosen, & J. Chetty (Eds.), **Navigating Computer Science Education in the 21st Century (pp. 21-40)**. IGI Global. https://doi.org/10.4018/979-8-3693-1066-3.ch002

Dois estudos estão sendo desenvolvidos: o primeiro deles apresenta a IPAVL e será submetido ao periódico *ACM Transactions on Computing Education*. O segundo apresenta o TAGGER e será enviado para avaliação no periódico *IEEE Transactions on Learning Technologies*:

1. Deus, W. S. de; Barbosa, E. F. *"An Academic Vocabulary List to Identify Open Educational Resources for Introductory Programming"*. In: **ACM Transactions on Computing Education**. *Em desenvolvimento*.

2. Deus, W. S. de; Morrison. B.; Barbosa, E. F. *"Identifying Open Educational Resources for Introductory Programming: A Case Study from TAGGER"*. In: **IEEE Transactions on Education**. *Em desenvolvimento*.

## Orientações

Durante o período desta pesquisa, o autor colaborou como orientador ou coorientador dos seguintes trabalhos de conclusão de curso:

1. Fernando César Balbino. AquaRElA - Um Editor de REA. 2022. Monografia - Universidade de São Paulo.

2. Luciano Bernardes de Paula. Análise de Repositórios de REAs em Relação ao Uso dos Padrões de Linked Open Data. 2020. Monografia - Universidade de São Paulo.

3. Marcelo Antonio de Carvalho Junior. O uso de metadados para identificação de Recursos Educacionais Abertos: uma abordagem experimental verificando a influência de metadados em imagens. 2020. Monografia - Universidade de São Paulo.

4. Marlucia Delfino Amaral. Estudo de Caso do acervo de Recursos Educacionais do conteúdo de Ciências da Natureza, disponíveis na Plataforma Integrada RED/MEC. 2020. Monografia - Universidade de São Paulo.

5. Rafael Campelo. Uso de recursos educacionais abertos no ensino básico: uma análise sob a perspectiva de professores de física. 2020. Monografia - Universidade de São Paulo.

6. Aline Teles Cristalino. Recursos Educacionais Abertos para apoiar o Ensino Remoto Emergencial: Uma perspectiva baseada em Formulário Digitais. 2020. Monografia - Universidade de São Paulo.

7. Lucas Osorio Alves da Silva. Novas Perspectivas nas Produções de Recursos Educacionais Abertos Orientados pelo Design Estratégico. 2020. Monografia - Universidade de São Paulo.

### Apresentações

Além das apresentações dos artigos em eventos científicos, em quatro oportunidades distintas foram feitas apresentações para a comunidade:

- *Open Education and Computing: a Pathway to Democratize Access to Education in Brazil*: Apresentação no Workshop da **Communications of the ACM - Regional Special Sections on Latin America 2024**, (2023).
- *Recursos Educacionais Abertos: Pensando desafios e possibilidades*. Apresentação na **Roda de Debates - Educação Aberta: REAs e MOOCs**", organizado pelo Curso de Pós-Graduação em Computação Aplicada à Educação e Tecnologias Educacionais do Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo (2023).
- *Recursos Educacionais Abertos: Definições e uso.* Apresentação para os discentes do **Programa de Pós-Graduação em Tecnologia e Gestão em Educação a Distância** da Universidade Federal Rural de Pernambuco (2021).
- *Recursos Educacionais Abertos: Desafios e Oportunidades.* Apresentação na **Roda de Debates - Educação Aberta: REAs e MOOCs**", organizado pelo Curso de Pós-Graduação em Computação Aplicada à Educação e Tecnologias Educacionais do Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo (2021).

Dois artigos produzidos na tese - *"A Systematic Mapping of the Classification of Open Educational Resources for Computer Science Education in Digital Sources"* (DEUS; BARBOSA, 2022) e *"The Use of Metadata in Open Educational Resources Repositories: An Exploratory Study"* (DEUS; BARBOSA, 2020) foram apresentados no *Open Education Week 2022*[2] pela professora Dra. Ellen Francine Barbosa. O evento foi realizado com o apoio da UNESCO e buscou debater a qualidade dos REA, uma das dimensões chave da área de ação da recomendação mais recente da Unesco (2019).

### Reconhecimentos

O artigo *A Dynamic Open Educational Resources Repository to Enhance Primary and Secondary Education* produzido por Balbino, Deus e Barbosa (2023) foi nomeado para premiação do melhor trabalho completo do evento *IEEE International Conference on Advanced Learning Technologies*[3].

Três orientações e/ou coorientações receberam as seguintes premiações da Especialização em Computação Aplicada à Educação e Tecnologias Educacionais do ICMC/USP:

[2] https://events.unesco.org/event?id=OER_Dynamic_Coalition_webinar_3752501155lang=1033
[3] https://tc.computer.org/tclt/icalt-2023-best-paper-award-nominations/

1. Prêmio de Melhor Trabalho de Conclusão de Curso do aluno Fernando César Balbino pela construção de um repositório dinâmico de REA.
2. Prêmio de Menção Honrosa da aluna Marlucia Delfino Amaral pela condução de um estudo de caso na plataforma MEC/RED.
3. Prêmio de Destaque Acadêmico do aluno Luciano Bernardes de Paula pela publicação parcial dos resultados no XXXI Simpósio Brasileiro de Informática na Educação

O autor recebeu a premiação de melhor revisor na Trilha 4 (Recursos e Ambientes Educacionais) pela sua atuação em duas edições do Simpósio Brasileiro de Educação em Computação (EduComp) nos anos de 2021 e 2022.

### *Dados Abertos*

Com o propósito de facilitar a disseminação dos resultados alcançados por esta pesquisa, os dados coletados ou gerados durante o período de investigação estão disponíveis para acesso e *download* no repositório: https://doi.org/10.5281/zenodo.12744288.

### *Comitê de Ética*

O projeto de doutorado foi submetido e aprovado pelo Comitê de Ética em Pesquisa com o título *Recursos Educacionais Abertos para o Ensino e Aprendizado de Programação* (Número do CAAE: 66316222.4.0000.5390). Assim, foram adotados diversos procedimentos para garantir segurança aos participantes da pesquisa bem como adequação aos códigos de ética para pesquisadores.

