



UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE TECNOLOGIA



HUMBERTO AUGUSTO PIOVESANA ZANETTI

MÉTODO DE ENSINO DE PROGRAMAÇÃO ORIENTADA A OBJETOS BASEADO EM
APRENDIZAGEM SIGNIFICATIVA E COMPUTAÇÃO FÍSICA

LIMEIRA
2022



UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE TECNOLOGIA



HUMBERTO AUGUSTO PIOVESANA ZANETTI

MÉTODO DE ENSINO DE PROGRAMAÇÃO ORIENTADA A OBJETOS BASEADO EM
APRENDIZAGEM SIGNIFICATIVA E COMPUTAÇÃO FÍSICA

Tese de Doutorado apresentada ao
Programa de Pós-Graduação em Tecnologia
da Faculdade de Tecnologia da
Universidade Estadual de Campinas para
obtenção do título de Doutor em
Tecnologia, na área de Sistemas de
Informação e Comunicação.

Orientador: Prof. Dr. Marcos Augusto Francisco Borges

Coorientador: Prof. Dr. Ivan Luiz Marques Ricarte

ESTE EXEMPLAR CORRESPONDE À
VERSÃO FINAL DA TESE DEFENDIDA
PELO ALUNO HUMBERTO AUGUSTO
PIOVESANA ZANETTI E ORIENTADO
PELO PROF. DR. MARCOS AUGUSTO
FRANCISCO BORGES.

LIMEIRA
2022

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Faculdade de Tecnologia
Felipe de Souza Bueno - CRB 8/8577

Z16m Zanetti, Humberto Augusto Piovesana Zanetti, 1983-
Método de ensino de programação orientada a objetos baseado em
aprendizagem significativa e computação física / Humberto Augusto Piovesana
Zanetti. – Limeira, SP : [s.n.], 2022.

Orientador: Marcos Augusto Francisco Borges.
Coorientador: Ivan Luiz Marques Ricarte.
Tese (doutorado) – Universidade Estadual de Campinas, Faculdade de
Tecnologia.

1. Ensino - Metodologia. 2. Programação orientada a objetos
(Computação). 3. Aprendizagem significativa. 4. Computação física. I. Borges,
Marcos Augusto Francisco, 1971-. II. Ricarte, Ivan Luiz Marques, 1962-. III.
Universidade Estadual de Campinas. Faculdade de Tecnologia. IV. Título.

Informações Complementares

Título em outro idioma: Method of object-oriented programming teaching based on
meaningful learning and physical computing

Palavras-chave em inglês:

Teaching - Methodology

Object-oriented programming (Computer science)

Meaningful learning

Physical computing

Área de concentração: Sistemas de Informação e Comunicação

Titulação: Doutor em Tecnologia

Banca examinadora:

Marcos Augusto Francisco Borges [Orientador]

Ana Francisca da Cunha Monteiro

Rodrigo Bonacin

Osvaldo Luiz de Oliveira

Vitor Rafael Coluci

Data de defesa: 24-10-2022

Programa de Pós-Graduação: Tecnologia

Identificação e informações acadêmicas do(a) aluno(a)

- ORCID do autor: <https://orcid.org/0000-0001-6108-148X>

- Currículo Lattes do autor: <https://lattes.cnpq.br/5769042434186463>

FICHA DE APROVAÇÃO

Abaixo se apresentam os membros da comissão julgadora da sessão pública de defesa de dissertação para o Título de Doutor em Tecnologia na área de concentração de Sistemas de Informação e Comunicação, a que submeteu o aluno Humberto Augusto Piovesana Zanetti, em 24 de outubro de 2022 na Faculdade de Tecnologia- FT/ UNICAMP, em Limeira/SP.

Prof. Dr. Marcos Augusto Francisco Borges

Presidente da Comissão Julgadora

Profa. Dra. Ana Francisca da Cunha Monteiro

IPCA (Instituto Politécnico do Cávado e do Ave - Portugal)

Prof. Dr. Rodrigo Bonacin

UNIFACCAMP (Centro Universitário de Campo Limpo Paulista)

Prof. Dr. Osvaldo Luiz de Oliveira

UNIFACCAMP (Centro Universitário de Campo Limpo Paulista)

Prof. Dr. Vitor Rafael Coluci

Faculdade de Tecnologia – FT/UNICAMP

Ata da defesa, assinada pelos membros da Comissão Examinadora, consta no SIGA/Sistema de Fluxo de Dissertação/Tese e na Secretaria de Pós-graduação da FT.

DEDICATÓRIA

Dedico esta tese aos meus pais Alberto e Celia, pelo seu amor e seus sacrifícios para que eu e meus irmão tivéssemos as oportunidades que eles não tiveram,

à minha esposa Flavia, que me apoiou em toda a minha jornada, sempre ao meu lado mesmo nos momentos mais difíceis

e à minha filha Betina, a luz que me guia.

AGRADECIMENTOS

Agradeço primeiramente ao meu orientador Prof. Dr. Marcos Augusto Francisco Borges, por toda confiança, apoio e parceria durante todo meu doutorado, pois sem ele esta pesquisa não seria possível. Também ao meu coorientador, Prof. Dr. Ivan Luiz Marques Ricarte, por toda dedicação, ajuda e compromisso, que foram essenciais para que eu pudesse alcançar o sucesso nesta tese. Foi uma honra e privilégio ter sido orientado por vocês.

Agradeço a todos meus colegas da Escola Técnica (ETEC) Rosa Perrone Scavone e da Faculdade de Tecnologia (FATEC) Deputado Ary Fossen. Agradecimento especial também aos meus alunos, não somente aqueles que participaram deste estudo, mas a todos aqueles que de alguma forma fizeram parte da minha carreira como professor. Por eles sigo em frente nessa profissão.

Por fim, agradeço a Universidade Estadual de Campinas (UNICAMP) e Faculdade de Tecnologia (FT), incluindo todos os professores, funcionários e colegas que conheci durante o período do doutorado. Também agradeço a todos os colegas e amigos que fiz no Laboratório de Informática, Aprendizado e Gestão (LIAG).

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

“Que você sempre se lembre de aproveitar a estrada, especialmente quando for difícil.”

Kobe Bryant

RESUMO

Um dos maiores desafios no ensino em cursos da área de Computação são disciplinas que abordam Programação Orientada a Objetos, especialmente para aquelas direcionadas à alunos iniciantes. Mesmo sendo um paradigma amplamente utilizado e abordado na formação profissional, o professor ainda encontra dificuldade em adotar estratégias e ferramentas que sejam efetivas para o aprendizado. Em disciplinas introdutórias, é importante estimular e amparar o exercício de abstração por parte dos alunos, abstração que é fundamental para a aprendizagem desse conteúdo. Novatos possuem dificuldades em conceitos fundamentais, como, por exemplo, a relação entre classe e objetos, muitas vezes devidas à uma abordagem didática que não consegue trazer tais conceitos para um ambiente que seja mais significativo ou que facilite sua compreensão. Esta tese propôs um método de ensino de Programação Orientada a Objetos que busca contribuir para o exercício de abstração dos alunos, da conceituação inicial até a aplicação e desenvolvimento de código-fonte, abordando os principais conceitos desse paradigma de programação. O método ComFAPOO (Computação Física para Aprendizagem de Programação Orientada a Objetos), baseado na teoria da Aprendizagem Significativa e nos artefatos da Computação Física, tem como estratégia a transição gradual, utilizando a técnica *Concreteness Fading*, que parte de modelos concretos, mais tangíveis e representativos, para modelos mais abstratos, como o código-fonte. Para a análise da eficácia do método, foram realizadas oficinas com 80 alunos de ensino técnico e superior, gerando resultados qualitativos e quantitativos. Foi possível identificar nesses resultados aspectos que comprovam as potencialidades do método no ensino de Programação Orientada a Objetos, assim como aspectos que podem ser melhorados e complementados em trabalhos futuros. O método ComFAPOO traz contribuições para área de pesquisa por propor uma solução que alinha a Aprendizagem Significativa e a Computação Física de maneira inédita, através de uma solução aplicável em sala de aula, a partir dos materiais didáticos desenvolvidos para este trabalho.

Palavras-chave: Método de Ensino, Programação Orientada a Objetos, Aprendizagem Significativa, Computação Física.

ABSTRACT

One of the biggest educational challenges in Computing courses is teaching subjects which address Object-Oriented Programming, especially for beginners. Even though this paradigm is widely used and taught in professional education, teachers still struggle to find and adopt effective learning strategies and tools. Considering that abstraction is crucial for learning this content, it is within reason to recognize how necessary it is for introductory subjects to stimulate and support the exercise of abstraction by students early on. Beginners present difficulties regarding fundamental concepts, such as, the relationship between classes and objects. Said difficulty is oftentimes the result of didactic approaches which fail to bring those concepts into a more meaningful environment that could facilitate their understanding. This thesis proposes an Object-Oriented Programming teaching method which seeks to contribute to the exercise of student abstraction, from initial conceptualization to the development and application of source codes and encompassing this programming paradigm's main concepts. The ComFAPOO Method (physical computing for Object-Oriented Programming learning) is based on David Ausubel's Theory of Meaningful Learning and on the artifacts of Physical Computing. Its strategy resides in a gradual transition through the usage of the Concreteness Fading technique, which departs from more concrete, tangible and representative models to more abstract ones, such as the source code. For the evaluation of the method's efficiency, workshops were held with 80 technical and higher education students, providing both quantitative and qualitative results for the analysis. The findings showed aspects that attest to the potentiality of the method in the teaching of Object-Oriented Programming, as well as aspects which may be improved and complemented in future works. The ComFAPOO Method brings contributions to the research field by proposing a solution which aligns Ausubel's Theory of Meaningful Learning and Physical Computing in an unprecedented way, through the means of creating a suitable solution for classrooms with the usage of the didactic materials developed for this work.

Keywords: *Teaching Method, Object-Oriented Programming, Meaningful Learning, Physical Computing*

LISTA DE FIGURAS

Figura 1.1 Etapas da pesquisa.....	23
Figura 1.2 Visão geral das etapas de pesquisa e artigos relacionados.....	27
Figura 2.1 Número de publicação entre 2012 e 2015 nos eventos WIE, SBIE e WEI	36
Figura 3.1 Relação dos artigos por ferramenta/plataforma	49
Figura 4.1 Mapa mental das demandas de pesquisa.....	65
Figura 5.1 Circuito simples com Arduino e LED.....	76
Figura 5.2 Exemplo de uma classe LED	77
Figura 5.3 Modelo teórico do <i>Concreteness Fading</i> (adaptado de Fyfe et al. [8])	79
Figura 5.4 Exemplo de uma montagem com LED e botão	82
Figura 5.5 A relação dos estágios do <i>Concreteness Fading</i> e artefatos didáticos	83
Figura 6.1 Os estágios do CoFa de Bruner (adaptado de Fyfe et al. (2014))	91
Figura 6.2 A estrutura do método ComFAPOO	91
Figura 6.3 Exemplo de aplicação do ComFAPOO seguindo o <i>framework</i> CoFa.....	93
Figura 7.1 Etapas do método ComFAPOO [Zanetti, Borges e Ricarte 2021].	100
Figura 7.2 Escalas para os questionários. Adaptado de Soares et al. (2013).....	101
Figura 7.3 A aplicação do <i>framework</i> CoFa. Adaptado de Zanetti e Borges [2021]	103
Figura 7.4 Alunos durante as práticas da oficina.	106
Figura 7.5 Distribuição das respostas das QPCs.	107
Figura 8.1 Modelo hierárquico da <i>diferenciação progressiva e reconciliação integradora</i> , adaptado de Moreira e Mansini (2001)	118
Figura 8.2 Componentes de uma interface física de comunicação com usuário, adaptado de Hartmann et al. (2005).....	119
Figura 8.3 Os estágios do CoFa, adaptado de Fyfe et al. (2014).....	121
Figura 8.4 As etapas do método ComFAPOO	126
Figura 8.5 O alinhamento dos princípios da AS com o modelo CoFa	130
Figura 8.6 Composição da prática didática por meio de ancoragem.....	131
Figura 8.7 Escala utilizada nos questionários. Adaptado de Soares et al. (2003)	133
Figura 8.8 Exemplos de telas do repositório do GitHub.	135
Figura 8.9 Participantes durante as oficinas.	136
Figura 8.10 Diagramas de caixa para o grupo TODOS.....	137
Figura 8.11 Diagramas de caixa para o grupo ET.	138

Figura 8.12 Diagramas de caixa para o grupo ES.	138
Figura 8.13 Distribuição das respostas das QPCs no grupo TODOS.....	139
Figura 8.14 Distribuição das respostas das QPCs no grupo ET.	140
Figura 8.15 Distribuição das respostas das QPCs no grupo ES.	140

LISTA DE TABELAS

Tabela 2.1 Critérios de inclusão e exclusão	33
Tabela 2.2 Formulário de extração	34
Tabela 2.3 Lista de estudos em ordem alfabética por autores	35
Tabela 2.4 Categorias de práticas pedagógicas	37
Tabela 2.5 Relação entre as práticas e os estudos	37
Tabela 2.6 Lista de habilidades	39
Tabela 2.7 Relação entre as habilidades e os estudos.....	40
Tabela 3.1 Critérios de inclusão e exclusão	46
Tabela 3.2 Descrição das categorias e relação dos artigos	47
Tabela 4.1 Resumo do resultado da busca pelos trabalhos.....	57
Tabela 4.2 Critérios de inclusão e exclusão	58
Tabela 4.3 Distribuição dos trabalhos por estratégia de ensino e público-alvo	60
Tabela 7.1 Relação de conceitos e objetos usados durante a oficina.	103
Tabela 7.2 Enunciados das <i>Questões sobre Percepção dos Conceitos</i> (QPC).....	104
Tabela 7.3 Enunciados das Questões sobre a Percepção Emocional (QPE).	105
Tabela 7.4 Respostas às Questões sobre a Percepção Emocional (QPE).....	108
Tabela 8.1 Organização dos conceitos e objetos	129
Tabela 8.2 Questões do conjunto Questões sobre Percepção de Conceitos (QPC).....	132
Tabela 8.3 Questões do conjunto <i>Questões sobre Percepção Emocional</i> (QPE).	133
Tabela 8.4 Respostas do grupo TODOS às <i>Questões sobre a Percepção Emocional</i> (QPE).141	
Tabela 8.5 Respostas do grupo ET às <i>Questões sobre a Percepção Emocional</i> (QPE).....	142
Tabela 8.6 Respostas do grupo ES às <i>Questões sobre a Percepção Emocional</i> (QPE).....	143

LISTA DE ABREVIATURAS E SIGLAS

ACM	<i>Association for Computing Machinery</i>
ADM	Ambiente dinâmico e/ou motivador
AS	Aprendizagem Significativa
CAAE	Certificado de Apresentação de Apreciação Ética
CAPES	Coordenação de Aperfeiçoamento de Pessoal de Nível Superior
CCPP	Compreensão de Conceitos ou Paradigmas de Programação
CD	Computação “Desplugada”
CF	Computação Física
CoFa	<i>Concreteness Fading</i>
ComFAPOO	Computação Física para Aprendizagem Orientada a Objetos
CSTA	<i>Computer Science Teachers Association</i>
DOI	<i>Digital Object Identifier System</i>
EC	Engajamento e Criação
EduComp	Simpósio Brasileiro de Educação em Computação
ETEC	Escola Técnica
FATEC	Faculdade de Tecnologia
FT	Faculdade de Tecnologia
IEEE	<i>Institute of Electrical and Electronic Engineers</i>
ISTE	<i>International Society for Technology in Education</i>
JD	Jogos Digitais
LED	<i>Light-Emitting Diode</i>
LP	Linguagem de Programação
LPV	Linguagem de Programação Visual
NSF	<i>National Science Foundation</i>
PBL	<i>Problem-Based Learning</i>
PC	Pensamento Computacional
POO	Programação Orientada a Objetos
PRISMA	<i>Preferred Reporting Items for Systematic Reviews and Meta-Analyses</i>
QA	Questões Abertas
QP	Questões de Pesquisa
QPC	Questões sobre a Percepção dos Conceitos
QPE	Questões sobre a Percepção Emocional

RBIE	Revista Brasileira de Informática na Educação
RP	Robótica Pedagógica
RSL	Revisão Sistemática da Literatura
SAM	<i>Self-Assessment Manikin</i>
SBC	Sociedade Brasileira de Computação
SBIE	Simpósio Brasileiro de Informática da Educação
UNICAMP	Universidade Estadual de Campinas
VIE	Visualização e Identificação de Erros
WAlgProg	Workshop de Ensino em Pensamento Computacional, Algoritmos e Programação
WEI	Workshop sobre Educação em Computação
WIE	Workshop de Informática na Escola

SUMÁRIO

1 INTRODUÇÃO.....	18
1.1 Problema.....	19
1.2 Objetivos da pesquisa	21
1.3 Método de pesquisa	22
1.4 Organização da tese	23
2 PENSAMENTO COMPUTACIONAL NO ENSINO DE PROGRAMAÇÃO: UMA REVISÃO SISTEMÁTICA DA LITERATURA BRASILEIRA.....	29
2.1 Introdução.....	30
2.2 Contexto da Pesquisa	31
2.2.1 Questões da Pesquisa	32
2.3 Métodos aplicados na Revisão Sistemática	33
2.3.1 Questões da Pesquisa	33
2.3.2 Ameaças à validade da pesquisa	34
2.4 Resultados obtidos e discussões.....	35
2.4.1 Visão geral dos estudos.....	35
2.4.2 Práticas pedagógicas nos estudos (QP1)	36
2.4.2.1 Análise e discussões.....	36
2.4.3 Níveis de escolaridade (QP2).....	37
2.4.3.1 Análise e discussões.....	38
2.4.4 Objetivos alcançados nos estudos (QP3).....	38
2.4.4.1 Análise e discussões.....	39
2.5 Conclusões.....	41
2.6 Referências	42
3 ENSINO DE PROGRAMAÇÃO UTILIZANDO COMPUTAÇÃO FÍSICA: UMA REVISÃO SISTEMÁTICA DA LITERATURA.....	43
3.1 Introdução.....	44
3.2 Metodologia.....	45
3.2.1 Questões de Pesquisa	45
3.2.1.1 Métodos aplicados na Revisão da Literatura	45
3.3 Resultados e discussões	46
3.3.1 Discussão sobre a QP1	46
3.3.2 Discussão sobre a QP2.....	48

3.4 Conclusões.....	50
3.5 Bibliografia.....	50
4 A TEORIA DE APRENDIZAGEM SIGNIFICATIVA NO ENSINO DE PROGRAMAÇÃO: UM MAPEAMENTO SISTEMÁTICO DA LITERATURA.....	53
4.1 Introdução.....	54
4.2 Contexto e objetivo do mapeamento	56
4.2.1 As questões de pesquisa.....	56
4.3 Métodos aplicados na Revisão Sistemática da Literatura	57
4.3.1 Análise temática dos trabalhos.....	58
4.4 Resultados obtidos e discussões.....	61
4.4.1 Os públicos-alvo presentes nos trabalhos (QP1)	61
4.4.2 As estratégias adotadas (QP2).....	62
4.4.3 Discussões gerais e lacunas para pesquisa	63
4.5 Considerações finais	65
4.6 Referências	66
5 POR QUE ESTIMULAR A APRENDIZAGEM SIGNIFICATIVA NO ENSINO DE PROGRAMAÇÃO ORIENTADA A OBJETOS?.....	71
5.1 Introdução.....	72
5.2 A Aprendizagem Significativa	73
5.3 A Aprendizagem Significativa e o ensino de Programação Orientada a Objetos	73
5.4 Desafios na adoção da Aprendizagem Significativa	75
5.5 Computação Física como instrumento didático	76
5.6 A teoria de <i>Concreteness Fading</i> e o ensino de programação	78
5.7 Computação Física e <i>Concreteness Fading</i> para promover a Aprendizagem Significativa	80
5.8 Considerações finais	83
5.9 Referências	84
6 MÉTODO DE ENSINO DE PROGRAMAÇÃO ORIENTADA A OBJETOS BASEADA EM COMPUTAÇÃO FÍSICA, APRENDIZAGEM SIGNIFICATIVA E <i>CONCRETENESS FADING</i> ..	88
6.1 Introdução.....	89
6.2 Referencial teórico.....	89
6.3 O método ComFAPOO	91
6.4 Conclusões.....	94
6.5 Referências	94

7 APLICAÇÃO DE UM MÉTODO PARA ENSINO DE PROGRAMAÇÃO ORIENTADA A OBJETOS POR MEIO DE APRENDIZAGEM SIGNIFICATIVA E COMPUTAÇÃO FÍSICA 96

7.1 Introdução.....	97
7.2 Dificuldades encontradas no ensino de Programação Orientada a Objetos	98
7.3 Definindo os fundamentos da Programação Orientada a Objetos	99
7.4 O método ComFAPOO	100
7.5 A aplicação da oficina e experimentação	105
7.6 Análise dos resultados	106
7.7 Considerações finais	109
7.8 Referências	109

8 COMFAPOO: MÉTODO DE ENSINO DE PROGRAMAÇÃO ORIENTADA À OBJETOS BASEADO EM APRENDIZAGEM SIGNIFICATIVA E COMPUTAÇÃO FÍSICA 113

8.1 Introdução.....	114
8.2 Fundamentação teórica	116
8.2.1 A teoria da Aprendizagem Significativa (AS).....	116
8.2.2 A Computação Física (CF) e suas aplicações	119
8.2.3 O modelo <i>Concreteness Fading</i> (CoFa)	120
8.3 Trabalhos relacionados à temática da pesquisa	121
8.4 O método ComFAPOO	124
8.4.1 Embasamento teórico aplicado ao método ComFAPOO	125
8.4.2 Descrição das etapas	126
8.4.3 Desenvolvimento do material didático e aplicação.....	129
8.4.4 Avaliações	131
8.5 Aplicação das oficinas	134
8.5.1 Descrição dos participantes.....	134
8.5.2 A dinâmica durante a oficina	135
8.6 Análise dos resultados	136
8.6.1 Análise das <i>Questões sobre a Percepção dos Conceitos</i> (QPC)	136
8.6.2 Análise das <i>Questões sobre a Percepção Emocional</i> (QPE).....	141
8.6.3 Análise qualitativa sobre os relatos dos participantes.....	143
8.7 Considerações finais	146
8.7.1 Limitações da pesquisa	146
8.7.2 Trabalhos futuros	147
8.7.3 Contribuições.....	148

8.8 Referências	149
9 DISCUSSÃO.....	154
9.1 Sobre a Revisão da Literatura	154
9.2 Sobre o Desenvolvimento do Método	155
9.3 Sobre a Aplicação das Oficinas e Resultados.....	157
10 CONCLUSÃO	159
REFERÊNCIAS.....	162
APÊNDICE A – MATERIAL DIDÁTICO DAS OFICINAS	167
APÊNDICE B – PLANOS DE AULA.....	204
APÊNDICE C – IMAGENS DAS OFICINAS.....	225
ANEXO A – PARECER DO COMITÊ DE ÉTICA.....	228
ANEXO B – AUTORIZAÇÕES DO USO DOS ARTIGOS	235

Capítulo 1

INTRODUÇÃO

A Programação Orientada a Objetos (POO) é um dos paradigmas de programação mais utilizados atualmente, tanto no mercado de trabalho quanto na academia. A primeira linguagem de programação a implementar esse paradigma foi SIMULA 67, nos anos 1960, que utilizava o conceito de “objeto”, que originou toda a estrutura do formato padrão da POO (NYGAARD, 1986). A linguagem que popularizou a POO nos anos 1970 foi o Smalltalk, sendo uma das principais linguagens a definir as características essenciais do paradigma (STEFIK e BROBROW, 1985). Trabalhos que datam de meados dos anos 1980 mostram que pesquisadores e professores da época já adotavam ou apresentavam argumentos para a adoção do paradigma em suas universidades, defendendo que a POO seria o futuro do desenvolvimento de software (RENTSCH, 1982; NYGAARD, 1986; KEMPF e STELZNER, 1987).

Com a crescente adoção do paradigma em disciplinas dos cursos de computação a partir dos anos 1980, surgiu a necessidade de discutir melhores práticas de ensino. Beck e Cunningham (1989) apresentam em seu trabalho a dificuldade em introduzir o novo paradigma a programadores novatos, em especial aqueles que já conhecem o paradigma estruturado. A principal dificuldade identificada pelos autores é a nova “forma de pensar” que o paradigma demanda. Knudsen e Madsen (1988) apontam que o ensino de POO não deve apenas centrar-se nas questões técnicas de uma linguagem de programação específica, mas sim em seus conceitos fundamentais do paradigma e aplicações.

As disciplinas introdutórias de programação usualmente fazem parte das primeiras dificuldades que alunos encontram no início de sua formação (BENNEDSEN e CASPERSEN, 2007). Com relação ao paradigma POO, esse cenário pode ainda se agravar, ao se considerar que, quando é abordado esse paradigma, os alunos podem ter dúvida e incertezas de disciplinas anteriores, como as que envolvem lógica de programação e programação estruturada (ROBINS, ROUNTREE e ROUNTREE., 2003; XINO GALOS, 2015). Atualmente, as dificuldades no ensino de POO ainda persistem, sendo um desafio para os educadores apresentar os conceitos fundamentais do paradigma para os iniciantes. Xinogalos (2015) aponta em sua pesquisa que

uma das grandes barreiras para os novatos está na relação entre os conceitos de “classe” e “objeto”, sendo algo impeditivo para o andamento de qualquer currículo de programação.

A habilidade de abstração é recomendada para alunos que estejam tendo o primeiro contato com POO, mas, muitas vezes, a capacidade de abstração e generalização do indivíduo ainda não é adequada naquele momento do curso (GOMES e MENDES, 2007; GOMES e MENDES, 2014). Segundo Or-Bach e Lavy (2004), programar no paradigma orientado a objetos não é uma habilidade técnica isolada, mas sim uma atividade cognitiva complexa que, quando apresentada aos iniciantes em Computação, pode trazer várias dificuldades no processo de aprendizagem e formação do aluno. Normalmente, o primeiro contato do aluno com o paradigma ocorre em um momento no qual o aluno precisa ter uma habilidade de abstração desenvolvida, para que possa compreender até os conceitos mais simples (HADAR, 2013).

Com esse cenário em vista, esta pesquisa apresenta uma proposta de método que objetiva auxiliar o processo de aprendizagem do aluno, por meio de práticas que estimulem e apoiem a habilidade do aluno em abstrair conceitos relacionados ao paradigma POO. O método, chamado de ComFAPOO (Computação Física para Aprendizagem Orientada a Objetos), fundamenta-se na teoria da Aprendizagem Significativa (AS) e utiliza recursos da Computação Física (CF).

1.1 Problema

Esta tese foi orientada por problemas de pesquisa que delimitaram toda sua investigação. Segundo Vilaça (2010), um problema de pesquisa se apresenta como questionamento sobre um determinado tema que direciona a pesquisa, podendo orientar toda a sistematização de procedimentos de estudo e de investigação.

Para Bennedsen e Caspersen (2006) o pensamento abstrato é essencial para a aprendizagem de computação, em especial programação, sendo requerido nas disciplinas, mas pouco estimulado durante as aulas. Metáforas e exemplos utilizados durante as aulas muitas vezes são inadequados, pois não representam o cotidiano do aluno ou se apoiam em conceitos que eles não possuem ou dominam (SAJANIEMI, BYCKLING e GERDT, 2007; LIBERMAN, BEERI e KOLIKANT, 2011).

O ensino de POO apresenta desafios relacionados à composição de aulas e práticas de programação que possam incentivar e motivar os alunos. Xinogalos (2015) mostra que ainda há a necessidade de estudar e apresentar propostas didáticas para o ensino de conceitos

fundamentais do paradigma orientado à objetos. Trabalhos como de Aureliano, Tedesco e Giraffa (2016) e Kaila et al. (2016) reforçam a necessidade em pesquisas que proponham atividades e tecnologias que possam auxiliar os alunos durante o processo de aprendizagem.

Por meio da revisão da literatura realizada ao longo desta pesquisa, foi possível observar que, no cenário atual de trabalhos acadêmicos sobre práticas de ensino de POO, as dificuldades no exercício de abstração por parte dos alunos e a falta de instrumentos que possam auxiliar nesse processo são pontos relevantes e problemas presentes no campo de pesquisa da temática. Além disso, ainda é bastante presente a demanda por novos métodos que possam engajar o aluno no processo de aprendizagem em POO. Sob esse contexto, foram determinados problemas de pesquisa, para nortear e conduzir este trabalho e o desenvolvimento do método ComFAPOO. Tais problemas são:

- **Problema 1: identificar quais recursos podem ajudar no exercício de abstração necessária para o ensino de POO.**

Recursos didáticos que sejam lúdicos, interativos e motivadores e, ao mesmo tempo, que sejam concretos e tangíveis não são comuns no ensino de programação. Tais recursos podem motivar e engajar os alunos, além de apoiar o desenvolvimento da capacidade de abstração deles (WEB, 2017; MICHAELSON, 2018; JAAKKOLA, e VEERMANS, 2020; LIAN, VAROY e GIACAMAN, 2022);

- **Problema 2: definir estratégias didáticas que possam auxiliar na compreensão e correlação entre os conceitos de POO.**

Estratégias de ensino de POO são temas de pesquisas, desde o início da adoção de POO em currículos de formação profissional (ABBASI et al., 2021; LIAN, VAROY e GIACAMAN, 2022). Além das dificuldades associadas ao pensamento abstrato, também são notáveis as dificuldades com a contextualização e correlações entre os conceitos que são apresentados (MÖSTROM et al., 2008; LIBERMAN, BEERI e KOLIKANT, 2011);

- **Problema 3: criar um método instrucional de apresentação do conteúdo didático.**

A apresentação do conteúdo em disciplinas de programação deve priorizar não apenas a exposição do conceito em si ou a linguagem de programação, mas sua aplicação e contribuição em um contexto mais amplo (KNUDSEN e MADSEN, 1988; XINO GALOS, 2015; GUTIÉRREZ, GUERRERO e LÓPEZ-OSPINA, 2022). A compreensão dos conceitos fundamentais e suas aplicações devem

realizadas antes da sua implementação em código-fonte (RAGONIS e BEN-ARI, 2005).

1.2 Objetivos da pesquisa

Com base no cenário do problema de pesquisa apresentado, o principal objetivo desta pesquisa é o desenvolvimento de um método de ensino de POO eficaz para alunos do ensino técnico e superior, que alinhe as premissas fundamentais da teoria da AS com recursos da CF. Este método visa apoiar o educador no processo de ensino, abordando conceitos fundamentais do paradigma POO, e estimular o exercício de abstração por parte do aluno. A questão geral de pesquisa deste trabalho foi “*é viável o desenvolvimento de um método de ensino de Programação Orientado a Objetos baseado em Aprendizagem Significativa e Computação Física que ampare o exercício de abstração do aluno?*”. A seguir, serão discutidos os problemas que nortearam a pesquisa.

Com a relação ao Problema 1, a investigação deste trabalho buscou artefatos que pudessem atender a requisitos de tangibilidade e que pudessem também ser programáveis, ou seja, pudessem ser associados a conceitos de POO. Além disso, fatores como simplicidade na compreensão e baixo custo foram critérios determinantes nessa busca.

Também se buscou atender a demanda por práticas didáticas que priorizem a construção do conhecimento de forma encadeada, conexa e com significado para o aluno, como apresentado no Problema 2. Por isso, este trabalho optou por basear-se em um referencial teórico que privilegiasse a construção do conhecimento por meio da correlação de conceitos.

A proposta do método de ensino deste trabalho buscou apresentar um meio de exposição dos conceitos de POO, iniciando pela compreensão dos conceitos por meio do uso de objetos concretos, mostrando gradativamente como tal conceito é implementado em código-fonte, atendendo às demandas do Problema 3. Essa transição também pode ser percebida por meio do sentido inverso, do modelo mais abstrato até o mais concreto, fortalecendo a percepção do aluno sobre o que foi apresentado.

Em uma visão geral, a solução proposta nesta pesquisa consiste em um método no qual o professor pode, por meio de objetos reais, tangíveis e programáveis (componentes eletrônicos) encontrar um recurso que ajude o aluno no exercício de abstração. O método ComFAPOO objetiva apoiar o aprendiz a compreender do conceito apresentado até sua

implementação em código-fonte, partindo de um objeto concreto para que possa ser analisado e modelado, até que, por fim, possa ser controlado por meio de programação.

Para alcançar o objetivo principal, objetivos específicos traçam o caminho até sua realização. Dentre os objetivos específicos buscados durante esta pesquisa, pode-se destacar:

- Compreensão do cenário a ser explorado: estudo com base na literatura em torno da temática, com a finalidade de entender quais eram as lacunas de pesquisa e possíveis soluções e tecnologias presentes em pesquisas recentes;
- Investigação e aplicação da fundamentação teórica: estudo dos principais referenciais teóricos que fundamentam a pesquisa, investigando quais são as possibilidades tecnológicas e metodológicas a serem aplicadas;
- Proposta de um método de ensino: apresentar uma solução metodológica que possa suprir as demandas identificadas, e que possa ser aplicada em um cenário real de ensino de POO;
- Experimentação e análise: aplicar o método juntamente com o público-alvo, extraíndo informações, tanto para a validação do método, como também para identificação de potenciais correções ou melhorias e trabalhos futuros;
- Produtos de ensino: criação de material didático que possa ser aplicado por educadores que buscam novas práticas e recursos de ensino de POO, tanto no ensino técnico como superior.

1.3 Método de pesquisa

Esta pesquisa se desenvolveu metodologicamente em 3 etapas principais, que podem ser definidas como Revisão da Literatura, Desenvolvimento do Método ComFAPOO e Aplicação das Oficinas. Ao longo dessas etapas, houve o desenvolvimento e/ou publicação de sete artigos que expõem seus resultados (que serão brevemente apresentados na Seção 1.4).

As etapas foram progressivamente construindo a pesquisa apresentada nesta tese, através de resultados preliminares, até alcançar o principal objetivo, que foi a criação do método ComFAPOO e sua validação junto ao público-alvo. A Figura 1.1 apresenta as etapas aplicadas nesta pesquisa.



Figura 1.1 Etapas da pesquisa.

O objetivo principal de cada uma das etapas da pesquisa foi:

- **Revisão da literatura:** nessa etapa foram realizados os mapeamentos da literatura para a identificação de estratégias e recursos que pudessem ajudar o aluno no exercício de abstração e prover ferramentas para o educador com esse fim. Essa etapa atende à demanda colocada pelo Problema 1, que direciona a busca por recursos didáticos para esse fim e que, ainda, fossem engajadores para os alunos;
- **Desenvolvimento do Método:** essa etapa atende à demanda dos Problema 2 e Problema 3, sendo a etapa na qual foram realizadas discussões e estudos para a integração de recursos e estratégias que pudessem amparar o ensino de POO. Fez parte desse processo a definição das dinâmicas e recursos que o método ComFAPOO iria conter;
- **Aplicação das oficinas:** nessa etapa ocorreram todas as ações de aplicações das oficinas para obtenção de dados e posterior análise sobre os resultados. Essa etapa se alinha com o Problema 3, pois com a aplicação é possível analisar não só a eficácia do método, mas também compreender quais aspectos atendem a proposta de ensino e quais podem ser melhorados, para posterior aprimoramento do método.

1.4 Organização da tese

Esta tese está organizada em dez capítulos, sendo formado por sete capítulos relacionados à artigos, além dos capítulos de Introdução, Discussão (Capítulo 9) e Conclusão (Capítulo 10). Cada um dos capítulos representados por artigos, é composto por trabalhos já publicados ou que estão em análise para serem publicados. A seguir serão detalhados os capítulos formados

por artigos (capítulos 2 a 8), suas contribuições e as relações com o desenvolvimento da pesquisa:

- **Capítulo 2: Pensamento Computacional no Ensino de Programação: Uma Revisão Sistemática da Literatura Brasileira.**

Neste capítulo é apresentado uma revisão sistemática, desenvolvida no início da pesquisa de doutorado apresentado nesta tese, fazendo parte da etapa da Revisão da Literatura. O resultado dessa revisão mostrou quais eram as principais linhas de pesquisas que alinhavam Pensamento Computacional e o ensino de programação. Embora Pensamento Computacional não tenha sido aplicado na tese final, esse trabalho mostrou que ainda havia poucos trabalhos que utilizavam recursos como CF e Robótica Pedagógica. Também, como resultado, mostrou que não havia trabalhos direcionados exclusivamente para o ensino de POO.

Este capítulo foi publicado como um artigo completo nos anais do *XXVII Simpósio Brasileiro de Informática da Educação (SBIE 2016)* (ZANETTI, BORGES e RICARTE, 2016).

- **Capítulo 3: Ensino de Programação utilizando Computação Física: uma Revisão Sistemática da Literatura.**

Este capítulo faz parte da etapa de Revisão da Literatura e é apresentado um estudo sobre a literatura atual em torno da adoção da CF no ensino de programação e quais as principais ferramentas utilizadas. Para a pesquisa em geral, este estudo contribui para a identificação das principais motivações para a adoção da CF, apontando os aspectos que os pesquisadores identificam como válido a aplicação de seus recursos. Esta revisão fundamentou a escolha pela CF como sendo o recurso didático utilizado na ComFAPOO e definiu quais aspectos esses recursos poderiam contribuir com o método proposto.

O capítulo foi publicado como um artigo curto na revista *Comunicações em Informática, volume 4, edição 1* (ZANETTI e BORGES, 2020).

- **Capítulo 4: A Teoria de Aprendizagem Significativa no Ensino de Programação: um Mapeamento Sistemático da Literatura**

Este capítulo é composto por um artigo completo que apresenta um mapeamento sistemático da literatura com o objetivo de apontar quais são os públicos-alvo das pesquisas que envolvam AS e quais estratégias são adotadas. Os resultados obtidos desse mapeamento, que faz parte da Revisão da

Literatura, ajudaram a identificar qual é o cenário atual do tema, além de apontar lacunas que poderiam ser exploradas pelo método proposto nesta tese. Com as análises feitas nesse artigo, foi possível identificar que havia uma tendência de orientar as premissas da AS no ensino programação em cursos de formação profissional, como em cursos técnicos e superiores, mesmo público em que o método ComFAPOO aborda. Também foi possível identificar lacunas de pesquisa para o ensino específico de POO e adoção de recursos tangíveis, como os ofertados pela CF.

O artigo que compõe o capítulo 4 foi publicado nos anais do *XXXIII Simpósio Brasileiro de Informática na Educação (SBIE 2022)* (ZANETTI, BORGES e RICARTE, 2022b).

- **Capítulo 5: Por que estimular a Aprendizagem Significativa no ensino de Programação Orientada a Objetos?**

O capítulo apresenta um artigo de opinião (*position paper*), com uma análise crítica-reflexiva sobre a adoção de premissas da AS no ensino de POO. Esse trabalho inicia a etapa de Desenvolvimento do Método e apresenta a concepção primordial do método ComFAPOO, expondo como as bases teóricas que fundamentam o método poderiam viabilizar o ensino do paradigma POO. Quatro aspectos importantes são apresentados com relação ao método que, no momento da escrita do artigo, estava em sua etapa de concepção: i) direcionar o ensino de POO seguindo as premissas apresentadas pela teoria da AS e sua justificativa; ii) alinhar os recursos da CF para criar recursos que pudessem amparar o ensino e o exercício de abstração; iii) adoção do modelo instrucional CoFa, como uma estratégia de aplicação e condução da prática didática e; iv) os desafios na aplicação da AS, justificando e motivando a proposição de um método para esse fim.

O texto do capítulo foi publicado nos anais do *Simpósio Brasileiro de Educação em Computação (EduComp 2021)* (ZANETTI e BORGES, 2021).