## 7.2 Limitações

Os capítulos anteriores trouxeram de forma contextualizada uma lista das ameaças e riscos à validade das investigações conduzidas. A seguir, são listadas algumas limitações da pesquisa em um sentido mais amplo.

Em primeiro lugar, devido à pandemia, não foi possível realizar encontros pessoais com participantes. A pandemia também impactou a participação de outros usuários, como alunos iniciantes em programação, tendo em vista as dificuldades de identificar potenciais colaboradores e realizar agendamentos de reuniões *online*.

Em segundo lugar, o foco da validação conduzida nos mecanismos foi baseado na identificação e na recuperação de REA voltados ao ensino introdutório de programação. Não foi avaliado de forma direta a eficiência de ensinar programação utilizando os REA que os mecanismos forneceram aos usuários nem mesmo o processo de reuso desses materiais por outros usuários.

Por fim, cabe destacar que os mecanismos apresentados foram validados utilizando diferentes cenários de aplicação. Mas, ferramentas específicas ainda podem ser desenvolvidas para a realização de testes mais detalhados, focando sobretudo em diferentes tipos de usuários que podem utilizar REA para o ensino ou o aprendizado de programação introdutória.

## 7.3 Trabalhos Futuros

Em relação aos trabalhos futuros, diversas possibilidades podem ser exploradas. As principais linhas de ação são apresentadas a seguir:

- **Uso dos mecanismos por alunos:** devido aos desafios impostos pela pandemia, não foi possível testar os mecanismos com alunos iniciantes em programação. Por isso mesmo, uma contribuição relevante é realizar o teste com esse perfil de usuário.
- **Ensino de programação em diferentes perspectivas:** Outra contribuição relevante é expandir o conceito de programação em diferentes níveis, sobretudo o ensino de nível avançado.
- **Analisar o reuso dos REA:** O reuso dos recursos por usuários, como a remixagem ou readaptação, também são pontos relevantes para pesquisas futuras. Logo, será possível analisar como alcançar os 5R's em conjunto com os mecanismos desenvolvidos.
- **Desenvolvimento de ferramentas:** Novas ferramentas educacionais, sobretudo softwares e aplicativos, podem ser desenvolvidos para integrar os mecanismos em diferentes etapas da identificação e da recuperação de REA. Nesse sentido, *plugins* que funcionem em plataformas abertas, criação de filtros ou ferramentas que apoiem o processo de elaboração de *strings* de buscas são contribuições futuras relevantes.
- **Inteligência Artificial:** A aplicação da Inteligência Artificial em união com os REA pode apresentar contribuições significativas em torno dos trabalhos a serem desenvolvidos. No futuro, pretende-se investigar como aplicar os 5R's de forma efetiva com o apoio da Inteligência Artificial. É importante o desenvolvimento de plataformas educacionais que buscam conectar produtores de conteúdos com usuários. Para isso, pretende-se evoluir o OER-Chat com novos recursos e possibilidades.

# REFERÊNCIAS

ACM/IEEE-CS. **Computing Curricula 2020**. 1. ed.. ed. Association for Computing Machinery (ACM) e IEEE Computer Society (IEEE-CS), 2020. Disponível em: <https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2020.pdf>. Citado nas páginas 29, 31 e 40.

AMIEL, T.; OREY, M.; WEST, R. Recursos educacionais abertos (rea): modelos para localização e adaptação. **ETD - Educação Temática Digital**, v. 12, p. 112–125, nov. 2010. Disponível em: <https://periodicos.sbu.unicamp.br/ojs/index.php/etd/article/view/1206>. Citado na página 37.

ARAUJO, A.; ANDRADE, W.; GUERRERO, D. Um mapeamento sistemático sobre a avaliação do pensamento computacional no brasil. In: **Anais dos Workshops do Congresso Brasileiro de Informática na Educação**. Uberlândia, Brasil: SBC, 2016. v. 5, n. 1, p. 1147–1158. ISSN 2316-8889. Disponível em: <https://doi.org/10.5753/cbie.wcbie.2016.1147>. Citado na página 82.

ARAUJO, R.; ZORZO, A.; NUNES, D.; MATOS, E.; STEINMACHER, I.; LEITE, j.; CORREIA, r.; MARTINS, S. **Referenciais de Formação para os Cursos de Graduação em Computação 2017**. 1. ed.. ed. Porto Alegre: Sociedade Brasileira de Computação, 2017. Disponível em: <https://www.sbc.org.br/documentos-da-sbc/summary/131-curriculos-de-referencia/1165-referenciais-de-formacao-para-cursos-de-graduacao-em-computacao-outubro-2017>. Citado nas páginas 31 e 40.

BAHAMÓN, J. C. A case study on the adoption of open educational resources in a c programming course. In: **Proceedings of the 53rd ACM Technical Symposium on Computer Science Education V. 2**. Providence, RI, USA. Proceedings [...]. New York, NY, USA: Association for Computing Machinery, 2022. p. 1128. Disponível em: <https://doi.org/10.1145/3478432.3499095>. Citado nas páginas 30 e 32.

BALBINO, F. C.; DEUS, W. S. de; BARBOSA, E. F. A dynamic open educational resources repository to enhance primary and secondary education. In: **2023 IEEE International Conference on Advanced Learning Technologies (ICALT)**. [S.l.: s.n.], 2023. p. 4–8. Citado na página 139.

BALDWIN, D. Can we "flip"non-major programming courses yet? In: **Proceedings of the 46th ACM Technical Symposium on Computer Science Education**. Kansas City, Missouri, USA. Proceedings [...]. New York, NY, USA: Association for Computing Machinery, 2015. p. 563–568. Disponível em: <https://doi.org/10.1145/2676723.2677271>. Citado nas páginas 30 e 43.

BARKER, P. **What is IEEE learning object metadata/IMS learning resource metadata**. Cetis, 2005. Disponível em: <http://www.dia.uniroma3.it/~sciarro/e-learning/WhatIsLOMscreen.pdf>. Citado na página 39.

BATTARBEE, K.; KOSKINEN, I. Co-experience: user experience as interaction. **CoDesign**, Taylor & Francis, v. 1, n. 1, p. 5–18, 2005. Disponível em: <https://doi.org/10.1080/15710880412331289917>. Citado na página 73.

BECKER, B. A.; QUILLE, K. 50 years of cs1 at sigcse: A review of the evolution of introductory programming education research. In: . 2019, Minneapolis, MN, USA. Proceedings [...]. New York, NY, USA: Association for Computing Machinery, 2019. p. 338–344. Disponível em: <https://doi.org/10.1145/3287324.3287432>. Citado nas páginas 31 e 46.

BEGEL, A.; KO, A. J. Learning outside the classroom. In: ____. **The Cambridge Handbook of Computing Education Research**. Cambridge: Cambridge University Press, 2019. (Cambridge Handbooks in Psychology), p. 749–772. Disponível em: <https://doi.org/10.1017/9781108654555.027>. Citado nas páginas 40 e 41.

BORGES, R. P.; OLIVEIRA, P. R. F.; LIMA, R. G. da R.; LIMA, R. W. de. A systematic review of literature on methodologies, practices, and tools for programming teaching. **IEEE Latin America Transactions**, v. 16, n. 5, p. 1468–1475, Maio 2018. Disponível em: <https://doi.org/10.1109/TLA.2018.8408443>. Citado na página 41.

BRASIL. Lei nº 14.533, de 11 de janeiro de 2023. institui a política nacional de educação digital e altera as leis nºs 9.394, de 20 de dezembro de 1996 (lei de diretrizes e bases da educação nacional), 9.448, de 14 de março de 1997, 10.260, de 12 de julho de 2001, e 10.753, de 30 de outubro de 2003. **Diário Oficial [da] República Federativa do Brasil**, Brasília, DF, 2023. Disponível em: <https://www.in.gov.br/web/dou/-/lei-n-14.533-de-11-de-janeiro-de-2023-457334986>. Citado na página 30.

BRETTHAUER, D. Open source software: A history. **UConn Libraries Published Works**, v. 21, p. 1–23, Março 2001. Disponível em: <https://digitalcommons.lib.uconn.edu/cgi/viewcontent.cgi?article=1009&context=libr_pubs>. Citado na página 36.

CHAN, H. C. B.; HO, Y. H.; TOVAR, E.; REISMAN, S. Enhancing the learning of computing/it students with open educational resources. In: **Proceedings of the IEEE 44th Annual Computers, Software, and Applications Conference**. 2020, Madrid, Spain. Proceedings [...].: [s.n.], 2020. p. 113–122. Disponível em: <https://doi.org/10.1109/COMPSAC48688.2020.00024>. Citado nas páginas 30 e 43.

CLIFF, S.; MELISSA, M. Axial coding. **The International Encyclopedia of Communication Research Methods**, John Wiley Sons, Inc., Nova Jersey, EUA, v. 62, p. 1–2, 2017. Disponível em: <https://doi.org/10.1002/9781118901731>. Citado na página 65.

COMAS-QUINN, A.; BORTHWICK, K. Sharing: Open educational resources for language teachers. In: ____. **Developing Online Language Teaching: Research-Based Pedagogies and Reflective Practices**. Londres, Reino Unido: Palgrave Macmillan UK, 2015. p. 96–112. ISBN 978-1-137-41226-3. Disponível em: <https://doi.org/10.1057/9781137412263_7>. Citado na página 37.

COMBEFIS, S.; MOFFARTS, G.; JOVANOV, M. Tlcs: A digital library with resources to teach and learn computer science. **Olympiads in Informatics**, IOI, v. 13, p. 3–20, Setembro 2019. Disponível em: <https://ioinformatics.org/journal/v13_2019_3_20.pdf>. Citado na página 42.

CORTINOVIS, R.; MIKROYANNIDIS, A.; DOMINGUE, J.; MULHOLLAND, P.; FARROW, R. Supporting the discoverability of open educational resources. **Education and Information Technologies**, v. 24, n. 5, p. 3129–3161, Setembro 2019. Disponível em: <https://doi.org/10.1007/s10639-019-09921-3>. Citado nas páginas 32 e 42.

COXHEAD, A. A new academic word list. **TESOL Quarterly**, v. 34, n. 2, p. 213–238, 2000. Disponível em: <http://www.jstor.org/stable/3587951>. Citado nas páginas 84 e 112.

CURIE, D. H.; JAISON, J.; YADAV, J.; FIONA, J. R. Analysis on web frameworks. **International Conference on Physics and Photonics Processes in Nano Sciences**, v. 1362, p. 1–6, 2019. Disponível em: <https://www.doi.org/10.1088/1742-6596/1362/1/012114>. Citado na página 92.

DESSì, D.; FENU, G.; MARRAS, M.; Reforgiato Recupero, D. Bridging learning analytics and cognitive computing for big data classification in micro-learning video collections. **Computers in Human Behavior**, v. 92, p. 468 – 477, 2019. ISSN 0747-5632. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0747563218301092>. Citado nas páginas 45, 46 e 126.

DEUS, W. S. de; BARBOSA, E. F. The use of metadata in open educational resources repositories: An exploratory study. In: **Proceedings of the IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)**. IEEE, 2020. p. 123–132. Disponível em: <https://doi.org/10.1109/COMPSAC48688.2020.00025>. Citado nas páginas 80 e 139.

____. A systematic mapping of the classification of open educational resources for computer science education in digital sources. **IEEE Transactions on Education**, v. 65, n. 3, p. 450–460, Agosto 2022. Disponível em: <https://doi.org/10.1109/TE.2021.3128019>. Citado nas páginas 48, 80 e 139.

DICHEV, C.; DICHEVA, D. Open educational resources in computer science teaching. In: **Proceedings of the 43rd ACM Technical Symposium on Computer Science Education**. 2012, New York, NY, USA. Proceedings [...]. Raleigh, United States: ACM, 2012. p. 619–624. Disponível em: <https://doi.org/10.1145/2157136.2157314>. Citado na página 32.

DICHEVA, D.; DICHEV, C. Finding open educational resources in computing. In: **IEEE 14th International Conference on Advanced Learning Technologies**. 2014, Washington, DCUnited States. Proceedings [...]. Atenas, Grécia: IEEE Computer Society, 2014. p. 22–24. Disponível em: <https://doi.org/10.1109/ICALT.2014.17>. Citado na página 31.

DIETTERICH, T. Overfitting and undercomputing in machine learning. **ACM Computing Surveys**, Association for Computing Machinery, New York, NY, USA, v. 27, n. 3, p. 326–327, Setembro 1995. Disponível em: <https://doi.org/10.1145/212094.212114>. Citado na página 105.