- **Capítulo 6: Método de Ensino de Programação Orientada a Objetos Baseada em Computação Física, Aprendizagem Significativa e Concreteness Fading.**

Este capítulo apresenta um artigo curto, sendo a primeira publicação do método ComFAPOO e compõe a etapa de Desenvolvimento do Método. Neste trabalho é mostrado a estrutura essencial do ComFAPOO, com suas etapas definidas e

todo o alinhamento com as premissas da AS, a utilização do ferramental da CF e a aplicação do modelo CoFa, como recurso instrucional.

O artigo foi publicado nos anais do evento *XII Conferência Internacional de Tecnologias de Informação e Comunicação na Educação Challenges 2021: Desafios do Digital*, sediado em Braga, Portugal (ZANETTI, BORGES e RICARTE, 2021).

- **Capítulo 7: Aplicação de um Método para Ensino de Programação Orientada a Objetos por meio de Aprendizagem Significativa e Computação Física.**

Neste artigo é apresentado uma aplicação prévia no método ComFAPOO, apresentando resultados preliminares e compondo a etapa de Aplicação do Método. Essa primeira intervenção possibilitou a compreensão da dinâmica e a eficiência do próprio método, assim como a validação dos questionários aplicados. Como resultado, foi possível notar tanto o potencial do ComFAPOO, como fragilidades a serem corrigidas em trabalhos futuros.

O artigo deste capítulo foi publicado nos anais do *XXX Workshop sobre Educação em Computação – 2022 (WEI 2022)* (ZANETTI, BORGES e RICARTE, 2022a).

- **Capítulo 8: ComFAPOO: Método de Ensino de Programação Orientada a Objetos Baseado em Aprendizagem Significativa e Computação Física**

Este capítulo é composto por um artigo completo, sendo a publicação que apresenta, com maiores detalhes, o método ComFAPOO e sua aplicação. O texto expõe tanto a composição do método, partindo de suas bases teóricas, até a formação do material didático disponibilizado nos *workshops* para os alunos. O texto apresenta, também, uma análise mais completa e detalhada dos resultados obtidos.

O artigo foi aceito para publicação na *Revista Brasileira de Informática na Educação (RBIE)*, em seu 30º volume, no ano de 2022.

A Figura 1.2 ilustra a relação entre os artigos e cada uma das etapas da pesquisa, como descrito na Seção 1.2. A etapa de Revisão da Literatura é composta por três artigos que concentram discussões sobre o mapeamento da literatura. Na etapa de Desenvolvimento do Método, dois artigos descrevem a definição do tema, sendo um artigo de opinião, com uma discussão sobre as bases fundamentais da ComFAPOO, e outro artigo que apresenta a estrutura

básica do método. Os dois artigos que formam a etapa de Aplicação de Oficinas apresentam detalhes sobre a aplicação do ComFAPOO, público-alvo, material didático desenvolvido e análises de resultados.

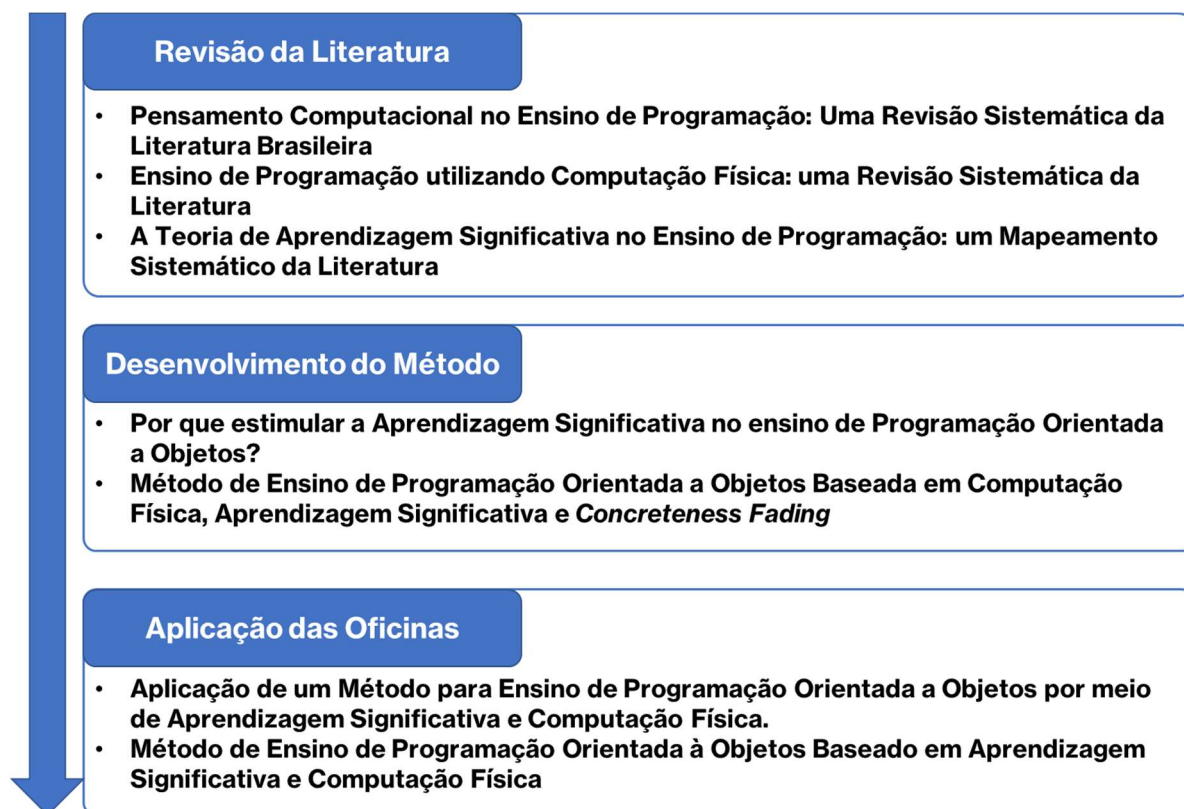


Figura 1.2 Visão geral das etapas de pesquisa e artigos relacionados.

A etapa de Revisão da Literatura foi desenvolvida logo no início do processo de definição de tema de pesquisa, com uma análise de trabalhos sobre Pensamento Computacional, temática que não foi abordada no restante da pesquisa. No entanto, esse estudo preliminar auxiliou na identificação de futuras demandas de pesquisa que foram focalizadas no desenvolvimento do método proposto. Outros mapeamentos da literatura foram realizados e tiveram o objetivo mapear as possíveis demandas de pesquisa e as contribuições em torno do uso de CF e trabalhos que seguissem premissas da AS no contexto de ensino de programação. Os resultados obtidos desses mapeamentos orientaram o direcionamento do método ComFAPOO em relação a todo o embasamento teórico e aplicação prática baseada nesses conceitos.

A etapa de Desenvolvimento do Método iniciou com um estudo preliminar e a construção de uma visão geral do método ComFAPOO. A partir da visão das demandas de pesquisa obtidas pela Revisão da Literatura, foi possível iniciar um estudo mais aprofundado

sobre as possibilidades e meios de alinhar as premissas da AS com o ensino de POO e como utilizar os recursos da CF para promover o aprendizado. Para a primeira versão do método, houve um estudo aprofundado sobre o uso de CF e AS de forma integrada no ensino de programação, além da adição do modelo CoFa, para suprir a ausência de um método instrucional não definido pela AS. Um método instrucional são processos e estratégias que auxiliam na estruturação de cursos ou práticas didáticas (FALCADE et al., 2016). Nessa etapa da pesquisa também houve uma análise de como poderiam ser realizadas práticas didáticas para esse fim, definido a estrutura primária do método ComFAPOO.

Por fim, a etapa de Aplicação de Oficinas ocorre a partir da definição da estrutura do método, e se inicia com a seleção dos conceitos de POO que serão abordados e o desenvolvimento do material didático. Também foram definidas as avaliações que seriam aplicadas para a validação do método. A aplicação com os alunos se iniciou com uma aplicação preliminar para a obtenção dos primeiros resultados. Em seguida, houve uma aplicação com um número maior de participantes, trazendo resultados mais completos e consistentes sobre o método ComFAPOO.

Por fim, esta tese é composta pelos seguintes apêndices e anexos:

- Apêndice A – Material didático das oficinas: apresenta o material didático aplicado nas oficinas e disponibilizado no repositório GitHub;
- Apêndice B – Planos de aula: apresenta os planos de aula desenvolvidos para servir como guia de apresentação por professores;
- Apêndice C – Imagens das oficinas: imagens das dinâmicas que ocorreram durante a aplicação das oficinas;
- Anexo A – Parecer do Comitê de Ética: apresenta o parecer do comitê de ética designado para esta pesquisa;
- Anexo B – Autorizações do uso dos artigos: apresenta os documentos comprobatórios do uso dos artigos que compõe esta tese.

Capítulo 2

PENSAMENTO COMPUTACIONAL NO ENSINO DE PROGRAMAÇÃO: UMA REVISÃO SISTEMÁTICA DA LITERATURA BRASILEIRA

RESUMO

Este artigo tem como objetivo apresentar uma revisão sistemática da literatura dos trabalhos relacionados ao Pensamento Computacional, publicados em eventos brasileiros de 2012 até o ano de 2015. Esta revisão foi realizada a partir de artigos publicados nos três importantes eventos nacionais: Simpósio Brasileiro de Informática na Educação (SBIE), Workshop de Informática na Escola (WIE) e Workshop sobre Educação em Informática (WEI). Também foram utilizados trabalhos publicados no evento Workshop de Ensino em Pensamento Computacional, Algoritmos e Programação (WAlgProg), a fim de investigar quais são as pesquisas mais recentes. Foram analisados 16 artigos, com o objetivo de apresentar uma visão crítica desta área de pesquisa emergente e definir quais as linhas de atuação dos pesquisadores no Brasil.

PUBLICAÇÃO

Artigo publicado nos anais do XXVII Simpósio Brasileiro de Informática da Educação (SBIE 2016). Este texto usa as normas e formatos de referências estabelecidos pela Sociedade Brasileira de Computação (SBC) para publicações de artigos. As numerações das seções, figuras e tabelas foram adaptadas do texto original para dar sequência à numeração desta tese.

2.1 Introdução

O ensino de programação de computadores não é mais uma prática exclusiva em cursos da área da Computação e de Engenharias. Em um mercado competitivo como o que temos atualmente, engenheiros, economistas e cientistas, por exemplo, devem saber utilizar ferramentas e métodos para criar modelos computacionais [Blikstein 2008]. Wing (2006) afirma que o Pensamento Computacional (PC) deve ser uma habilidade básica a ser ensinada às crianças, assim como ler, escrever e aritmética. É possível encontrar práticas de ensino de programação dentro das escolas do ensino fundamental e médio, sendo aplicadas como atividade extracurricular. Essas práticas podem desenvolver habilidades para auxiliar os alunos na resolução de problemas, ser um elemento ativo na construção do conhecimento e entender o contexto tecnológico em que vivemos.

PC, segundo Wing (2006), é um método para solucionar problemas, conceber sistemas e compreender o comportamento humano inspirado em conceitos da Ciência da Computação. Blikstein (2008) afirma que PC é saber usar o computador como instrumento do poder cognitivo e operacional humano, a fim de aumentar nossa produtividade, inventividade e criatividade. O trabalho de Mestre et al. (2015) justifica que as habilidades estimuladas pelo PC estão diretamente relacionadas à resolução de problemas, pois envolvem a capacidade de compreender as situações propostas e criar soluções através de modelos matemáticos, científicos ou sociais.

O desenvolvimento do PC em alunos por meio do ensino de programação demonstra resultados positivos. Brennan e Resnick (2012) mostram que uma abordagem baseada em PC pode ser utilizada no aprendizado de programação, principalmente para jovens e usando ferramentas que motivem e estimulem a criatividade. Para Wing (2006), PC é baseado em abstração e decomposição de problemas ou sistemas complexos, habilidades altamente recomendadas para alunos de programação.

Este artigo tem como objetivo principal apresentar uma visão sobre a pesquisa relacionada a PC no Brasil nos últimos anos. Para isso, foram considerados somente estudos direcionados às práticas educacionais que estimulem ou envolvam o PC, com o objetivo de ensinar programação de computadores, ou que utilizem métodos e ferramentas que usem

conceitos de programação. Este trabalho é parte de uma pesquisa mais ampla, que investiga a abrangência na adoção do PC no ensino de programação, e visa contribuir para pesquisas futuras, mostrando onde os esforços da comunidade estão focados nessa linha de pesquisa.

O presente trabalho apresenta resultados de uma Revisão Sistemática da Literatura (RSL), assim como o processo de desenvolvimento, mostrando os protocolos da pesquisa até chegar aos resultados finais. A Seção 2.2 descreve o contexto da pesquisa e a Seção 2.3 explica os métodos utilizados na revisão sistemática. A Seção 2.4 apresenta a síntese dos resultados obtidos e discussões sobre os resultados. Por fim, a Seção 2.5 apresenta as conclusões da pesquisa.

2.2 Contexto da Pesquisa

As pesquisas relacionadas ao PC vêm crescendo, tanto no Brasil como no exterior. Desde a criação do termo “*Computational Thinking*” (Pensamento Computacional, em inglês) pela pesquisadora Jeannette Wing em 2006, vários pesquisadores cujo foco da pesquisa é o ensino de programação, veem o PC como uma estratégia de grande valia. Trabalhos como os de Cuny, Snyder e Wing (2010), Barr e Stephenson (2011) e Brennan e Resnick (2012) mostram a eficácia da adoção do PC como método de sistematizar o pensamento na formulação de soluções, de maneira incremental e recursiva.

Pensar computacionalmente não é sinônimo de “programar”. No entanto, Blikstein (2008) aponta que umas das etapas fundamentais do PC é saber programar um computador para realizar tarefas cognitivas e de maneira automatiza, para que este conhecimento seja um suporte ao raciocínio humano. De Paula, Valente e Burn (2014) afirmam que o PC é uma maneira específica de se pensar e de analisar uma situação ou um artefato, sendo independente do uso de tecnologia, mas argumentam que as ideias teóricas, que são a base do conhecimento, podem ser aliadas aos aspectos práticos da programação. Nesse contexto, se faz relevante uma revisão sistemática que possa indicar como a pesquisa brasileira avança no uso de PC no ensino de programação.

2.2.1 Questões da Pesquisa

O objetivo desta RSL é identificar e avaliar os estudos publicados nos principais eventos e revistas brasileiras associadas à Sociedade Brasileira de Computação (SBC), que apresentem alguma contribuição no ensino de programação de computadores e PC. Com esse objetivo em vista, pretende-se responder à questão principal desta pesquisa: “*O que se tem feito nas pesquisas realizadas no Brasil sobre Pensamento Computacional no contexto de ensino de programação?*”.

Com base na questão principal apresentada, outras três questões de pesquisa (QP) mais específicas foram definidas:

- QP1: *Quais são as práticas pedagógicas mais realizadas?*
- QP2: *Em quais níveis de escolaridade as pesquisas estão sendo feitas?*
- QP3: *Quais são os objetivos a serem alcançados com essas práticas?*

A QP1 serve para mapear quais são as práticas (ações) mais comuns realizadas em pesquisas no Brasil, mostrando também quais são os principais instrumentos (tecnologias e ferramentas) utilizados. Na QP2, o objetivo é identificar quais são os níveis de escolaridade (ensino fundamental, médio ou superior) nos quais estão ocorrendo as pesquisas, traçando quais são as tendências em cada nível. Por fim, a QP3 delimita quais são os objetivos definidos nas práticas realizadas nas pesquisas. Objetivo, nesse contexto, representa o propósito ou o que se pretende alcançar na realização da prática.

A resposta à QP3 resulta na identificação de quais são os objetivos que os pesquisadores desejam alcançar com o PC, podendo trazer informações relevantes para definir uma tendência da pesquisa nacional e relacionar com o cenário mundial. Para a pesquisa deste artigo, os objetivos serão categorizados segundo o trabalho de Barr e Stephenson (2011), que define nove habilidades fundamentais no processo de PC e na resolução de problemas. Essas habilidades foram especificadas a partir das premissas definidas pela *International Society for Technology in Education* (ISTE)¹, *Computer Science Teachers Association* (CSTA)² e *National Science Foundation* (NSF)³ [ISTE, 2016]. O PC não se limita exclusivamente a essas habilidades e nem todas essas devem ser abordadas para caracterizar o processo de PC. Os artigos analisados pela revisão sistemática satisfazem um ou mais desses critérios, através dos

¹ ISTE - <http://www.iste.org/>

² CSTA - <http://www.csta.acm.org/>

³ NSF - <http://www.nsf.gov/>

resultados obtidos na pesquisa descrita. Com isso, será possível sumarizar os objetivos definidos e alcançados pelos trabalhos e a relação direta com as premissas no uso de PC.

2.3 Métodos aplicados na Revisão Sistemática

A revisão bibliográfica foi realizada seguindo algumas abordagens da metodologia PRISMA (*Preferred Reporting Items for Systematic Reviews and Meta-Analyses*), descrita em Galvão, Pansani e Harrad (2015). A busca por artigos abrangeu as bases de dados dos próprios mecanismos de busca dos anais eletrônicos dos eventos Simpósio Brasileiro de Informática na Educação (SBIE), Workshop de Informática na Escola (WIE), Workshop sobre Educação em Informática (WEI) e I Workshop de Ensino em Pensamento Computacional, Algoritmos e Programação (WAlgProg), no período de 2007 a 2015, mas os trabalhos sobre o tema surgiram apenas a partir do ano de 2012. Para todas as bases, a chave de pesquisa utilizada foi “pensamento computacional” ou “*computational thinking*”, tanto no título, quanto no resumo (ou *abstract*). Foram incluídos nas buscas artigos publicados na Revista Brasileira de Informática na Educação, mas não foi encontrado nenhum resultado de pesquisa que atendesse os critérios de inclusão para esta RSL.

2.3.1 Questões da Pesquisa

O critério de inclusão dos trabalhos selecionados foi definido como a capacidade do mesmo em responder pelo menos umas das questões da pesquisa. Após esse primeiro filtro, foram definidos outros critérios de inclusão, como mostra a Tabela 2.1. Os critérios de exclusão definiram diretrizes para excluir os trabalhos que não trariam contribuições à pesquisa, e também podem ser vistos na Tabela 2.1. Não foram selecionados artigos que apenas faziam uma revisão da literatura, sendo considerados aqueles que descrevem ou aplicam alguma prática pedagógica ou descrevem o uso de ferramentas de ensino.

Tabela 2.1 Critérios de inclusão e exclusão.

Critérios de Inclusão	Critérios de Exclusão
I1. Artigos completos (quatro ou mais páginas)	E1. Artigos no formato de revisão sistemática da literatura
I2. Publicados após o ano de 2006	E2. Estudos que não aprofundem o PC na prática de ensino de programação
I3. Artigos que abordem pelo menos uma prática didática e/ou proponha alguma metodologia e/ou	

alguma ferramenta de ensino de programação. I4. No trabalho deve haver práticas de programação e PC	E3. Trabalhos que não contemplem nenhum dos componentes da QP3
--	--

Após as buscas nas bases de dados, houve a aplicação dos filtros com os critérios de inclusão e exclusão através de uma leitura rápida (*scanning*), com ênfase nas seções de resultados e/ou conclusões. Com isso, foram selecionados 6 artigos do WEI, 3 do SBIE, 4 do WIE e 3 do WAlgProg, em um total de 16 artigos. Com a seleção finalizada, todos os trabalhos foram lidos na íntegra e feito o fichamento, seguindo as diretrizes PRISMA.

Pelo fichamento, foi possível criar um formulário de extração, o qual descreve e identifica o artigo e a sua contribuição junto à revisão. Esse formulário possui as seguintes informações: identificador, título, autores, fonte do artigo (evento e ano) e atendimento aos critérios específicos de cada questão de pesquisa. A Tabela 2.2 mostra o formato do formulário de extração.

Tabela 2.2 Formulário de extração.

Identificador	
Título	
Autores	
Fonte do Artigo	{evento/revista, ano}
Atendimento a questão QP1	[C. “Desplugada”, Jogos Digitais, Ling. Prog, Ling. Prog. Visual, Rob. Pedag.]
Atendimento a questão QP2	[Fundamental, Médio/Técnico, Superior]
Atendimento a questão QP3	[H1, H2, H3, H4, H5, H6, H7, H8, H9]

A lista dos estudos e demais informações se encontram disponíveis para consulta em <<https://goo.gl/3OT4kN>>.

2.3.2 Ameaças à validade da pesquisa

Esta revisão sistemática utilizou um processo de seleção de estudos através de critérios de inclusão e exclusão com o objetivo de ser coerente e fiel ao tema proposto. Os critérios de inclusão e exclusão foram cuidadosamente projetados e discutidos entre os autores para minimizar o risco de exclusão de estudos relevantes. A busca foi realizada nas principais bases de dados dos eventos brasileiros, mas é possível que estudos relevantes possam não ter sido incluídos.

Outro fator que ameaça a validade da pesquisa pode estar presente nas decisões subjetivas ocorridas durante o processo de seleção e extração de alguns estudos que não

apresentavam uma descrição clara, consequentemente dificultando a aplicação objetiva dos critérios e análise, principalmente em relação à questão QP3. Para minimizar essa ameaça, a seleção e extração foi feita de forma iterativa e colaborativa pelos autores, sendo discutidos os possíveis conflitos em interpretações individuais.

2.4 Resultados obtidos e discussões

Esta seção apresenta uma visão geral dos estudos que foram selecionados pelos processos de extração e filtragem apresentados na Seção 3. Em seguida, serão apresentados e discutidos de forma detalhada os resultados relacionados às questões de pesquisa apresentados na Seção 2.

2.4.1 Visão geral dos estudos

Os artigos selecionados são do período entre 2012 e 2015. O intuito inicial era pesquisar artigos publicados após o ano de 2006, ano da criação do termo *Computational Thinking*, mas durante a pesquisa foi identificado que o surgimento de trabalhos nas fontes pesquisadas ocorreu apenas a partir do ano de 2012. A Tabela 2.3 apresenta a lista de estudos (em ordem alfabética por autores), e seu respectivo evento e ano.

Tabela 2.3 Lista de estudos em ordem alfabética por autores.

ID	Autores	Evento	Ano
E01	de Campos, Gleider M. et al.	WIE	2014
E02	de Carvalho, M. L. B., Chaimowicz, L. e Moro, M. M.	WEI	2013
E03	de França, R. S., dos Santos Ferrerira, V. A. F., de Almeida, L. C. F. e do Amaral, H. J. C.	WEI	2014
E04	de França, R. S. e Tedesco, P. C. A. R.	SBIE	2014
E05	de França, R. S. e Tedesco, P. C. A. R.	WEI	2015
E06	Falcão, T. P. e Barbosa, R.	SBIE	2015
E07	de França, R. S. e Tedesco, P. C. A. R.	WAlgProg	2015
E08	Gomes, T. e Melo, J.	WEI	2013
E09	Ramos, F. e Da Silva Teixeira, L.	WIE	2015
E10	Rodriguez, C., Zem-Lopes, A. M., Marques, L., e Isotani, S.	WIE	2015
E11	Santos, G., Silva, W., Cavalheiro, S., Foss, L., Aguiar, M., Pernas, A. M. e Reiser, R.	WIE	2015
E12	Scaico, P., Duarte, A., Alves, G. M., Maia, M., de Oliveira, F. M., Neto, S. V. M., Dantas, V.	WEI	2012
E13	Schoeffel, P. M., Varela, G. M., Durigon, L. R., Albuquerque, G. C. e Niquelati, M. S.	WAlgProg	2015
E14	Silva, A. e Soares, L. S.	WEI	2015
E15	Viel, F., Raabe, A. e Zeferino, C.	WIE	2014
E16	Zanetti, H. A. P. e Oliveira, C. L. V.	WAlgProg	2015

A Figura 2.1 mostra a distribuição dos números de artigos que abordam PC e ensino de programação por ano e evento. Neste gráfico, não foram incluídos os artigos do WAlgProg, pois o evento teve sua primeira ocorrência no ano de 2015.

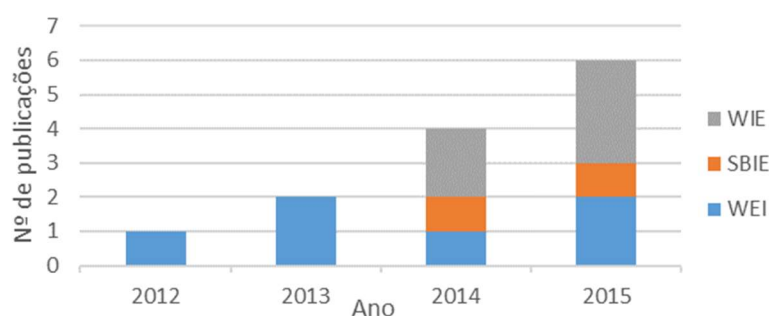


Figura 2.1 Número de publicação entre 2012 e 2015 nos eventos WIE, SBIE e WEI.

Como podemos observar, é crescente o número de publicações que abordam o ensino de programação e PC nos últimos anos, quando considerados todos os eventos. É importante ressaltar também o surgimento de publicações nos eventos SBIE e WIE, a partir de 2014, e o aumento de publicações no WIE no ano seguinte.

2.4.2 Práticas pedagógicas nos estudos (QP1)

Para esta pesquisa, o contexto da prática pedagógica pode ser interpretado como uma interferência feita pelo professor no desenvolvimento ou aprendizagem de um grupo de alunos. É um procedimento que pode ser adotado no momento em que é apresentado algum problema no aprendizado, com o objetivo de compreendê-lo, explicitá-lo ou corrigi-lo, podendo trazer novos elementos para que os alunos possam pensar e elaborar soluções de maneira diferenciada [Freire e Prado 1996].

2.4.2.1 Análise e discussões

A categorização dessas práticas que foi utilizada nesta RSL foi feita baseando-se nas ações mais comuns utilizadas em trabalhos recuperados pelas buscas, e nas práticas mais comumente abordados em trabalhos nos eventos pesquisados. Foram definidas cinco categorias, apresentadas na Tabela 2.4 com uma breve descrição. Essa categorização tem como objetivo

prover um mecanismo de mapeamento para futuro pesquisadores que queiram relacionar o uso de PC em ensino de programação e uma prática em específico. Após a discussão dos autores, as categorias definidas foram: Computação “Desplugada” (CD); Jogos Digitais (JD); Linguagem de Programação (LP); Linguagem de Programação Visual (LPV); e Robótica Pedagógica (RP).

Tabela 2.4 Categorias de práticas pedagógicas.

Categoria	Descrição
Computação “Desplugada” (CD)	Métodos que procuram promover o ensino de computação sem o uso de computador, utilizando atividades lúdicas para atingir pessoas de todas as idades [CSUnplugged 2016].
Jogos Digitais (JD)	Utilização de jogos digitais para o ensino de conceitos de computação ou programação de computadores.
Linguagem de Programação (LP)	Utilização de linguagens de programação de alto nível, como paradigma imperativo (e.g. C ou Pascal) ou orientada a objetos (e.g. Java ou Python).
Linguagem de Programação Visual (LPV)	Utilização de linguagens de programação visual, como Scratch ou App Inventor.
Robótica Pedagógica (RP)	Utilização de artefatos robóticos, reais ou virtuais, como ferramenta para de ensino de programação.

Essa análise baseou-se na descrição contida nos textos dos trabalhos pesquisados, de modo a identificar qual foi a prática escolhida e claramente definida no estudo ou, quando não houve essa descrição clara da prática adotada, quais ferramentas foram adotadas, denotando assim a categoria da prática. Nos casos de proposta de metodologias de ensino, foram verificadas em qual categoria a proposta contemplava. A Tabela 2.5 mostra os resultados encontrados nessa questão de pesquisa

Tabela 2.5 Relação entre as práticas e os estudos.

Prática	Estudo (ID)	Qtd.
Computação “Desplugada”	E01, E02, E03, E11, E13, E14	6
Jogos Digitais	E05, E06, E07, E09, E10, E12	6
Uso de Linguagem de Programação	E12, E15	2
Uso de Linguagem de Programação Visual	E02, E03, E08, E09, E10, E14, E16	7
Robótica Pedagógica	E03, E12, E13, E16	4
Não declarada	E04	1

2.4.3 Níveis de escolaridade (QP2)

Esta questão busca mostrar o cenário das pesquisas feitas no Brasil nos diferentes níveis de escolaridade. Os níveis estão delimitados por ensino fundamental, ensino médio/técnico e

ensino superior e os estudos foram classificados de acordo com o nível de escolaridade para os quais as pesquisas se destinaram.

2.4.3.1 Análise e discussões

Após o levantamento, foi possível perceber que a maioria dos estudos foram voltados ao ensino médio/técnico, totalizando 10 estudos. Os estudos de nível fundamental, somam 6 estudos e, por fim, apenas 1 estudo no ensino superior. É importante destacar que apenas 1 estudo [E03] destinava a pesquisa para dois níveis distintos de escolaridade (fundamental e médio/técnico).

Pelo levantamento de dados, é possível perceber uma tendência de ensino de programação aplicando PC no nível médio/técnico em 10 estudos [E02, E03, E04, E05, E07, E08, E09, E10, E15 e E16]. Alunos nesse nível de escolaridade possuem uma capacidade de pensamento e abstração que possibilita uma melhor assimilação de conceitos de programação e maturidade para desenvolver habilidades fundamentais para o PC, como abstração, decomposição de problemas e análise de dados. Além disso, há uma grande gama de instrumentos de ensino de programação voltados aos alunos do ensino médio, como jogos digitais e plataformas para robótica.

O segundo nível mais abordado nas pesquisas foi o fundamental, com 6 estudos [E01, E03, E06, E11, E13, E14] predominantemente com práticas usando LPV, RP e CD. Houve apenas um estudo aplicado ao nível superior [E12], com práticas de LP, RP e JD.

2.4.4 Objetivos alcançados nos estudos (QP3)

O objetivo desta questão de pesquisa é verificar quais as habilidades são mais estimuladas no ensino de programação com PC. Nesse contexto, foram utilizadas as habilidades definidas pela ISTE (2016), para categorizar quais habilidades são presentes e abordadas por pesquisadores brasileiros. A Tabela 2.6 mostra as habilidades e uma breve descrição das mesmas.

Tabela 2.6 Lista de habilidades.

Habilidade	Descrição	Habilidade	Descrição
(H1) Coleção de dados	Processo de recolher as informações adequadas.	(H6) Algoritmos e Procedimentos	Série de passos ordenados e encadeados para resolver um problema ou atingir algum fim.
(H2) Análise de Dados	Encontrar o sentido dos dados, encontrar padrões e tirar conclusões.	(H7) Automação	Usar computadores ou máquinas fazer tarefas repetitivas ou tediosas.
(H3) Representação de dados	Representação e organização de dados em gráficos adequados, cartas, palavras ou imagens.	(H8) Simulação	Representação ou modelo de um processo. Simulação envolve também as experiências em andamento utilizando modelos.
(H4) Decomposição do Problema	Quebrar tarefas em partes menores e gerenciáveis.	(H9) Paralelização	Organizar recursos para, simultaneamente, realizar tarefas para alcançar um objetivo comum.
(H5) Abstração	Reduzir a complexidade para definir ideia principal		

Baseado nos textos presentes no estudo, os autores desta pesquisa identificaram que alguns estudos abordam integralmente ou parcialmente algumas habilidades em suas propostas e, algumas vezes, foi identificada a ausência de algumas habilidades. Para uma demonstração de aplicação dessas habilidades nos estudos, foi definida uma classificação de três modos: *i*) aborda claramente (C) – quando a habilidade está claramente descrita nos objetivos das práticas ou nos resultados e/ou discussões do estudo; *ii*) apresenta indícios (I) – quando se identifica através dos resultados e discussões do estudo que a habilidade foi abordada durante a prática e; *iii*) não identificado (N) – quando a habilidade não foi identificada pelos autores.

2.4.4.1 Análise e discussões

A Tabela 2.7 apresenta uma distribuição matricial que possibilita a visualização clara de quais habilidades (colunas) são abordadas nos estudos levantados (linhas), identificados com a classificação (C, I ou N) referente à abordagem da habilidade no cruzamento entre as linhas e colunas. Como é possível observar, cada estudo aborda ao menos duas habilidades e, na maioria, de maneira clara. Há algumas exceções [E03, E04 e E13], mas majoritariamente entre os estudos há uma presença significativa dessas habilidades nas práticas apresentadas, sendo colocadas como objetivos a serem alcançados junto aos alunos. Como já supracitado, um estudo [E07] não contempla uma prática e sim um modelo de aplicação, mas define claramente quais habilidades são abordadas.

Tabela 2.7 Relação entre as habilidades e os estudos.

Estudo	H1	H2	H3	H4	H5	H6	H7	H8	H9	Estudo	H1	H2	H3	H4	H5	H6	H7	H8	H9
E01	I	I	C	N	C	C	N	C	N	E09	C	C	C	C	C	C	C	C	N
E02	C	I	N	C	C	C	C	I	N	E10	C	C	C	C	C	C	C	C	N
E03	N	N	C	N	C	C	N	C	N	E11	N	N	C	N	C	C	C	C	N
E04	N	C	N	C	N	N	N	N	N	E12	I	I	I	C	C	C	C	C	N
E05	N	N	N	C	I	C	N	C	N	E13	N	N	N	C	C	C	N	N	N
E06	I	I	I	C	C	C	N	C	N	E14	C	C	C	C	C	C	N	N	N
E07	C	C	C	C	C	C	N	C	N	E15	C	C	C	N	C	C	N	C	N
E08	C	C	C	C	C	C	N	C	N	E16	C	C	C	C	C	C	N	C	N

Legenda: C – Aborda claramente, I – Apresenta indícios, N – Não identificado

As habilidades H5 e H6 aparecem na maioria dos trabalhos (93,75%), seguidas por H8 (81,35%), H2, H3 e H4 (75%), H1 (68,75%), H7 (31,25%) e, por fim, a habilidade H9, que não aparece em nenhum dos estudos resgatados nesta RSL. A Tabela 4 mostra que há uma predominância de habilidades fundamentais para o ensino de programação, como a abstração (H5) e desenvolvimento de algoritmos (H6), assim como habilidades de coleta, interpretação e representação de dados (H1, H2 e H3, respectivamente).

O uso de modelos e simulação também está presente em grande parte dos estudos, devido às práticas adotadas na maioria dos trabalhos, em especial as que utilizam JD e RP. Essas práticas fornecem ferramental que facilita a criação de modelos intermediários à solução final, enriquecendo o ambiente de aprendizado. A habilidade de decomposição de problemas (H4) também é bastante significativa e aparente nos estudos, que se apresenta como uma premissa essencial no ensino de programação, principalmente nas etapas iniciais, onde os alunos precisam planejar sua solução do problema proposto.

A baixa incidência da habilidade de automação (H7) é devida à abordagem que os pesquisadores adotam em seus estudos. O processo de automação é muito relevante na programação, mas dificilmente o termo “automatizar” é utilizado nesse contexto. Nessa RSL foram identificadas descrições que indicavam que a modularidade de alguns procedimentos de laços de repetição, como práticas usando LP, sugeriam a automação de processos repetitivos. Dentre os estudos levantados não foi possível identificar de maneira clara a intenção de qualquer pesquisador estimular a habilidade de paralelização (H9). O conceito de paralelização é explorado em atividades com alunos mais experientes e, provavelmente, esse pode ser o motivo da ausência nos estudos.

2.5 Conclusões

Este estudo teve como objetivo levantar e analisar pesquisas existentes com relação a aplicação do PC no ensino de programação, delimitando o cenário no Brasil. Para cumprir este objetivo foi realizada uma revisão sistemática da literatura cobrindo os principais eventos associados à SBC.

Como apresentado na análise sobre a QP1, vemos que o uso de LPV, seguido por JD e CD, são as práticas mais abordadas nos estudos, sendo que alguns estudos utilizavam essas práticas em conjunto. Em práticas utilizando LPV, a ferramenta mais comum foi o *Scratch*, projeto criado pelo *Massachusetts Institute of Technology* (MIT), muito difundido no meio acadêmico e que apresenta resultados positivos. O uso de JD aparece como instrumento de motivação aos alunos, pelo caráter lúdico, desafiador e motivador. A utilização da CD está presente principalmente nos momentos de introdução a noções básicas de computação e programação, sendo um recurso lúdico e de baixo custo.

De acordo com os resultados obtidos é possível observar uma tendência de pesquisas no ensino médio/técnico e no ensino fundamental, como apontam as análises referentes a QP2. O ensino de programação nesses níveis pode ser apoiado por diversas ferramentas, que são de interesse dos alunos nessas faixas etárias correspondentes, como, por exemplo, jogos digitais, programação visual e plataformas robóticas.

Os resultados obtidos na QP3 mostram quais são as habilidades mais abordadas em pesquisas que se apoiam em PC no ensino de programação. É possível notar que habilidades indispensáveis ao programar, e que caracterizam o PC, são comuns entre os trabalhos. Habilidades como coleta, análise e representação de dados, decomposição de problemas, abstração e desenvolvimento de algoritmos encontram-se na maioria dos estudos.

Como trabalho futuro desta revisão sistemática, espera-se pesquisar e desenvolver novas soluções para o ensino de programação alinhado ao PC, a fim de colaborar com essa área de pesquisa. Após a análise da QP3, é possível notar a demanda por novos métodos que explorem habilidades necessárias para programar, como abstração, decomposição de problemas, entre outros, e, portanto, a continuidade deste trabalho será direcionada para este objetivo.

2.6 Referências

- Barr, V. e Stephenson, C. (2011). Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community?. *Acm Inroads*, 2(1), 48-54.
- Blikstein, P. (2008). “O Pensamento Computacional e a Reinvenção do Computador na Educação”. Disponível em: < <http://bit.ly/1lXlbNn>>. Acesso em: 10 de abr. 2016.
- Brennan, K. e Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American Educational Research Association*, Vancouver, Canada.
- CSUnplugged (2016). Computer Science without a Computer. Disponível em: < <http://csunplugged.org/>>. Acesso em: 25 abr. 2016.
- Cuny, J., Snyder, L. e Wing, J. M. (2010). Demystifying computational thinking for non-computer scientists. Manuscrito não publicado. Disponível em: <http://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf>.
- De Paula, B. H., Valente, J. A. e Burn, A. (2014). O uso de jogos digitais para o desenvolvimento do currículo para a Educação Computacional na Inglaterra. *Currículo sem Fronteiras*, v. 14, n. 3, p. 46-71.
- Freire, F. M. e Prado, M. E. B. B. (1996). Professores construcionistas: a formação em serviço. In *Actas do III Congresso Ibero-Americano de Informática Educativa*. Disponível em: <http://lsm.dei.uc.pt/ribie/docfiles/txt200352145836PROFESSORES%20CONSTRUCIONISTAS.pdf>. Acesso em: 08 jan 2014.
- Galvão, T. F., Pansani, T. S. A. e Harrad, D. (2015). Principais itens para relatar Revisões sistemáticas e Meta-análises: A recomendação PRISMA. *Epidemiologia e Serviços de Saúde*, v. 24, p. 335-342.
- ISTE (2016). Computational thinking teacher resources. Disponível em: https://csta.acm.org/Curriculum/sub/CurrFiles/472.11CTTeacherResources_2ed-SP-vF.pdf. Acesso em: 28 jan. 2016.
- Mestre, P., Andrade, W., Guerrero, D., Sampaio, L., da Silva Rodrigues, R. e Costa, E. (2015) Pensamento Computacional: Um estudo empírico sobre as questões de matemática do PISA. In: *Anais dos Workshops do Congresso Brasileiro de Informática na Educação*. p. 1281.
- Wing, J. M. (2006). “Computational Thinking”. *Communications of the ACM*. March, Vol. 49, No. 13.