DRESCH, A.; LACERDA, D. P.; ANTUNES, J. A. V. Design science research. In: ____. **Design Science Research: A Method for Science and Technology Advancement**. Cham: Springer International Publishing, 2015. p. 67–102. ISBN 978-3-319-07374-3. Disponível em: <https://doi.org/10.1007/978-3-319-07374-3_4>. Citado na página 33.

DUNCAN, C.; BELL, T.; TANIMOTO, S. Should your 8-year-old learn coding? In: **Proceedings of the 9th Workshop in Primary and Secondary Computing Education**. 2014, New York, NY, USA. Proceedings [...]. Berlin, Alemanha: ACM, 2014. p. 60–69. Disponível em: <http://doi.acm.org/10.1145/2670757.2670774>. Citado nas páginas 30, 40 e 42.

DURAK, H. Y. The effects of using different tools in programming teaching of secondary school students on engagement, computational thinking and reflective thinking skills for problem solving. **Technology, Knowledge and Learning**, v. 25, n. 1, p. 1–17, Agosto 2018. Disponível em: <https://doi.org/10.1007/s10758-018-9391-y>. Citado na página 41.

FALKNER, N.; VIVIAN, R.; PIPER, D.; FALKNER, K. Increasing the effectiveness of automated assessment by increasing marking granularity and feedback units. In: **Proceedings**

**of the 45th ACM Technical Symposium on Computer Science Education**. 2014, Atlanta, Georgia, USA. Proceedings [...]. New York, NY, USA: ACM, 2014. p. 9–14. Disponível em: <http://doi.org/10.1145/2538862.2538896>. Citado na página 41.

GARCÍA, M. M.; RODRÍGUEZ, R. P.; RIFÓN, L. A.; FERRO, M. V. Towards a multi-label classification of open educational resources. In: **2015 IEEE 15th International Conference on Advanced Learning Technologies**. [s.n.], 2015. p. 407–408. Disponível em: <https://doi.org/10.1109/ICALT.2015.55>. Citado na página 38.

GARDNER, D.; DAVIES, M. A New Academic Vocabulary List. **Applied Linguistics**, v. 35, n. 3, p. 305–327, Agosto 2013. Disponível em: <https://doi.org/10.1093/applin/amt015>. Citado na página 112.

GARSHOL, L. M. Metadata? thesauri? taxonomies? topic maps! making sense of it all. **Journal of Information Science**, v. 30, n. 4, p. 378–391, 2004. Disponível em: <https://doi.org/10.1177/0165551504045856>. Citado nas páginas 38 e 39.

GONÇALVES, C. A. F.; FERREIRA, D. C.; CUNHA, J. M. D. J.; RODRIGUES, R. F. T.; RODRIGUES, V. L. R. O uso do estrangeirismo na língua portuguesa. **Revista Acadêmica Interinstitucional**, n. 10, p. 1–32, Março 2011. Disponível em: <https://fals.com.br/novofals/revela/REVELA%20XVII/artigoexper_05revela10.pdf>. Citado na página 48.

GREGOR, S. The nature of theory in information systems. **MIS quarterly**, v. 30, n. 3, p. 611–642, 2006. Disponível em: <https://doi.org/10.2307/25148742>. Citado nas páginas 77 e 78.

GUNARATHNE, W. K. T. M.; CHOOTONG, C.; SOMMOOL, W.; OCHIRBAT, A.; CHEN, Y.; REISMAN, S.; SHIH, T. K. Web-based learning object search engine solution together with data visualization: The case of merlot ii. In: **2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)**. [S.l.: s.n.], 2018. v. 01, p. 1026–1031. ISBN 978-1-5386-2667-2. Citado na página 44.

HALSTEAD-NUSSLOCH, R.; RUTHERFOORD, R. Tips for sources of cost-free and open educational resources to reduce textbook costs in it courses. In: **Proceedings of the 20th Annual SIG Conference on Information Technology Education**. Tacoma, United States: ACM, 2019. (SIGITE '19), p. 174–174. ISBN 978-1-4503-6921-3. Disponível em: <10.1145/3349266.3351359>. Citado na página 42.

HAVEMANN, L. Open educational resources. In: ____. **Encyclopedia of Educational Philosophy and Theory**. Singapore: Springer Singapore, 2017. p. 1706–1712. Disponível em: <https://doi.org/10.1007/978-981-287-588-4_218>. Citado na página 29.

HERTZ, M. What do "cs1"and "cs2"mean? investigating differences in the early courses. In: . 2010, Milwaukee, Wisconsin, USA. Proceedings [...]. New York, NY, USA: Association for Computing Machinery, 2010. p. 199–203. Disponível em: <https://doi.org/10.1145/1734263.1734335>. Citado nas páginas 31 e 113.

KABIR, R.; KABIR, S.; AMIN, S. Isolating informative blocks from large web pages using html tag priority assignment based approach. **Electrical Computer Engineering: An International Journal**, v. 4, n. 3, p. 61–72, 2015. Disponível em: <http://dx.doi.org/10.14810/ecij.2015.4305>. Citado nas páginas 92 e 103.

KASTRATI, Z.; IMRAN, A. S.; KURTI, A. Transfer learning to timed text based video classification using cnn. In: **Proceedings of the 9th International Conference on Web Intelligence, Mining and Semantics**. New York, NY, USA: Association for Computing Machinery, 2019. (WIMS2019). ISBN 9781450361903. Disponível em: <https://doi.org/10.1145/3326467.3326483>. Citado nas páginas 45, 46 e 126.

KHANDKAR, S. H. **Open coding**. Calgary, Alberta, CA: University of Calgary, 2009. Open Coding by Shahedul Huq Khandkar. Disponível em: <https://pages.cpsc.ucalgary.ca/~saul/wiki/uploads/CPSC681/open-coding.pdf>. Citado na página 65.

KIM, M. The creative commons and copyright protection in the digital era: Uses of creative commons licenses. **Journal of Computer-Mediated Communication**, Oxford University Press Oxford, UK, v. 13, n. 1, p. 187–209, 2007. Disponível em: <https://academic.oup.com/jcmc/article/13/1/187/4583060>. Citado na página 36.

KITCHENHAM, B.; CHARTERS, S. Guidelines for performing systematic literature reviews in software engineering. Keele University and University of Durham, Keele, Staffs, UK. Durham, UK, Jul 2007. Disponível em: <http://cdn.elsevier.com/promis_misc/525444systematicreviewsguide.pdf>. Citado na página 43.

LAHTINEN, E.; ALA-MUTKA, K.; JARVINEN, H.-M. A study of the difficulties of novice programmers. **SIGCSE Bulletin**, v. 37, n. 3, p. 14–18, Junho 2005. Disponível em: <https://doi.org/10.1145/1151954.1067453>. Citado na página 39.

LANDIS, J. R.; KOCH, G. G. The measurement of observer agreement for categorical data. **biometrics**, JSTOR, v. 33, n. 1, p. 159–174, Março 1977. Disponível em: <http://www.jstor.org/stable/2529310?origin=JSTOR-pdf>. Citado nas páginas 114 e 120.

LEI, L.; LIU, D. A new medical academic word list: A corpus-based study with enhanced methodology. **Journal of English for Academic Purposes**, v. 22, p. 42–53, 2016. ISSN 1475-1585. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1475158516300078>. Citado nas páginas 84 e 87.

LINDBERG, R. S.; LAINE, T. H.; HAARANEN, L. Gamifying programming education in k-12: A review of programming curricula in seven countries and programming games. **British Journal of Educational Technology**, Wiley Online Library, v. 50, n. 4, p. 1979–1995, Setembro 2019. Disponível em: <https://doi.org/10.1111/bjet.12685>. Citado nas páginas 30 e 40.

LOKAR, M.; PRETNAR, M. A low overhead automated service for teaching programming. In: **Proceedings of the 15th Koli Calling Conference on Computing Education Research**. 2015, Koli, Finlândia. Proceedings [...]. New York, NY, USA: ACM, 2015. (Koli Calling '15), p. 132–136. Disponível em: <http://doi.org/10.1145/2828959.2828964>. Citado na página 41.

MARCOLINO, A.; BARBOSA, E. A survey on problems related to the teaching of programming in brazilian educational institutions. In: **Proceedings of the 47th IEEE/ASE Frontiers in Education Conference**. Indianapolis, United States: IEEE, 2017. (FIE'47), p. 1–9. Disponível em: <https://doi.org/10.1109/FIE.2017.8190495>. Citado nas páginas 42 e 43.

MAZZARDO, M. D. **Recursos educacionais abertos: inovação na produção de materiais didáticos dos professores do ensino médio**. Tese (Doutorado) — Universidade Aberta, Novembro 2018. Disponível em: <https://repositorioaberto.uab.pt/handle/10400.2/7788>. Citado nas páginas 35 e 36.

MCGREAL, R. A typology of learning object repositories. In: ____. **Handbook on Information Technologies for Education and Training**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. p. 5–28. ISBN 978-3-540-74155-8. Disponível em: <https://doi.org/10.1007/978-3-540-74155-8_1>. Citado na página 37.

MEDEIROS, R. P.; RAMALHO, G. L.; FALCÃO, T. P. A systematic literature review on teaching and learning introductory programming in higher education. **IEEE Transactions on Education**, v. 62, n. 2, p. 77–90, May 2019. Disponível em: <https://doi.org/10.1109/TE.2018.2864133>. Citado nas páginas 29, 30 e 40.

MEHMOOD, E.; ABID, A.; FAROOQ, M. S.; NAWAZ, N. A. Curriculum, teaching and learning, and assessments for introductory programming course. **IEEE Access**, v. 8, p. 125961–125981, 2020. Citado na página 42.

MONGE, A.; QUINN, B. A.; FADJO, C. L. Engagecsedu: Cs1 and cs2 materials for engaging and retaining undergraduate cs students. In: **Proceedings of the 46th ACM Technical Symposium on Computer Science Education**. 2015, Kansas City, Missouri, USA. Proceedings [...]. New York, NY, USA: Association for Computing Machinery, 2015. p. 271. Disponível em: <https://doi.org/10.1145/2676723.2691875>. Citado na página 30.

MOURIÑO-GARCÍA, M.; PÉREZ-RODRÍGUEZ, R.; ANIDO-RIFÓN, L.; FERNÁNDEZ-IGLESIAS, M. J.; DARRIBA-BILBAO, V. M. Cross-repository aggregation of educational resources. **Computers Education**, v. 117, p. 31–49, Fevereiro 2018. ISSN 0360-1315. Disponível em: <https://doi.org/10.1016/j.compedu.2017.09.014>. Citado nas páginas 31, 44, 46, 103, 105, 106 e 126.

NAVARRETE, R.; LUJÁN-MORA, S. Bridging the accessibility gap in open educational resources. **Universal Access in the Information Society**, v. 17, n. 4, p. 755–774, Novembro 2018. Disponível em: <https://doi.org/10.1007/s10209-017-0529-9>. Citado na página 29.

OHASHI, Y.; KUMENO, F.; YAMACHI, H.; TSUJIMURA, Y. Readiness of japanese elementary school teachers to begin computer-programming education. In: **2018 IEEE International Conference on Teaching, Assessment, and Learning for Engineering**. 2018, Proceedings [...]. Wollongong, Australia: IEEE, 2018. p. 807–810. Disponível em: <https://doi.org/10.1109/TALE.2018.8615358>. Citado nas páginas 30 e 40.

OLIVEIRA, T. de; STRINGHINI, D.; CORREA, D. G. M. Strategies focused on the teaching of programming logic: A systematic review of brazilian literature. In: **2018 XIII Latin American Conference on Learning Technologies (LACLO)**. 2018, São Paulo, SP, Brasil. Proceedings [...].: [s.n.], 2018. p. 292–298. Disponível em: <https://doi.org/10.1109/LACLO.2018.00059>. Citado na página 41.

O'NEIL, C. **Algoritmos de destruição em massa**. 1. ed.. ed. [S.l.]: Editora Rua do Sabão, 2021. Citado na página 32.

ONU. **Objetivo 4. Assegurar a educação inclusiva e equitativa e de qualidade, e promover oportunidades de aprendizagem ao longo da vida para todas e todos**. ONU, 2023. Os Objetivos de Desenvolvimento Sustentável no Brasil. Disponível em: <https://brasil.un.org/pt-br/sdgs/4>. Citado na página 32.