Capítulo 3

ENSINO DE PROGRAMAÇÃO UTILIZANDO COMPUTAÇÃO FÍSICA: UMA REVISÃO SISTEMÁTICA DA LITERATURA

RESUMO

Esta pesquisa tem como objetivo discutir sobre a teoria, as aplicações e as principais estratégias didáticas para ensino de Programação utilizando recursos de Computação Física. Pesquisas sobre esse tema são recentes. No meio acadêmico, ainda há questões a serem discutidas sobre o alinhamento entre as atuais pesquisas. Essas questões podem causar má interpretação ou dúvidas ao se escolher ou adotar práticas sobre o tema. Nesta Revisão Sistemática da Literatura, foram avaliados 15 artigos, com o objetivo de apresentar uma visão crítica sobre as pesquisas mais recentes e suas aplicações

PUBLICAÇÃO

Artigo publicado na revista *Comunicações em Informática vol. 4 ed. 1*. Este texto usa as normas e formatos de referências estabelecidos pela revista. As numerações das seções, figuras e tabelas foram adaptadas do texto original para dar sequência à numeração desta tese.

3.1 Introdução

Atualmente, a tecnologia faz parte da rotina diária de todos e é muito acessível. Jovens estudantes dominam boa parte dos recursos tecnológicos e computacionais, devido a sua vivência com tecnologia desde a infância. É cada vez mais esperado que a escola forneça um ambiente em que a tecnologia esteja presente, não só como uma ferramenta de ensino, mas em dinâmicas nas quais esses jovens deixem de ser meros “consumidores” de tecnologia e se tornem “criadores”. Nesse contexto, o papel fundamental da escola e dos educadores é orientar e conduzir práticas que catalisem a motivação e a criação com esses jovens.

A programação de computadores tem potencial para ser uma atividade educativa que pode auxiliar de maneira efetiva a aquisição de habilidades cognitivas fundamentais para diversas outras tarefas, como a atitude crítica-reflexiva e a resolução de problemas [13]. Também auxilia o aluno a explicitar um modelo mental de solução, podendo expressar de maneira quase fiel a sequência de resolução e verificar a causa-e-efeito [14].

Computação Física (CF) é a integração da computação com o mundo físico, através de sensores e atuadores, amparados por sistemas embarcados, comunicando-se via redes de computadores e provendo ambientes automatizados [22]. A CF tem como objetivo conectar o mundo virtual com o real, com a criação de novas interfaces intuitivas entre objetos e seres humanos. Utilizar da CF em práticas de ensino pode oferecer recursos didáticos que ajudem a promover a criatividade e a compreensão da aplicação de conceitos [15].

Atividades envolvendo componentes eletrônicos e robótica, alinhados com o ensino de programação, podem diminuir as dificuldades de alunos iniciantes no processo do aprendizado de programação. Por exemplo, utilizando um robô móvel e uma arena com obstáculos, para demonstrar conceitos como estruturas condicionais (*if-else*) e de repetição (*for* e *while*) [23].

3.2 Metodologia

Para este trabalho, foi realizada uma Revisão Sistemática da Literatura (RSL), que permitiu identificar e analisar principais aspectos da área de pesquisa que envolve CF e ensino de Programação.

3.2.1 Questões de Pesquisa

O principal objetivo desta RSL é avaliar estudos realizados no período de 2010 a 2019, que apresentem contribuições ao tema de uso da CF no ensino de programação. A principal questão a ser respondida com este presente trabalho é: “*Quais são as abordagens didáticas aplicadas para o ensino de programação utilizando ferramentas e artefatos de Computação Física?*”. Com base na questão principal apresentada, outras 2 questões de pesquisa (QP) mais específicas foram definidas: **QP1:** *Quais as motivações para o uso de CF no ensino de programação?*; **QP2:** *Quais são as principais plataformas e ferramentas de CF utilizadas nas práticas?*

A QP1 tem como objetivo identificar quais são as principais motivações que levam a adoção da CF para práticas de ensino de programação. O objetivo da QP2 é mapear quais são as ferramentas e plataformas mais utilizadas nas pesquisas.

3.2.1.1 Métodos aplicados na Revisão da Literatura

A QP1 tem como objetivo identificar quais são as principais motivações que levam a adoção da CF para práticas de ensino de programação. O objetivo da QP2 é mapear quais são as ferramentas e plataformas mais utilizadas nas pesquisas.

A presente RSL foi realizada seguindo a metodologia PRISMA (*Preferred Reporting Items for Systematic Reviews and Meta-Analyses*) [3]. As bases de dados utilizadas para a busca de artigo foram ACM, Google Scholar, IEEE, Science Direct, Scopus e Springer, em trabalhos publicados entre os anos de 2010 e 2019. A chave (*string*) de pesquisa utilizava termos em português e inglês, como “computação física” (“*physical computing*”), ensino ou aprendizagem (“*teach*”, “*teaching*”, “*learning*”) e programação (“*programming*”, “*coding*”).

Os critérios de inclusão dos trabalhos selecionados foram definidos a partir da capacidade de atender pelo menos uma das questões de pesquisa e, após isso, seguir os outros critérios de inclusão, mostrados na Tabela 3.1. Foi feita a aplicação dos critérios de inclusão e exclusão através de uma leitura rápida (*scanning*), dando ênfase nos resumos, seções de resultados e/ou conclusões. Maiores detalhes da metodologia adotada estão disponíveis no site do projeto.⁴

Tabela 3.1 Critérios de inclusão e exclusão.

Critérios de Inclusão	Critérios de Exclusão
I1. Publicados após o ano de 2010, e até 2019. I2. Artigos com práticas pedagógicas e/ou proponha alguma metodologia ou ferramenta didática. I3. Abordar práticas de programação e CF	E1. Artigos que apresentem apenas revisão da literatura. E2. Não abordar ensino de programação ou CF. E3. Trabalhos que não contemplem nenhuma das questões de pesquisa

Com isso, foram selecionados 15 artigos que atendem o propósito desta RSL, que foram publicados entre 2013 e 2018, embora tenham sido buscadas pesquisas sobre o tema desde 2010, só surgiram trabalhos relevantes no período indicado. Após essa seleção, foi realizado um fichamento através de um formulário de extração com o propósito de identificar as contribuições de cada artigo.

3.3 Resultados e discussões

Esta seção apresenta uma visão geral dos artigos selecionados na Seção 3.3.2 e as discussões das questões apresentadas na Seção 3.3.1.

3.3.1 Discussão sobre a QP1

Foi analisado se há relatos ou descrições nos artigos que apontem quais foram as motivações que levaram a adoção da CF nas práticas descritas ou no desenvolvimento de alguma plataforma ou ferramenta. Dentre as motivações descritas, foi possível categorizar 5 áreas distintas, que conseguem trazer um consenso e agrupamentos coerentes entre os trabalhos.

⁴ Endereço o site: <https://tinyurl.com/projetoCFAS>

A categorização utilizada nesta RSL foi feita baseando-se nos relatos e descrições mais comuns nos artigos recuperados pelas buscas. Essa categorização tem como objetivo prover uma referência para o mapeamento de futuros pesquisadores interessados em CF e ensino de programação. As áreas definidas foram: “Ambiente dinâmico e/ou motivador” (ADM); “Compreensão de Conceitos ou Paradigmas de Programação” (CCPP); “Engajamento e Criação” (EC); “Resolução de Problemas” (RP); e “Visualização e Identificação de Erros” (VIE). O Tabela 3.2 traz uma breve descrição de cada categoria e a relação de artigos.

Tabela 3.2 Descrição das categorias e relação dos artigos.

Id.	Descrição	Artigos
ADM	Relatos que indicavam que a presença de CF tinha o objetivo de criar um ambiente mais dinâmico e motivacional, diferente do tradicional, despertando maior interesse pelos alunos.	[1], [2], [6], [9], [10], [11], [12], [16], [17], [21]
CCPP	Artigos que procuram meios de trazer instrumentos didáticos para ajudar os alunos na abstração e visualização de conceitos abstratos relacionados à programação.	[1], [5], [6], [7], [10], [12], [18], [19]
EC	Participação ativa dos alunos e estímulo à criação de novos artefatos e soluções usando CF, estimulando a cultura <i>maker</i>	[5], [20]
RP	Trabalhos que abordam a habilidade de tomada de decisão e resolução de problemas, estimulando a participação do aluno na construção de soluções.	[10], [21]
VIE	Estudo que utilizam a CF para proporcionar um ambiente que possa trazer <i>feedback</i> que sejam melhores interpretados e os erros visualizados de maneira mais tangível.	[10], [16]

Os artigos que se enquadram na categoria ADM foram a maioria, pois a busca por um ambiente mais dinâmico e motivacional é algo muito buscado em práticas didáticas que envolvam programação. O ambiente tradicional de programação pode ser pouco estimulante, potencializando possíveis problemas e dúvidas que os alunos tenham [8]. Para esses trabalhos, a adoção de recursos da CF pode ser um aliado para esse propósito, devido a uma maior ludicidade das atividades na visão dos alunos.

A segunda categoria com mais artigos, a CCPP, aborda um problema bastante comum entre os iniciantes em programação, a pouca capacidade de abstração. A aprendizagem de programação demanda habilidades como abstração, capacidade de identificar generalização e pensamento crítico [4]. Meios que ajudem a visualização, de maneira mais concreta, fora do computador, dos resultados de certos conceitos de programação podem ser bastante positivos.

Há um alinhamento com os artigos da categoria VIE, pois a busca por instrumento que tornem os conceitos de programação mais tangíveis auxilia na interpretação e na percepção de erros que estão ocorrendo, e consequentemente, a busca por soluções.

Os trabalhos da categoria EC dão ênfase na relação de engajamento do aluno junto a prática e ao incentivo de utilizar a tecnologia não apenas como um consumidor, mas sim como um criador.

Os trabalhos que se enquadram na categoria RP tinham como objetivo proporcionar um cenário que explorasse a habilidade de tomada de decisão e resolução de problemas, que são essenciais para a programação.

É importante notar que alguns artigos aparecem em mais de uma categoria, denotando algumas intersecções. Os artigos [1], [6], [10] e [12] aparecem em ADM e CCPP, mostrando que trabalhos buscam ambientes motivadores com o objetivo de trazer maneiras de ajudar na abstração de conceitos complexos. O trabalho A3 aparece nas categorias CCP e EC, por tratar sobre a criação de artefatos a partir elementos da CF e programação, para promover a visualização de conceitos abstratos. Por fim, uma análise possível de interseção de categorias é o artigo [10], presente em 4 categorias distintas (ADM, CCPP, RP e VIE), tratando-se de uma pesquisa que apresenta práticas detalhadas, lúdicas, motivacionais e focadas em trazer conceitos abstratos de maneira mais tangível através de puzzles.

3.3.2 Discussão sobre a QP2

O objetivo principal desta questão foi fazer um levantamento de quais são as principais plataformas ou ferramentas utilizadas, podendo se identificar, inclusive, se há novas ferramentas sendo desenvolvidas. Buscou-se identificar o cenário atual e quais tecnologias são mais adotadas nas pesquisas e as novas demandas.

As tecnologias utilizadas ou desenvolvidas nos trabalhos selecionados foram: a plataforma de prototipagem Arduino, o kit de robótica *Lego® Mindstorms*, a plataforma de tecnologias vestíveis *Lilypad Arduino* e as placas de eletrônica *Makey Makey* e *micro:bit*. Também houve 3 trabalhos que desenvolveram suas próprias tecnologias. A Figura 3.1 mostra um gráfico com relação total dos artigos e seus identificadores, além dos números de artigos.

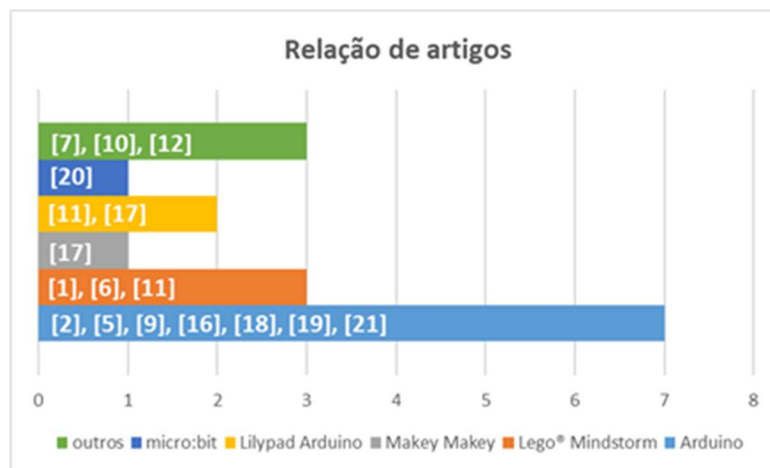


Figura 3.1 Relação dos artigos por ferramenta/plataforma.

A presença do Arduino na maioria dos trabalhos era esperada, por se tratar de uma plataforma de CF acessível, tanto por disponibilidade no mercado, quanto devido ao seu caráter de projeto aberto de hardware. Também existe muito material e literatura voltado à plataforma. A versatilidade do Arduino é outro ponto positivo na sua adoção, pois pode ser incluído em projetos simples e complexos, envolvendo práticas de robótica e Internet das Coisas.

O baixo custo e fácil manutenção também são atrativos para instituições de ensino. Para esse estudo, a plataforma *Lilypad* (baseado em Arduino) foi separada das demais, devido à sua aplicação específica em tecnologias vestíveis. Os dois estudos que usam *Lilypad* possuem práticas com um foco estritamente em desenvolvimento de projetos objetivando a criação de dispositivos vestíveis (*wearable*) e ambos usam outras tecnologias conjuntamente: um usa também *Makey Makey* e outro *Lego® Mindstorms*.

O kit didático *Lego® Mindstorms* aparece em 3 artigos, sendo utilizado em práticas de robótica e programação, que é justamente o objetivo principal dessa ferramenta. Mesmo com seu alto custo no mercado, seus benefícios são notáveis nos estudos, como a versatilidade de construções de modelos automatizados e a facilidade de programação. As demais plataformas didáticas, *Makey Makey* e *micro:bit*, que foram desenvolvidas para criar pequenos projetos e ter uma interface de programação intuitiva, aparecem em apenas 1 artigo para cada ferramenta. Ambas são ferramentas com recursos limitados e custo relativamente alto, se comparadas com as plataformas citadas anteriormente.

Houve 3 trabalhos que desenvolveram tecnologias próprias para serem utilizadas em práticas com CF: *Talkoo kit* no artigo [7], *Bots & (Main)Frames* no artigo [10] e *i*CaTch* no artigo [12]. A *Talkoo kit* tem como objetivo criar um ambiente de ensino de conceitos de computação, com facilidade de montagem de projeto através de módulos *plug-and-play* e com

ambiente de programação visual. O projeto *Bots & (Main)Frames* propõe um ambiente de desafios (*puzzle*) para o ensino de programação, utilizando blocos físicos baseados em tecnologia tangível. Por fim, a plataforma *i*CaTch* apresenta um kit didático usando CF e tecnologia vestíveis, para estimular o ensino de programação e criação de projetos de dispositivos vestíveis.

3.4 Conclusões

Este estudo teve como principal objetivo fazer o levantamento e análise de pesquisas que abordam recursos da CF no ensino e aprendizagem de programação. Para alcançar esse objetivo, foi realizada uma RSL buscando por trabalhos nas principais bases de trabalhos acadêmicos publicados a partir de 2010.

Foi observada uma maior concentração de artigos de pesquisas que buscam a CF como um recurso didático com propósito de criar um ambiente motivador, que promova maior engajamento por parte do aluno e que apresente elementos que auxilia no aprendizado de conceitos de programação, muitas vezes abstratos. Esses aspectos são recorrentes em diversos trabalhos que objetivam novas estratégias ou recursos para o ensino de programação. Também foi identificado que há poucos trabalhos focados em desenvolvimento de habilidades necessárias ao programador, como resolução de problemas ou identificação e correção de erros.

A plataforma mais utilizada nas pesquisas foi o Arduino, seguida pelo kit didático *Lego® Mindstorms*, ferramentas essas presentes há mais de uma década no mercado. A plataforma *Lilypad*, baseada em Arduino, apresenta uma solução para tecnologias vestíveis, podendo trazer soluções criativas e acessíveis. As ferramentas *Makey Makey* e *micro:bit*, mais recentes, também foram citadas.

3.5 Bibliografia

- [1] Anfurrutia, F. I. et al. (2016). Incorporating educational robots and visual programming environments in introductory programming courses. In 2016 International Symposium on Computers in Education (SIIE) (pp. 1-4). IEEE. DOI: 10.1109/SIIE.2016.7751835.
- [2] Arakliotis, S.; Nikolos, D. G.; Kalligeros, E. (2016). LAWRIIS: A rule-based arduino programming system for young students. In 2016 5th International Conference on Modern

- Circuits and Systems Technologies (MOCASST) (pp. 1-4). IEEE. DOI: 10.1109/MOCASST.2016.7495150.
- [3] Galvão, T. F.; Pansani, T. D. S. A.; Harrad, D. (2015). Principais itens para relatar Revisões sistemáticas e Meta-análises: A recomendação PRISMA. *Epidemiologia e Serviços de Saúde*, 24, 335-342.
- [4] Gomes, A.; Mendes, A. J. (2007). Learning to program-difficulties and solutions. In *International Conference on Engineering Education–ICEE* (Vol. 2007).
- [5] Jang, Y.; Lee, W.; Kim, J. (2015). Assessing the usefulness of object-based programming education using arduino. *Indian Journal of Science and Technology*, 8(S1), 89-96. DOI: 10.17485/ijst/2015/v8iS1/57701.
- [6] Jin, K. H.; Haynie, K.; Kearns, G. (2016). Teaching elementary students programming in a physical computing classroom. In *Proceedings of the 17th annual conference on information technology education* (pp. 85-90). DOI: 10.1145/2978192.2978238.
- [7] Katterfeldt, E. S. et al. (2018). Physical computing with plug-and-play toolkits: Key recommendations for collaborative learning implementations. *International Journal of Child-Computer Interaction*, 17, 72-82.
- [8] Lahtinen, E.; Ala-Mutka, K.; Järvinen, H. M. (2005). A study of the difficulties of novice programmers. *Acm sigcse bulletin*, 37(3), 14-18. DOI: 10.1145/1067445.1067453.
- [9] Martin, C.; Hughes, J.; Richards, J. (2017). Learning Experiences in Programming: The Motivating Effect of a Physical Interface. In: *Proceedings of the 9th International Conference on Computer Supported Education*, SCITEPRESS, pp. 162–172. DOI: 10.5220/0006375801620172.
- [10] Melcer, E. F. (2017). Exploring the Effects of Physical Embodiment in a Puzzle-Based Educational Programming Game. In *Proceedings of the 2017 ACM SIGCHI Conference on Creativity and Cognition* (pp. 532-538). DOI: 10.1145/3059454.3078704.
- [11] Merkouris, A.; Chorianopoulos, K.; Kameas, A. (2017). Teaching programming in secondary education through embodied computing platforms: Robotics and wearables. *ACM Transactions on Computing Education (TOCE)*, 17(2), 1-22. DOI: 10.1145/3025013.
- [12] Ngai, G. et al. (2013). Designing i* CATch: A multipurpose, education-friendly construction kit for physical and wearable computing. *ACM Transactions on Computing Education (TOCE)*, 13(2), 1-30. DOI: 10.1145/2483710.2483712.
- [13] Papert, S. (1994) *A máquina das crianças*. Porto Alegre: Artmed.

- [14] Pea, R. D.; Kurland, D. M. (1984). On the cognitive effects of learning computer programming. *New ideas in psychology*, 2(2), 137-168. DOI: 10.1016/0732-118X(84)90018-7.
- [15] Przybylla, M.; Henning, F.; Schreiber, C.; Romeike, R. (2017). Teachers' Expectations and Experience in Physical Computing. In *International Conference on Informatics in Schools: Situation, Evolution, and Perspectives* (pp. 49-61). Springer, Cham. DOI: 10.1007/978-3-319-71483-7_5.
- [16] Przybylla, M.; Romeike, R. (2014). Physical Computing and Its Scope--Towards a Constructionist Computer Science Curriculum with Physical Computing. *Informatics in Education*, 13(2), 241-254. DOI: 10.15388/infedu.2014.05.
- [17] Richard, G. T.; Kafai, Y. B. (2015). Making physical and digital games with e-textiles: a workshop for youth making responsive wearable games and controllers. In *Proceedings of the 14th international conference on interaction design and children* (p. 399-402). DOI: 0.1145/2771839.2771926.
- [18] Rubio, M. A. et al. (2014). Enhancing an introductory programming course with physical computing modules. In *2014 IEEE Frontiers in Education Conference (FIE) Proceedings* (1-8). IEEE. DOI: 10.1109/FIE.2014.7044153.
- [19] Rubio, M. A.; Hierro, C. M.; Pablo, A. P. D. M. (2013). Using arduino to enhance computer programming courses in science and engineering. In *Proceedings of EDULEARN13 conference* (1-3). IATED Barcelona, Spain.
- [20] Sentance, S. et al. (2017). Teaching with physical computing devices: the BBC micro: bit initiative. In *Proceedings of the 12th Workshop on Primary and Secondary Computing Education* (pp. 87-96). DOI: 10.1145/3137065.3137083.
- [21] Sohn, W. (2014). Design and evaluation of computer programming education strategy using Arduino. *Advanced Science and Technology Letters*, 66(1), 73-77. DOI: 10.14257/astl.2014.66.18.
- [22] Stankovic, J. A. et al. (2005). Opportunities and obligations for physical computing systems. *Computer*, 38(11), 23-31. DOI: 10.1109/MC.2005.386.
- [23] Zanetti, H., e Oliveira, C. (2015). Práticas de ensino de Programação de Computadores com Robótica Pedagógica e aplicação de Pensamento Computacional. In *Anais dos Workshops do Congresso Brasileiro de Informática na Educação* (Vol. 4, No. 1, p. 1236).

Capítulo 4

A TEORIA DE APRENDIZAGEM SIGNIFICATIVA NO ENSINO DE PROGRAMAÇÃO: UM MAPEAMENTO SISTEMÁTICO DA LITERATURA

RESUMO

Esta pesquisa tem como objetivo discutir as aplicações e as estratégias didáticas para ensino de programação baseada na teoria de Aprendizagem Significativa, desenvolvida por David Ausubel. Pesquisas sobre esse tema são recentes e há poucos relatos de aplicação da teoria no contexto de ensino de programação, com ainda menos pesquisas estudando a convergência do resultado desses relatos. Essa falta de referências e exemplos pode gerar insegurança ou dúvidas no professor e pesquisador que busca práticas baseadas na Aprendizagem Significativa no contexto de ensino de Programação. Neste mapeamento sistemático da literatura, foram avaliados 26 artigos publicados desde 2010, com o objetivo de apresentar uma visão crítica sobre as pesquisas mais recentes. Com este estudo foi possível mapear as principais estratégias adotadas e os públicos-alvo em pesquisas que seguem premissas da Aprendizagem Significativa no ensino de programação.

PUBLICAÇÃO

Artigo publicado nos anais do XXXIII Simpósio Brasileiro de Informática na Educação (SBIE 2022). Este texto usa as normas e formatos de referências estabelecidos pelos anais do simpósio. As numerações das seções, figuras e tabelas foram adaptadas do texto original para dar sequência à numeração desta tese.

4.1 Introdução

As disciplinas de “Programação de Computadores” e “Algoritmos” estão presentes em todos os cursos de formação técnica e superior na área de Informática e Computação. Elas trabalham conhecimentos fundamentais e essenciais para o desenvolvimento acadêmico dos alunos durante o decorrer de formações na área. As disciplinas introdutórias de programação apresentam um primeiro obstáculo aos iniciantes, devido ao seu caráter técnico e exigência de um alto nível de abstração.

O desenvolvimento de algoritmos e a prática de programação são conhecimentos difíceis de ser ensinados, exigindo uma grande dedicação do aluno e abordagens adequadas para sua aprendizagem. Muitas aplicações dessas disciplinas focam em ensinar questões técnicas da linguagem de programação sem focar em práticas para a resolução de problemas [Gomes e Mendes 2007]. Uma das grandes dificuldades dos iniciantes em programação é correlacionar conceitos anteriormente aprendidos e sedimentados com os novos apresentados, de modo a serem utilizados em conjunto nas soluções. Moström *et al.* (2008) destacam o problema em determinar quais recursos são necessários e qual a sequência de utilização deles para a composição de uma solução algorítmica. Gomes e Mendes (2007) enfatizam que os alunos têm dificuldades em estabelecer relações entre problemas anteriormente vivenciados e conhecimentos prévios com os novos conhecimentos e suas aplicações em novos problemas.

Trabalhos como o de Piteira e Costa (2013) discutem a dificuldade de alunos de cursos de computação em entender certas estruturas de dados e suas aplicações na resolução de problemas reais. A ausência de um elo que conecte o recurso apresentado ao contexto do problema a ser resolvido pode levar à desmotivação do aluno. Várias pesquisas, como de Möstrom *et al.* (2008), Gasaymeh *et al.* (2016) e Almeida *et al.* (2017), mostram que estratégias de ensino que promovam a aplicação de conceitos de forma mais concreta e significativa podem auxiliar no desenvolvimento da habilidade de abstração e na associação de conceitos para construção de uma solução.

Ausubel (2012) propôs a teoria da Aprendizagem Significativa (AS), na qual defende que a aprendizagem se torna mais significativa à medida que um novo conhecimento ou conteúdo é agregado às estruturas de conhecimento previamente existentes em um aprendiz, incorporando mais significado à sua estrutura cognitiva. Essa teoria questiona o aprendizado

“mecânico” ou repetitivo, no qual um novo conhecimento é isolado ou tem associações arbitrárias com os anteriores.

Pelizzari *et al.* (2002) ressaltam o impacto positivo que a AS proporciona na construção e enriquecimento da estrutura cognitiva do aprendiz, do ponto de vista dos conceitos anteriormente apresentados e sua utilização na aquisição de novos conceitos e suas conexões. Esse processo de correlação e integração de conceitos, por meio da identificação de suas similaridades e diferenças, é fundamental para a ocorrência de uma aprendizagem mais significativa. Na AS, essas etapas são chamadas de *diferenciação progressiva*, apresentando inicialmente conceitos mais gerais e evoluindo, progressivamente, para mais específicos, e de *reconciliação integrativa*, que buscando correlacionar os conceitos apresentados [Moreira e Mansini 2002].

A AS pode ser base para métodos e estratégias no contexto do ensino de programação, com um objetivo claro de promover meios de aprendizagem contextualizados, progressivos e com associações dos conceitos que são aplicados em qualquer linguagem de programação. A preocupação por buscar meios de aprendizagem de programação que possam ser mais significativos, compreensíveis e contextualizados não é recente. Shneiderman (1977) explorou a temática, associando a AS para discutir abordagens didáticas que pudessem auxiliar na compreensão da sintaxe e semântica das linguagens de programação.

Este artigo tem o objetivo de realizar um mapeamento sistemático da literatura e analisar trabalhos que exploram estratégias de ensino que se baseiem nas premissas da AS. Essas pesquisas são direcionadas para o ensino-aprendizagem de programação de computadores, lógica de programação e algoritmos, discutindo os objetivos desses trabalhos e potenciais contribuições que eles podem trazer para a área. O mapeamento tem o objetivo de identificar trabalhos publicados que possam atestar a eficácia e resultados da aplicação dessas práticas baseadas em AS, e proporcionar meios definir quais estratégias são aplicadas. Por meio desta análise, objetiva-se também encontrar lacunas de pesquisas e demandas para criação de um método que siga diretrizes da AS.

O texto está organizado da seguinte forma: a Seção 4.2 apresenta o contexto e objetivos que conduziram esse mapeamento sistemático; a Seção 4.3 mostra como foi feito o processo de busca e seleção dos trabalhos analisados, assim a categorização dos mesmos; a Seção 4.4 apresenta resultados e discussões sobre os trabalhos e questões de pesquisa, assim como apontamentos de lacunas para novas pesquisas; e, por fim, na Seção 4.5, são feitas as considerações finais.

4.2 Contexto e objetivo do mapeamento

Atualmente, muitos pesquisadores e educadores procuram novas maneiras de tornar o ensino de programação mais dinâmico, motivador e significativo ao aluno [Raposo e Dantas 2016]. Transmitir um novo conhecimento requer planejamento, usando como base o que já foi apresentado (e aprendido), buscando consolidar o conhecimento, tornando-o mais atraente e significativo ao aluno. Segundo Moreira (2012), o papel do educador é proporcionar, de maneira gradativa e progressiva, recursos para que o aluno possa construir sua estrutura cognitiva, organizando e relacionando o conhecimento previamente adquirido com os novos, que estão em processo de aquisição.

O mapeamento sistemático, segundo Dermeval, Coelho & Bittencourt (2019), tem como objetivo responder questões de pesquisa sem grande aprofundamento, apenas para que se tenha uma visão geral sobre o tema. A escolha por esta forma de revisão, por meio de mapeamento, se deu pela necessidade de categorizar trabalhos de interesse para trabalhos futuros que possam colaborar com a temática envolvida. Este mapeamento tem como objetivo principal analisar trabalhos que possam apontar quais são as práticas encontradas na literatura que seguem como premissa diretrizes da AS para o ensino de programação, assim como quais são os públicos-alvo atingidos. O resultado dessa análise visa contribuir com futuros trabalhos que possam desenvolver métodos de ensino em programação que possam se apoiar nos pilares da AS, promovendo meios de auxiliar a aquisição de conhecimento por parte do aluno.

4.2.1 As questões de pesquisa

A definição das questões de pesquisa (QP), que delimitam essa pesquisa, foi baseada em dois aspectos que são importantes em práticas de ensino de programação: o público-alvo a ser atendido e quais estratégias são adotadas. Esses aspectos foram utilizados para categorizar os trabalhos e com isso, conduzir as análises e discussões. Para este mapeamento, duas questões foram formuladas:

- **QP1:** *Qual é o público-alvo presente em pesquisas que utilizam práticas de ensino de programação baseado em Aprendizagem Significativa?*
- **QP2:** *Quais estratégias são utilizadas em práticas de ensino de programação que promovam a Aprendizagem Significativa?*

4.3 Métodos aplicados na Revisão Sistemática da Literatura

Para esta RSL, foi utilizado o protocolo PRISMA (*Preferred Reporting Items for Systematic reviews and Meta-Analyses*), uma metodologia que estabelece procedimentos, como a lista de itens que devem ser considerados para a revisão (PRISMA *checklist*) e o fluxo de exclusão e inclusão de artigos (PRISMA *flow diagram*) [Page *et al.* 2021].

Para a construção da *string* de busca, foram escolhidos três conceitos principais para definição dos termos, que são: a própria AS e a inclusão do termo “teoria”, para tirar a desambiguação com o termo geral “aprendizagem significativa”; a inclusão do nome do criador da teoria, também por critérios de desambiguação e, por fim, o conceito de ensino de programação. A *string* de busca padrão foi inicialmente definida a partir dos principais termos em inglês e seus sinônimos, sendo definida na seguinte forma: *((("meaningful learning" OR "meaningful learning theory") AND ("david ausubel" OR "ausubel")) AND ("introductory programming education" OR "introductory programming" OR "teach* programming" OR "learn* programming" OR "computer programming" OR "novice programming" OR "coding education" OR "coding" OR "software engineering education" OR "cs1" OR "computer science education" OR "introductory computer science"))*. Essa *string* também foi adaptada para as línguas portuguesa e espanhola.

Foram escolhidas: a base de dados *Scopus*, a base de pesquisa Periódicos CAPES e o motor de busca Google Acadêmico, para os mecanismos de consulta e busca por trabalhos. A busca foi feita por publicações a partir de 2010, buscando uma análise mais contemporânea do cenário envolvendo a pesquisa. A **Tabela 4.1** apresenta a quantidade de trabalhos por etapa e o total selecionado para a análise.

Tabela 4.1 Resumo do resultado da busca pelos trabalhos

Bases de Dados	Qtd. de trabalhos	Após 1ª triagem	Após 2ª triagem
Periódicos CAPES	136	26	12
Google Acadêmico	114	48	21
Scopus	204	29	10
Total c/ trabalhos duplicados			43
Total s/ trabalhos duplicados			26

Os critérios de inclusão e exclusão estão apresentados resumidamente na **Tabela 4.2**. A prioridade foi a análise de trabalhos que abordassem práticas didáticas que envolvam programação como objetivo principal ou nas quais a programação fosse de suma importância

para a execução da atividade. Toda essa fase de inclusão e exclusão ocorreu em duas etapas: 1) com a primeira triagem, aplicando o filtro por data (aplicando o critério I1) e a seleção de artigos completos (aplicando o critério I2); 2) com a segunda triagem, com a leitura primeiramente do título, palavras-chave e resumo (aplicando os critérios E1, E2 para exclusão) e, posteriormente, o texto do artigo (aplicando os critérios I3 e I4 de inclusão e critérios E3, E4 e E5 de exclusão). Também foi realizada uma filtragem por trabalhos em que o termo “aprendizagem significativa” estivesse relacionado à teoria desenvolvida por David Ausubel, como um critério de desambiguação, sendo que os alguns trabalhos em que o termo não fazia referência à teoria da AS, foram descartados (que foi a base para definir o critério de exclusão E3).

Tabela 4.2 Critérios de inclusão e exclusão

Critérios de Inclusão	Critérios de Exclusão
I1. Publicados a partir do ano de 2010.	E1. Apresentem apenas revisão da literatura.
I2. Trabalhos completos	E2. Sem fonte de publicação
I3. Abordem práticas pedagógicas e/ou proponham alguma metodologia e/ou ferramenta didática e/ou avaliem instrumentos didáticos.	E3. Usem o termo “aprendizagem significativa”, mas sem relação com a teoria de Ausubel.
I4. Abordem os temas ensino de programação e AS.	E4. Não contemplem a questão de pesquisa
	E5. Abordem práticas de Computação sem programação

Após a seleção dos trabalhos e leitura dos textos, foi realizado o processo de classificação, extração e mapeamento dos dados. Em uma planilha, foram organizados os trabalhos primeiramente por público-alvo. Após isso, foram anotados e relacionados quais eram as práticas e ferramentas descritas em cada um dos trabalhos. Com essas anotações, foi possível a incidência comum entre os artigos e definidos os agrupamentos finais para a análise.

4.3.1 Análise temática dos trabalhos

Com o objetivo de investigar tendências e lacunas dentro da temática de aplicação de AS no ensino de programação, foi realizada uma análise temática com o objetivo de identificar temas e abordagens na área. Segundo Souza (2019), a análise temática utiliza uma estratégia indutiva e baseada nas informações presentes nos textos, levando o pesquisador a descobrir padrões semelhantes e tendências entre os trabalhos analisados, podendo criar classificações. Nesta análise, foram utilizadas duas temáticas, que são convergentes nos trabalhos: a do público-alvo e a de estratégias de ensino.

Em uma classificação por seu público-alvo, foi possível notar que há trabalhos para *Ensino Superior* (17 trabalhos, 65,4%), *Ensino Médio* (4 trabalhos – 15,4%), *Ensino Fundamental* (3 trabalhos – 11,5%) e exclusivamente para o *Ensino Técnico* (2 trabalhos – 7,7%). A identificação de um público-alvo ajuda a entender qual é o objetivo da prática no trabalho analisado. Trabalhos que focalizam o ensino superior e técnico possuem um perfil de formação profissional, enquanto nos trabalhos dos demais grupos a aprendizagem de programação tem outros objetivos.

Para esse mapeamento, é considerada estratégia de ensino qualquer aplicação de algum método ou prática de ensino, ou adição de alguma ferramenta ou recurso de apoio para o educador. Dentre os estudos selecionados, foram identificadas algumas estratégias de ensino, sendo que em alguns trabalhos é possível notar mais do que uma estratégia utilizada. Pode-se observar que a maioria dos trabalhos utilizaram *Mapas Conceituais* (7 trabalhos - 26,9%), um recurso utilizado para representação de conhecimento e associação de conceitos. Outra estratégia que é associada ao relacionamento de conceitos foi a *Apresentação gradativa de conceitos* (4 trabalhos – 15,4%), sem adotar uma ferramenta ou outra estratégia de ensino específica. A apresentação gradativa do conhecimento é uma das premissas da AS, mas, para análise feita neste artigo, foi considerado como uma estratégia em particular, pois alguns trabalhos deixavam clara sua adoção. Na AS, a apresentação gradativa de conceitos está baseada principalmente nos princípios de diferenciação progressiva e a reconciliação integradora [Moreira, 2012]. *Recursos Multimídias* e *Novas Linguagens de Programação* são utilizados em 3 trabalhos cada (11,5%). Recursos como *Jogos Digitais*, *Robótica Pedagógica*, *Aprendizado Baseado em Problemas (PBL - Problem-Based Learning)* e *Scratch*, comuns em práticas didáticas, aparecem em apenas 2 trabalhos cada uma (7,7%). Apenas um trabalho (3,8%) aplica *Metodologias Ativas*. A **Tabela 4.3** organiza os trabalhos separados por estratégias de ensino aplicadas na pesquisa e os respectivo público-alvo.

Tabela 4.3 Distribuição dos trabalhos por estratégia de ensino e público-alvo

	Fundamental	Médio	Técnico	Superior
Apresentação gradativa	-	-	-	[Berssanette e Francisco 2018], [Buriticá 2011], [Buriticá 2013], [Buriticá 2014]
Jogos Digitais	-	-	-	[Barbosa, Fernandes e Campos 2011], [Hernandez <i>et al.</i> 2010]
Novas Linguagens	[Wang, Mendori e Xiong 2014]	[Alexandron <i>et al.</i> 2017], [Silva <i>et al.</i> 2020]	-	-
Mapas Conceituais	-	-	-	[Bishop <i>et al.</i> 2017], [Brito, Rodriguez e Pellicer 2016], [Menon e Kovalchick 2020], [Minakshi e Sona 2018], [Mühling 2016], [Sharma e Chawla 2019], [Soto <i>et al.</i> 2013]
Metodologias Ativas	-	-	-	[Shitsuka <i>et al.</i> 2018]
PBL	-	-	-	[Aires <i>et al.</i> 2021], [Lima, Diniz e Eliasquevici 2014]
Recursos Multimídia	-	-	[Astolfi e Lopes Junior 2015], [Astolfi e Lopes Junior 2016]	[Matthews, Hin e Choo 2014]
Robótica Pedagógica	[Kaloti-Hallak, Armoni e Ben-Ari 2019]	[Yepes, Barone e Porciuncula 2021]	-	-
Scratch	[Suárez 2015]	[Gomez-Zermeno e Mejia 2020]	-	-

A presença majoritária de trabalhos em nível de escolaridade profissionalizante (superior e técnico) aponta uma preocupação de pesquisadores em apresentar conceitos de programação nessa etapa de formação. Em nível superior, há poucos trabalhos que abordam conceitos de Programação Orientada a Objetos (POO) e/ou conceitos mais avançados, como Estrutura de Dados e Programação Funcional, sendo apenas 6 estudos (Bishop *et al.* (2017); Brito, Rodriguez e Pellicer (2016); Buriticá (2013); Buriticá, (2014); Hernandez *et al.* (2010); e Müling (2016)). Os demais trabalhos destinados ao ensino superior abordavam conceitos fundamentais de programação. Os 2 trabalhos para o público de formação técnica envolviam POO, e todos os trabalhos de nível médio e fundamental abordavam os conceitos básicos de Algoritmos e Programação Estruturada.

4.4 Resultados obtidos e discussões

Nesta seção são discutidos os resultados obtidos na análise dos trabalhos que compõem essa RSL. A Seção 4.4.1 discute sobre o público-alvo; na Seção 4.4.2 são apresentadas discussões sobre as estratégias adotadas; e, por fim, na Seção 4.4.3 há uma discussão mais abrangente em torno do tema e o apontamento de lacunas que podem ser base para novas pesquisas.