PANDIT, N. R. The creation of theory: A recent application of the grounded theory method. **The qualitative report**, Fort Lauderdale, v. 2, n. 4, p. 1–15, 1996. Disponível em: <https://doi.org/10.46743/2160-3715/1996.2054>. Citado nas páginas 64, 65, 75 e 124.

PANT, G. Deriving link-context from html tag tree. In: **Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery**. New York, NY, USA: Association for Computing Machinery, 2003. p. 49–55. Disponível em: <https://doi.org/10.1145/882082.882094>. Citado na página 91.

PAWLOWSKI, J. M.; BICK, M. Open educational resources. **Business & Information Systems Engineering**, v. 4, n. 4, p. 209–212, Agosto 2012. Disponível em: <https://doi.org/10.1007/s12599-012-0219-3>. Citado na página 29.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V. *et al.* Scikit-learn: Machine learning in python. **the Journal of machine Learning research**, JMLR. org, v. 12, p. 2825–2830, 2011. Citado na página 105.

PIEDRA, N.; CHICAIZA, J.; LóPEZ, J.; MARTíNEZ, O.; CARO, E. T. An approach for description of open educational resources based on semantic technologies. In: **IEEE EDUCON 2010 Conference**. Proceedings [...] Madrid. Espanha: IEEE, 2010. p. 1111–1119. Disponível em: <https://doi.org/10.1109/EDUCON.2010.5492453>. Citado na página 31.

PIMENTEL, M.; FILIPPO, D.; SANTORO, F. M. Design science research: fazendo pesquisas científicas rigorosas atreladas ao desenvolvimento de artefatos computacionais projetados para a educação. In: ____. **Metodologia de Pesquisa Científica em Informática na Educação: Concepção de Pesquisa.** Porto Alegre: SBC, 2019. p. 1–29. Disponível em: <https://metodologia.ceie-br.org/livro-1/>. Citado na página 33.

PULKER, H. **Teacher's Practices through Adaptation of Open Educational Resources for Online Language Teaching**. Tese (Doutorado) — The Open University, May 2019. Disponível em: <http://oro.open.ac.uk/62109/>. Citado nas páginas 29 e 32.

QIAN, Y.; LEHMAN, J. Students' misconceptions and other difficulties in introductory programming: A literature review. **ACM Transactions on Computing Education**, v. 18, n. 1, Outubro 2017. Disponível em: <https://doi.org/10.1145/3077618>. Citado nas páginas 31 e 41.

RALPH, P. Toward methodological guidelines for process theories and taxonomies in software engineering. **IEEE Transactions on Software Engineering**, v. 45, n. 7, p. 712–735, 2019. Disponível em: <https://doi.org/10.1109/TSE.2018.2796554>. Citado nas páginas 79, 80 e 83.

RATHOD, N.; CASSEL, L. Building a search engine for computer science course syllabi. In: **Proceedings of the 13th ACM/IEEE-CS Joint Conference on Digital Libraries**. New York, NY, USA: Association for Computing Machinery, 2013. p. 77–86. Disponível em: <https://doi.org/10.1145/2467696.2467723>. Citado na página 105.

ROBINS, A. V. Novice programmers and introductory programming. In: ____. **The Cambridge Handbook of Computing Education Research**. Cambridge University Press, 2019. (Cambridge Handbooks in Psychology), p. 327–376. Disponível em: <https://doi.org/10.1017/9781108654555.013>. Citado na página 40.

RODÉS-PARAGARINO, V.; GEWERC-BARUJEL, A.; LLAMAS-NISTAL, M. Use of repositories of digital educational resources: State-of-the-art review. **IEEE Revista Iberoamericana de Tecnologias del Aprendizaje**, v. 11, n. 2, p. 73–78, Maio 2016. Disponível em: <https://doi.org/10.1109/RITA.2016.2554000>. Citado na página 31.

ROESLER, D. When a bug is not a bug: An introduction to the computer science academic vocabulary list. **Journal of English for Academic Purposes**, v. 54, p. 1–11, 2021. ISSN 1475-1585. Disponível em: <https://doi.org/10.1016/j.jeap.2021.101044>. Citado nas páginas 84 e 87.

RUIZ-INIESTA, A.; JIMÉNEZ-DÍAZ, G.; GÓMEZ-ALBARRÁN, M. A semantically enriched context-aware OER recommendation strategy and its application to a computer science OER repository. **IEEE Transactions on Education**, v. 57, n. 4, p. 255–260, Nov 2014. Disponível em: <https://doi.org/10.1109/TE.2014.2309554>. Citado nas páginas 44 e 46.

SAMPSON, D. G.; ZERVAS, P.; CHLOROS, G. Ask-lom-at 2.0: A web-based tool for educational metadata authoring of open educational resources. In: **IEEE International Conference on Technology for Education**. 2011, Chennai, Tamil Nadu, India. Proceedings [...]. Washington, D.C. Estados Unidos.: IEEE, 2011. p. 76–80. Disponível em: <https://doi.org/10.1109/T4E.2011.20>. Citado na página 39.

SCHERER, R.; SIDDIQ, F.; VIVEROS, B. The cognitive benefits of learning computer programming: A meta-analysis of transfer effects. **Journal of Educational Psychology**, American Psychological Association, October 2018. Disponível em: <https://psycnet.apa.org/record/2018-52944-001>. Citado na página 42.

SCHERER, R.; SIDDIQ, F.; VIVEROS, B. S. A meta-analysis of teaching and learning computer programming: Effective instructional approaches and conditions. **Computers in Human Behavior**, v. 109, p. 1–18, Agosto 2020. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0747563220301023>. Citado na página 30.

SELLTIZ, C.; JAHODA, M.; DEUTSCH, M.; COOK, S. W. **Research methods in social relations**. Nova York: Henry Holt, 1991. Disponível em: <2>. Citado nas páginas 123 e 124.

SHU-HSIANG, C.; JAITIP, N.; ANA, D. J. From vision to action – a strategic planning process model for open educational resources. **Procedia - Social and Behavioral Sciences**, v. 174, p. 3707 – 3714, Fevereiro 2015. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1877042815011623>. Citado na página 29.

SORVA, J.; KARAVIRTA, V.; MALMI, L. A review of generic program visualization systems for introductory programming education. **Transactions on Computing Education**, v. 13, n. 4, p. 15:1–15:64, Novembro 2013. Disponível em: <http://doi.org/10.1145/2490822>. Citado na página 41.