4.4.1 Os públicos-alvo presentes nos trabalhos (QP1)

A partir das análises apresentadas na seção 3.1 observa-se a maior presença de trabalhos no Ensino Superior. Nesse nível de formação, a apresentação de conceitos de programação é mais aprofundada e a necessidade do aluno em ter base de conhecimentos prévios para ir adiante nos estudos é mandatório. O paradigma de POO e as aplicações de Estruturas de Dados, por exemplo, exigem um domínio nas noções básicas de programação. Mas a identificação de poucos trabalhos que abordem conceitos avançados de programação pode indicar que ainda há demanda de pesquisa de aplicação da AS no ensino formal de programação.

A maior presença de trabalhos dedicados ao ensino nos níveis de formação profissional (Superior e Técnico) apontam uma preocupação por novas estratégias de ensino nesses níveis de formação. A falta de engajamento, dificuldades em abstrair e aprender conceitos alunos em cursos de formação em Computação é presente, como apontam Souza, Batista e Barbosa (2016). Com essa análise é possível identificar que há uma demanda de pesquisa em que o objetivo seja atingir a formação profissional. Em cursos formais de programação, é comum a alta taxa de evasão, principalmente em disciplinas iniciais. Também são notáveis fatores como desmotivação e falta de engajamento em alunos iniciantes, sendo uma questão relevante a ser resolvida pela academia [Hoed 2016].

A formação profissional em Computação em nível superior, presente na maior parte dos trabalhos analisados, possui o desafio de motivar e engajar um público adulto. Várias pesquisas apresentam soluções dinâmicas e lúdicas para o público mais jovens (crianças e adolescentes), que não se enquadram em um público mais maduro [Souza, Moretti & Podestá 2008; Medeiros, da Silva & Aranha 2013]. A base fundamental da AS, que é a correlação de conceitos aplicados por meio de um instrumento que auxilie sua visualização, como Mapas Conceituais (presente nos trabalhos), pode trazer um recurso que ampare essa percepção do

aluno. Mas falta ainda a associação de um processo instrucional que seja mais efetivo na associação das diretrizes da AS, como mostram alguns trabalhos analisados, que definem uma apresentação gradativa dos conceitos, mas não se alinham a nenhum método conhecido na academia. Poucos trabalhos trazem esse alinhamento, como aqueles que usam PBL e Metodologias Ativas.

4.4.2 As estratégias adotadas (QP2)

Sobre as estratégias adotadas nos trabalhos, a utilização de Mapas Conceituais teve a maior incidência. Esse recurso é comumente utilizado em trabalhos que abordam AS, tanto como ferramenta de ensino (apresentação dos conceitos e suas relações), como também artifício de discussão sobre o resultado da aprendizagem. Menon e Kovlachick (2020), Mühling (2016), Sharma e Chawla (2019) e Soto *et al.* (2013) apresentam explicitamente Mapas Conceituais que representam e correlacionam conceitos de programação de forma hierárquica para os alunos, criando um modelo do conhecimento que se espera ser adquirido.

Os trabalhos enquadrados na *Apresentação gradativa de conceitos* são pesquisas que mostram a intenção do pesquisador-educador em apresentar conceitos de programação de forma gradativa, para auxiliar a construção de um modelo mental em que os conceitos iniciais se conectem e façam sentido para o conceito seguinte. Os trabalhos desse grupo, mesmo deixando clara a intenção de apresentação gradativa, não apresentam de maneira explícita a utilização de uma ferramenta ou estratégia encontrada na literatura para o auxílio à tarefa. Em alguns casos, como em Buriticá (2013), os artigos descrevem sua prática didática como uma metodologia nova.

Entre as pesquisas que apresentam Novas Linguagens de Programação, destaca-se o de Silva *et al.* (2020), que apresenta a linguagem e ambiente de programação Calango, baseada em comandos em português, e o trabalho de Wang, Mendori e Xiong (2014), que desenvolvem uma linguagem de programação visual baseada em Mapas Conceituais. Em todos os trabalhos classificados desta forma, é possível identificar a demanda por criar ferramentas que possam ser mais lúdicas e que necessitam de menor grau de abstração na utilização de conceitos de programação.

As pesquisas que envolvem Jogos Digitais e Robótica Pedagógica também apresentam propostas de práticas com o objetivo de engajar os alunos de ensino superior e mostrar conceitos de programação em meios mais concretos. Os trabalhos envolvendo robótica

tinham como público alunos de cursos de engenharia. A pesquisa de Barbosa, Fernandes e Campos (2011) propõe uma metodologia que mostra conceitos de programação através de um jogo no formato de desafios (*puzzles*). Já o trabalho de Hernandez *et al.* (2010) utiliza uma ferramenta de desenvolvimento de jogos digitais (*game engine*) para ensinar conceitos de POO.

Apenas três trabalhos alinhavam metodologias de ensino com AS. Em Aires *et al.* (2021) são discutidas práticas de desenvolvimento de aplicativos móveis baseadas em PBL. Em Lima, Diniz e Eliasquevici (2019) é proposta uma metodologia voltada ao ensino de Algoritmos usando resolução de problemas. Shitsuka *et al.* (2018) utilizam uma metodologia ativa em uma turma de Engenharia, devido à demanda dos alunos por mais dinamismo nas aulas de programação.

Em todos os trabalhos que utilizavam *Recursos Multimídias*, o principal objetivo era a associação de conceitos de maneira menos abstrata, usando recursos visuais, como vídeos e imagens. Os trabalhos com *Scratch* envolviam públicos de formação não profissional, tendo o ensino de programação como um recurso para criação de outras atividades didáticas.

Observa-se que há poucos trabalhos que utilizam estratégias que possam trazer a programação para um ambiente que ampare o exercício de abstração, como é necessário em POO, por exemplo. A utilização de Robótica Pedagógica, presente em apenas dois trabalhos, pode ser um recurso lúdico, interativa e multidisciplinar [Santos *et al.* 2019]. Os poucos trabalhos que abordam o recurso são voltados apenas para o ensino fundamental e médio, não abordando como esse ferramental poderia ser utilizado também para a formação profissional.

Poucos trabalhos propõem novas metodologias ou ferramentas para o ensino de programação, mostrando uma oportunidade para investigações na área com esse objetivo. A maior parte dos trabalhos aplicaram estratégias de apresentar conceitos de programação por meios que não fossem diretamente através de código-fonte. Essa preocupação aponta que usar as vantagens da AS pode favorecer na apresentação de conceitos que tenham maior significado, principalmente através de conceitos básicos que formam ou complementam novos conceitos, de forma bem aparente (por exemplo, usando mapas conceituais) ou de forma gradativa.

4.4.3 Discussões gerais e lacunas para pesquisa

Este mapeamento sistemático mostra que ainda não se esgotam as possibilidades de novas pesquisas em torno no tema. Os resultados oriundos no mapeamento por público-alvo apontam uma maioria de trabalhos voltados à formação profissional. A correlação de conceitos de forma

hierárquica e correlacionada, como defendida na AS, se mostra fortemente relacionada à estrutura didática encontrada no ensino de programação e áreas correlacionadas. Por exemplo, o encadeamento de conceitos fundamentais em POO ou Estrutura de Dados podem ser apresentados de forma ordenada e gradativa, com um reforço nas correlações de cada conceito e seu subsequente item: quando se é apresentada uma *classe* e sua relação direta com *objeto*, ou até mesmo as diferentes estruturas de dados, como *árvores*.

A apresentação dos conceitos de forma gradativa é uma diretriz da AS, mas como se trata de uma teoria educacional, não define um sistema metodológico de aplicação. Na AS, o processo de apresentação do conteúdo se inicia com um conceito inicial, que seja de domínio do aluno ou que seja inclusivo para ele, de fácil compreensão (chamado de ideia-âncora ou *subsunçor*) [Moreira e Mansini 2002]. Poucos trabalhos exploram artifícios que possam não somente ajudar na compreensão inicial, mas também em toda a cadeia de conhecimento. Ferramentas oriundas da Robótica Pedagógica e Computação Física, que apresentam artefatos tangíveis, interativos e programáveis, poderiam construir todo um cenário didático e aplicar conceitos de programação, tanto básicos quanto avançados (com em POO).

Uma demanda aparente na análise, e vinculada com o fato de ausência de um processo metodológico, é a de aplicação e desenvolvimento de métodos que conduzam a AS. Houve poucos trabalhos com a iniciativa de criar um meio sistemático de condução de práticas. É comum na comunidade acadêmica, vinculado ao ensino de programação, a busca por métodos que sejam replicáveis em sala de aula. Ainda existe demanda por trabalhos que proponham novas soluções nesse sentido.

Por meio do levantamento realizado nesta pesquisa, é notável a baixa incidência de trabalhos que alinhem métodos que amparem e estimulem o exercício de abstração com a AS. Futuras pesquisas podem propor métodos e estratégias que possam utilizar os conceitos de ideia-âncora e a progressão da apresentação de conceitos de maneira efetiva em atividades e práticas de programação. O uso de recurso tangíveis, como apresentados na Robótica Pedagógica, ou os recursos interativos oriundos de Jogos Digitais, podem promover ambientes que apoiem as principais premissas da AS, conduzindo práticas motivadores junto aos alunos.

A Figura 4.1 apresenta um mapa mental das possíveis demandas sugeridas por este mapeamento, podendo ser explorados em novas pesquisas. Vale ressaltar que essas demandas não limitam a ocorrência de outras, mas baseiam-se nos resultados obtidos neste mapeamento, podendo conduzir pesquisadores a novos questionamentos.

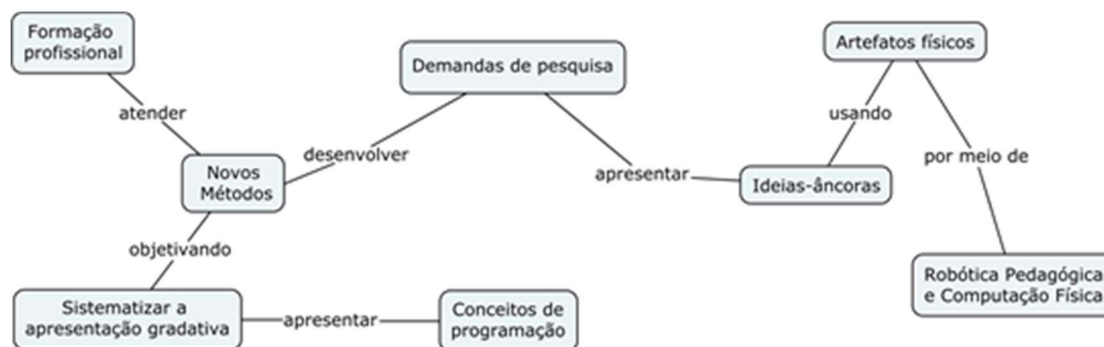


Figura 4.1 Mapa mental das demandas de pesquisa

4.5 Considerações finais

Estudos sobre a teoria da AS juntamente com o ensino de programação ainda são escassos, mesmo sendo uma teoria presente em diversas pesquisas de ensino em outras áreas, como matemática e ciência. A motivação em adotar premissas da AS vem principalmente das dificuldades que os educadores ainda encontram na apresentação de conceitos de programação para alunos iniciantes. Como contribuição, este artigo apresentou uma análise temática com o objetivo de orientar pesquisadores e novas investigações.

As estratégias de ensino de programação abordadas nessa análise trazem um cenário geral da pesquisa na área e algumas demandas por novas pesquisas. Por exemplo, a escassez de trabalhos que utilizem Robótica Pedagógica, Computação Física, Interfaces Tangíveis, entre outras ferramentas que possam ajudar na demonstração de conceitos com um menor nível de abstração e maior interatividade. Entre os trabalhos analisados, também se nota uma baixa incidência de pesquisas que abordem o ensino de programação posterior a Algoritmos e Lógica de Programação, como em POO ou Estrutura de Dados, que são comuns em formação profissional. Novas investigações podem sugerir aplicações, estratégias e ferramentas que possam ser aplicadas em cursos formais de programação, que sofrem um impacto direto quando há problemas na aprendizagem, como evasão e baixo aproveitamento acadêmico.

Pela análise realizada neste trabalho, é possível perceber que a utilização de conceitos da teoria da AS contribuem em práticas didáticas no ensino de programação, pois agregam recursos que podem auxiliar em alguns problemas comumente encontrados pelos alunos, como no exercício de abstração e na associação de conceitos. As estratégias apresentadas nos trabalhos utilizam-se de prerrogativas fundamentais da AS para levar novos recursos que possam auxiliar no processo de apresentação e construção do conhecimento por

parte dos alunos. Há também trabalhos que conseguem alinhar a AS com alguns recursos como Jogos Digitais, Robóticas, PBL, entre outros, para criar estratégias e métodos de ensino.

Por fim, esta pesquisa não esgota todas as discussões acerca do uso da AS em ensino de programação, mas apresenta um cenário atual na área. Como trabalho futuro, a partir das análises realizadas, sugere-se o aprofundamento da pesquisa, com o intuito de criar um embasamento teórico para o desenvolvimento de metodologias de ensino que alinhem AS e com outras estratégias de ensino, promovendo novas contribuições na área de pesquisa.

4.6 Referências

- Aires, J. P., Aires, S., Pereira, M. J., & Alves, L. M. (2021). Active methodologies in incoming programming classes. In *Second International Computer Programming Education Conference (ICPEC 2021)* (Vol. 91). Schloss Dagstuhl--Leibniz-Zentrum für Informatik.
- Alexandron, G., Armoni, M., Gordon, M., & Harel, D. (2017). Teaching scenario-based programming: An additional paradigm for the high school computer science curriculum, Part 1. *Computing in Science & Engineering*, 19(5), 58-67.
- Astolfi, G., & Junior, D. L. (2015). Investigação sobre conhecimentos prévios de alunos do curso Técnico em Informática a partir da aplicação de organizadores prévios. *Aprendizagem Significativa. Revista/Meaningful Learning Review*, 15-28.
- Astolfi, G., & Junior, D. L. (2016,). Ensino de linguagem de programação com ênfase na aprendizagem significativa. In *Anais do XXIV Workshop sobre Educação em Computação* (pp. 2106-2115). SBC.
- Almeida, T., Netto, J. F., da Silva, R., & Custódio, T. (2017). Laboratório remoto de robótica como elemento motivador para a aprendizagem de programação. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)* (Vol. 28, No. 1, p. 665).
- Ausubel, D. P. (2012). *The acquisition and retention of knowledge: A cognitive view*. Springer Science & Business Media.
- Barbosa, L. S., Fernandes, T. C., & Campos, A. M. (2011). Takkou: Uma ferramenta proposta ao ensino de algoritmos. In *XVIII Workshop sobre Educação em Computação (WEI 2011)*.
- Berssanette, J. H., & Frencisco, A. C. (2018). Proposta de abordagem prática para o ensino de Programação baseada em Ausubel. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)* (Vol. 29, No. 1, p. 398).

- Bishop, M., Dai, J., Dark, M., Ngambeki, I., Nico, P., & Zhu, M. (2017). Evaluating secure programming knowledge. In *IFIP World Conference on Information Security Education* (pp. 51-62). Springer, Cham.
- Bosse, Y. (2020). Padrões de dificuldades relacionadas com o aprendizado de programação. Tese de Doutorado. Universidade de São Paulo. São Paulo, SP.
- Brito, M. G. L., Rodriguez, L. R. R. & Pellicer, Y. S. (2016). Concept Maps as a means university teaching assistant. In *International Conference on Concept Mapping*. Springer, Cham.
- Buriticá, O. I. T. (2011). Planeación de la asignatura programación i en un programa de ingeniería de sistemas a partir de la teoría del aprendizaje significativo. *Revista Educación en Ingeniería*, 6(12), 102-114.
- Buriticá, O. I. T. (2013). Estrategia metodológica para aproximar los paradigmas funcional, estructurado y orientado a objetos en ingeniería de sistemas a partir de aprendizaje significativo. *Avances Investigación en Ingeniería*, 10(2), 49-63.
- Buriticá, O. I. T. (2014). Relaciones de aprendizaje significativo entre dos paradigmas de programación a partir de dos lenguajes de programación. *Tecnura*, 18(41), 91-102.
- Dermeval, D., Coelho, J. A. D. M., & Bittencourt, I. I. (2019). Mapeamento sistemático e revisao sistemática da literatura em informática na educação. *JAQUES, Patrícia Augustin; PIMENTEL, Mariano; SIQUEIRA, Sean; BITTENCOURT, Ig.(Org.) Metodologia de Pesquisa em Informática na Educação: Abordagem Quantitativa de Pesquisa. Porto Alegre: SBC.*
- Gasaymeh, A., AlJa'afreh, I. A., Al-Dmour, A., & Abu-Alrub, M. (2016). Higher education students' preferences for applying the principles of constructivism in learning programming languages with the use of ICTs. *Journal of Studies in Education*, 6(3), 168-187.
- Gomes, A., & Mendes, A. J. (2007). Learning to program-difficulties and solutions. In *International Conference on Engineering Education–ICEE* (Vol. 7).
- Gomes, A., & Mendes, A. (2014). A teacher's view about introductory programming teaching and learning: Difficulties, strategies and motivations. In *2014 IEEE Frontiers in Education Conference (FIE) Proceedings* (pp. 1-8). IEEE.
- Gómez-Zermeño, M. G. & Mejía, D. B. (2020). Development of Significant Learning through Scratch Programming Logic of Secondary School Students. *The International Journal of Technologies in Learning*, 27(2), 21-36.

- Hernandez, C. C., Silva, L., Segura, R. A., Schimiguel, J., Ledón, M. F. P., Bezerra, L. N. M., & Silveira, I. F. (2010). Teaching programming principles through a game engine. *CLEI electronic journal*, 13(2), 1-8.
- Hoed, R. M. (2016). Análise da evasão em cursos superiores: o caso da evasão em cursos superiores da área de Computação.
- Kaloti-Hallak, F., Armoni, M., & Ben-Ari, M. (2019). The Effect of Robotics Activities on Learning the Engineering Design Process. *Informatics in Education*, 18(1), 105-129.
- Lima, Á., Diniz, M., & Eliasquevici, M. (2019). Metodologia 7Cs: Uma Nova Proposta de Aprendizagem para a Disciplina Algoritmos. In *Anais do XXVII Workshop sobre Educação em Computação* (pp. 429-443). SBC.
- Matthews, R., Hin, H. S., & Choo, K. A. (2015). Practical use of review question and content object as advanced organizer for computer programming lessons. *Procedia-Social and Behavioral Sciences*, 172, 215-222.
- Medeiros, R. P. (2019). *Hello, world: uma análise sobre dificuldades no ensino e na aprendizagem de introdução à programação nas universidades*. Tese de doutorado. Universidade Federal de Pernambuco. Recife, PE.
- Medeiros, T. J., da Silva, T. R., & da Silva Aranha, E. H. (2013). Ensino de programação utilizando jogos digitais: uma revisão sistemática da literatura. *Renote*, 11(3).
- Menon, P., & Kovalchick, L. (2020). Using a Concept Map to Represent the Composition of Knowledge in an Introductory Programming Course. *Information Systems Education Journal*, 18(3), 4-17.
- Minakshi, S., & Sonal, C. (2018). Using Concept Map Network Based CLE for Teaching Learning and Evaluating the Knowledge Acquired by Learners. In *International Conference on Advances in Computing and Data Sciences* (pp. 148-157). Springer, Singapore.
- Moreira, M.A. (2012). O que é afinal, aprendizagem significativa? *Quriculum*. Laguna, v. 2, n. 3, p. 1-27.
- Moreira, M. A., & Masini, E. F. S. (2002). *Aprendizagem significativa: a teoria de David Ausubel*. Centauro.
- Moström, J. E., Boustedt, J., Eckerdal, A., McCartney, R., Sanders, K., Thomas, L., & Zander, C. (2008). Concrete examples of abstraction as manifested in students' transformative experiences. In *Proceedings of the Fourth international Workshop on Computing Education Research* (pp. 125-136).
- Mühling, A. (2016). Aggregating concept map data to investigate the knowledge of beginning CS students. *Computer Science Education*, 26(2-3), 176-191.

- Page, M. J., Moher, D., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., & McKenzie, J. E. (2021). PRISMA 2020 explanation and elaboration: updated guidance and exemplars for reporting systematic reviews. *Bmj*, 372.
- Pelizzari, A., Kriegel, M. D. L., Baron, M. P., Finck, N. T. L., & Dorocinski, S. I. (2002). Teoria da aprendizagem significativa segundo Ausubel. *revista PEC*, 2(1), 37-42.
- Piteira, M., & Costa, C. (2013). Learning computer programming: study of difficulties in learning programming. In *Proceedings of the 2013 International Conference on Information Systems and Design of Communication* (pp. 75-80).
- Raposo, E. H. S., & Dantas, V. (2016). O Desafio da Serpente-Usando gamification para motivar alunos em uma disciplina introdutória de programação. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)* (Vol. 27, No. 1, p. 577).
- Santos, R., Sousa, B., Raiol, A., Cerqueira, P., & Bezerra, F. (2019). Uma Proposta de Método de Ensino e Relatos de Experiências com a Robótica Educacional. In *Anais do XXVII Workshop sobre Educação em Computação* (pp. 111-120). SBC.
- Sharma, M. & Chawla S. (2018). Using Concept Map Network Based CLE for Teaching Learning and Evaluating the Knowledge Acquired by Learners. In *International Conference on Advances in Computing and Data Sciences* (pp. 148-157). Springer, Singapore.
- Shitsuka, D. M., Pereira, A. S., Shitsuka, R., & Boghi, C. (2019). Aprendizagem ativa de programação em turmas de engenharia: uma pesquisa-ação. *Research, Society and Development*, 8(3), 01-19.
- Shneiderman, B. (1977). Teaching programming: A spiral approach to syntax and semantics. *Computers & Education*, 1(4), 193-197.
- Silva, G., Santos, G., Canedo, E. D., Rissoli, V., Praciano, B., & Andrade, G. (2020). Impact of Calango language in an Introductory Computer Programming Course. In *2020 IEEE Frontiers in Education Conference (FIE)* (pp. 1-9). IEEE.
- Soto, B. D. G., Ortiz, J. F. V., Moreno, I. R., & Ortiz, L. E. V. (2013). Experiencias de éxito en la aplicación de mapas conceptuales en la carrera de Ingeniería en Computación, México. *JETT*, 4(1), 73-84
- Souza, D. M., da Silva Batista, M. H., & Barbosa, E. F. (2016). Problemas e dificuldades no ensino de programação: Um mapeamento sistemático. *Revista Brasileira de Informática na Educação*, 24(1), 39.
- Souza, L. K. D. (2019). Pesquisa com análise qualitativa de dados: conhecendo a Análise Temática. *Arquivos Brasileiros de Psicologia*, 71(2), 51-67.

- Sousa, V. N. D., Moretti, A. C., & Podestá, V. A. D. (2008). Programação da grade de horário em escolas de ensino fundamental e médio. *Pesquisa Operacional*, 28(3), 399-421.
- Suárez, M. G. (2015). Efectos del proceso de aprender a programar con “Scratch” en el aprendizaje significativo de las matemáticas en los estudiantes de educación básica primaria. *Escenarios*, 13(2), 87-102.
- Wang, J., Mendori, T., & Xiong, J. (2014). A language learning support system using course-centered ontology and its evaluation. *Computers & Education*, 78, 278-293.
- Yepes, I., Barone, D. A. C., & Porciuncula C. M. D. (2021). Use of Drones as Pedagogical Technology in STEM Disciplines. *Informatics in Education*.

Capítulo 5

POR QUE ESTIMULAR A APRENDIZAGEM SIGNIFICATIVA NO ENSINO DE PROGRAMAÇÃO ORIENTADA A OBJETOS?

RESUMO

Este artigo apresenta uma análise crítico-reflexiva sobre a adoção da teoria da Aprendizagem Significativa no ensino-aprendizagem de Programação Orientada a Objetos. Este texto apresenta a visão dos autores de como a teoria de David Ausubel pode ser aplicada com resultados positivos no processo de construção do conhecimento, em especial no ensino de Programação Orientada a Objetos por meio de práticas e recursos que possam trazer mais significado ao aluno, como por exemplo, o uso Computação Física e outros recursos didáticos.

PUBLICAÇÃO

Artigo publicado nos anais do Simpósio Brasileiro de Educação em Computação (EduComp 2021). Este texto usa as normas e formatos de referências estabelecidos pela revista. As numerações das seções, figuras e tabelas foram adaptadas do texto original para dar sequência à numeração desta tese.

5.1 Introdução

O ensino de programação para iniciantes possui muitos desafios. Essa discussão é foco de muitas pesquisas e ainda é uma área com muitas possibilidades de exploração. Butler e Morgan [5] argumentam que um dos principais problemas encontrados pelos alunos é combinar conceitos abstratos encontrados na programação e os processos de raciocínio lógico. Trabalhos como de Hadar [11] e Xinogalos [30], que abordam o ensino de Programação Orientada a Objetos (POO) para iniciantes, apontam que uma das principais deficiências dos aprendizes de POO é conseguir relacionar os conceitos de forma lógica e racional. No trabalho de Hadar [11] é possível notar que muitos alunos não conseguem associar “objetos” ao enunciado proposto, por não entender a função básica do conceito de classe e sua instância (o próprio objeto). Na pesquisa de Xinogalos [30] é mostrado que a falta de compreensão de um conceito e sua correlação com os demais faz com que as dúvidas não sejam sanadas ao longo de todo o processo de aprendizagem.

A Aprendizagem Significativa (AS) é uma teoria que defende que a aquisição de conhecimento tem como premissa básica a influência do conhecimento prévio do aprendiz como sendo um fator fundamental. Segundo a AS, o indivíduo, no processo de aprendizagem, necessita organizar todo o conhecimento em uma estrutura cognitiva. Segundo Ausubel [4], essa estrutura cognitiva é uma organização, um construto, que se relaciona com algo concreto. A estrutura é particular para cada aprendiz, que, de uma forma individual, a cria para aprender de forma significativa. Moreira [16] afirma que essa estrutura é caracterizada pela interação entre os conhecimentos novos e aqueles relevantes já existente na estrutura cognitiva do sujeito que aprende. Para a AS, a interação do conhecimento prévio com novos conhecimentos e a relação entre eles deve ser organizada de maneira lógica para a composição da estrutura cognitiva do indivíduo.

Muitas pesquisas têm como objetivo buscar novos meios de proporcionar um ensino motivador. Este trabalho traz uma breve discussão sobre as motivações e desafios que podem ser associados à adoção da AS, sugerindo alguns recursos que podem ser utilizados para ajudar nesse propósito.

5.2 A Aprendizagem Significativa

A teoria da AS é pautada nos conhecimentos relevantes previamente adquiridos e existentes na estrutura cognitiva do sujeito. Alguns desses conhecimentos, relevantes para a aprendizagem de novos conhecimentos, podendo ser representados como um símbolo, imagem, conceito ou modelo mental, são definidos como ideia-âncora ou *subsunçor* [4] [16]. Moreira [17] define que “*subsunçor* é o nome que se dá a um conhecimento existente na estrutura de conhecimentos do indivíduo, que permite dar significado a um novo conhecimento que lhe é apresentado ou por ele descoberto”.

Na concepção do seu criador, David Ausubel, a AS é a aprendizagem por recepção significativa, ou seja, requer a apresentação de materiais e conceitos que sejam potencialmente significativos para o aprendiz. O material envolvido na aprendizagem tem que estar relacionado de forma não arbitrária (não aleatória, plausível e sensível), tendo um significado lógico ao aluno [4]. Moreira [16] complementa, afirmando que, no contexto da AS, deve haver uma interação de ideias e conhecimentos com aquilo que o aluno já sabe: não com qualquer conhecimento prévio, mas com algum conhecimento especificamente relevante existente na estrutura cognitiva do sujeito.

Darroz [7] afirma que a AS contrasta com a maneira tradicional, definida por Ausubel como sendo a aprendizagem mecânica, na medida em que, na primeira, uma nova informação interage com um *subsunçor* existente na estrutura cognitiva, e na segunda, a nova informação não interage com os conhecimentos já adquiridos, não contribuindo com a progressão e desenvolvimento do aprendiz. Segundo Moreira [16], mesmo que em alguns momentos a aprendizagem mecânica se faça necessária, ela deve objetivar uma aprendizagem mais significativa ao aprendiz.

5.3 A Aprendizagem Significativa e o ensino de Programação Orientada a Objetos

O paradigma POO pode ser todo organizado em conceitos, denotando diferentes *subsunçores* e suas relações. De acordo com a AS um problema que pode ocorrer no ensino de POO é não mostrar qual é a relação dos conceitos novos com aqueles que estão sendo apresentados no momento. Por exemplo, o *subsunçor* “classe” deve ser vinculado de maneira explícita ao

subsunçor “objeto” no momento de sua apresentação, para que o aprendiz consiga correlacionar os conceitos de forma natural e significativa.

Conforme a AS, a exposição de um conteúdo novo, sem estar associado ao anterior, pode não fazer sentido ao aluno. Segundo Astolfi e Junior [2], o educador tem como função promover mecanismos para que o aprendiz possa progressivamente complementar e enriquecer sua estrutura cognitiva, além de reorganizar e relacionar seus conhecimentos. A atuação do educador deve ser focada em apresentar conceitos mais genéricos primeiro, depois descrevendo os mais específicos, mostrando qual é a estrutura hierárquica do conhecimento e fortalecendo a consolidação da estrutura cognitiva do aluno.

A diferenciação progressiva, um dos princípios fundamentais da AS, é um processo muito adequado para o ensino de POO. Este princípio estabelece que a aprendizagem significativa deve ser um processo contínuo, no qual novos conceitos adquirem mais significância à medida que são alcançadas novas relações [18]. Os conceitos não são aprendidos de uma forma completa, mas sim são enriquecidos constantemente e modificados ao longo de diversas interações. Esse processo se assemelha muito com que identificamos como “experiência”. Ser experiente em algo denota que o sujeito consegue adquirir novos conhecimentos e associá-los aos que já estão incorporados, tornando-os mais explícitos, significativos e amplos, em um processo de contínua reorganização cognitiva. Moreira [17] reforça que a diferenciação progressiva é o processo de atribuir novos significados a um dado *subsunçor*, resultando em um *subsunçor* mais elaborado e diferenciado, servindo como âncora para novos conhecimentos

Quando um estudante de Computação que está aprendendo conceitos fundamentais de POO, e já compreende o conceito de “classe” (um *subsunçor*), por exemplo, quando for abordado o conceito de “herança” (novo *subsunçor*), o *subsunçor* “classe” irá adquirir mais significado, continuando a se modificar e sofrendo uma diferenciação progressiva baseada nessa nova relação. A cada vez que um aluno avança nos estudos, o *subsunçor* “classe” irá incorporar novos significados, dado a sua aplicação em novos contextos, por meio da experiência (uso contínuo e interativo) do *subsunçor* por parte do aluno.

A experiência do aluno está diretamente relacionada ao que ele realmente aprende, ou seja, quando ele reconhece de maneira autônoma novas relações conceituais. A reconciliação integradora é um processo que se refere à dinâmica da estrutura cognitiva, junto com a diferenciação progressiva, que consiste em eliminar ambiguidades, resolver inconsistências e criar uma ordem hierárquica na construção do conhecimento [17]. Ausubel [4] explica que é fundamental que o aluno identifique que os conceitos anteriormente aprendidos e presentes na

estrutura cognitiva são semelhantes, ou essencialmente diferentes, aos novos conceitos que estão sendo apresentados. Por exemplo, é de grande valia compreender que o *subsunçor* “classe” agrega significado tanto ao *subsunçor* “herança” quanto ao *subsunçor* “classe abstrata”, mas que “herança” e “classe abstrata” possuem ideias que as diferenciam, e não são obrigatoriamente complementares.

5.4 Desafios na adoção da Aprendizagem Significativa

Toda tentativa de criar novos métodos de ensino possui inúmeros desafios. Na estrutura de uma disciplina de programação, sempre há um início no qual deve-se pressupor que os alunos ainda não saibam o fundamental. Este seria o momento de apresentar um primeiro *subsunçor* a ser compreendido pelos alunos [2]. Também se faz necessário apresentar aos alunos, por meio de algum instrumento, de uma maneira menos abstrata e lógica, todas as relações de conceitos, contribuindo para a evolução de sua estrutura cognitiva.

Quando não há o conhecimento prévio ou não há *subsunçores* adequados, pode-se utilizar de organizadores prévios para fornecer o primeiro *subsunçor* necessário [4]. O professor, nesse caso, deve apelar para o meio tradicional de ensino, com o objetivo de tornar a aprendizagem significativa após a introdução desse primeiro *subsunçor* [15]. Na pesquisa de Pimentel e Osmar [20] são utilizados recursos de aprendizagem convencionais (aprendizagem mecânica) para dar uma aula introdutória de programação, para que os alunos, a partir disso, possam iniciar a estruturação e relação dos novos conceitos.

Outra importante etapa do processo de aplicação da AS no ensino de programação é definir como apresentar a relação e estrutura dos *subsunçores*. Novak e Gowin [18] baseiam sua pesquisa na adoção de mapas conceituais como instrumento para representar relações significativas no formato de proposições. A construção do conhecimento deve ser organizada e hierarquizada, de forma a oferecer uma compreensão mais simples dos relacionamentos entre os conceitos envolvidos [19].

Para um processo de aprendizagem de um novo paradigma de programação, ou mesmo de uma nova linguagem de programação, a organização dos conceitos envolvidos é fundamental para o entendimento de qual a relação de cada conceito aprendido em relação aos demais, facilitando a compreensão lógica. A percepção da relação entre os conceitos é fundamental para uma evolução progressiva e coerente por parte do aprendiz. Butler e Morgan [5] apontam uma fragilidade no processo de aprendizagem de programação, especificamente

na dificuldade que o aluno tem quando é requerido o pensamento abstrato, correlação de conceitos e as aplicações lógicas em resolução de problemas. Astolfi e Junior [3] mostram a eficácia em identificar conhecimentos prévios dos alunos para conduzir as práticas pedagógicas, minimizando as deficiências conceituais, denotando a necessidade em criar meios que trazer aos alunos ferramentas e métodos possam auxiliar logo do início da apresentação dos novos conceitos.

5.5 Computação Física como instrumento didático

Uma maneira de trazer um maior significado ao aluno é promovendo a diminuição da complexidade e da abstração dos conceitos envolvidos, criando meios para que o aprendiz possa fazer correlações com o mundo real, analogias com o cotidiano, interagindo com objetos reais. A Computação Física (CF) tem como objetivo proporcionar a conexão entre o mundo virtual e o real, por meio de interfaces intuitivas e programáveis entre objetos e seres humanos [21]. Segundo Stankovic et al. [25], a CF integra a computação com o mundo físico utilizando componentes eletrônicos, como sensores e atuadores, por meio de sistemas embarcados, comunicando-se por redes de computadores e criando ambientes automatizados. A Figura 5.1 ilustra um exemplo simples de um circuito contendo um LED, sendo controlado por uma placa Arduino.

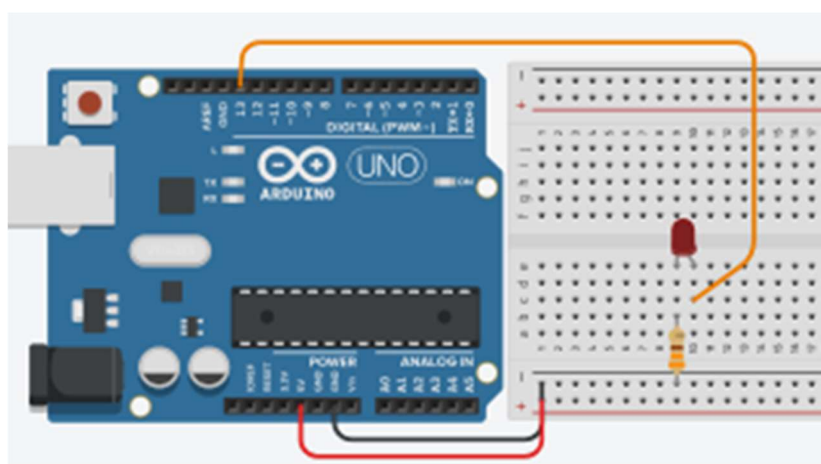


Figura 5.1 Circuito simples com Arduino e LED

O ambiente tangível da CF proporciona meios de diminuir as dificuldades de aprendizado, principalmente dos iniciantes em programação. Como mostra o trabalho de

Zanetti e Oliveira [31], a utilização de um robô móvel e uma arena de obstáculos ajudou a compreensão em conceitos básicos de programação, como estruturas condicionais (*if-else*) e de repetição (*for* e *while*). Anfurrutia et al. [1], também apresentam um trabalho usando Robótica Pedagógica em ensino básico de programação e programação visual, utilizando a linguagem de programação Java e conceitos fundamentais da POO, com resultados positivos no estímulo a resolução de problemas e construção de algoritmos,

Jang, Kim e Lee [13] apresentam um experimento, utilizando a plataforma Arduino, para ensinar o paradigma da POO, por meio de programação de pequenos circuitos com componentes eletrônicos, como LEDs (*Light Emitting Diode* ou Diodo Emissor de Luz), botões e sensor de luminosidade, fazendo analogias entre os objetos reais e as classes desenvolvidas utilizando linguagem de programação. Kafai et al. [14] apresenta uma pesquisa em que usa componentes eletrônicos em vestuário (Tecnologia Vestível ou *Wearable Technology*) para ensinar programação, mostrando projetos de vestimentas, representando suas características e interagindo com elas por meio de uma linguagem visual (*Scratch*), fazendo uma transição entre o concreto (contendo uma placa Arduino) e linguagem de programação.

A POO tem, como princípio fundamental, a racionalização e construção do software por meio de uma analogia a objetos. Usando componentes eletrônicos, é possível fazer um paralelo direto entre o objeto real e sua “representação” virtual (o código-fonte que o descreve). Por exemplo, ao se criar uma classe LED, pode-se definir seus atributos e métodos com base em sua constituição real e nas ações que se pode fazer com o componente.

Uma possível descrição de classe para o controle do LED presente no circuito da Figura 5.1 está apresentada na Figura 5.2. Os atributos cor, pino (pino que está conectado ao Arduino) e estado (ligado/desligado) representam valores que representam características reais do objeto junto ao sistema. Os métodos acender(), apagar() e piscar() representam ações reais que podem ser feitas com o LED.

LED
- Pino: int - Cor: String - Estado: boolean
- Acender() - Apagar() - Piscar()

Figura 5.2 Exemplo de uma classe LED

O uso de CF em educação tem seu primeiro registro nos anos 1970, tendo sido usado para explorar conceitos de Inteligência Artificial, pela exploração de um mundo físico e tangível [21] [23]. Os recursos da CF oferecem aos alunos várias maneiras de aprender computação (software, hardware, comunicação em rede e interação humano-computador), além de proporcionar práticas mais dinâmicas e motivadoras [22].

5.6 A teoria de *Concreteness Fading* e o ensino de programação

A teoria do *Concreteness Fading* (CoFa) de Bruner [6] propõe apoiar a transferência do conhecimento, facilitando as conexões entre as representações de conceitos a serem alcançados ao longo de uma progressão do mais concreto para o mais abstrato. Na CoFa, as representações concretas são utilizadas no primeiro momento, para que possa tornar o conceito mais acessível e tangível ao aluno, fornecendo uma conexão com o conhecimento prévio e com cenários realistas [12].

Além da transição do concreto para o abstrato, o uso de representações concretas pode servir como referência após a transição para as representações mais abstratas. Essas abstrações podem ser mais ambíguas ou simplificadas. Poder estar em contato com a representação concreta pode auxiliar na compreensão [8]. A transição de uma representação concreta para uma mais abstrata pode diminuir equívocos que poderiam surgir na análise do aluno se fosse baseada apenas na representação abstrata, pois com o objeto concreto, sua análise é mais direta e empírica [9].