SUCUNUTA, M.; RIOFRIO, G.; TOVAR, E. Information retrieval model for open educational resources. In: **2019 IEEE Global Engineering Education Conference (EDUCON)**. [s.n.], 2019. p. 1255–1261. Disponível em: <https://doi.org/10.1109/EDUCON.2019.8725125>. Citado na página 38.

TEW, A. E.; GUZDIAL, M. Developing a validated assessment of fundamental cs1 concepts. In: **Proceedings of the 41st ACM Technical Symposium on Computer Science Education**. Milwaukee, Wisconsin, USA. Proceedings [...]. New York, NY, USA: Association for Computing Machinery, 2010. p. 97–101. Disponível em: <https://doi.org/10.1145/1734263.1734297>. Citado nas páginas 39, 40 e 82.

TODORINOVA, L.; WILKINSON, Z. T. Incentivizing faculty for open educational resources (oer) adoption and open textbook authoring. **The Journal of Academic Librarianship**, v. 46,

n. 6, p. 102220, Dezembro 2020. ISSN 0099-1333. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0099133320301117>. Citado na página 29.

TOVAR, E.; CHAN, H.; REISMAN, S. Promoting merlot communities based on oers in computer science and information systems. In: **Proceedings of the 41st Annual Computer Software and Applications Conference**. Turin, Italy. Proceedings [...]: IEEE, 2017. p. 700–706. Disponível em: <https://doi.org/10.1109/COMPSAC.2017.290>. Citado nas páginas 30, 31, 45 e 46.

UNESCO. **Forum on the Impact of Open Courseware for Higher Education in Developing Countries: Final report**. Paris: UNESCO, 2002. Final Report. Disponível em: <https://unesdoc.unesco.org/ark:/48223/pf0000128515>. Citado nas páginas 29 e 35.

____. **Recommendation on Open Educational Resources (OER)**. Paris: UNESCO, 2019. UNESCO's General Conference (40th session). Disponível em: <https://unesdoc.unesco.org/ark:/48223/pf0000373755/PDF/373755eng.pdf.multi.page=3>. Citado nas páginas 29, 31, 32, 35, 36 e 139.

VALIENTE, M.-C.; SICILIA, M.-A.; GARCIA-BARRIOCANAL, E.; RAJABI, E. Adopting the metadata approach to improve the search and analysis of educational resources for online learning. **Computers in Human Behavior**, v. 51, p. 1134 – 1141, 2015. Computing for Human Learning, Behaviour and Collaboration in the Social and Mobile Networks Era. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0747563215001442>. Citado na página 38.

WAITE, J.; CURZON, P.; MARSH, W.; SENTANCE, S. Difficulties with design: The challenges of teaching design in k-5 programming. **Computers Education**, Elsevier, v. 150, p. 103838, 2020. ISSN 0360-1315. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0360131520300385>. Citado na página 42.

WANG, J.; LIANG, S.-l.; GE, G.-c. Establishment of a medical academic word list. **English for Specific Purposes**, v. 27, n. 4, p. 442–458, 2008. Disponível em: <https://doi.org/10.1016/j.esp.2008.05.003>. Citado nas páginas 84 e 86.

WILEY, D. **The Access Compromise and the 5th R**. Open Content Blog, 2014. The Access Compromise and the 5th R. Digital post. Disponível em: <https://opencontent.org/blog/archives/3221>. Citado na página 36.

WILEY, D.; BLISS, T. J.; MCEWEN, M. Open educational resources: A review of the literature. In: **Handbook of Research on Educational Commun. and Technology**. New York: Springer, 2014. p. 781–789. Disponível em: <https://doi.org/10.1007/978-1-4614-3185-5_63>. Citado nas páginas 31 e 38.

WOOD, L.; HORS, A. L.; APPARAO, V.; BYRNE, S.; CHAMPION, M.; ISAACS, S.; JACOBS, I.; NICOL, G.; ROBIE, J.; SUTOR, R. *et al.* **Document object model (DOM) level 1 specification**. online: W3C recommendation, 1998. Document object model (DOM) level 1 specification. Disponível em: <https://www.w3.org/TR/1998/REC-DOM-Level-1-19981001/>. Citado nas páginas 19, 89 e 90.

YIN, R. K. **Estudo de Caso: Planejamento e métodos - Quarta Edição**. Porto Alegre, Brazil: Bookman editora, 2010. ISBN 978-1-4129-6099-1. Citado nas páginas 97, 98, 99 e 103.

APÊNDICE

A

# TABELAS COMPLEMENTARES

Este Apêndice apresenta duas tabelas complementares para a leitura e análise da tese.

A Tabela 24 apresenta a lista dos metadados identificados nas diferentes plataformas, conforme apresentado no Capítulo 3. Alguns caracteres dos metadados não puderam ser processados. Por isso, o sinal “¿” foi usado para destacar tais ocorrências.

A Tabela 25 lista os artigos de revisão e mapeamento sistemáticos que foram selecionados para montagem da IPAVL, conforme apresentado no Capítulo 5.