Uma das principais habilidades demandadas do aluno iniciante em POO é a capacidade de abstração e a resolução de problemas. A CoFa pode proporcionar um meio de explorar o uso de representações concretas que sejam significativas aos alunos, promovendo uma maior compreensão sobre os conceitos aprendidos e dando mais autonomia na análise de problemas. Giraffa, Moraes e Uden [10] afirmam que, no processo de aprendizagem de programação, os alunos precisam imaginar e compreender termos abstratos que muitas vezes não possuem equivalentes na vida real. Wrenn e Krishnamurthi [29] utilizam em sua pesquisa ambientes de programação que mostram, em tempo real (por meio de uso de gráficos e animações), o que ocorre durante a execução de um código, criando um meio de relacionar o modelo mais abstrato de um programa com um modelo mais real e concreto.

No trabalho de Suh, Lee e Xia [27] é discutido que a utilização da teoria de CoFa no ensino de programação pode ser efetiva, ajudando na compreensão de conceitos abstratos,

desde fundamentos de lógica como variáveis e estruturas condicionais, até conceitos mais complexos, como recursividade e algoritmos de ordenação, que são comumente abordados em aulas relacionadas à programação.

A proximidade com o construto real pode estimular a independência e confiança do aluno em explorar o objeto conseguindo, desta forma, ter maior domínio sobre o problema ou conceito. Trory, Howland e Good [28] realizaram um experimento com crianças entre 9 e 10 anos para ensinar conceitos fundamentais para algoritmos de problema de caminho mínimo, utilizando uma maquete com linhas dispostas em localidades diferentes e arestas indicando as distâncias entre elas. É relatado que a presença de um cenário “real” análogo a um grafo possibilitou que crianças pudessem compreender a aplicação desses algoritmos, considerados complexos até mesmo dentro de cursos de Computação.

Ambos os experimentos supracitados, utilizam a abordagem do *framework* de 3 estágios definida por Bruner [6], na adoção do CoFa: ativo (*enactive*), que proporciona a interação com o ambiente; icônico (*iconic*), que utiliza uma representação mais visual e pictográfica, e; simbólico (*symbolic*), que utiliza uma linguagem e a apresentação abstrata. Fyfe et al. (2014) apresentam um modelo teórico da CoFa, definindo ações básicas que devem ocorrer em cada estágio. A Figura 5.3 apresenta o modelo resumido a adaptado do original.

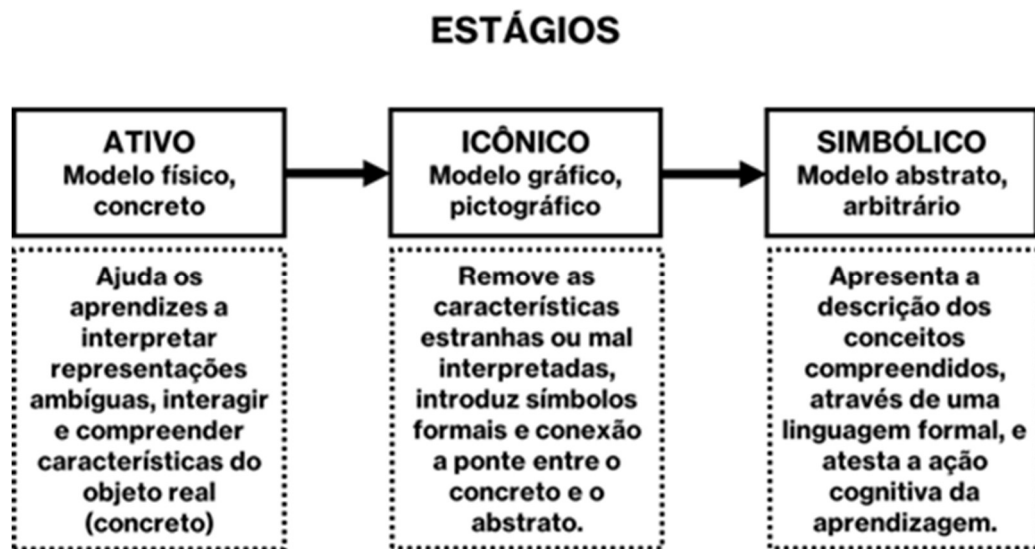


Figura 5.3 Modelo teórico do *Concreteness Fading* (adaptado de Fyfe et al. [8])

O trabalho de Suh, Lee e Law [26] apresenta uma revisão sistemática da literatura, na qual umas das questões que fundamentam a pesquisa está relacionada a adoção do *framework* de Bruner [6] e suas derivações, indicando que a adoção dos três estágios pode proporcionar melhores resultados. Este *framework* pode ser relacionado ao ensino de POO. Por exemplo, o

estágio de apresentação do objeto concreto poderia discutir e apresentar todas as características e ações possíveis que aquele objeto pode fazer, possibilitando ao aluno interagir com o objeto (estágio ativo). Em um segundo momento, pode-se levar esse levantamento preliminar das características do objeto concreto para o desenvolvimento de modelos gráficos, como um digrama de classes (estágio icônico). Por fim, após o desenvolvimento desses modelos gráficos, poderia ser iniciado o trabalho em código-fonte, utilizando uma linguagem de programação (estágio simbólico).

5.7 Computação Física e *Concreteness Fading* para promover a Aprendizagem Significativa

Recursos da CF são baseados em objetos tangíveis e interação com usuário, por meio da utilização de componentes e plataformas eletrônicos, podendo ser ferramentas a ser utilizadas como representações concretas. Usar construtos interativos e programáveis tem a potencialidade de criar atividades que sejam dinâmicas e motivadoras e que possam apoiar a transição à conceitos mais abstratos, como afirma o trabalho de Przybilla et al. [21]. Nesse mesmo trabalho são apresentadas as etapas da construção de uma prática envolvendo CF:

1. Configuração: preparar componentes de hardware, ferramentas e demais materiais;
2. Introdução e Inspiração: introduzir os conceitos fundamentais do hardware, apresentar o tema da prática e os conceitos básicos que serão usados para a programação;
3. Criação: formar grupos, incentivar brainstorms entre os alunos e idealizar projetos, com o professor apresentando sugestões sobre o hardware e programação;
4. Compartilhar e Refletir: apresentar os projetos elaborados, estimular os participantes a trocar experiências e relatar erros e acertos durante a realização da prática.

As etapas descritas acima podem ser adotadas em uma aplicação do primeiro estágio da CoFa (estágio ativo), para que o aprendiz tenha o contato com representação concreta. Em uma aula de introdução a POO, componentes da CF poderiam ser materiais de estudo para definição de conceitos como objetos e seus atributos e métodos. A facilidade em interagir com esses artefatos eletrônicos e robóticos pode colaborar com a exploração e

motivação, despertando o interesse do aluno, como corrobora os estudos realizados por Qiu et al. [24] e Pryzybilla [22].

Atualmente, existem plataformas acessíveis que facilitam o controle e programação de componentes eletrônicos, como as placas de prototipagem Arduino e Raspberry Pi. Um professor pode criar experimentos que se apoiem com construtos reais e concretos, podendo evoluir a apresentação dos conceitos para o meio mais abstrato. Com o uso da programação é possível criar uma maneira na qual o aluno veja os dois modelos (concreto e abstrato) juntos, com a transição sendo feita de imediato.

Em uma proposta didática, por exemplo, poderia ser criado um pequeno circuito no qual o professor poderia debater e expor as características do objeto, acendendo e apagando o LED, mostrando quais são seus atributos, como cor, estado (ligado/desligado) entre outros. Após isso, o professor apresenta um modelo mais abstrato, o diagrama de classe, referenciando o que foi apresentado na etapa anterior. Por fim, são usados recursos de uma linguagem de programação para transcrever e formalizar o modelo mais abstrato, que surge nessa transição gradual.

Considerando as premissas da AS, a influência do conhecimento prévio do aluno, o recurso de *subsunções* e a interação com novos conhecimentos para formar a estrutura cognitiva do indivíduo, unir os recursos da CF e a estratégia proposta pela CoFa pode proporcionar um método eficiente para promover a AS no ensino de POO. As barreiras que surgem para os aprendizes de POO, como a necessidade de abstrair conceitos e entender a relação entre objetos em código (como em um sistema com um ou mais objetos interagindo), podem ser desmotivadores para o aluno [10]. A analogia com componentes eletrônicos interagindo em um cenário real pode ser melhor compreendida pelo aluno.

Um exemplo do uso de analogias, seria um pequeno circuito como o apresentado na Figura 5.4, com um LED e um botão (tipo *push button*). A partir desse circuito, é possível sugerir como *subsunções* a apresentação das características e ações que os componentes possuem, mostrando os componentes reais e uma apresentação oral. Em um segundo momento, pode-se apresentar alguma relação entre os dois componentes, como o acender do LED por meio da interação do botão. Mesmo que o aluno não tenha conhecimento em eletrônica, ele pode entender os princípios básicos de cada um dos objetos envolvidos e compreender que eles podem “conversar”, “trocar mensagem”, interagir. Essa percepção, quando for apresentada em modelos mais abstratos, pode ter uma retenção mais adequada e clara por parte do aluno, por já ter experimentado e compreendido essas relações em um cenário real. Em uma prática usando esse cenário, seria possível compreender todos os estágios definidos pela CoFa e ainda utilizar os conceitos fundamentais da AS.

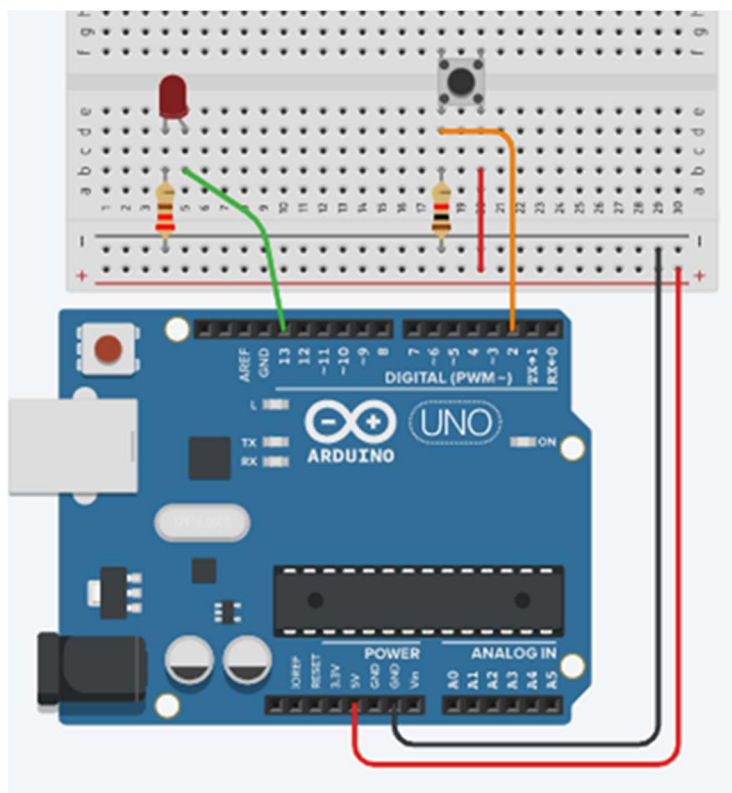


Figura 5.4 Exemplo de uma montagem com LED e botão

A Figura 5.5 apresenta um diagrama que ilustra um exemplo de prática e o processo de apresentação ao aluno. No estágio “ativo” pode-se abordar os *subsunçores* básicos (com os objetos reais) e interagir com um cenário real. No estágio icônico, podem ser discutidos os conceitos em diagramas e representações mais pictográficas. Por fim, no estágio simbólico, é introduzida a linguagem de programação e as estruturas básicas para o desenvolvimento do código-fonte.

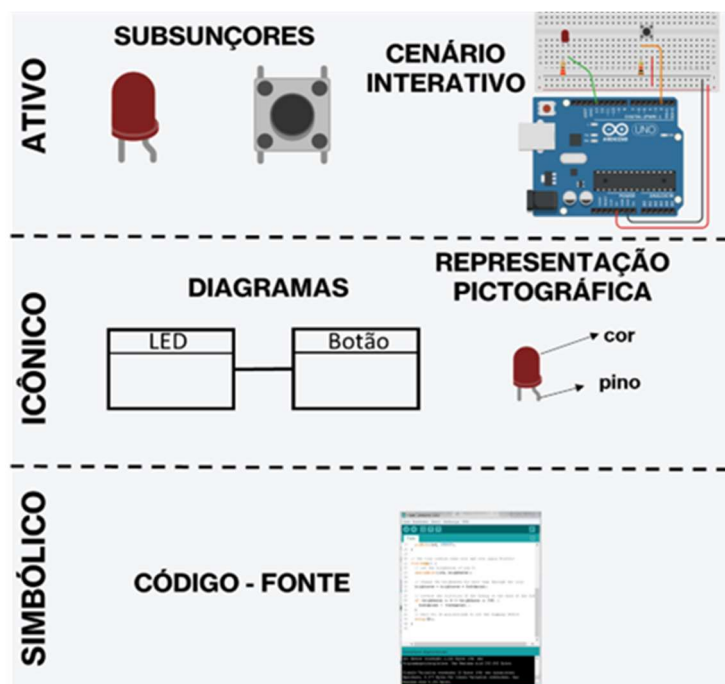


Figura 5.5 A relação dos estágios do Concreteness Fading e artefatos didáticos

5.8 Considerações finais

Há muitos desafios que envolvem o ensino de POO, principalmente aos iniciantes na área de Computação, com várias pesquisas sendo feitas sobre o tema. São discutidos desde aspectos motivacionais, até capacidade de abstração e construção cognitiva. Promover a AS em um cenário de ensino de programação pode ser adequado, se bem planejado e com recursos que possam incentivar o aluno a interagir com novos conhecimentos.

O objetivo deste artigo foi discutir temas relacionados à AS e sua aplicação no contexto de ensino de POO. O trabalho apresenta em quais aspectos a teoria de David Ausubel poderia ser adequada para fornecer um amparo metodológico e instrumental de como apresentar conceitos de programação. O trabalho também discute como potencializar a motivação do aluno diante de uma solução que pode trazer conceitos muito abstratos, por meio de uma correlação mais lógica e alcançável, baseada em conhecimentos pré-existentes.

Há desafios que devem ser superados na aplicação da AS. Os educadores e pesquisadores devem buscar novas soluções que permitam integrar os conceitos fundamentais da teoria da AS em suas aulas, seguindo o conteúdo didático para a formação do aprendiz. Recursos como os organizadores prévios e os mapas conceituais, podem ser instrumentos eficazes para estruturar um ambiente favorável para essa integração.

Cabe ao professor buscar a maneira mais adequada à uma aprendizagem mais significativa. Por exemplo, os materiais apresentados para discutir algum *subsunçor* podem ter um caráter menos abstrato, do que, por exemplo, quando apresentados diretamente em uma linguagem de programação (código-fonte). Essa prática não precisa ser feita de maneira isolada, podendo agregar outros métodos de ensino como CoFa, e utilizando recursos tecnológicos mais dinâmicos e interativos, como artefatos da CF.

As discussões apresentadas neste texto objetivam mostrar que a CF e a CoFa podem auxiliar na adoção da AS, oferecendo um cenário didático atraente ao aluno, seguindo as principais premissas definidas pela AS. A união dessas 3 linhas pode gerar instrumentos interessantes e efetivos para a dinâmica de ensino de POO, ensino que demanda meios de estimular e apoiar a capacidade de abstração dos alunos.

Baseado no *framework* de Bruner [6], os recursos da CF podem ser efetivamente adotados como instrumentos de criação dos objetos concretos necessários no início do processo (estágio ativo). Além disso, provêm recursos interativos e várias plataformas que facilitam sua programação e implementação de pequenos circuitos ou projetos.

Explorar o paradigma Orientado a Objetos diretamente em um modelo muito abstrato (como a linguagem de programação), pode ser frustrante ao aluno iniciante. Adotar um processo que se apoie na CoFa permite uma elaboração de uma prática pedagógica relevante e estruturada em etapas em que o objetivo seja a gradual compreensão de conceitos. O exercício de relacionar objetos reais, com suas características físicas e ações pode ser um suporte interessante para que se possa levar esses conceitos aos níveis mais abstratos.

5.9 Referências

- [1] Anfurrutia, F. I. et al. (2016). Incorporating educational robots and visual programming environments in introductory programming courses. In *2016 International Symposium on Computers in Education (SIIE)* (pp. 1 4). IEEE. DOI: 10.1109/SIIE.2016.7751835.
- [2] Astolfi, G., Junior, D. L. (2015). Investigação sobre conhecimentos prévios de alunos do curso Técnico em Informática a partir da aplicação de organizadores prévios. *Aprendizagem Significativa. In Revista/Meaningful Learning Review*, pp. 15-28.
- [3] Astolfi, G., Junior, D. L. (2016). Ensino de linguagem de programação com ênfase na aprendizagem significativa. In *Anais do XXIV Workshop sobre Educação em Computação* (pp. 111-120). SBC. DOI: <http://dx.doi.org/10.5753/wei.2016.9654>.

- [4] Ausubel, D. P. (2003). Aquisição e retenção de conhecimentos: uma perspectiva cognitiva. Lisboa: Plátano, 1.
- [5] Butler, M., e Morgan, M. (2007). Learning challenges faced by novice programming students studying high level and low feedback concepts. *Proceedings ascilite Singapore*, (99-107).
- [6] Bruner, J. S. (1966). Toward a theory of instruction. Cambridge, MA: Belknap Press of Harvard University Press.
- [7] Darroz, L. M. (2018). Aprendizagem significativa: a teoria de David Ausubel. *Revista Espaço Pedagógico*, 25(2), 576-580. DOI: <http://dx.doi.org/10.5335/rep.v25i2.8180>
- [8] Fyfe, E. R., McNeil, N. M., Son, J. Y., e Goldstone, R. L. (2014). Concreteness fading in mathematics and science instruction: A systematic review. *Educational psychology review*, 26(1), 9-25. DOI: <http://dx.doi.org/10.1007/s10648-014-9249-3>
- [9] Gentner, D., e Smith, L. A. (2013). Analogical learning and reasoning. Oxford University Press. DOI: <http://dx.doi.org/10.1093/oxfordhb/9780195376746.013.0042>
- [10] Giraffa, L. M., Moraes, M. C., e Uden, L. (2014). Teaching object-oriented programming in first-year undergraduate courses supported by virtual classrooms. In *The 2nd International Workshop on Learning Technology for Education in Cloud* (pp. 15-26). Springer, Dordrecht. DOI: http://dx.doi.org/10.1007/978-94-007-7308-0_2.
- [11] Hadar, I. (2013). When intuition and logic clash: The case of the object-oriented paradigm. *Science of Computer Programming*, 78(9), 1407-1426. DOI: <http://dx.doi.org/10.1016/j.scico.2012.10.006>.
- [12] Jaakkola, T., e Veermans, K. (2020). Learning electric circuit principles in a simulation environment with a single representation versus “concreteness fading” through multiple representations. *Computers & Education*, 148. DOI: <https://doi.org/10.1016/j.compedu.2020.103811>.
- [13] Jang, Y.; Lee, W.; Kim, J. (2015). Assessing the usefulness of object based programming education using Arduino. *Indian Journal of Science and Technology*, 8(S1), 89 96. DOI: 10.17485/ijst/2015/v8iS1/57701.
- [14] Kafai, Y. B., Lee, E., Searle, K., Fields, D., Kaplan, E., e Lui, D. (2014). A crafts-oriented approach to computing in high school: Introducing computational concepts, practices, and perspectives with electronic textiles. *ACM Transactions on Computing Education (TOCE)*, 14(1), 1-20. DOI: <http://dx.doi.org/10.1145/2576874>.

- [15] Moreira, M. A. (2003). Linguagem e aprendizagem significativa. *In Conferência de encerramento do IV Encontro Internacional sobre Aprendizagem Significativa*, Maragogi, AL, Brasil (Vol. 8).
- [16] Moreira, M. A. (2006). Aprendizagem Significativa: da visão clássica à visão crítica (Meaningful learning: from the classical to the critical view). *In Conferência de encerramento do V Encontro Internacional sobre Aprendizagem Significativa*, Madrid, Espanha.
- [17] Moreira, M. A. (2010). O que é afinal aprendizagem significativa? Aula Inaugural do Programa de Pós-Graduação em Ensino de Ciências Naturais, Instituto de Física, Universidade Federal do Mato Grosso, Cuiabá, MT.
- [18] Novak, J. D. e Gowin, D. B. (1995). Aprendendo a aprender. Lisboa: Plátano Edições Técnicas.
- [19] Novak, J. D. (2002). Meaningful learning: The essential factor for conceptual change *In limited or inappropriate propositional hierarchies leading to empowerment of learners*. Science education, 86(4), 548-571.
- [20] Pimentel, E., Omar, N. (2008). Ensino de algoritmos baseado na aprendizagem significativa utilizando o ambiente de avaliação Netedu. SBC, 79.
- [21] Przybylla, M.; Henning, F.; Schreiber, C.; Romeike, R. (2017). Teachers' Expectations and Experience in Physical Computing. *In International Conference on Informatics in Schools: Situation, Evolution, and Perspectives* (pp. 49 61). Springer, Cham. DOI: http://dx.doi.org/10.1007/978-3-319-71483-7_5.
- [22] Przybylla, M. (2018). From embedded systems to physical computing: Challenges of the “digital world” in secondary computer science education. Tese de doutorado. Universität Potsdam, Postdam, Alemanha.
- [23] Perlman, R. (1974). TORTIS (Toddler's Own Recursive Turtle Interpreter System). Massachusetts Institute of Technology, LOGO-9, v. 11, 1974.
- [24] Qiu, K., Buechley, L., Baafi, E., e Dubow, W. (2013). A curriculum for teaching computer science through computational textiles. *In Proceedings of the 12th international conference on interaction design and children* (pp. 20-27). DOI: <http://dx.doi.org/10.1145/2485760.2485787>.
- [25] Stankovic, J. A. et al. (2005). Opportunities and obligations for physical computing systems. Computer, 38(11), 23 31. DOI: 10.1109/MC.2005.386.

- [26] Suh, S., Lee, M., e Law, E. (2020). How do we design for concreteness fading? survey, general framework, and design dimensions. *In Proceedings of the Interaction Design and Children Conference* (pp. 581-588).
- [27] Suh, S., Lee, M., e Xia, G. (2020). Coding strip: A pedagogical tool for teaching and learning programming concepts through comics. *In 2020 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)* (pp. 1-10). IEEE.
- [28] Trory, A., Howland, K., & Good, J. (2018, June). Designing for concreteness fading in primary computing. *In Proceedings of the 17th ACM Conference on Interaction Design and Children* (pp. 278-288). DOI: <http://dx.doi.org/10.1145/3202185.3202748>.
- [29] Wrenn, J., e Krishnamurthi, S. (2019, July). Executable examples for programming problem comprehension. *In Proceedings of the 2019 ACM Conference on International Computing Education Research* (pp. 131-139). DOI: <http://dx.doi.org/10.1145/3291279.3339416>.
- [30] Xinogalos, S. (2015). Object-oriented design and programming: an investigation of novices' conceptions on objects and classes. *ACM Transactions on Computing Education (TOCE)*, 15(3), 1-21. DOI: <http://dx.doi.org/10.1145/2700519>.
- [31] Zanetti, H., e Oliveira, C. (2015). Práticas de ensino de programação de Computadores com Robótica Pedagógica e aplicação de Pensamento Computacional. *In Anais dos Workshops do Congresso Brasileiro de Informática na Educação* (Vol. 4, No. 1, p. 1236). DOI: <http://dx.doi.org/10.5753/cbie.wcbie.2015.1236>.

Capítulo 6

MÉTODO DE ENSINO DE PROGRAMAÇÃO ORIENTADA A OBJETOS BASEADA EM COMPUTAÇÃO FÍSICA, APRENDIZAGEM SIGNIFICATIVA E *CONCREteness FADING*

RESUMO

A habilidade de abstração é um dos requisitos principais para um aluno na fase inicial de aprendizagem em Programação Orientada a Objetos (POO). Um dos maiores desafios dos professores desse paradigma de programação é proporcionar ambientes que possam ajudar o aluno no exercício de abstração, de modo conciso e, ao mesmo tempo, motivador. A busca por meios que possam apoiar as práticas pedagógicas em POO estimula a pesquisa e criação de métodos que possam apoiar o trabalho do educador e proporcionar um cenário de aprendizagem estimulante ao aluno. Com essa demanda por recurso didáticos, foi criada a metodologia ComFAPOO (“Computação Física para Aprendizagem de Programação Orientada a Objetos), que se baseia nos objetivos didáticos-pedagógicos da teoria da Aprendizagem Significativa e no ferramental presente na Computação Física. Para o processo de aplicação, é utilizado o método de ensino *Concreteness Fading*, que objetiva apresentar conceitos através de uma transição fluída entre modelos concretos (construtos eletrônicos programáveis), passando por modelos representativos (digramas e gráficos), até modelos mais abstratos (código-fonte), auxiliando o aluno no processo de abstração e domínio do problema e solução.

PUBLICAÇÃO

Artigo publicado nos anais do evento XII Conferência Internacional de Tecnologias de Informação e Comunicação na Educação *Challenges 2021: Desafios do Digital*, sediado em Braga, Portugal. Este texto usa as normas e formatos de referências estabelecidos pelos anais da conferência. As numerações das seções, figuras e tabelas foram adaptadas do texto original para dar sequência à numeração desta tese.

6.1 Introdução

A habilidade de abstrair é essencial para que um programador tenha a capacidade de criar modelos, definir padrões de projetos e implementar soluções, criando modelos formais abstratos (a programação, por exemplo) (Kramer, 2007). Essa habilidade é uma das mais difíceis de se desenvolver entre os aprendizes de Programação Orientada a Objetos (POO). Abidin e Zawawi (2020) indicam, com sua pesquisa, que o uso de ambientes educacionais mais atrativos e dinâmicos pode auxiliar no processo de domínio e compreensão dos conceitos fundamentais do paradigma POO.

Para a compreensão do paradigma POO, é necessário que o aprendiz entenda o propósito de uma classe e sua instância, como uma representação de um modelo de algo real ou de um conceito. Knudsen e Madsen (1988) afirmam que no ensino de POO, o programa deve ser pensado com um modelo que simula uma parte real ou imaginária do mundo, sendo importante reconhecer a aplicação desse modelo, entender qual é a sua função. Analisando Borges (2000), Börstler, Bruce e Michiels (2003) e Woon e Bau (2017), entre outros, observa-se que embora existam propostas de soluções diferenciadas para introduzir o conceito de objeto no paradigma de POO em um contexto prático ou em um cenário real, ainda são comuns abordagens de ensino com foco na sintaxe de uma linguagem de programação.

Este trabalho tem como objetivo apresentar a método ComFAPOO (Computação Física para Aprendizagem de Programação Orientada a Objetos), que apresenta uma solução que integra o ferramental da Computação Física (CF), conceitos fundamentais da teoria da Aprendizagem Significativa (AS) e o *framework* de apresentação didática do *Concreteness Fading* (CoFa) para promover um aprendizado mais significativo e auxilie no processo de abstração.

6.2 Referencial teórico

Segundo Ausubel (2003), a AS é uma teoria que se guia pelas premissas de que a aquisição do conhecimento tem influência direta do conhecimento prévio do aprendiz e a interação com os novos conhecimentos deve ser organizada de maneira lógica, para uma composição cognitiva de maneira progressiva. Nessa teoria, a estrutura cognitiva do indivíduo pode ser formada

através da interação com novos conhecimentos e com aqueles relevantes previamente existentes (Moreira, 2008).

Ausubel (2003) define que os conhecimentos relevantes previamente adquiridos pelo aluno podem ser expostos como símbolos, imagens, conceitos ou modelos mentais, sendo chamados de ideia-âncora ou subsunçor. Caso o aprendiz não tenha o *subsunçor* adequado, o professor deve apresentar o conceito necessário para que possa ser criada essa “âncora” que irá apoiar os demais conhecimentos (Moreira, 2008). Essa ideia de *subsunçor* pode ser aplicada no aprendizado de programação, no qual sempre há um conhecimento inicial que se deve pressupor que os alunos ainda não saibam, sendo esse o primeiro *subsunçor* (Astolfi e Junior 2016).

A CF reúne um conjunto de recursos que integram componentes de hardware (mecânico, eletrônico) e software (programáveis). Segundo Desportes (2018), esses recursos permitem a criação de soluções que podem interagir com o aluno e o mundo ao seu redor, levando a um maior engajamento, estimulando a criatividade e a aprendizagem baseada em projetos. A natureza tangível e o incentivo à exploração permitem a integração de várias soluções do domínio da Computação em um ambiente experimental e motivador (Deitrick et al., 2015).

O CoFa é um *framework* instrucional baseado na apresentação de conceitos complexos, através de uma transição gradual (*fading*) de representações mais concretas para mais abstratas (Suh, Lee e Law, 2020). Segundo Bruner (1966), a progressão gradual da representação concreta para a mais abstrata possibilita ao aprendiz conhecer o conceito com algo mais representativo e tátil (concreto), podendo interagir com ele, antes de iniciar a utilização de interpretações abstratas. Bruner apresenta o CoFA (Figura 6.1) como sendo formado por três estágios distintos que descrevem essa progressão gradual das representações: “ativo” (*enactive*), “icônico” (*iconic*) e “simbólico” (*symbolic*).

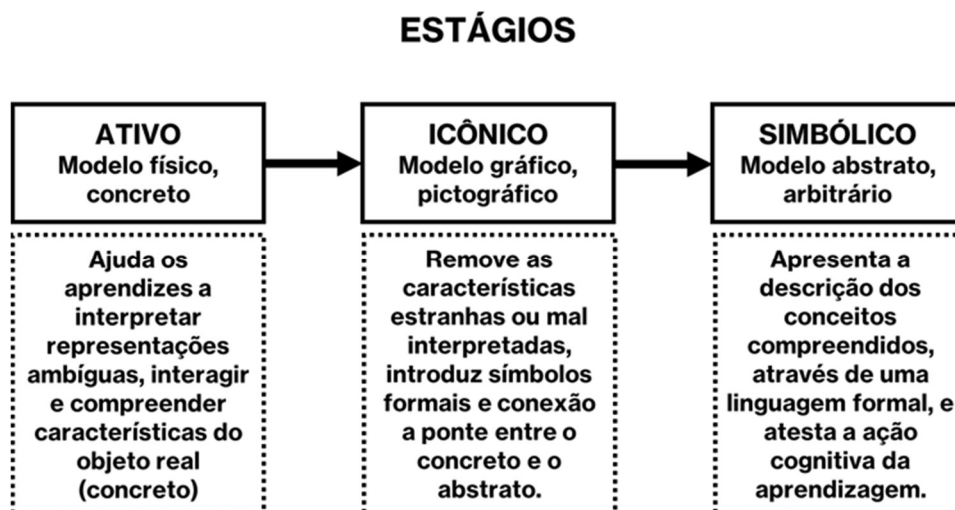


Figura 6.1 Os estágios do CoFa de Bruner (adaptado de Fyfe et al. (2014))

6.3 O método ComFAPOO

O método ComFAPOO tem como objetivo servir como base para a criação de práticas didáticas que possam estimular a interação com objetos reais programáveis, a aplicação dessas atividades junto aos alunos e uma avaliação, para compreender qual o impacto na construção do conhecimento. O método está estruturado em 4 etapas: Concepção, Desenvolvimento, Aplicação e Avaliação. A etapa Concepção está diretamente relacionada aos objetivos defendidos pela AS, enquanto as etapas Desenvolvimento e Aplicação utilizam o modelo teórico da CoFa e recursos da CF. A Figura 6.2 apresenta a estrutura do ComFAPOO, suas etapas e atividades.

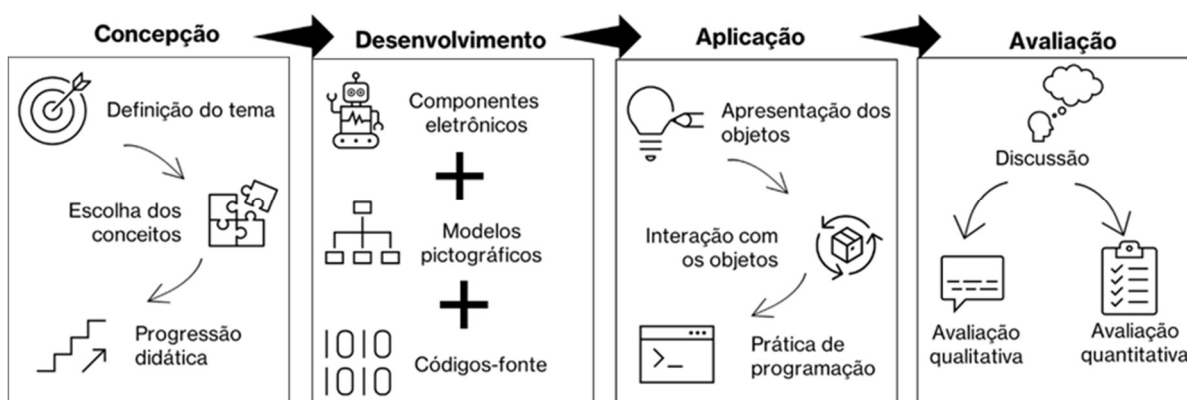


Figura 6.2 A estrutura do método ComFAPOO

A etapa de Concepção define quais são objetivos a serem alcançados em uma aula ou prática didática específica. O tema dessa prática pode abordar mais de um conceito de programação, esclarecendo a relação dos conceitos abordados. As atividades são:

- Definição do tema: compreender e mapear quais são os principais objetivos a serem alcançados com a prática, levando em consideração o contexto didático;
- Escolha dos conceitos: determinar quais serão os conceitos abordados e se eles são complementares ou relacionáveis.
- Progressão didática: é necessário mapear a sequência de apresentação e as relações diretas entre os conceitos que serão abordados.

Na etapa de Desenvolvimento, devem ser definidos os instrumentos que serão usados e o contexto de apresentação dos conceitos escolhidos na etapa anterior. As atividades dessa etapa são:

- Componentes eletrônicos: a seleção de quais componentes concretos, e quais construtos eles irão formar, deverá seguir a definição da progressão estipulada na etapa anterior;
- Modelos pictográficos: seguindo o *framework* CoFa (estágio icônico), a criação de representações pictográficas intermediárias, entre o objeto concreto e sua concepção mais abstrata, deve refletir não somente os conceitos a ser apreendidos, mas também o “comportamento” real do objeto, conduzindo a abstração do aluno a uma representação menos tangível;
- Códigos-fonte: no último estágio da CoFa (simbólico), haverá a abordagem direta de código-fonte. O educador deve criar exemplos que podem explorar de forma gradativa a transição do modelo anterior para um mais abstrato.

Na etapa de Aplicação ocorre a condução da atividade pelo professor junto ao alunado. Nessa etapa, o professor irá seguir a progressão definida na etapa de Concepção, usando todos os recursos definidos na etapa de Desenvolvimento. As atividades são:

- Apresentação dos objetos: consiste em apresentar todos os objetos e construtos que serão utilizados, de uma maneira mais técnica, mas não aprofundada. Essa apresentação tem como objetivo fazer com que o aluno entenda o comportamento geral do objeto;
- Interação com os objetos: o educador deve aprofundar a discussão sobre o objeto, fazendo associações entre suas características e seu comportamento, de uma maneira análoga aos atributos e métodos de uma classe, mostrando uma

transição do objeto real para uma representação mais abstrata e interagindo com o objeto;

- Práticas de programação: apresenta o modelo mais abstrato, o código-fonte. Esse modelo teve uma transição que pode situar o aluno em um cenário de maior domínio, por conta das atividades anteriores. A apresentação do código-fonte também pode ser de forma gradual.

A etapa de Aplicação do ComFAPOO usa o *framework* CoFa. A Figura 6.3 ilustra uma prática, com apresentação de objetos concretos, suas representações pictográficas e, finalmente, a representação mais abstrata, o código-fonte.

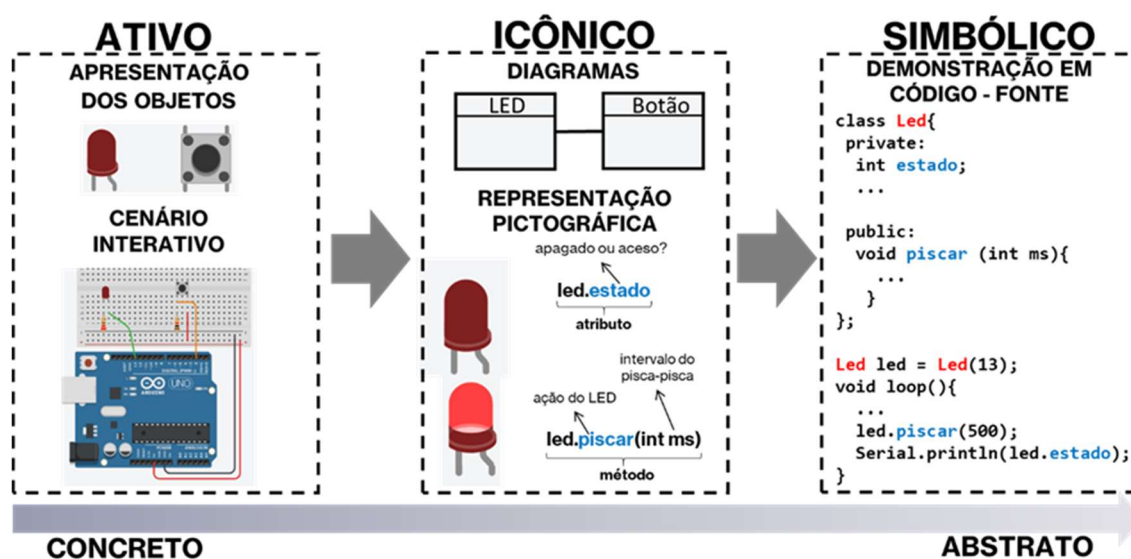


Figura 6.3 Exemplo de aplicação do ComFAPOO seguindo o *framework* CoFa

Na última etapa do ComFAPOO, a de **Avaliação**, é feita uma avaliação do domínio dos conceitos apresentados. Essa etapa não está diretamente relacionada aos 3 pilares do ComFAPOO (AS, CF e CoFa), mas é fundamental para validação do método. Essa avaliação não objetiva pontuar erros e acertos, mas sim o envolvimento do aluno na construção do conhecimento. Ao final de cada prática, o professor deverá conduzir uma breve discussão (atividade Discussão) e depois aplicar duas avaliações:

- Avaliação quantitativa através Autoavaliação de Emoções (SAM - *Self-Assessment Manikin*): segundo Bradley e Lang (1994), esse método consiste em um sistema baseado em classificação pictográfica (com uma escala gradativa), para avaliar diretamente a Satisfação, Motivação e Controle em resposta à prática;

- Avaliação qualitativa: será apresentado aos alunos um questionário com perguntas dissertativas e abertas, para que se possa identificar suas particularidades e experiências individuais, que possam apontar problemas e direcionar possíveis melhorias.

6.4 Conclusões

Este texto tem como objetivo descrever o método ComFAPOO, que tem como principal objetivo melhorar a eficácia do ensino de POO. Para tal, o ComFAPOO baseia-se em três pilares: AS, CF e CoFA. A AS é muito estudada no ensino básico e fundamental, mas há poucos trabalhos que a aplicam no contexto de aprendizagem de programação para a formação profissional. Ao trazer significado ao ensino, objetiva-se fazer com que o aluno tenha apreço por aquilo que está descobrindo e vivenciando. A CF como recurso didático oferece instrumentos que podem trazer reconhecimento e valor por parte de quem aprende. O *framework* CoFa possibilitou agregar um procedimento organizado e sistemático para amparar o professor no desenvolvimento e aplicação de práticas didáticas.

Devido às restrições causadas pela pandemia mundial de Covid19, ainda não foi possível aplicar o ComFAPOO com alunos para uma análise prática do método. Realizar uma análise em processos de ensino não presenciais, como os que atualmente estão sendo conduzidos, para um contexto que propõe a CF e o CoFA, seria contraditório às próprias premissas da pesquisa.

6.5 Referências

- Abidin, Z. Z., & Zawawi, M. A. A. (2020). OOP-AR: Learn Object Oriented Programming Using Augmented Reality. *International Journal of Multimedia and Recent Innovation*, 2(1), 60-75.
- Astolfi, G., & Junior, D. L. (2016). Ensino de linguagem de programação com ^Enfase na aprendizagem significativa. In *Anais do XXIV Workshop sobre Educação em Computação* (pp. 111-120). SBC.
- Ausubel, D. P. (2003). Aquisição e retenção de conhecimentos: uma perspectiva cognitiva. *Lisboa: Plátano, 1*.

- Borges, M. A. (2000). Avaliação de uma metodologia alternativa para a aprendizagem de programação. In *VIII Workshop de Educação em Computação–WEI* (No. 8, p. 15).
- Bradley, M. M., & Lang, P. J. (1994). Measuring emotion: the self-assessment manikin and the semantic differential. *Journal of behavior therapy and experimental psychiatry*, 25(1), 49-59.
- Bruner, J. S. (1966). *Toward a theory of instruction* (Vol. 59). Harvard University Press.
- Börstler, J., Bruce, K., & Michiels, I. (2003). Sixth workshop on pedagogies and tools for learning object oriented concepts. In *ECOOP* (Vol. 17, pp. 84-87).
- Deitrick, E., Shapiro, R. B., Ahrens, M. P., Fiebrink, R., Lehrman, P. D., & Farooq, S. (2015). “Using distributed cognition theory to analyze collaborative computer science learning”. In *Proceedings of the eleventh annual International Conference on International Computing Education Research* (pp. 51-60).
- Desportes, K. S. (2018). *Physical computing education: Designing for student authorship of values-based learning experiences* (Doctoral dissertation, Georgia Institute of Technology).
- Fyfe, E. R., McNeil, N. M., Son, J. Y., e Goldstone, R. L. (2014). Concreteness fading in mathematics and science instruction: A systematic review. *Educational psychology review*, 26(1), 9-25
- Knudsen, J. L., & Madsen, O. L. (1988). Teaching object-oriented programming is more than teaching object-oriented programming languages. In *European Conference on Object-Oriented Programming* (pp. 21-40). Springer, Berlin, Heidelberg.
- Kramer, J. (2007). Is abstraction the key to computing?. *Communications of the ACM*, 50(4), 36-42.
- Moreira, M. A. (2008). Organizadores previos y aprendizaje significativo. *Revista chilena de educación científica*, 7(2), 23-30.
- Suh, S., Lee, M., & Law, E. (2020). How do we design for concreteness fading? survey, general framework, and design dimensions. In *Proceedings of the Interaction Design and Children Conference* (pp. 581-588).
- Woon, H. C., & Bau, Y. T. (2017). Difficulties in Learning C++ and GUI Programming with QT Platform: View of Students. In *Proceedings of the 2017 International Conference on E-commerce, E-Business and E-Government* (pp. 15-19).

Capítulo 7

APLICAÇÃO DE UM MÉTODO PARA ENSINO DE PROGRAMAÇÃO ORIENTADA A OBJETOS POR MEIO DE APRENDIZAGEM SIGNIFICATIVA E COMPUTAÇÃO FÍSICA

RESUMO

Este artigo apresenta um relato de experiência e seus resultados sobre uma oficina de ensino de Programação Orientado a Objetos por meio de um método que utiliza premissas da Aprendizagem Significativa e Computação Física, ComFAPOO (Computação Física para Aprendizagem de Programação Orientada a Objetos). Este método propõe uma estratégia de ensino que busca amenizar as principais dificuldades de alunos iniciantes no ensino paradigma orientado a objetos. Os resultados de sua aplicação mostram que o método auxiliou os alunos participantes na compreensão de conceitos fundamentais da Programação Orientado a Objetos.

PUBLICAÇÃO

Artigo publicado nos anais do evento do XXX Workshop sobre Educação em Computação – 2022 (WEI 2022). Este texto usa as normas e formatos de referências estabelecidos pelos anais do workshop. As numerações das seções, figuras e tabelas foram adaptadas do texto original para dar sequência à numeração desta tese.

7.1 Introdução

O ensino de Programação Orientada a Objetos (POO) ainda é um desafio, sendo foco de pesquisas recentes, principalmente sobre as dificuldades que são encontradas no processo [Gutiérrez, Guerrero e López-Ospina 2022; Abassi et al. 2021]. Desde o início do ensino desse paradigma, pesquisadores já relatavam problemas encontrados por seus alunos, como aponta o trabalho de Knudsen e Madsen (1988), demonstrando que ensinar POO é mais do que ensinar linguagens e as técnicas de programação, mas sim ensinar seus conceitos fundamentais. Beck e Cunningham (1989) apresentam uma reflexão sobre o ensino de programação e a dificuldade de seus alunos em migrar de um paradigma procedural para um com maior nível de abstração, como o paradigma POO.

O professor encontra seu primeiro desafio na criação de uma proposta de um currículo que possa abordar os fundamentos do paradigma POO. Bennedsen e Schulte (2007) fazem uma análise através de entrevistas com diversos professores e pesquisadores ao redor do mundo, não apresentando uma concepção única sobre o que é o ensino básico de POO, mas identificando como consenso que, no início do aprendizado, o aluno deve entender como usar o “objeto” antes de implementar qualquer tipo de técnica.

O método ComFAPOO (Computação Física para Aprendizagem de Programação Orientada a Objetos) foi criada com o objetivo de apoiar os alunos a enfrentar essas dificuldades no processo de aprendizado de POO [Zanetti, Borges e Ricarte 2021]. Este artigo apresenta o relato de experiência na aplicação de oficinas de ensino de POO usando o método ComFAPOO, com 14 alunos de um curso técnico em Informática, com o objetivo avaliar seus efeitos e sua viabilidade como instrumento de ensino. A Seção 7.2 apresenta uma revisão bibliográfica sobre as dificuldades encontradas no ensino de POO. A Seção 7.3 apresenta as referências utilizadas para definir os fundamentos de POO que foram apresentados durante a oficina. Na Seção 7.4 é apresentada uma breve descrição do método. A Seção 7.5 descreve o perfil dos alunos e a dinâmica de aplicação da oficina. A Seção 7.6 apresenta os resultados e discussões sobre eles. Por fim, a Seção 7.7 apresenta as considerações finais.

7.2 Dificuldades encontradas no ensino de Programação Orientada a Objetos

Há muitos desafios associados ao ensino de POO. Gutiérrez, Guerrero e López-Ospina (2020) mostram que as dificuldades dos alunos vão desde a aplicação de conceitos básicos de programação, a relação entre classe e objeto e o desenvolvimento abstrato, até conceitos mais avançados, como não compreender a aplicação de herança, polimorfismo e reuso de código. Xinogalos (2015) reforça que as dificuldades primárias se concentram em compreender os conceitos de classe e objeto, fazendo com que o início do processo de aprendizagem seja desmotivador. Liberman, Beeri e Kolikant (2011) apresentam uma análise sobre as dificuldades dos alunos em compreender herança e polimorfismo, ressaltando que o principal problema é a falta de relação da aplicação desses conceitos em um cenário real.

Assim como em qualquer processo de aprendizagem, deficiências encontradas logo no início da aprendizagem de POO pode levar a uma cadeia de novos problemas de aprendizado, sendo intensificado a cada conceito novo apresentado. Logo nos conceitos iniciais de POO, o aluno já é estimulado a utilizar a habilidade de abstração e, ao mesmo tempo, representar um modelo em código. Lian, Varoy e Giacaman (2022) e Abbasi et al. (2021) apresentam, em suas pesquisas, uma listagem de equívocos e dificuldades comuns em cursos de POO, havendo ocorrências tanto com conceitos mais básicos quanto com os mais avançados.

Bennedsen e Caspersen (2006) mostram que mesmo alunos que possuem uma habilidade de abstração alta (observada por avaliações matemáticas) não apresentam uma habilidade de programação de mesmo nível. É apontado que, muitas vezes, o curso de programação tem o foco muito no instrumento (linguagem) e pouco no projeto (conceitos). Möstrom et al. (2008) argumentam que o problema para os alunos não é a capacidade de abstração, mas sim, a contextualização desses modelos e objetos em um cenário que faça sentido.

Conceitos abstratos podem ser melhor absorvidos se aplicados em cenários realistas ou envolverem objetos reais, que possam trazer interação e significado. O uso de ferramentas instrucionais que não sejam apenas baseadas em código-fonte pode amenizar a compreensão dos conceitos até mais avançados. Em seu trabalho, Möstrom et al. (2008) utilizaram objetos concretos como base para o exercício de abstração de conceitos de Computação. Trory, Howland e Good (2018) demonstram que o uso de objetos concretos, que podem ser palpáveis e interativos, auxiliam no exercício de abstração de conceitos matemáticos e computacionais.

7.3 Definindo os fundamentos da Programação Orientada a Objetos

Definir uma estrutura curricular para uma aula não é uma tarefa trivial. O ensino do paradigma POO ainda tem várias dúvidas, associadas a quais conceitos abordar e qual sequência didática é mais adequada. É possível notar que, desde o início da adoção do paradigma em cursos de Computação, a discussão sobre seus fundamentos e métodos de ensino é abordada na literatura. Pugh, LaLonde e Thomas (1987) discutem, em seu trabalho, o impacto da inserção do paradigma POO em cursos de Computação em meados dos 1980, com ênfase no trabalho do educador e como adequar um currículo para os iniciantes.

Uma análise mais estruturada é apresentada por Rosson e Alpert (1990), que definem *Abstração*, *Classe*, *Encapsulamento*, *Ocultamento de Informação*, *Herança*, *Instância*, *Passagem de Mensagem*, *Abstração de dados*, *Método*, *Objeto*, *Modelagem de Objeto* e *Polimorfismo* como os principais conceitos que compõem o paradigma de POO. A análise taxonômica feita por Henderson-Sellers e Edwards (1994) é baseada em um “Triângulo da Orientação a Objetos”, sendo que em cada um dos vértices do triângulo estão os conceitos de *Encapsulamento e Ocultamento de Informação* (*Encapsulation and Information Hiding*), *Abstração* (*Abstraction*) e *Herança e Polimorfismo* (*Inheritance and Polymorphism*).

Armstrong (2006) faz uma análise sobre quais são os fundamentos da POO, através de uma investigação em 239 trabalhos na área pelo termo “desenvolvimento orientado a objetos” (*object-oriented development*). Como resultado, foram definidos 30 conceitos diferentes, e foi verificado que havia conceitos que eram citados em mais de 50% dos estudos, sendo a base de sua taxonomia. Nessa taxonomia, os conceitos foram divididos em “estrutura” (*structure*) e “comportamento” (*behavior*). O grupo “estrutura” é composto por *Abstração*, *Classe*, *Encapsulamento*, *Herança* e *Objeto*, enquanto do grupo “comportamento” é composto por *Método*, *Passagem de Mensagem* e *Polimorfismo*.

Baseando-se nas propostas de Rosson e Alpert (1990), Henderson-Sellers e Edwards (1994) e Armstrong (2006) sobre os fundamentos do paradigma POO, é possível definir uma proposta de currículo que contemple os conceitos fundamentais de POO. Para o método ComFAPOO, foi adotada a taxonomia proposta por Armstrong (2006), que se baseia nas de Rosson e Alpert (1990) e Henderson-Sellers e Edwards (1994), apresentando um embasamento amparado por uma pesquisa sobre uma extensa coleção de trabalhos acadêmicos.

7.4 O método ComFAPOO

O método ComFAPOO apresenta uma proposta de ensino de POO utilizando as premissas da Aprendizagem Significativa (AS). A AS é uma teoria criada por David Ausubel que se guia pela apresentação gradativa de conceitos para a aquisição do conhecimento por parte do aluno [Ausubel 2003]. Segundo essa teoria, a estrutura cognitiva do indivíduo é formada pela interação de novo conhecimentos com aqueles que ele já possui ou que pode ser apresentado inicialmente pelo professor (ideia-âncora ou *subsunçor*) [Moreira 2006].

A ComFAPOO também usa os recursos Computação Física (CF) e o design instrucional do *Concreteness Fading* (CoFa). Segundo Desportes (2018), a CF utiliza um conjunto de recursos de hardware que podem ser programáveis, podendo trazer soluções que estimulem o aluno a interagir com o mundo ao seu redor, engajando e estimulando a criatividade e a resolução de problemas por meio de projetos. O *framework* instrucional CoFa segue a premissa da progressão gradual da representação de algo concreto para mais abstrato, fornecendo ao aluno algo mais representativo e tátil antes de apresentar formas mais abstratas [Bruner 1996]. A ComFAPOO é baseada em quatro etapas (*Concepção*, *Desenvolvimento*, *Aplicação* e *Avaliação*), apresentadas com detalhes em Zanetti, Borges e Ricarte (2021). Essas etapas são apresentadas na Figura 7.1.

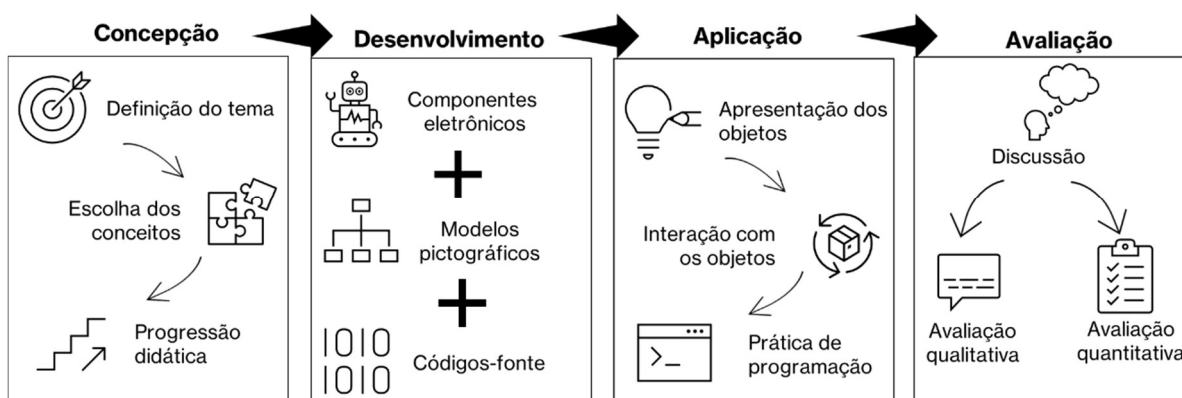


Figura 7.1 Etapas do método ComFAPOO [Zanetti, Borges e Ricarte 2021].

Segundo Bradley e Lang (1994), o método SAM consiste em um sistema baseado em classificação pictográfica com o objetivo de avaliar diretamente a Satisfação, Motivação e Domínio em resposta a um evento. Essa avaliação foi integrada à ComFAPOO, permitindo que

se possa identificar fatores emocionais relacionados à prática didática realizada. A Figura 7.2 mostra as escalas utilizadas em questionários do método SAM.

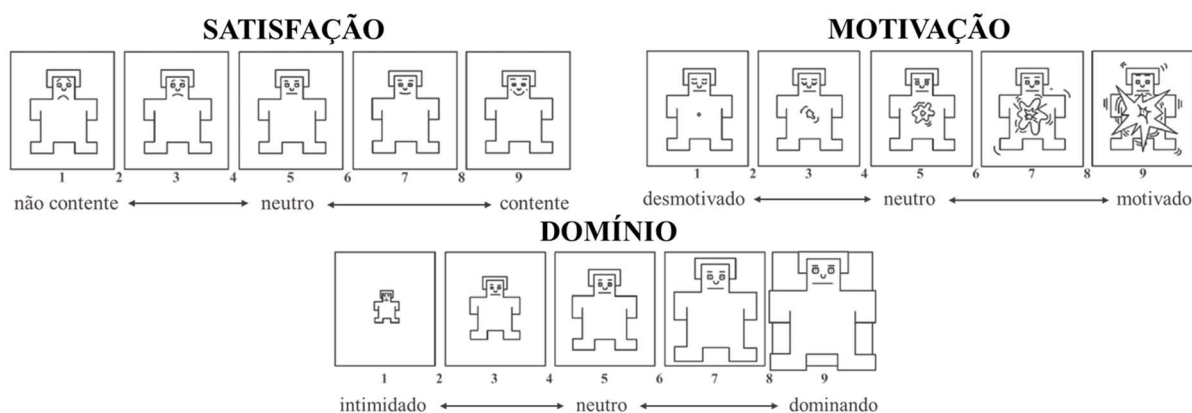


Figura 7.2 Escalas para os questionários. Adaptado de Soares et al. (2013)

Como instrumento de avaliação, também é proposto um questionário para averiguar a percepção dos alunos com relação aos conceitos fundamentais de POO antes e depois das práticas (pré e pós). Esse questionário é composto por afirmações que devem ser respondidos por meio da escala Likert, que associa os valores de 1 a 5 às relações de discordância (*discordo totalmente* ou *discordo parcialmente*), neutralidade (*indiferente*) e concordância (*concordo totalmente* ou *concordo parcialmente*) [Severino Júnior e Costa 2014].

Após a aplicação do questionário inicial, as oficinas que compõem a ComFAPOO têm início com a apresentação de objetos concretos. A ComFAPOO sugere a aplicação de oficinas utilizando a plataforma de Arduino, desenvolvida para facilitar o desenvolvimento e programação de sistemas automatizados [McRoberts 2018]. O Arduino é utilizado tanto para facilitar a demonstração das propriedades e ações dos objetos que serão apresentados, como para as práticas de programação.

No início de cada oficina são introduzidos os objetos que serão abstraídos, através da apresentação de suas propriedades e ações como, por exemplo, mostrar que um LED (*Light-Emitting Diode*) muda de estado (aceso e apagado) e pode ter as ações de “acender” e “apagar”. Depois da apresentação do objeto, é discutido quais são as propriedades que esse objeto pode ter ou assumir, fazendo um exercício de abstração, conduzindo a discussão da construção da classe que representa o objeto. Por fim, é exibido o código-fonte que representa esses comportamentos e características e se faz uso dos objetos abstratos, ou seja, os objetos na programação.

Como exemplo, a prática que se baseia na apresentação do objeto LED para demonstrar a relação de classe e objeto, se inicia com a demonstração de quais características que o LED possui, como seu estado (ligado ou desligado), assim como ações que ele pode realizar, como acender, apagar e piscar. Além da demonstração real do objeto e a percepção disso pelo aluno, também é utilizado material gráfico, como sendo um primeiro estágio de transição do objeto concreto para uma representação mais abstrata. Após esse levantamento de características e ações, é feita a representação da classe (representação mais abstratas), através de seus atributos e métodos. Após essa transição, é demonstrado um exemplo de código-fonte que represente a classe, mostrando suas relações com os modelos anteriores. Por fim, é feita a instância do objeto, fazendo com que aquele modelo abstrato controle o objeto real, ilustrando o uso de seus atributos e métodos.

A aplicação de cada uma das partes da ComFAPOO, além de seguir uma sequência gradativa de conceitos conforme recomenda a AS, também de baseou no *framework* CoFa para a apresentação dos objetos concretos até sua representação mais abstrata. O estágio *Ativo* da CoFa auxilia os alunos a interpretar, interagir e compreender as características de um objeto concreto (real); o estágio *Icônico* tem como objetivo retirar características que se desnecessárias ou mal interpretadas, e introduz símbolos formais e faz a ponte entre a representação mais concreta e a mais abstrata; e o estágio *Simbólico*, o mais abstrato, apresenta os conceitos compreendidos usando uma linguagem mais formal e atesta a ação cognitiva da aprendizagem [Bruner 1996]. A Figura 7.3 apresenta um exemplo da apresentação do objeto LED relacionando as etapas do *framework* CoFa.

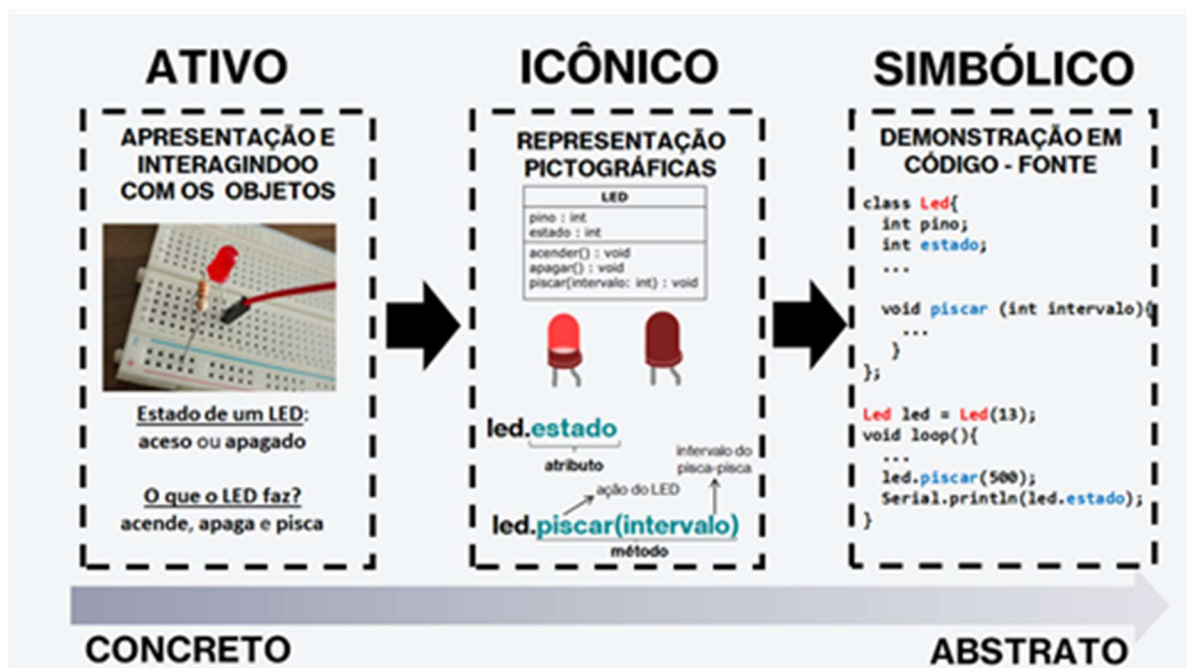


Figura 7.3 A aplicação do framework CoFa. Adaptado de Zanetti e Borges [2021].

A oficina do método ComFAPOO é definida com uma divisão do conteúdo em cinco partes. Cada uma das partes apresenta o conteúdo de forma gradual, correlacionando os conceitos e trazendo significado, seguindo as orientações da AS. Cada uma dessas partes é estruturada para apresentar um ou mais conceitos da POO e, entre elas, também foi planejada uma progressão gradual entre esses conceitos. A escolha dos conceitos para estruturar o currículo da oficina foi baseada nas análises apresentadas na Seção 7.3 deste artigo. A Tabela 7.1 descreve de maneira breve os conceitos de cada uma das partes e os objetos que são utilizados.

Tabela 7.1 Relação de conceitos e objetos usados durante a oficina.

Parte	Conceitos abordados	Objetos utilizados
1	Abstração, classe e objeto	LED e botão
2	Encapsulamento	LED e botão
3	Herança	Sensor de luminosidade e sensor de temperatura
4	Polimorfismo	LED e botão
5	Troca de mensagens entre objetos	Semáforo (LED e sensor de luminosidade)

Em cada uma das partes da oficina são realizadas montagens de circuitos com diferentes componentes eletrônicos, para que os alunos possam ver o resultado da programação em tempo real atuando com o objeto real. Também é estimulada a realização de tarefas que

demandam a adição de novos componentes e alterações no código-fonte, para que possam experimentar tanto o trabalho com os objetos concretos como com sua programação.

O questionário aplicado antes da aplicação das oficinas (pré-teste) é formado por 12 questões para serem respondidas por meio da escala Likert, variando de *discordo totalmente* (1) a *concordo totalmente* (5). Essas questões têm como objetivo determinar a percepção de domínio e compreensão dos alunos com relação aos conceitos fundamentais de POO abordados durante a oficina. Esse conjunto de questões será referido neste trabalho como QPC (*Questões sobre Percepção dos Conceitos*). Também foram utilizadas as mesmas questões no questionário a ser respondido após a oficina (pós-teste) para verificar se houve alguma melhoria da percepção dos alunos com relação aos conceitos de POO. As questões são apresentadas na Tabela 7.2.

Tabela 7.2 Enunciados das *Questões sobre Percepção dos Conceitos* (QPC).

Enunciados	
QPC1. De forma geral, você possui domínio dos conceitos fundamentais de POO (Classe, Objetos, Herança, Polimorfismo e outros).	QPC7. Você compreende o conceito de Encapsulamento e sua aplicação.
QPC2. Você compreende o conceito de Abstração aplicado à POO.	QPC8. Você compreende o conceito de Herança e sua aplicação.
QPC3. Você compreende o conceito de Classe.	QPC9. Você compreende o conceito de Polimorfismo e sua aplicação.
QPC4. Você compreende o conceito e a utilização de Objetos.	QPC10. Você compreende a necessidade de utilização de mais de um Objeto em um programa orientado a objetos.
QPC5. Você compreende o conceito de método construtor.	QPC11. Você compreende a troca de mensagens entre os Objetos.
QPC6. Você compreende o conceito de instância.	QPC12. Você se sente capaz de desenvolver um programa orientado a objetos.

O questionário pós-teste também contou com nove questões a serem respondidas usando a escala SAM, organizadas para verificar três diferentes aspectos da ComFAPOO: 1) a dinâmica apresentada durante a oficina; 2) o ato de programar/codificar durante as práticas e; 3) os recursos utilizados durante a oficina. Esse conjunto de questões será referido neste artigo como QPE (*Questões sobre a Percepção Emocional*). A Tabela 7.3 apresenta as questões relacionadas à escala SAM.

Tabela 7.3 Enunciados das Questões sobre a Percepção Emocional (QPE).

	Satisfação	Motivação	Domínio
Dinâmica	QPE1. Qual o seu nível de Satisfação com relação à dinâmica utilizada no ComFAPOO?	QPE2. Qual o seu nível de Motivação com relação à dinâmica do ComFAPOO?	QPE3. Qual o seu nível de Domínio com relação à dinâmica do ComFAPOO?
Programar	QPE4. Qual o seu nível de Satisfação em programar (codificar) durante as práticas do ComFAPOO?	QPE5. Qual o seu nível de Motivação em programar (codificar) durante as práticas do ComFAPOO?	QPE6. Qual o seu nível de Domínio em programar (codificar) durante as práticas do ComFAPOO?
Recursos	QPE7. Qual é seu nível de Satisfação com relação aos materiais utilizados (recursos didáticos) durante as práticas do ComFAPOO?	QPE8. Qual é seu nível de Motivação com relação aos materiais utilizados (recursos didáticos) usados durante as práticas do ComFAPOO?	QPE9. Qual é seu nível de Domínio com relação aos materiais utilizados (recursos didáticos) usados durante as práticas do ComFAPOO?

7.5 A aplicação da oficina e experimentação

Esta seção apresenta uma experimentação. As oficinas propostas pelo método foram realizadas com 14 alunos do 3º módulo do curso Técnico em Desenvolvimento de Sistemas da Escola Técnica Rosa Perrone Scavone, situada na cidade de Itatiba-SP. Todos os alunos estavam no último módulo do curso, com idades variando entre 16 e 17 anos, já tendo cursado disciplinas que utilizam o paradigma de POO.

O material didático da oficina foi disponibilizado em um repositório online hospedado no GitHub⁵, rede social para desenvolvedores e de gerenciamento de repositórios. As atividades ocorreram nas dependências da escola, durante o horário de aula. Foram realizadas as montagens dos componentes eletrônicos e a programação no ambiente de desenvolvimento do Arduino. Este projeto foi avaliado e aprovado pelo comitê de ética vinculado à Universidade Estadual de Campinas (UNICAMP)⁶.

⁵ Repositório disponível em <<https://github.com/humbertozeanetti/comfapoo>>

⁶ Certificado de Apresentação de Apreciação Ética (CAAE) de número 29622720.8.0000.5404.

A oficina iniciou-se com a aplicação do questionário contendo as QPCs para verificar a percepção do aluno sobre seu nível de domínio com relação aos fundamentos de POO, disponibilizado via a ferramenta online Google *Forms*. Seguindo as diretrizes do método ComFAPOO, foram realizadas todas as partes que compõem as atividades didáticas. Após o término das práticas, foi submetido novamente o mesmo questionário, como os mesmos itens avaliados antes da oficina. Também foi acrescido mais um questionário de avaliação, com as QPEs, usando o método SAM, para averiguar as respostas emocionais sobre o método aplicada. A Figura 7.4 mostra os alunos na oficina.

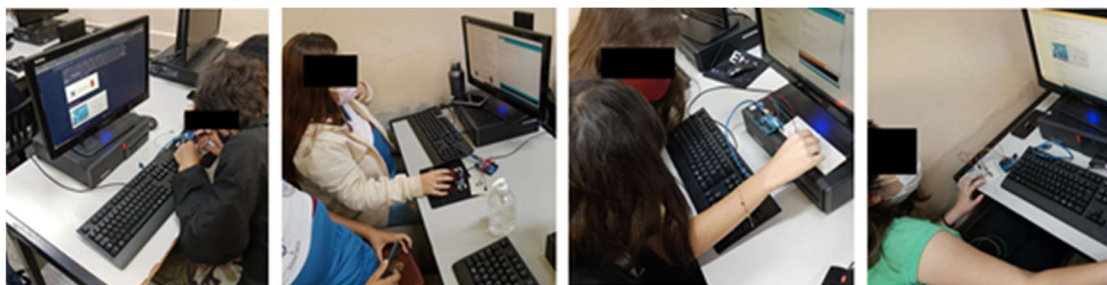


Figura 7.4 Alunos durante as práticas da oficina.

7.6 Análise dos resultados

Com relação às QPCs, apresentadas na Tabela 7.2, a Figura 7.5 apresenta a dispersão das respostas de cada uma das questões. À esquerda, observa-se o gráfico de barras com a dispersão das respostas (1 a 5) usando a escala Likert no pré-teste. É possível notar que há uma ampla presença de respostas relacionadas a *discordo totalmente*, *discordo parcialmente* e *indiferente*. Isso indica que os alunos apresentavam, antes da aplicação da ComFAPOO, baixa percepção de domínio e compreensão dos conceitos fundamentais de POO. No gráfico de barras à direita, que representa o pós-teste, após a aplicação da ComFAPOO, a maior parte da dispersão de respostas está entre *concordo parcialmente* e *concordo totalmente*.

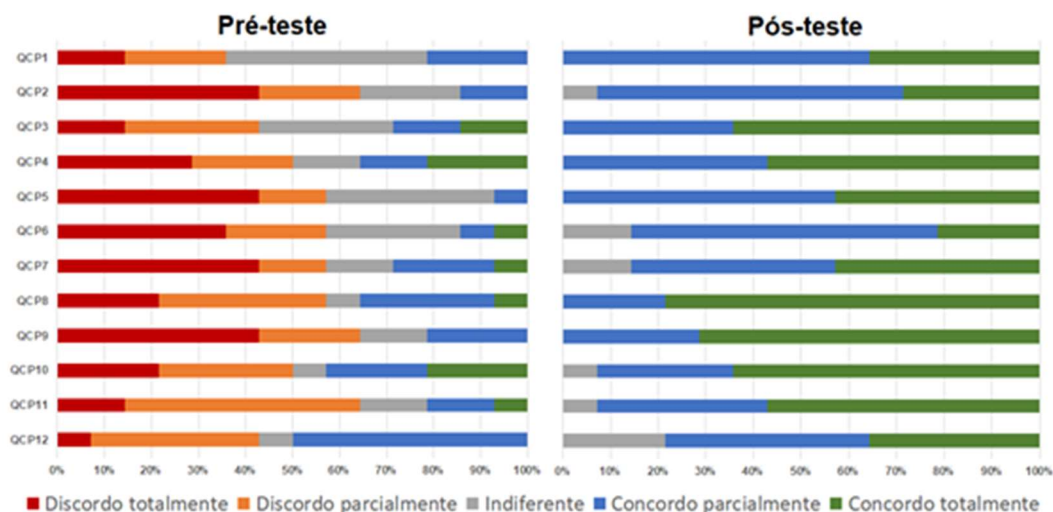


Figura 7.5 Distribuição das respostas das QPCs.

A partir dos gráficos do pós-teste é possível notar que houve uma melhor percepção por parte dos alunos com relação ao seu conhecimento sobre as bases fundamentais de POO. Houve poucas incidências de *indiferente* e nenhuma de *discordo totalmente* ou *discordo parcialmente*. Com isso, a aplicação da oficina utilizando o método ComFAPOO pode ter proporcionado uma sensível melhoria na compreensão dos fundamentos de POO.

Uma análise mais detalhada mostra que, na questão QPC1, que questionava a compreensão de forma geral sobre o conhecimento do aluno sobre POO, não houve nenhuma avaliação abaixo de *concordo parcialmente*, mostrando que entre todos há uma segurança sobre o que são as bases fundamentais do paradigma. A compreensão dos conceitos de *Classe* e *Objeto*, conceitos que comumente levam a dúvidas logo no início da aprendizagem, também apresenta melhoria na percepção dos alunos após oficina (QPC3 e QPC4). Conceitos tidos como mais complexos, como *Herança* e *Polimorfismo* (QPC8 e QPC9, respectivamente) também tiveram apenas avaliação de concordâncias. Apesar disso, a QPC12, que questiona se o aluno se sente capaz de desenvolver um sistema orientado a objetos, foi a que teve a maior ocorrência de avaliações *indiferente*.

Em relação às QPEs, a Tabela 7.4 apresenta a distribuição das respostas sobre a escala de nove valores do método SAM. As QPEs estão organizadas pelos diferentes aspectos (*Dinâmica*, *Programar* e *Recursos*) e a incidência de cada uma das escalas (1 a 9). Juntamente com a identificação da QPE, aparece uma indicação se a questão corresponde a (S)atisfação, (M)otivação ou (D)omínio. A última linha da tabela mostra a mediana, para mostrar a tendência central das avaliações por questão.

Tabela 7.4 Respostas às Questões sobre a Percepção Emocional (QPE).

	Dinâmica			Programar			Recursos		
Escala SAM	QPE1 (S)	QPE2 (M)	QPE3 (D)	QPE4 (S)	QPE5 (M)	QPE6 (D)	QPE7 (S)	QPE8 (M)	QPE9 (D)
9	8	4	1	3	4	1	9	8	3
8	3	5	3	3	3	2	3	2	7
7	2	1	6	1	2	5	1	2	3
6	0	2	3	3	3	3	1	1	0
5	1	2	1	3	2	3	0	1	1
4	0	0	0	1	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0
Mediana	9,0	8,0	7,0	6,5	7,5	7,0	9,0	9,0	8,0

Com relação ao aspecto referente à *Dinâmica* apresentada na ComFAPOO, é apresentada uma avaliação muito positiva com relação à *Satisfação*, *Motivação* e *Domínio*, com uma predominância com avaliações 7 ou superior. A interação com objetos reais pode ser um grande motivador na execução da prática e no envolvimento do aluno, podendo ter sido o fator mais relevante na composição dessa avaliação

A percepção sobre o ato de *Programar* foi o que teve a menor avaliação com respeito a *Satisfação*, *Motivação* e *Domínio*. As QPC4, QPC5 e QPC6 apresentaram a maior distribuição de incidência dentre todas as questões. Essa percepção por parte do aluno pode ter sofrido influência da linguagem de programação utilizada. O ato de programar a partir da linguagem do Arduino pode ter influenciado essa percepção do aluno. A plataforma utiliza uma linguagem baseada em C/C++, que possui uma sintaxe mais formal e prolixa quando comparada com a programação em blocos (como Scratch) ou algumas linguagens de programação *script*, como Python.

A percepção sobre os Recursos utilizados teve a avaliação de maior concentração entre os maiores valores da escala. A escolha pelo ferramental da CF para o método ComFAPOO objetivou trazer uma motivação maior ao aluno, por meio de uma percepção mais adequada com a relação entre um objeto concreto e um objeto abstrato, e usando objetos que poderiam ser facilmente compreendidos. Além disso, a facilidade de controlar esses objetos por meio da plataforma Arduino pode ter colaborado com as avaliações positivas nesse quesito. Este resultado pode fornecer subsídios para a adoção de outras plataformas e linguagens, além do Arduino.

7.7 Considerações finais

O principal objetivo da experimentação relatada neste artigo não foi comprovar a eficiência do método, mas sim ser um estudo de viabilidade, tanto de sua estrutura didática como também na aplicação dos questionários e sua validação por meio da análise desses resultados. De posse dessa primeira avaliação, foi identificada que a aplicação da ComFAPOO é viável e há bons indícios de sua efetividade; o projeto então seguirá fazendo a aplicação da ComFAPOO com um número maior de participantes.

Os resultados preliminares mostram que a ComFAPOO tem potencial para contribuir com a melhoria e compreensão dos fundamentos de POO por parte dos participantes das oficinas, por meio da análise feita sobre as QPCs. Por meio da análise dos QPEs, nota-se uma ampla aceitação com relação à dinâmica e recursos adotados na prática, e potenciais demandas para estudos de novas linguagens e plataformas de programação a serem adotadas.

Como trabalho futuro, é prevista a continuidade da aplicação da oficina com um número maior de participantes e, conseqüentemente, a captação de um maior volume de dados, a fim de desenvolver uma análise estatística mais consistente. Além disso, será analisado a possibilidade de revisão do formato do experimento para que haja grupo de controle com uma abordagem de ensino diferente do método descrito neste trabalho. Também será avaliada a adoção de instrumentos de análise qualitativa e a utilização de novas plataformas de desenvolvimento e linguagens, devido à análise prévia deste trabalho.

7.8 Referências

- Abbasi, S., Tabbassum, K., Kazi, H., Tunio, S., & Qureshi, S. (2021). Investigating Student's Obstacles While Learning Object Orientation. *International Journal of Scientific & Technology Research*, 6, 7-11.
- Armstrong, D. J. (2006). The quarks of object-oriented development. *Communications of the ACM*, 49(2), 123-128.
- Ausubel, D. P. (2003). Aquisição e retenção de conhecimentos: uma perspectiva cognitiva. *Lisboa: Plátano*, 1.
- Beck, K., & Cunningham, W. (1989). A laboratory for teaching object oriented thinking. *ACM Sigplan Notices*, 24(10), 1-6.