Tabela 24 – Metadados identificados em cada plataforma analisada

| Plataforma | Metadados |
|---|---|
| iOER | Access Rights, Accessibility Control, Accessibility Feature, Accessibility Hazard, Assessment Type, Created on, Creator, description, Disability Topic, Education Levels, Educational Use, End User, Explore Careers, Group Type, Guidance Scenarios, ¿title, Illinois Pathways, Jobs, keywords, Language, Layoff Assistance, Media Type, Network & Connect, Publisher, Qualify for Jobs, Region, Resource Type, Resources, Rights, Subject, Submitter, title, Training & Credentials, Usage Rights, WDQI, WIOA Works, Workforce and Education Partner, workNet Areas |
| LivreSaber | citation_abstract_html_url, citation_authors, citation_date, citation_isbn, citation_keywords, citation_language, citation_pdf_url, citation_title, DC.creator, DC.description, DC.identifier, DC.language, DC.source, DC.subject, DC.title, DC.type, DCTERMS.abstract, DCTERMS.available, DCTERMS.dateAccepted, DCTERMS.issued, DCTERMS.temporal |
| Merlot | abstract, Accessibility Information Available, Author, Cost Involved, Creative Commons, Date Added to MERLOT, Date Modified in MERLOT, Disciplines, ¿title, Keyword, Keywords, Language, Languages, Material Type, Mobile Compatibility, Primary Audience, Source Code Available, Submitter, Technical Format, Technical Requirements, title |
| Metafinder | abstract, additionalFields_name, additionalFields_value, authors_type, authors_value, collectionCode, collectionSequence, dates_original, dates_type, dates_value, dedupOrder, deepRank, deepRankUrl, documentFormat, documentType, doi, duplicates_abstract, duplicates_additionalFields_name, duplicates_additionalFields_value, duplicates_authors_type, duplicates_authors_value, duplicates_collectionCode, duplicates_collectionSequence, duplicates_dates_original, duplicates_dates_type, duplicates_dates_value, duplicates_dedupOrder, duplicates_deepRank, duplicates_deepRankUrl, duplicates_documentFormat, duplicates_documentType, duplicates_doi, duplicates_filters, duplicates_fullTextSnippet, duplicates_fullTextUrl, duplicates_imageUrl, duplicates_isbn, duplicates_issn, duplicates_issue, duplicates_keywords, duplicates_language, duplicates_libraryRecord, duplicates_metaDataUrl, duplicates_metaRank, duplicates_pages, duplicates_peerReviewed, duplicates_quickRank, duplicates_rank, duplicates_resultId, duplicates_searchId, duplicates_snippet, duplicates_sources_type, duplicates_sources_value, duplicates_title, duplicates_url, duplicates_volume, filters, fullTextSnippet, fullTextUrl, imageUrl, isbn, issn, issue, keywords, Language, libraryRecord, metaDataUrl, metaRank, pages, peerReviewed, quickRank, rank, resultId, searchId, snippet, sources_type, sources_value, title, url, volume |
| OER Commons | Author, Curriculum Standards, Date Added, Description, Grades, Keyword, Language, Level, License, Material Type, Media Format, Provider Set, Provider, Remix of, Subject, Technical Requirements, title |
| Skills Commons | abstract, Accessibility Evaluation Report, Accessibility Features, Accessibility Statement, Additional License(s), Additional Public Access To Materials, Bachelors Degree, Contrast Ratio of at Least 4.5, Copyright Owner, Course Note, Credential Type, Credential, Credit Type, Custom Quality Rubric, Date Quality Peer Review Rating Submitted, Date, Derivative Work from Other's Materials, Educational Level of Materials, Extension, Formal Accessibility Policy, Industry Partner, Industry Sector, Institution, Interactivity Type, Language, Number of courses in the program, Number of weeks per course, Occupation, Other Material Types, Primary License, Primary Material Type, Program Delivery Format, Project Name, Quality Assurance Organization, Quality Note, Quality of Online/Hybrid Course Design assured by, Quality of Subject Matter was assured by, SkillsCommons' Material Reused, Subjects, TAACCCT Round, Time Required, title |
| RA | Aparece nas colecções, authorProfile.affiliation, dc.contributor, dc.contributor.author, dc.contributor.editor, dc.contributor.other, dc.date.accessioned, dc.date.available, dc.date.issued, dc.description, dc.description.abstract, dc.description.sponsorship, dc.description.version, dc.format, dc.format.extent, dc.format.mimetype, dc.identifier.citation, dc.identifier.isbn, dc.identifier.issn, dc.identifier.other, dc.identifier.uri, dc.language.iso, dc.peerreviewed, dc.publisher, dc.relation, dc.relation.ispartof, dc.relation.isversionof, dc.relation.publisherversion, dc.relation.requires, dc.relation.uri, dc.rights, dc.rights.license, dc.rights.uri, dc.subject, dc.subject.mesh, dc.subject.ods, dc.title, dc.title.alternative, dc.type, dcterms.educationLevel, dcterms.type, degois.publication.firstPage, degois.publication.issue, degois.publication.lastPage, degois.publication.location, degois.publication.title, thesis.degree.level |
| Temoa | ?title, Approval status, Audience education level, Author(s), Content language, description, Description of educational resources provided, Genre, Granularity, Media type, Popular tags for this resource, Provider, Subject general, Subject keywords, Subjects general, title, Year created |

Fonte: Dados da Pesquisa.

Tabela 25 – Lista de artigos selecionados durante a revisão terciária da literatura

| **Título** | **Id** |
|---|---|
| A Review of Introductory Programming Research 2003-2017 | S01 |
| A Systematic Literature Review on Teaching and Learning Introductory Programming in Higher Education | S02 |
| A Systematic Review of Approaches for Teaching Introductory Programming and Their Influence on Success | S04 |
| A Systematic Review of Introductory Programming Languages for Novice Learners | S05 |
| Blended learning models for introductory programming courses: A systematic review | S06 |
| Computer-supported Collaborative Learning in Programming Education: A Systematic Literature Review | S07 |
| Curriculum, Teaching and Learning, and Assessments for Introductory Programming Course | S08 |
| Effective digital contents for computer programming learning: A systematic literature review | S09 |
| Emotions and programming learning: systematic mapping | S10 |
| Enhancing Students' Ability in Learning Process of Programming Language using Adaptive Learning Systems: A Literature Review | S11 |
| Failure Rates in Introductory Programming Revisited | S12 |
| Flipped Classroom In Programming Course: A Systematic Literature Review | S13 |
| Game elements for learning programming: A mapping study | S14 |
| Intelligent Tutoring Systems for Programming Education: A Systematic Review | S15 |
| Learning styles in programming education: A systematic mapping study | S16 |
| Online Tools to Support Novice Programming: A Systematic Review | S17 |
| Pedagogy of teaching introductory text-based programming in terms of computational thinking concepts and practices | S18 |
| Plagiarism in programming assessments: A systematic review | S19 |
| Regulation of Learning Interventions in Programming Education: A Systematic Literature Review and Guideline Proposition | S20 |
| Systematic literature review: Teaching novices programming using robots | S21 |
| The Role of Anxiety When Learning to Program: A Systematic Review of the Literature | S22 |
| The use of games on the teaching of programming: A systematic review | S23 |
| Empirical Evidence of the Usage of Programming Languages in the Educational Process | S24 |
| An Initial Analysis of the Research on Interest and Introductory Programming A Systematic Review of this Literature | S25 |

Fonte: Dados da Pesquisa.


ICMC
USP
SÃO CARLOS