- Bennedsen, J., & Caspersen, M. E. (2006). Abstraction ability as an indicator of success for learning object-oriented programming?. *ACM Sigcse Bulletin*, 38(2), 39-43.
- Bennedsen, J., & Schulte, C. (2007). What does “objects-first” mean? An international study of teachers’ perceptions of objects-first. In *Proceedings of the Seventh Baltic Sea Conference on Computing Education Research-Volume 88* (pp. 21-29).
- Bradley, M. M., & Lang, P. J. (1994). Measuring emotion: the self-assessment manikin and the semantic differential. *Journal of behavior therapy and experimental psychiatry*, 25(1), 49-59.
- Bruner, J. S. (1966). *Toward a theory of instruction* (Vol. 59). Harvard University Press.
- Desportes, K. S. (2018). *Physical computing education: Designing for student authorship of values-based learning experiences* (Doctoral dissertation, Georgia Institute of Technology).
- Gutiérrez, L. E., Guerrero, C. A., & López-Ospina, H. A. (2022). Ranking of problems and solutions in the teaching and learning of object-oriented programming. *Education and Information Technologies*, 1-35.
- Henderson-Sellers, B., & Edwards, J. (1994). *BOOKTWO of object-oriented knowledge: The working object: Object-Oriented Software Engineering: Methods and Management*. Prentice-Hall, Inc..
- Knudsen, J. L., & Madsen, O. L. (1988). Teaching object-oriented programming is more than teaching object-oriented programming languages. In *European Conference on Object-Oriented Programming* (pp. 21-40). Springer, Berlin, Heidelberg.
- Liberman, N., Beerli, C., & Ben-David Kolikant, Y. (2011). Difficulties in learning inheritance and polymorphism. *ACM Transactions on Computing Education (TOCE)*, 11(1), 1-23.
- Lian, V., Varoy, E., & Giacaman, N. (2022). Learning Object-Oriented Programming Concepts through Visual Analogies. *IEEE Transactions on Learning Technologies*.
- McRoberts, M. (2018). *Arduino básico*. Novatec Editora.
- Moreira, M. A. (2006). Aprendizagem Significativa: da visão clássica à visão crítica (Meaningful learning: from the classical to the critical view). In *Conferência de encerramento do V Encontro Internacional sobre Aprendizagem Significativa, Madrid, Espanha, setembro de*. sn.
- Moström, J. E., Boustedt, J., Eckerdal, A., McCartney, R., Sanders, K., Thomas, L., & Zander, C. (2008, September). Concrete examples of abstraction as manifested in students’ transformative experiences. In *Proceedings of the Fourth international Workshop on Computing Education Research* (pp. 125-136).

- Pugh, J. R., LaLonde, W. R., & Thomas, D. A. (1987). Introducing object-oriented programming into the computer science curriculum. In *Proceedings of the eighteenth SIGCSE technical symposium on Computer science education* (pp. 98-102).
- Rosson, M. B., & Alpert, S. R. (1990). The cognitive consequences of object-oriented design. *Human-Computer Interaction*, 5(4), 345-379.
- Severino Júnior, D. D. S., & Costa, F. J. (2014). Mensuração e escalas de verificação: uma análise comparativa das escalas de Likert e Phrase Completion. *PMKT–Revista Brasileira de Pesquisas de Marketing, Opinião e Mídia*, 15(1-16), 61.
- Soares, A. P., Pinheiro, A. P., Costa, A., Frade, C. S., Comesaña, M., & Pureza, R. (2013). Affective auditory stimuli: Adaptation of the international affective digitized sounds (IADS-2) for European Portuguese. *Behavior research methods*, 45(4), 1168-1181.
- Trory, A., Howland, K., & Good, J. (2018). Designing for concreteness fading in primary computing. In *Proceedings of the 17th ACM Conference on Interaction Design and Children* (pp. 278-288).
- Xinogalos, S. (2015). Object-oriented design and programming: an investigation of novices' conceptions on objects and classes. *ACM Transactions on Computing Education (TOCE)*, 15(3), 1-21.
- Zanetti, H. A., & Borges, M. A. (2021). Por que estimular a Aprendizagem Significativa no ensino de Programação Orientada a Objetos?. In *Anais do Simpósio Brasileiro de Educação em Computação* (pp. 290-295). SBC.
- Zanetti, H. A. P., Borges, M. A. F & Ricarte, I. L. M. (2021). Método de ensino de programação Orientada a Objetos baseado em Computação Física, Aprendizagem Significativa e Concreteness Fading. In A. J. Osório, M. J. Gomes, A. Ramos, & A. L. Valente (Eds.), *Challenges 2021, desafios do digital: Livro de atas* (pp. 23-32). Universidade do Minho. Centro de Competência.

Capítulo 8

COMFAPOO: MÉTODO DE ENSINO DE PROGRAMAÇÃO ORIENTADA À OBJETOS BASEADO EM APRENDIZAGEM SIGNIFICATIVA E COMPUTAÇÃO FÍSICA

RESUMO

Uma das maiores dificuldades dos alunos iniciantes em Programação Orientada a Objetos é o exercício de abstração, algo necessário para a compreensão dos conceitos mais fundamentais. Embora esse paradigma seja ministrado há décadas, ainda é um desafio para o professor encontrar ferramentas e métodos para amparar suas práticas de ensino que sejam efetivas e motivadoras para o aluno. Tendo em vista essa necessidade, este artigo apresenta um método de ensino de programação baseado na teoria da Aprendizagem Significativa e nos recursos da Computação Física, incluindo uma análise de sua aplicação em oficinas de ensino de Programação Orientada a Objetos. O método ComFAPOO (Computação Física para Aprendizagem de Programação Orientada a Objetos) busca auxiliar nas principais dificuldades de alunos iniciantes no ensino paradigma orientado a objetos. A principal estratégia do método é propor uma transição gradual de modelos concretos (componentes eletrônicos) para modelos mais abstratos (código-fonte), que possam interagir por meio da programação, elucidando conceitos abstratos em representações mais tangíveis e representativas. Foram realizadas oficinas com 80 alunos de ensino técnico e superior e os resultados qualitativos e quantitativos dessa aplicação apontam para a efetividade do método e sua viabilidade como instrumento didático no auxílio na compreensão de conceitos fundamentais da Programação Orientada a Objetos.

PUBLICAÇÃO

Artigo aceito pela Revista Brasileira de Informática na Educação (RBIE), em seu 30º volume em 2022. Este texto usa as normas e formatos de referências estabelecidos pela revista. As numerações das seções, figuras e tabelas foram adaptadas do texto original para dar sequência à numeração desta tese.

8.1 Introdução

Pesquisas que discutem desafios para ensinar Programação Orientada a Objetos (POO) não são recentes (Knudsen & Madsen, 1988; Korson & McGregor, 1990; Kölling, 1999). Disciplinas que abordam o paradigma muitas vezes são as que trazem maior taxa de reprovação ou de evasão em cursos profissionais, como em nível técnico e superior (Piva Jr & Freitas, 2010). O paradigma POO frequentemente é apresentado após disciplinas fundamentais de programação, como Algoritmos ou Lógica de Programação e, com isso, alunos que tenham dificuldades oriundas dessas disciplinas se veem diante de uma nova visão de desenvolvimento de programas, podendo aumentar ainda mais as dificuldades e dúvidas.

Um grande impacto que ocorre com iniciantes em POO é a atividade de abstração que é necessária desde o primeiro contato com o paradigma. Or-Bach & Lavy (2004) indicam que a capacidade de abstrair é importante em qualquer atividade que envolva programação, especialmente em POO, pois toda a arquitetura do código-fonte é baseada em abstrações. É fundamental que, desde o início de uma disciplina de POO, o aluno tenha a capacidade de abstração para entender os conceitos mais fundamentais, como por exemplo, a relação entre classe e objeto (Eckerdal & Thuné, 2005).

Decker & Hirshfield (1994) apontam um problema com forte impacto para os iniciantes em POO, que é a troca do paradigma estruturado para o novo paradigma. Os alunos podem carregar dificuldades e dúvidas nas disciplinas iniciais de Algoritmos, que podem ser intensificadas com o ensino de POO. Mesmo para os alunos experientes em programação, o novo paradigma traz novos desafios.

Alunos iniciantes têm dificuldades em entender conceitos básicos da POO e suas aplicações em classes e objetos, pois, muitas vezes, não encontram relações com o mundo real ou com a realidade em que vivem (Yan, 2009). O ensino tradicional, diretamente explorando o código-fonte e seus editores com base de texto, sem antes entender ou interagir com o objeto que irá representar, pode não ser atrativo para o estudante, pois em seu dia a dia, eles interagem com interfaces mais tangíveis (como mouse ou tela do celular) e por meio de interfaces gráficas (Corral et al., 2014).

Apesar de um amplo conjunto de trabalhos voltados ao ensino de programação do paradigma POO, esta pesquisa identificou que poucos destes trabalhos utilizam recurso da CF com o objetivo de amparar o exercício de abstração, em especial nos conceitos fundamentais do paradigma. As pesquisas que se amparam nas premissas da teoria da AS no ensino de POO,

com o objetivo de apresentar o conteúdo de forma gradual e focalizando nas relações dos conceitos, não fazem o alinhamento com o ferramental da CF, que pode ser base tanto para representações iniciais de conceitos quanto para demonstrar as relações entre eles.

Assim sendo, este trabalho tem como objetivo desenvolver um método de ensino que, por meio dos recursos da CF, possa promover uma aprendizagem de POO seguindo as bases fundamentais da AS, de uma maneira sistemática e organizada, que possa ser reproduzida e conduzida por educadores em práticas didáticas. Tendo em vista a demanda por soluções que visam prover novas formas de ensino de POO, motivando o aluno e amparando o exercício de abstração, o presente trabalho apresenta o método ComFAPOO (**Com**putação **F**ísica para **A**prendizagem de **P**rogramação **O**rientada a **O**bjetos) e uma análise de sua aplicação junto a alunos de ensino técnico e superior. Com a aplicação do método por meio de práticas didáticas (*workshops*), foi possível buscar evidências experimentais da eficácia do ComFAPOO, assim como identificar oportunidades para melhorias futuras.

O ComFAPOO é baseado em três pilares: premissas da teoria da Aprendizagem Significativa (AS), recursos da Computação Física (CF) e o processo instrucional do modelo *Concreteness Fading* (CoFa). O método proporciona a construção de correlações entre conceitos fundamentais da POO, além de estimular e ajudar no exercício de abstração. Para a validação dessa eficácia foram realizadas, por meio de questionários definidos pelo método, duas análises quantitativas, sendo uma referente à percepção dos alunos com relação ao conhecimento adquirido e outra para verificar a percepção de motivação e satisfação na adoção do ComFAPOO. Também foram aplicados questionários com respostas dissertativas abertas, buscando identificar aspectos negativos e positivos no método.

Este artigo está organizado em seis seções, incluindo esta Introdução. A Seção 8.2 apresenta uma breve fundamentação teórica sobre as principais bases conceituais do método ComFAPOO. A Seção 8.3 apresenta e discute alguns trabalhos relacionados à temática desta pesquisa. A Seção 8.4 descreve o método ComFAPOO, apresentando suas etapas e ilustrando com exemplos de atividades que o compõem. A Seção 8.5, por sua vez, apresenta como foi feita a aplicação de um *workshop* junto a alunos de nível médio e superior. Na Seção 8.6 são discutidos os resultados obtidos com a aplicação do método. Por fim, na Seção 8.7, são feitas as considerações finais e descritas as contribuições e limitações deste trabalho.

8.2 Fundamentação teórica

Nesta seção serão apresentadas brevemente as bases do método ComFAPOO: a teoria da AS, que visa conduzir o aprendizado do aluno por meio de significado e correlações de conceitos; a CF, que proporciona os recursos para tornar tangíveis e programáveis objetos concretos; CoFa, um processo instrucional de apresentação de conceitos de forma gradativas, partindo do meios concretos para suas representações mais abstratas.

8.2.1 A teoria da Aprendizagem Significativa (AS)

A teoria da AS, desenvolvida por David Ausubel, apoia-se na aprendizagem por meio de materiais e conceitos que sejam potencialmente relevantes e significativos aos alunos. Ausubel (2003) afirma que o que determina se o conteúdo é significativo ou não é a percepção particular do aluno, por meio de conhecimento prévios, ou até mesmo questões sociais. Cabe ao educador prover recursos que possam ter o potencial significativo e que possam ser relevantes para a construção do conhecimento.

Segundo Novak (2002), a AS depende de uma estrutura organizada de conceitos que, facilmente, possam se relacionar com conceitos previamente apresentados ou adquiridos. Deve haver uma interação entre os conceitos novos e os conhecimentos já adquiridos pelos alunos, interação essa que seja especificamente relevante na estrutura cognitiva do aprendiz (Moreira, 2006). A teoria da AS não é um processo que pode ser diretamente aplicado em uma prática de ensino, mas fornece diretrizes que podem conduzir o educador a aplicar técnicas e instrumentos para promovê-la, para dar condições para que essa aprendizagem ocorra (Moreira, 2003).

A AS é uma proposta que contrapõe uma aprendizagem “mecânica”. A *aprendizagem mecânica* é aquela em que o aprendizado não prioriza os significados pessoais e a relação de novos conceitos com anteriores. O novo conhecimento é armazenado de maneira arbitrária (sem relação previamente definida) e literal pelo indivíduo, que pode até ser capaz de reproduzir esse conhecimento que, de fato, não tem significado para ele (Moreira, 2006). Segundo Moreira (2003), ao contrário da aprendizagem mecânica, a AS privilegia a interação entre conhecimentos existentes e novos, sendo conduzidos por meio de uma *linguagem*, ou seja, um mecanismo que leve esse conjunto de conhecimentos até o alcance do aluno.

O conhecimento prévio relevante e específico, existente na estrutura cognitiva do sujeito e que permite dar significado a um novo conhecimento, é chamado de ideia-âncora ou *subsunçor*⁷. O conceito de “ancoragem” é a organização e a integração de um novo conhecimento na estrutura mental já existente. Durante esse processo, os novos conhecimentos ganham significado, ganhando maior estabilidade cognitiva (Moreira, 2010). O educador desempenha o papel daquele que não apenas expõe o novo conhecimento, mas que também estimula a correlação e motiva o aluno a participar do processo. Segundo Moreira e Mansini (2001), a AS pressupõe: a) que o conteúdo a ser aprendido seja potencialmente significativo para o aluno, sendo relacionável à sua estrutura de conhecimento; b) que o aluno esteja disposto a relacionar o novo conhecimento à sua estrutura cognitiva.

O processo de assimilação, defendido pela AS, é chamado também de processo de “*subsunção*”, por meio do *princípio de assimilação*. (Moreira & Mansini, 2001; Ausubel, 2003). Esse princípio descreve a visão de Ausubel para a organização e assimilação do novo conhecimento na estrutura cognitiva do sujeito. Na visão da AS, a aprendizagem deve ser feita de forma gradativa, com os conceitos sendo assimilados pelos aprendizes após serem apresentados, desenvolvidos e correlacionados por meio de interações.

Ausubel (2003) defende que a assimilação de conceitos é facilitada quando os elementos mais gerais são apresentados logo no início e, depois, progressivamente diferenciados, em detalhes e especificidade. Esse processo é definido como *diferenciação progressiva*, com as ideias mais gerais e mais inclusivas (menos complexas e com melhor assimilação) são apresentadas primeiro e depois desenvolvidas e incorporadas em novos contextos ou relacionadas com outras. Essa integração de conceitos deve fazer parte da estratégia e material instrucional do professor, sempre deixando claro quais são similaridades com demais conceitos e como eles se correlacionam. Essa etapa de integração de conceitos é chamada de *reconciliação integrativa* (Moreira & Mansini, 2001). Moreira e Mansini (2001) descrevem esses conceitos da seguinte forma:

- *diferenciação progressiva*: processo pelo qual o material instrucional deve ser preparado para que as ideias mais gerais e com maior potencial de assimilação sejam apresentadas antes, para que depois sejam progressivamente diferenciadas. Essa diferenciação pode ser por meio de detalhamento específico ou conexões com outros conceitos;

⁷ Para esse artigo serão utilizados os termos “*subsunçor*” e “*subsunçores*”, que são mais recorrentes na literatura em torno da teoria da Aprendizagem Significativa.

- *reconciliação integrativa*: processo pelo qual o material didático deve explorar a relação das ideias e conceitos apresentados, apontar similaridades e diferenças entre eles, reconciliando as divergências entre eles. Nessa etapa, o objetivo é integrar todos os conceitos, definindo suas relações e contribuições para estrutura do conhecimento em geral.

A Figura 8.1 apresenta uma representação esquemática do modelo hierárquico de *diferenciação progressiva* (em vermelho) e de *reconciliação integrativa* (em azul).

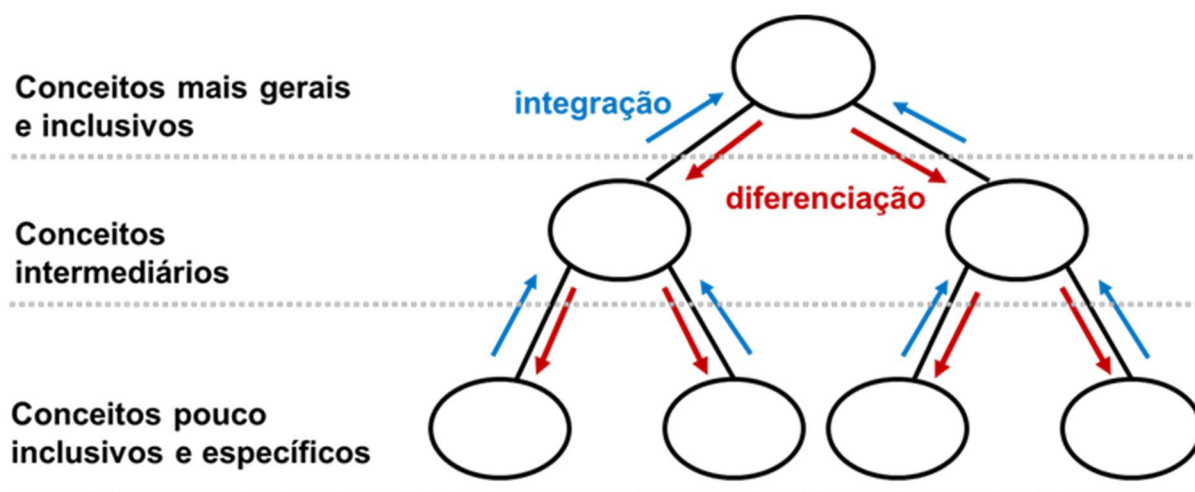


Figura 8.1 Modelo hierárquico da *diferenciação progressiva* e *reconciliação integradora*, adaptado de Moreira e Mansini (2001)

Segundo Ausubel (2003) e Moreira e Mansini (2001), a *diferenciação progressiva* é um princípio que deve ser levado em consideração no momento de desenvolver e programar o conteúdo que será apresentado ao aluno, como os conceitos e ideias mais gerais e inclusivas sendo apresentadas primeiro. A organização do conteúdo na mente de um indivíduo se apresenta como uma estrutura hierárquica, cuja os conceitos mais simples e menos abstratos se encontram no topo, e incorporam novos conhecimentos menos inclusivos e mais diferenciados.

Essas relações entre esses novos conhecimentos e conceitos com conhecimento e conceitos prévios devem ser exploradas durante a didática aplicada. Não só os aspectos comuns e complementares que esses conceitos têm, mas também suas diferenças e aplicações. Esse processo de *reconciliação integrativa* é essencial para a composição da estrutura cognitiva do aluno e juntamente com a *diferenciação progressiva* são princípios programáticos da organização da matéria e conteúdo a ser ensinado, levando à consolidação do conhecimento (Moreira, 2006).

Seguindo esses princípios, na prática, a programação de um conteúdo didático se dá por uma série de organizadores hierarquizados, apresentados de forma ordenada e

encadeada, com conceitos iniciais levando a novos, e a relação entre eles sendo clara aos alunos. Esses organizadores são elementos que apresentam um conceito, podendo ser qualquer objeto didático. Segundo Moreira e Mansini (2001), cada organizador fornece uma “ancoragem” para um novo organizador, criando uma sequência lógica e inclusiva de novos conceitos. Essa prática tem como objetivo impedir que haja o isolamento de conceitos, ou seja, sempre um conceito terá um elo com o que o precede, ou será a âncora para o posterior.

8.2.2 A Computação Física (CF) e suas aplicações

A utilização da CF combina a utilização de recursos de *hardware* (baseado em eletrônica) e a dinâmica da programação de *software*, envolvendo aspectos ligados à Engenharia e Ciência da Computação, para criar artefatos (Desportes, 2018). O termo “computação física” foi usado pela primeira vez por O’Sullivan e Igoe (2004), e referia-se a utilizar componentes físicos que pudessem ser controlados por programação. Brasileiro (2013) afirma que projetos com CF “[...] utilizam entradas e saídas de sensores e atuadores para dialogar com ambientes físicos, especialmente protótipos que interagem com componentes eletrônicos de percepções visuais, sonoras e táteis”. Hartmann et al. (2005) apresentam um diagrama que ilustra toda a estrutura essencial de uma interface física de comunicação com o usuário, como mostra a Figura 8.2.

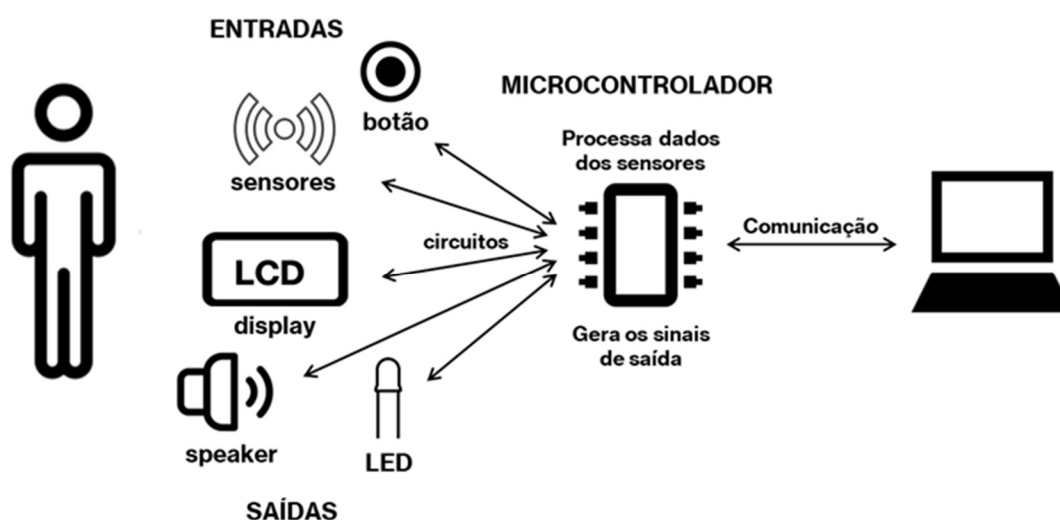


Figura 8.2 Componentes de uma interface física de comunicação com usuário, adaptado de Hartmann et al. (2005)

O ferramental da CF é voltado principalmente para o público que não possui conhecimentos técnicos avançados, muitas vezes podendo ser iniciantes tanto em eletrônica quanto em programação. Projetos usando CF podem ser feitos por alunos de diversas áreas de conhecimento, como designers e artistas, para criar interatividade em obras artísticas, ou *hobbyistas*, pessoas que eventualmente desejam criar algo automatizado (Mellis et al., 2007; Brasileiro, 2013).

Um dos principais objetivos da CF é criar um diálogo entre o mundo físico e o mundo virtual, por meio de interfaces que possam ser facilitadas para pessoas leigas tecnicamente. Plataformas *open-source*, como a *Arduino*, proporcionam meios de programação e prototipagem com componentes eletrônicos, deixando oculta toda a complexidade de se trabalhar com um meio microcontrolado (Mellis et al., 2007). Esse aspecto é um atrativo para o uso dessas plataformas em aplicações educacionais, podendo engajar o aluno para atuar com programação e eletrônica, promovendo a proatividade e criatividade (Desportes, 2018).

A característica tangível dos recursos da CF favorece a interatividade e a compreensão de vários aspectos relacionados com a programação. Przybilla e Romeike (2017) afirmam que a tangibilidade é um aspecto que faz com que os alunos se envolvam na atividade didática e mostrem mais autonomia na execução de projetos. A utilização de recursos físicos auxilia na compreensão da relação do código-fonte e a sua aplicação, pois o objeto concreto reage (ou não) ao que foi programado, fazendo com que o aluno identifique se o conceito aplicado na programação está coerente ou, em caso de erros, tenha recursos para identificar esses erros (Sentance et al., 2017).

8.2.3 O modelo *Concreteness Fading* (CoFa)

O CoFa é um modelo instrucional no qual as representações se alteram de uma representação mais concreta para uma mais abstrata durante a aprendizagem, objetivando combinar as vantagens de cada tipo de representação e promover a construção de conhecimento através dessa progressão (Jaakola & Veermans, 2018; Suh, Lee & Law, 2020).

O termo “*fading*”, que em uma tradução mais direta seria “desvanecimento” ou “dissipação”, indica essa progressão gradual da representação concreta para a mais abstrata, utilizando ambas as representações para entender os conceitos envolvidos. Essa trajetória de “dissipar” ou “diluir” o conhecimento iniciando pelo objeto concreto possibilita ao aluno ter contato com algo mais representativo, tátil, antes de discutir e apresentar símbolos ou

interpretações abstratas. Esse tipo de estratégia é bastante usado no ensino de matemática e física, como defende o trabalho de Bruner (1966), criador do modelo teórico do CoFa.

De acordo com Bruner (1996 apud Fyfe & Nathan, 2019), CoFa é “*uma teoria instrucional que se destina a facilitar as conexões entre várias representações de um conceito de destino ao longo de uma progressão de mais concreto para mais abstrato, com o objetivo de apoiar a transferência do conceito*”. Bruner especifica que essa transição deve ser feita através de múltiplas representações em uma sequência específica, sendo que seu framework inicial proposto contempla três fases consecutivas, através de representações “físicas” (*physical*), “pictográficas” (*pictorial*) e “simbólicas” (*symbolic*).

Jaakkola & Veermans (2020) afirmam que, no CoFa, são usadas múltiplas representações em uma determinada ordem, sendo as representações concretas em primeiro lugar, depois alguma representação que faça uma “conexão” entre o concreto e o mais abstrato e, por fim, modelos e representações mais abstratas. O modelo proposto por Bruner (1996), que sugere essa transição do concreto para o abstrato, pode ser visto na Figura 8.3.

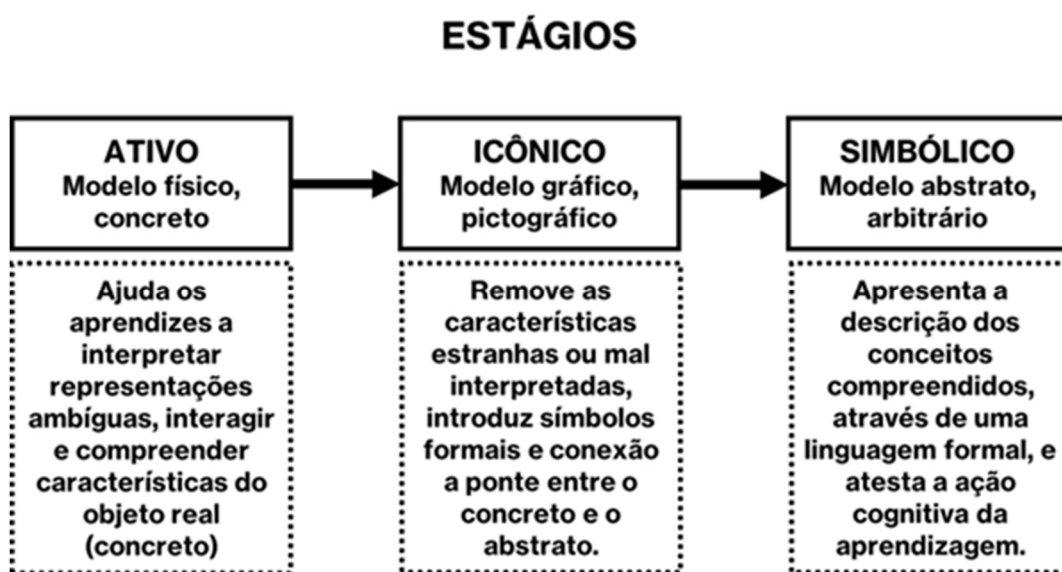


Figura 8.3 Os estágios do CoFa, adaptado de Fyfe et al. (2014).

8.3 Trabalhos relacionados à temática da pesquisa

O método ComFAPOO visa apoiar o ensino de fundamentos de POO, de uma maneira efetiva, utilizando recursos da CF e as premissas fundamentais da AS. A busca por trabalhos que utilizam CF e AS para ensino de POO possibilita identificar lacunas e deficiências que não são

supridas pelos trabalhos existentes, fazendo com que a ComFAPOO possa trazer inovações e colaborações em torno da temática. No início desta pesquisa, foram realizados alguns mapeamentos sistemáticos da literatura⁸, com o objetivo de identificar as principais aplicações da CF no ensino de programação e buscar por métodos de ensino que se baseiam na AS como diretrizes fundamentais. Foram realizadas consultas nas principais bases de bibliotecas digitais, como *Scopus Elsevier*, ACM (*Association for the Computing Machinery*) *Digital Library* e IEEE (*Institute of Electrical and Electronic Engineers*) *Xplore*, e motores de busca, como Periódicos CAPES e Google Acadêmico, tendo como resultado um conjunto de trabalhos com o objetivo de trazer à luz alguns questionamentos que pudessem contribuir para a construção do método ComFAPOO. Para a discussão apresentada nesta seção, serão apresentados trabalhos que abordem o ensino de POO que tenham como foco o ensino de programação formal, onde a prática com código-fonte seja abordada, ou que utilizem recursos que possam auxiliar o aluno no exercício de abstração.

Entre os trabalhos que utilizam recurso da CF, o de Jang, Lee e Kim (2015) apresenta um estudo de caso com alunos de ensino fundamental, usando Arduino para o ensino de conceitos de POO. Nesse trabalho, foi desenvolvida uma interface de software que auxilia os alunos a entender quais são as características dos objetos eletrônicos envolvidos e utiliza uma linguagem visual de programação. Como resultado dessa pesquisa, foi verificado que o ambiente visual utilizado contribuiu com a compreensão por parte dos alunos no exercício de abstração e assimilação dos conceitos de POO.

A pesquisa de Rubio, Hierro e Pablo (2013) relata o desenvolvimento de um currículo e práticas laboratoriais utilizando Arduino para o ensino introdutório de programação, para alunos de Engenharias e Ciência da Computação. Como resultado, foram comparados diferentes grupos de alunos, um usando o currículo proposto e, outro, um método tradicional (sem Arduino). O trabalho mostra que os alunos que participaram do novo currículo tiveram um desempenho melhor, além uma satisfação maior durante as práticas de programação.

Em ambas as pesquisas, a presença de recursos da CF foi um fator relevante e efetivo no ensino de programação, além de ser um elemento motivador para o engajamento do aluno. Em nenhum dos trabalhos mapeados previamente a esta pesquisa foi identificada a utilização de conceitos de POO juntamente com as práticas formais de programação (como código-fonte e diagramas de classe). Mesmo no trabalho de Jang, Lee e Kim (2015), que

⁸ Alguns publicados, como em Zanetti e Borges (2020)

apresenta uma interface de programação visual, não há uma completa transição do objeto real para seu modelo abstrato (código-fonte), como parte do exercício de abstração.

Corral et al. (2014) desenvolveu uma abordagem de ensino usando um método de *gamificação*, que utiliza blocos táteis com um display de LED, que respondiam (com mensagens, mudança de cores e projeções de imagens) diretamente à programação realizada pelos alunos. Contudo, os blocos, mesmo sendo táteis, não auxiliam no exercício de abstração, pois servem apenas como uma interface mais lúdica para a programação, não havendo diferenças físicas que possam ser notadas facilmente pelos alunos.

Com relação a pesquisas que explorem práticas de ensino de programação baseadas em premissas da AS, há dois trabalhos que trazem discussões sobre estratégias metodológicas. Buriticá (2013) apresenta uma aplicação de um método de ensino para ensino de paradigmas de programação, incluindo POO, cujo objetivo principal é promover a AS. Buriticá (2014) discute a aplicação de um método de ensino baseado em AS para ensino de programação funcional e estruturada, utilizando as linguagens C/C++.

Berssanette e Francisco (2018) utilizam algumas das bases fundamentais da AS, como a apresentações de *subsunçores* (conceito fundamentais prévios) e a demonstração da inter-relação entre conceitos. Astolfi e Junior (2015) apresentam um relato sobre uma intervenção com alunos de ensino técnico em uma disciplina de programação, onde são apresentados organizadores prévios como recursos didáticos, dando apoio à *diferenciação progressiva* e *reconciliação integrativa*. Em Lima, Diniz e Eliasquevici (2019) também é apresentado um método de ensino de lógica de programação respaldado nos princípios da AS, por meio de um jogo de cartas.

Apenas no trabalho de Buriticá (2013) está presente o ensino de POO de forma mais clara, mas não é o foco principal da pesquisa. Nas demais pesquisas, é explorado o ensino formal de programação, mas apenas em Astolfi & Junior (2015) está presente uma solução que aborde a utilização de *subsunçores* e a progressão do conhecimento. Em nenhum dos trabalhos é vista a adoção de algum processo que possa apoiar a didática nos princípios *diferenciação progressiva* e *reconciliação integrativa*.

8.4 O método ComFAPOO

O método ComFAPOO apresenta uma proposta para o ensino de conceitos fundamentais de POO, alinhando premissas de aprendizagem definidas pela AS e o ferramental instrucional da CF. Esse método foi desenvolvido para atender a demanda por recursos que pudessem dar melhor amparo ao exercício de abstração necessário na aprendizagem do paradigma orientado a objetos, além de prover uma dinâmica e recursos que pudessem motivar os alunos. O método busca fazer com que o aluno tenha contato com objetos concretos que possam ser controlados por meio de programação, refletindo as ações do “objeto de programação” no objeto concreto, no qual se pode interagir e ver o resultado em tempo real.

O ComFAPOO estrutura-se para que sejam apresentadas etapas que envolvem o objeto concreto, que tem como função elucidar todas suas características e ações, depois apresentar modelo mais abstratos, como diagramas de classes e representações gráficas dos objetos para que se realce os aspectos que devem ser abstraídos e, por fim, o modelo mais abstrato, o código-fonte, que especifica todas as características em forma de classe, com seus atributos e métodos.

Para que a utilização de componentes eletrônicos fosse viável, foi adotada a plataforma de prototipagem *Arduino*, consolidada tanto no meio educacional como profissional. Pela adoção dessa plataforma, a linguagem adotada para que fossem desenvolvidos os códigos-fonte foi a linguagem C++, nativa no ambiente padrão do *Arduino*. Na realidade, a programação nativa do Arduino baseia-se em comandos que são oriundos de uma classe que possui atributos e métodos que facilitam o desenvolvimento de programas para controle das interfaces do microcontrolador, mas essa classe é implementada em C++ e podem-se agregar todos os recursos da linguagem na programação.

Antes do desenvolvimento dos conteúdos e materiais que compõem o método, foi realizado um mapeamento da literatura para que fosse definido quais conceitos fundamentais relacionado à POO seriam escolhidos. Para isso, buscou-se na literatura trabalhos que discutissem e trouxessem sugestões de quais seriam os “fundamentos da POO”. Com isso, foi adotado a taxonomia proposta por Anderson (2006), que define os seguintes conceitos como fundamentais ao POO: *Abstração*, *Classe*, *Objetos*, *Método*, *Passagem de Mensagem*, *Encapsulamento*, *Herança* e *Polimorfismo*. Mais detalhes sobre esse mapeamento são encontrados no trabalho Zanetti, Borges e Ricarte (2022).

8.4.1 Embasamento teórico aplicado ao método ComFAPOO

Como anteriormente descrito, o método ComFAPOO foi desenvolvido sobre alguns pilares teóricos. Os princípios de *diferenciação progressiva* e *reconciliação integradora* são as bases fundamentais para o desenvolvimento do conteúdo e sequência didática implementada no método, além do sistema de aplicação de *subsunçores* e de ancoragem. Para cada uma das práticas propostas, foi desenvolvido um conjunto de material instrucional baseado na apresentação gradativa e incentivando a identificação das correlações que a AS defende. Um exemplo da aplicação da AS em umas das etapas do método é apresentado na Subseção 8.4.2.

A apresentação gradativa dos conceitos foi proposta para que o aluno pudesse ter o contato inicial com algum objeto concreto antes de fazer o exercício de abstrair sobre ele, podendo interagir e, conseqüentemente, compreender sobre o domínio no qual se insere tal objeto. Para esses objetos reais, foi escolhido o ferramental presente na CF, e para a apresentação gradual do modelo concreto para o mais abstrato, foi aplicado o CoFa.

A adoção de componentes eletrônicos para integrar o ComFAPOO se deu pela possibilidade de poder controlar objetos físicos que possam dar respostas instantâneas por meio de programação, fazendo com que o modelo mais abstrato (o código-fonte) seja o controle do modelo concreto, ajudando na correlação e validando o processo de abstração envolvido. Esse “diálogo” entre o modelo concreto e abstrato fortalece a relação entre o código-fonte e sua aplicação, trazendo mais significado ao ato de programar.

Ainda sobre o processo gradual de apresentação de conceitos, a escolha do modelo instrucional CoFa se deu pela estrutura bem definida de estágios de apresentação, partindo de modelos e objetos concretos, passando por representações mais abstratas, até atingir a concepção objetivada, a mais abstrata. A estrutura apresentada pela CoFa delineou as principais etapas de apresentação dos conteúdos da ComFAPOO, com os componentes eletrônicos compreendendo o estágio *ativo*, os diagramas e representações gráficas, o estágio *icônico*, até alcançar o estágio *simbólico*, o código-fonte. Portanto, cada base teórica contribui nos seguintes componentes e atividades do ComFAPOO:

- princípios da teoria da AS: estruturação do encadeamento e apresentação dos conceitos de forma gradual, selecionando quais são os *subsunçores* (conceitos iniciais) e toda a cadeia de conceitos subseqüentes e suas relações;

- ferramentas da CF: apresentação de objetos concretos que podem ser controlados via programação, trazendo interações e recursos de visualização e depuração em tempo real;
- o modelo da CoFa: auxilia a apresentação de concretos de forma gradual, do concreto ao abstrato, sendo um suporte para o exercício de abstração necessário para o paradigma orientado a objetos.

8.4.2 Descrição das etapas

O método ComFAPOO foi estruturado em 4 etapas distintas, que não necessariamente ocorrem sequencialmente, mas delimitam o grupo de tarefas que deve ser feito. Essas quatro etapas são *Concepção*, *Desenvolvimento*, *Aplicação* e *Avaliação* e foram originalmente descritas em Zanetti, Borges e Ricarte (2021). Na **Figura 8.4** são apresentadas as etapas do ComFAPOO.

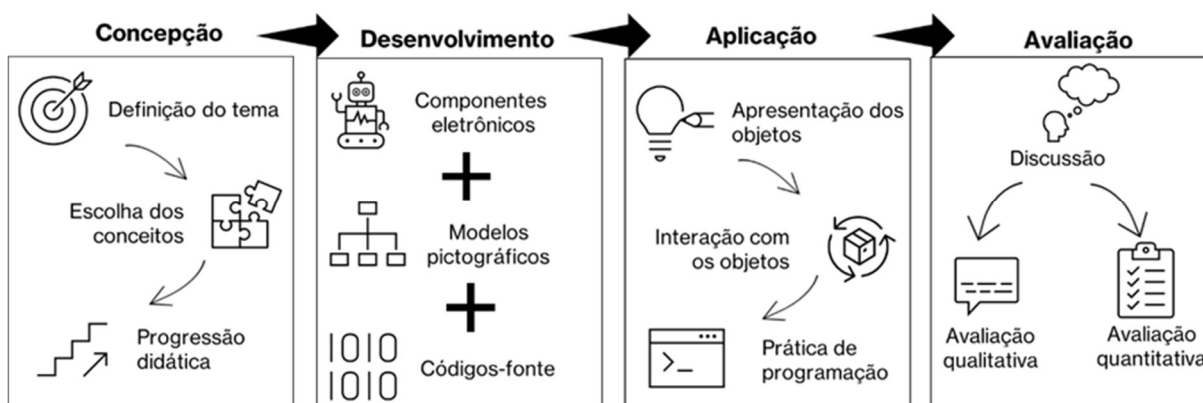


Figura 8.4 As etapas do método ComFAPOO

A etapa de *Concepção* tem como principal função elencar quais são os objetivos a serem alcançados em uma determinada prática. Nessa etapa também são definidos os conceitos que serão aplicados e qual é a progressão didática a ser realizada. Essa etapa é feita para cada uma das práticas a serem aplicadas. As atividades que compõem essa etapa são:

- Definição do tema: mapear, definir e compreender quais são os objetivos a serem alcançados com a prática. Esses objetivos são alinhados com os conceitos de POO que serão escolhidos;
- Escolha dos conceitos: escolher quais serão os conceitos relacionados à POO que farão parte da prática, e se eles são complementares ou relacionáveis;

- Progressão didática: mapeamento da sequência em que serão apresentados os conceitos e suas relações.

A etapa de *Concepção* tem uma forte correlação com os princípios fundamentais da AS, em especial a *diferenciação progressiva* e a *reconciliação integradora*. Nessa etapa, o encadeamento dos conceitos e suas relações são definidos e, na etapa seguinte, a de *Desenvolvimento*, são alinhados com o material didático. Na etapa de *Desenvolvimento*, essencialmente, é definido e desenvolvido o material didático a ser aplicado posteriormente. Com o mapeamento dos conceitos, são definidos quais são os objetos concretos (componentes eletrônicos), principalmente os que farão o papel de *subsunçor*. Além de integrar os princípios da AS, nessa etapa as atividades são definidas para atender o modelo proposto pela CoFa, onde são feitos todos os modelos pictográficos (nível intermediário de abstração) e os exemplos de código-fonte (forma mais abstrata) que são utilizados. Essa etapa é composta pelas seguintes atividades:

- Componentes eletrônicos: seleção dos componentes que serão utilizados, identificando que neles podem ser representados os conceitos de POO definidos para a prática. A escolha do componente ou componentes também segue a diretriz de se tornar o *subsunçor* da tarefa;
- Modelos pictográficos: criação de representações pictográficas que sejam intermediárias entre o modelo concreto e o abstrato (como define o CoFa no estágio *icônico*). O objetivo desses modelos é auxiliar os alunos, mostrando quais são as características e comportamentos do objeto concreto, apoiando o processo de abstração do aluno, ajudando na construção de um modelo mais abstrato (código-fonte);
- Códigos-fonte: representa o estágio simbólico do CoFa, criando exemplos que podem ser apresentados de forma gradativa (trechos de códigos-fonte que são acrescidos de novas funções), correlacionando com os modelos anteriormente apresentados. Esses códigos-fonte devem refletir o comportamento esperado pelo objeto concreto.

A etapa de *Aplicação* compreende as atividades relacionadas à condução da prática em si, seguindo a progressão e recursos definidos nas etapas anteriores. Seguindo também as premissas da AS, a condução deve ser gradual, buscando correlação com os conceitos previamente apresentados. Além de se basear no modelo CoFa, também é primordial seguir o processo de *ancoragem*. As atividades dessa etapa são:

- Apresentação dos objetos: consiste em apresentar os objetos e modelos concretos para que os alunos possam verificar quais são suas características e comportamento, ocultando conceitos técnicos aprofundados;
- Interação com os objetos: realizar uma discussão sobre o objeto apresentado, fazendo associações entre suas características e comportamentos, de uma maneira que seja análoga a atributos e métodos que serão abordados ao longo da prática, fazendo uso dos modelos intermediários (pictográficos);
- Práticas de programação: atividade que tem como objetivo apresentar o modelo mais abstrato, ou seja, o código-fonte, já que houve uma transição, iniciada pela apresentação do modelo concreto. Esse código-fonte pode ser apresentado de forma gradual, explorando poucas linhas de programação, e expandindo até atingir um código que represente os conceitos e controle o objeto concreto de modo esperado.

Exemplos da construção detalhada de parte do material didático e meio de aplicação serão abordados na seção 4.3.

Por fim, a etapa de *Avaliação* realiza uma avaliação sobre o domínio dos conceitos apresentados pelos alunos. Embora essa etapa não se relacione com os pilares teóricos do método ComFAPOO, ela é de grande importância para a validação do método. As avaliações não têm como função apresentar questões que mostrem erros e acertos, mas sim o envolvimento e internalização do conhecimento. Também, por meio das avaliações, é esperado identificar aspectos relacionados à motivação e envolvimento com o método, além de deficiências que podem ser melhoradas em novas aplicações. Essa etapa é composta pelas seguintes atividades:

- Avaliação quantitativa: para o método, é determinada uma avaliação por meio do *Self-Assessment Manikin* (SAM) para uma autoavaliação no aspecto emocional do aluno. Também foi agregado um questionário pré-teste e pós-teste, com o objetivo de identificar se houve melhoria na concepção de conceitos relacionados à POO;
- Avaliação qualitativa: são apresentadas aos alunos questões dissertativas abertas para que se possa obter relatos para identificar particularidades sobre o método e experiências individuais, com o objetivo de apontar problemas e indicar possíveis melhorias.

O método ComFAPOO apresenta uma divisão de todo o conteúdo relacionado aos fundamentos de POO em cinco partes. Cada uma dessas partes segue o mesmo processo

descrito no exemplo anterior, tanto para o desenvolvimento do material didático como em sua aplicação. Cada uma dessas partes tem como objetivo trabalhar com um ou mais conceitos de POO. Também foi elaborado um roteiro para que a apresentação gradual ocorra entre as partes. As partes são organizadas conforme mostra a **Tabela 8.1**.

Tabela 8.1 Organização dos conceitos e objetos

Parte	Conceitos abordados	Objetos
1	Abstração, Classe e Objeto	LED e botão
2	Encapsulamento	LED e botão
3	Herança	Sensores de luminosidade e temperatura
4	Polimorfismo	LED e botão
5	Troca de Mensagens	Semáforo (com LED e sensor de luminosidade)

A presente versão do ComFAPOO atende o formato de “oficinas”, ou seja, intervenções didáticas pontuais, que não formam a estrutura curricular de uma disciplina convencional em uma aplicação de um semestre, por exemplo. Para comprovação da efetividade do método foram realizadas oficinas (apresentadas na Seção 5), para que se colocasse em teste a atual versão da ComFAPOO. Com isso, a etapa de *Avaliações* foi desenvolvida para atender, prioritariamente, esta pesquisa, pois as avaliações realizadas têm o objetivo de apresentar resultados que possam ser analisados, tanto para a comprovação do método quanto para apontar possível melhorias em versões futuras. As avaliações utilizadas são descritas na seção 8.4.4. A partir dos dados obtidos nesta pesquisa, próximas versões do ComFAPOO poderão compreender o conteúdo programático que possa ser aplicado durante um semestre letivo em disciplina introdutória de POO, e consequentemente, sugerir um novo modelo de avaliação, que seja mais adequado para o novo formato.

8.4.3 Desenvolvimento do material didático e aplicação

O material didático, desenvolvido nas etapas de *Concepção* e *Desenvolvimento* do ComFAPOO, reflete as bases teóricas fundamentais do método. Com relação à divisão de artefatos a serem apresentados, o modelo CoFa é adotado como guia. Seus estágios, que gradativamente apresentam conceitos passando gradativamente da forma concreta para mais abstratas, apresentam um alinhamento com os princípios de *subsunção*, *diferenciação progressiva* e a *reconciliação integradora*. A Figura 8.5 mostra um exemplo da aplicação do material didático, seguindo uma apresentação gradativa e o alinhamento do CoFa com a AS.

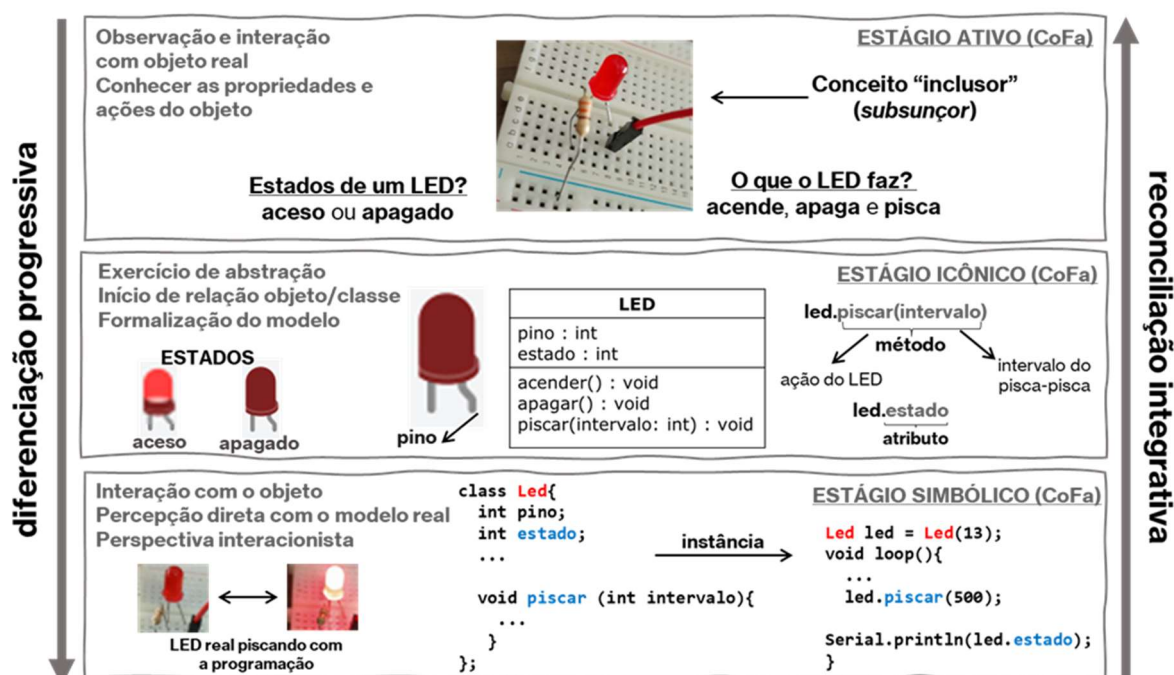


Figura 8.5 O alinhamento dos princípios da AS com o modelo CoFa

Como mostra a **Figura 8.5**, a distribuição dos passos para a apresentação de todo o material segue os três estágios do CoFa, onde cada estágio possui seus próprios objetivos. No primeiro estágio, o *ativo*, é apresentado o objeto real, e por meio de discussões e interações, deve ser compreendido o que ele faz e iniciando o exercício de abstração. No exemplo ilustrado, é apresentado o objeto LED, mostrando que suas ações são acender, apagar e piscar, e suas propriedades estão relacionadas com seu estado (aceso e apagado) e o pino conectado ao Arduino.

No estágio seguinte, o *icônico*, é realizada a maior parte do processo de abstração. Nesse caso, todas as características do LED serão traduzidas para atributos e métodos, ocorrendo uma transição do modelo mais concreto, para uma menos concreto, mais abstrato. A utilização de diagramas e informações visuais ajudam nessa transição de modelos. Nesse momento também é importante começar a formalizar uma construção que se aproxime à estrutura do código-fonte (como o uso de um diagrama de classe).

O último estágio, o *simbólico*, apresenta os modelos mais abstratos, os códigos-fontes. Nesse estágio, tudo o que foi abstraído e apresentando até o momento, sofre mais uma transição, agora para o modelo mais abstrato. Essa apresentação do código-fonte pode ocorrer também de forma gradual, fazendo com que a programação controle apenas algumas características. Por exemplo, a classe "Led" pode apenas conter o método construtor e o método "acender" e, instanciando o objeto, fazer com que o LED real acenda. Depois, podem ser

adicionados novos recursos, verificando-se gradualmente sua interação com o objeto real, sendo visualizado pelo aluno em tempo real.

Os princípios da *diferenciação progressiva* e da *reconciliação integradora* estão alinhados ao processo. A Figura 8.6 representa os conceitos de POO e os artefatos didáticos, seus relacionamentos e como eles promovem as premissas da AS durante a prática didática, por meio do recurso de *ancoragem*, mostrando o encadeamento dos conceitos durante a prática. A diferenciação (em vermelho), a partir do modelo mais concreto, ocorre enquanto a transição migra para o mais abstrato, com todo o apoio didático disponível. A integração (em azul) pode ser compreendida como o exercício do aluno reconstruir todo o caminho feito até o código-fonte, quando ele se depara com a programação e o objeto em código-fonte sendo a responsável por controlar o objeto físico. A *reconciliação integrativa* pode ser executada várias vezes, por exemplo, na apresentação gradual do código-fonte.

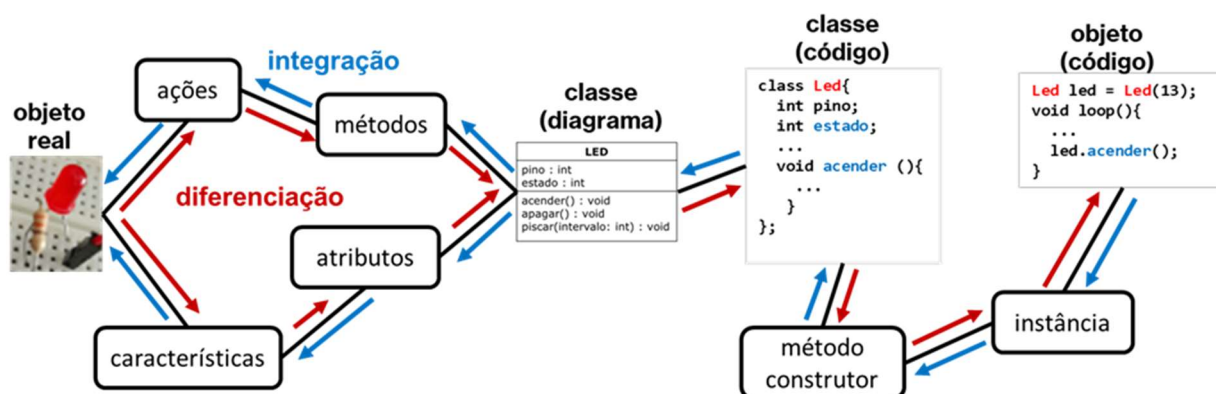


Figura 8.6 Composição da prática didática por meio de ancoragem.

8.4.4 Avaliações

O ComFAPOO propõe três avaliações, duas quantitativas e uma qualitativa, para que se possa ter meios de medir a aquisição do conhecimento (por meio de uma percepção pessoal) com o questionário *Questões sobre Percepção dos Conceitos* (QPC), que utiliza uma escala Likert de respostas. Para identificar questões emocionais, foi aplicado o questionário *Questões sobre a Percepção Emocional* (QPE), usando uma escala pictográfica. O questionário *Questões Abertas* (QA), com questões dissertativas, objetiva a coleta de relatos que possam apontar pontos fracos e fortes na aplicação do método.

Na atual versão do método, esta etapa atende, prioritariamente, a validação do método proposto. É previsto que a etapa de *Avaliação* seja reestruturada em novas versões do

método, que atendam uma grade curricular no formato de disciplina semestral, como é apresentado na Seção 7.2. Contudo, o questionário QPC pode ser utilizado como um instrumento de avaliação para educadores na versão atual, pois apresenta recurso de validação se a prática realizada teve impacto junto aos alunos. Por sua vez, os questionários QPE e QA objetivam exclusivamente a verificação da efetividade do método.

O questionário QPC foi aplicado antes do início das oficinas (pré-teste) e é formado por 12 questões para serem respondidas por meio da escala Likert, variando de *discordo totalmente* (1) a *concordo totalmente* (5). As questões foram elaboradas com o objetivo de determinar a percepção de domínio e compreensão dos alunos com relação aos conceitos durante as práticas. Esse mesmo questionário foi novamente aplicado após as oficinas (pós-teste), para que houvesse uma comparação da percepção do aluno com o pré-teste. As questões que compõem o QPC são apresentadas na Tabela 8.2.

Tabela 8.2 Questões do conjunto Questões sobre Percepção de Conceitos (QPC)

Enunciados	
QPC1. De forma geral, você possui domínio dos conceitos fundamentais de POO (Classe, Objetos, Herança, Polimorfismo e outros).	QPC7. Você compreende o conceito de Encapsulamento e sua aplicação.
QPC2. Você compreende o conceito de Abstração aplicado à POO.	QPC8. Você compreende o conceito de Herança e sua aplicação.
QPC3. Você compreende o conceito de Classe.	QPC9. Você compreende o conceito de Polimorfismo e sua aplicação.
QPC4. Você compreende o conceito e a utilização de Objetos.	QPC10. Você compreende a necessidade de utilização de mais de um Objeto em um programa orientado a objetos.
QPC5. Você compreende o conceito de método construtor.	QPC11. Você compreende a troca de mensagens entre os Objetos.
QPC6. Você compreende o conceito de instância.	QPC12. Você se sente capaz de desenvolver um programa orientado a objetos.

Junto com o questionário QPC de pós-teste, também foi aplicado o questionário QPE, usando uma escala pictográfica SAM, que foi organizada em três aspectos diferentes 1) a *Dinâmica* apresentada durante a oficina; 2) o ato de *Programar*/codificar durante as práticas e; 3) os *Recursos* utilizados durante a oficina. Segundo Bradley e Lang (1994), o método de avaliação SAM consiste em um sistema baseado em classificação pictográfica (manequins) que representam o aspecto emocional do indivíduo em resposta a um evento. Essas “emoções” são classificadas como *Satisfação* (prazer ou satisfação envolvidos), *Motivação* (engajamento e proatividade) e *Domínio* (controle da situação, autoridade ou sapiência). A Figura 8.7 mostra os modelos pictográficos utilizados no QPE.

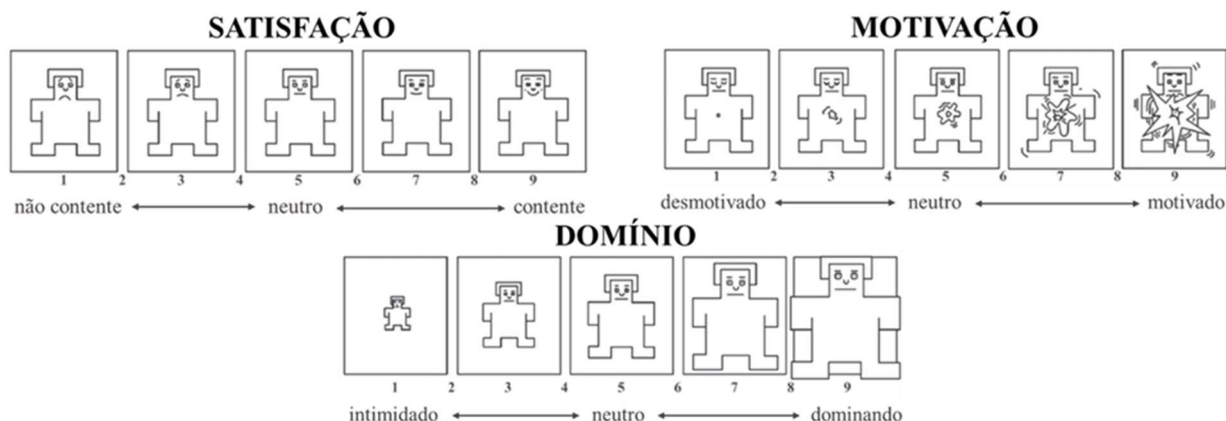


Figura 8.7 Escala utilizada nos questionários. Adaptado de Soares et al. (2003)

O questionário QPE é composto por nove questões, que estão divididas pelos aspectos Dinâmica, Programar e Recursos, e cada um desses aspectos possui uma questão associada a cada “emoção” (*Satisfação, Motivação e Domínio*). A Tabela 8.3 mostra as questões e sua organização.

Tabela 8.3 Questões do conjunto *Questões sobre Percepção Emocional (QPE)*.

	Satisfação	Motivação	Domínio
Dinâmica	QPE1. Qual o seu nível de Satisfação com relação à dinâmica utilizada no ComFAPOO?	QPE2. Qual o seu nível de Motivação com relação à dinâmica do ComFAPOO?	QPE3. Qual o seu nível de Domínio com relação à dinâmica do ComFAPOO?
Programar	QPE4. Qual o seu nível de Satisfação em programar (codificar) durante as práticas do ComFAPOO?	QPE5. Qual o seu nível de Motivação em programar (codificar) durante as práticas do ComFAPOO?	QPE6. Qual o seu nível de Domínio em programar (codificar) durante as práticas do ComFAPOO?
Recursos	QPE7. Qual é seu nível de Satisfação com relação aos materiais utilizados (recursos didáticos) durante as práticas do ComFAPOO?	QPE8. Qual é seu nível de Motivação com relação aos materiais utilizados (recursos didáticos) usados durante as práticas do ComFAPOO?	QPE9. Qual é seu nível de Domínio com relação aos materiais utilizados (recursos didáticos) usados durante as práticas do ComFAPOO?

Por fim, o questionário com questões abertas (QA) compreende oito questões. As seis primeiras questões são sobre a dinâmica e recursos utilizados durante a oficina. As duas últimas questões, mais amplas, se relacionam sobre os aspectos positivos e negativos do método ComFAPOO. As duas questões abertas (QA) consideradas para essa análise são: QA1 – “*Quais foram os pontos positivos do método ComFAPOO?*” e QA2 – “*Quais foram os pontos negativos do método ComFAPOO?*”.

8.5 Aplicação das oficinas

Nesta seção, é apresentada uma descrição da dinâmica adotada para a realização de oficinas conduzidas para avaliar a efetividade do método e verificar sua adequação para o ensino de conceitos fundamentais em POO. As oficinas ocorreram no período compreendido entre março e maio de 2022, de forma presencial e utilizando as dependências das instituições de ensino dos alunos participantes. Todos os trâmites necessários para a aplicação das oficinas foram avaliados e aprovados pelo comitê de ética vinculado à Universidade Estadual de Campinas (UNICAMP)⁹.

8.5.1 Descrição dos participantes

Os participantes compreendem dois níveis de formação profissional distintos, um grupo de alunos de ensino técnico e outro de ensino superior. Os alunos de ensino técnico, no total de 49 indivíduos, com idades variando de 16 a 18 anos, frequentam o curso de Ensino Médio Integrado ao Técnico de Desenvolvimento de Sistemas e o Técnico de Informática, na Escola Técnica Rosa Perrone Scavone da cidade de Itatiba/SP. Os participantes que cursam o ensino superior (total de 31 indivíduos) são alunos dos cursos de Tecnologia em Análise e Desenvolvimento de Sistemas e do curso Tecnologia em Sistemas Embarcados, da Faculdade de Tecnologia Deputado Ary Fossen, localizada na cidade de Jundiaí/SP, com idades que variam entre 18 e 66 anos.

Todos os alunos tiveram contato prévio com o paradigma POO, em disciplinas regulares em seus respectivos cursos. Outro fator comum entre esses grupos foi que, até o semestre letivo anterior ao que foi aplicado as oficinas, todos os cursos estavam em regime de educação à distância, devido à pandemia de Covid-19. Todos os indivíduos tiveram pelo menos dois semestres letivos nessa modalidade de ensino, retornando ao ensino presencial a partir do primeiro semestre. Na seção de considerações finais deste artigo são apresentadas algumas discussões sobre esse cenário e as consequências para a validade desta pesquisa.

⁹ Certificado de Apresentação de Apreciação Ética (CAAE) de número 29622720.8.0000.5404.

8.5.2 A dinâmica durante a oficina

As oficinas ocorreram de forma presencial nos laboratórios localizado nas dependências da Escola Técnica (ETEC) e Faculdade de Tecnologia (FATEC), durante o período letivo e horários de aula. O início da oficina foi por meio da aplicação do questionário pré-teste, com as QPCs, descritas na Seção 4.4., sendo disponibilizadas via a plataforma *Google Forms*. Também ocorreu a apresentação do repositório, hospedado na plataforma *GitHub*¹⁰, contendo o material didático que foi utilizado durante todas as práticas. A Figura 8.8 mostra alguns exemplos de telas (*prints*) do repositório com o material didático.



Figura 8.8 Exemplos de telas do repositório do GitHub.

Seguindo todas as diretrizes do método ComFAPOO, foram realizadas as cinco práticas, utilizando *kits* didáticos contendo placas Arduino e componentes eletrônicos, disponibilizados pelas instituições de ensino. Durante as práticas os alunos faziam as montagens (previamente disponibilizadas), verificavam, por meio da exposição do professor, algumas interações que mostravam o comportamento e as características dos componentes eletrônicos, e executavam trechos de código-fonte para que eles pudessem ver o resultado em suas próprias montagens.

Após o término da última parte da oficina, foi submetido um novo questionário (pós-teste), com os mesmos itens avaliados antes do início oficina. Também foi aplicado um questionário de avaliação, com as QPEs, usando o método de avaliação SAM, para uma análise complementar sobre o método ComFAPOO. A Figura 8.9 apresenta algumas imagens dos participantes durante o andamento das oficinas.

¹⁰ Repositório disponível em <https://github.com/humbertozanetti/comfapoo>

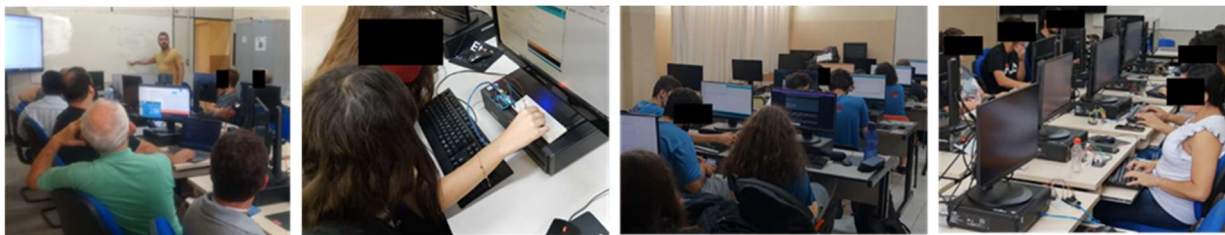


Figura 8.9 Participantes durante as oficinas.

8.6 Análise dos resultados

Nesta seção será apresentada uma análise dos resultados obtidos nas oficinas descritas na Seção 8.5, utilizando o método ComFAPOO. Para a análise serão destacados três agrupamentos diferentes: grupo **TODOS**, contendo todos os participantes sem distinção entre cursos de ensino técnico e superior; o grupo **ET**, com os participantes dos cursos de ensino técnico e; grupo **ES**, com os indivíduos dos cursos de ensino superior. Essa distinção entre esses grupos objetiva a comparação em aspectos que são convergentes entre esses agrupamentos e para identificar diferenças que possam surgir em cada um dos grupos.

A avaliação sobre as QPCs objetiva a verificação da efetividade do ComFAPOO como um método que possa auxiliar a compreensão de conceitos relacionados à POO, trazendo evidências que possam contribuir com o desenvolvimento do próprio método e possivelmente indicar falhas. A análise sobre as QPEs vem como um instrumento auxiliar para que se identifiquem aspectos emotivos e aceitação sobre a utilização do método, permitindo que seja percebido se os instrumentos e didática escolhidos no desenvolvimento do método podem ser melhorados ou alterados para uma maior aceitação na sua adoção junto a um grupo de alunos.

8.6.1 Análise das *Questões sobre a Percepção dos Conceitos* (QPC)

A análise sobre as QPCs foi por meio de questionários que utilizam uma escala Likert de concordância, com os valores podendo variar de 1 a 5. Para evidenciar a efetividade do método ComFAPOO foi realizado o teste *t* para cada uma das QPC, para evidenciar a diferença entre as amostras das respostas antes da aplicação do método (pré-teste) e após a aplicação do método (pós-teste). Para isso foi proposta a seguinte hipótese nula:

- H_0 : não existe diferença na percepção da compreensão dos conceitos avaliados pelas QPCs após a aplicação do método ComFAPOO ($H_0: \mu_1 = \mu_2$).

Para contrapor a hipótese nula, a hipótese alternativa se apresenta como:

- H_1 : existe diferença na percepção dos conceitos avaliados pelas QPCs após a aplicação do método ComFAPOO ($H_1: \mu_1 \neq \mu_2$).

O nível de significância adotado foi de 5% ($\alpha = 0.05$) e em todos dos grupos (TODOS, ET e ES) e em todas as QPCs (QPC1 até QPC12) o resultado do **valor p** (*p-value*) foi menor que .001 (*p-value* < .001), assim rejeitando a hipótese nula H_0 . Para uma análise gráfica, foi adotado o diagrama de caixa (*boxplot*), para que se possa verificar a variação dos dados entre pré-teste e pós-teste. A diferença na dispersão dos quartis torna possível evidenciar não somente a concentração de valores, mas também a diferença entre as amostras.

A Figura 8.10 ilustra a análise gráfica feita para o grupo TODOS. A apresentação dos diagramas de caixa está organizada entre pré-teste (Pré), à esquerda, e pós-teste (Pós), à direita, de cada uma das QPC. Em todos os gráficos de todos os grupos, é possível observar que há uma diferença significativa nas distribuições das QPCs com relação aos quartis de Pré e Pós, sendo notável a diferença entre as amostras e a melhor percepção conceitos por parte dos integrantes. Também se nota que em todas as QPCs a mediana dos grupos Pós está entre os valores 4 (*concordo parcialmente*) e 5 (*concordo totalmente*). A Figura 8.11 apresenta os diagramas do grupo ET. Assim como evidenciado no agrupamento TODOS, também é notório a diferença amostral entre Pré e Pós no grupo ET, além da melhoria da percepção por parte dos indivíduos. Essa constatação também pode-se aplicar para o grupo ES, como mostra a Figura 8.12.

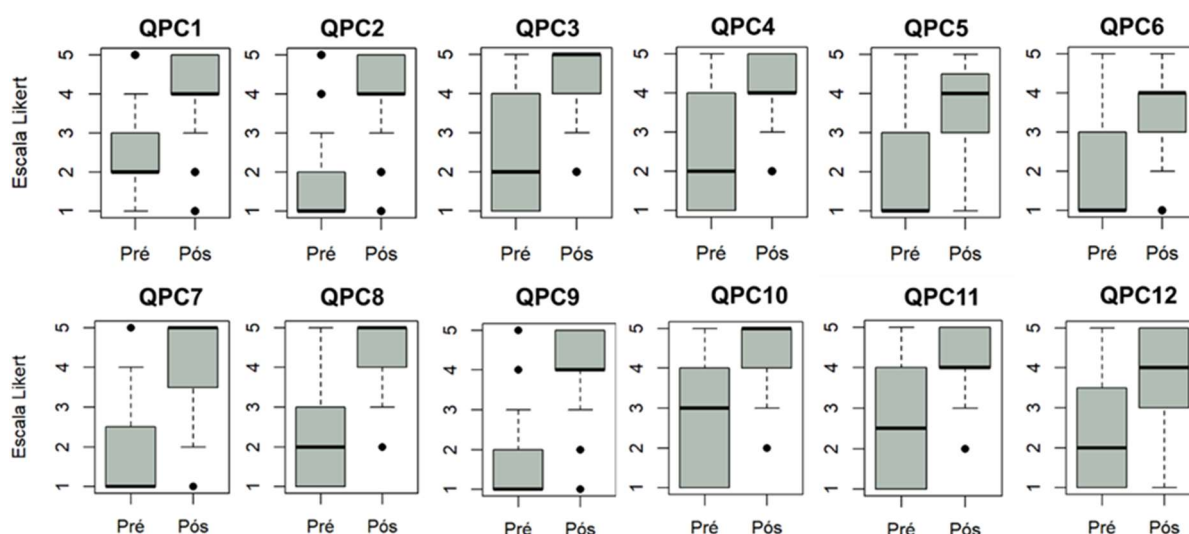


Figura 8.10 Diagramas de caixa para o grupo TODOS.

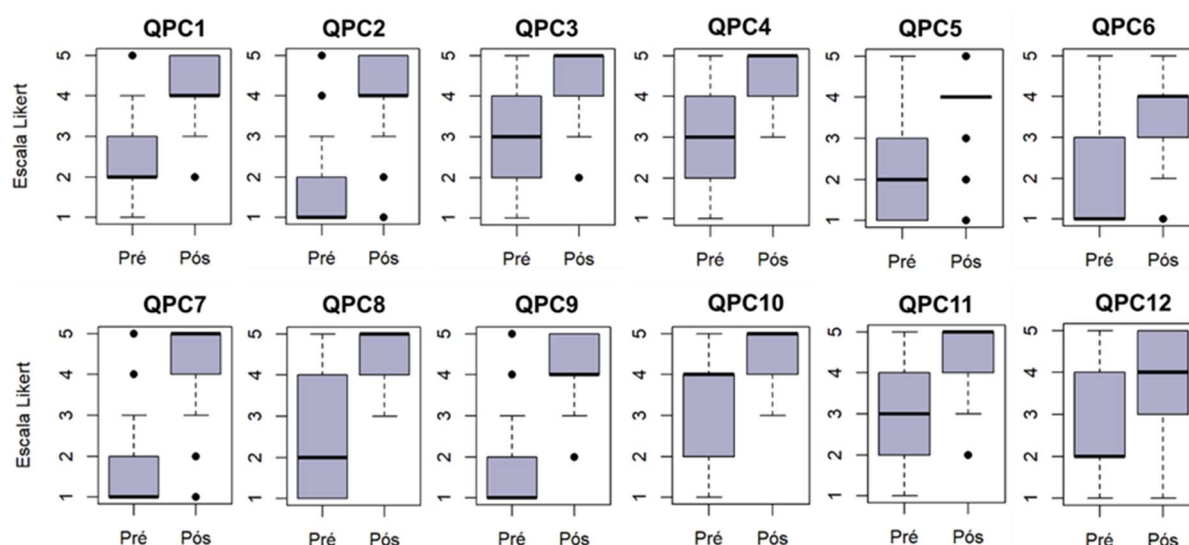


Figura 8.11 Diagramas de caixa para o grupo ET.

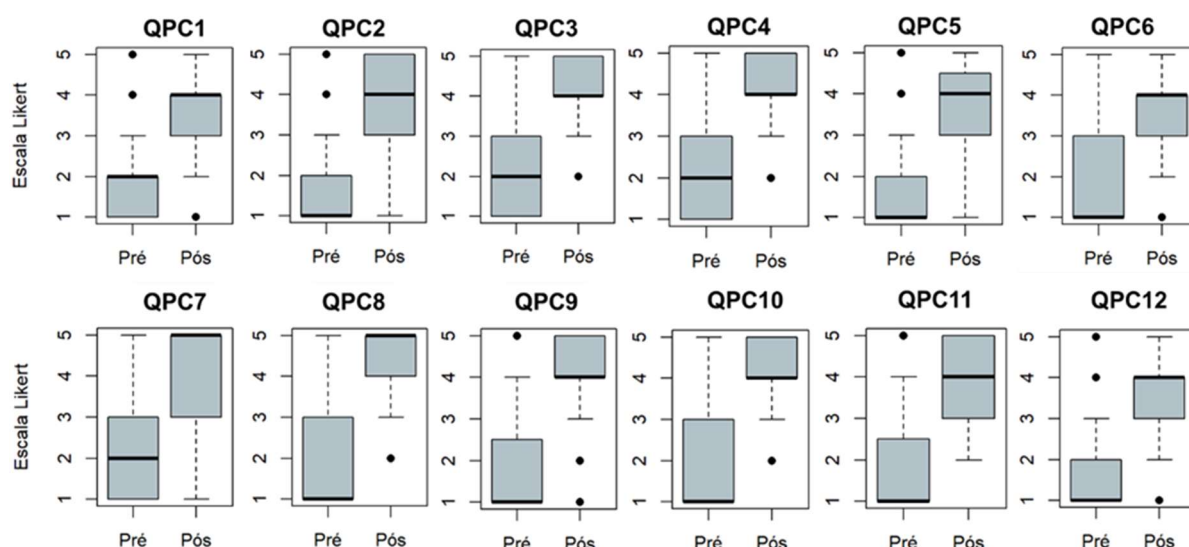


Figura 8.12 Diagramas de caixa para o grupo ES.

A Figura 8.13 apresenta dois gráficos de barras com a dispersão das respostas do grupo TODOS, sendo, à esquerda, o gráfico correspondente ao pré-teste e, à direita, o pós-teste. Assim como apontava os diagramas de caixa, há grande presença de *discordo totalmente*, *discordo parcialmente* e *indiferente* no pré-teste. No pós-teste é possível notar uma maior presença de avaliações *concordo parcialmente* e *concordo totalmente*. Questões relacionadas à conceitos que necessitam maior habilidade de abstração, como Classe (QPC3), Objeto (QPC4), Herança (QPC8) e Polimorfismo (QPC9), estão entre as com melhores desempenhos na

avaliação pós-teste. Questões que envolvem componentes de aplicação prática, como método construtor (QPC5) e instância (QPC6), estão entre as piores avaliações, mesmo apresentando uma evolução em comparação ao pré-teste. A QPC1, questão mais abrangente sobre o conhecimento geral sobre o paradigma POO apresenta notável evolução com quase 80% de suas respostas sendo *concordo totalmente* ou *concordo parcialmente*.

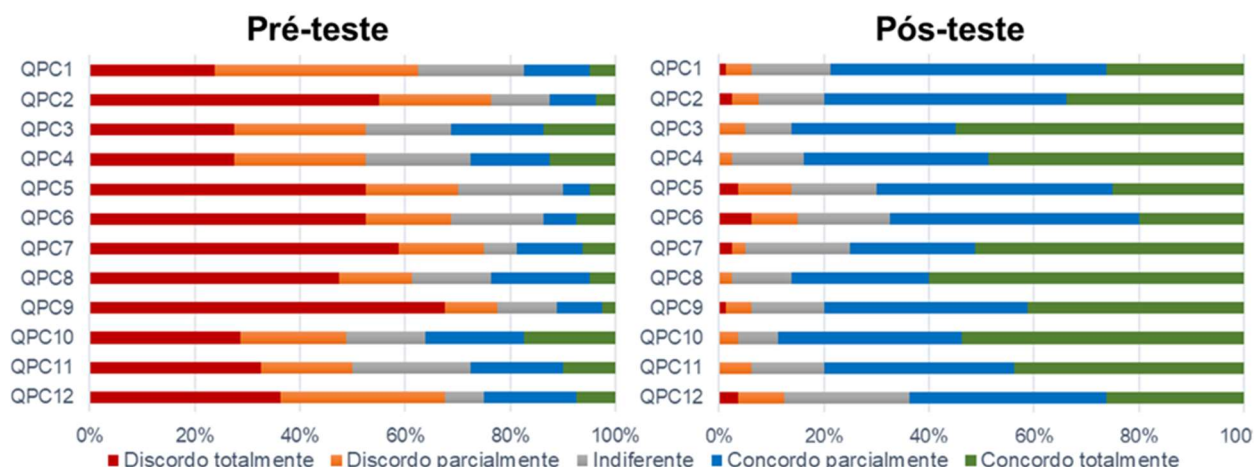


Figura 8.13 Distribuição das respostas das QPCs no grupo TODOS.

Para uma análise mais abrangente, o mesmo gráfico foi feito para os grupos ET e ES, a fim de identificar particularidades entre os grupos. A Figura 8.14 apresenta a dispersão das respostas do grupo ET. Em algumas questões no pós-teste (QPC4, QPC8 e QPC10), não se identifica nenhuma avaliação *discordo totalmente* ou *discordo parcialmente*. As QPC5 e QPC6, relacionadas aos conceitos de método construtor e instância, assim como na análise geral, se mostram com as piores avaliações, mesmo assim indicam melhoria significativa quando se compara ao pré-teste. A QPC4, relacionada ao conceito de Objeto, que teve a pior avaliação no pré-teste, no pós-teste apresenta a melhor avaliação. As percepções sobre os conceitos de Encapsulamento, Herança, Polimorfismo, uso de mais de um Objeto e troca de mensagens (QPC7, QPC8, QPC9, QPC10 e QPC11, respectivamente) estão entre as mais bem avaliadas.

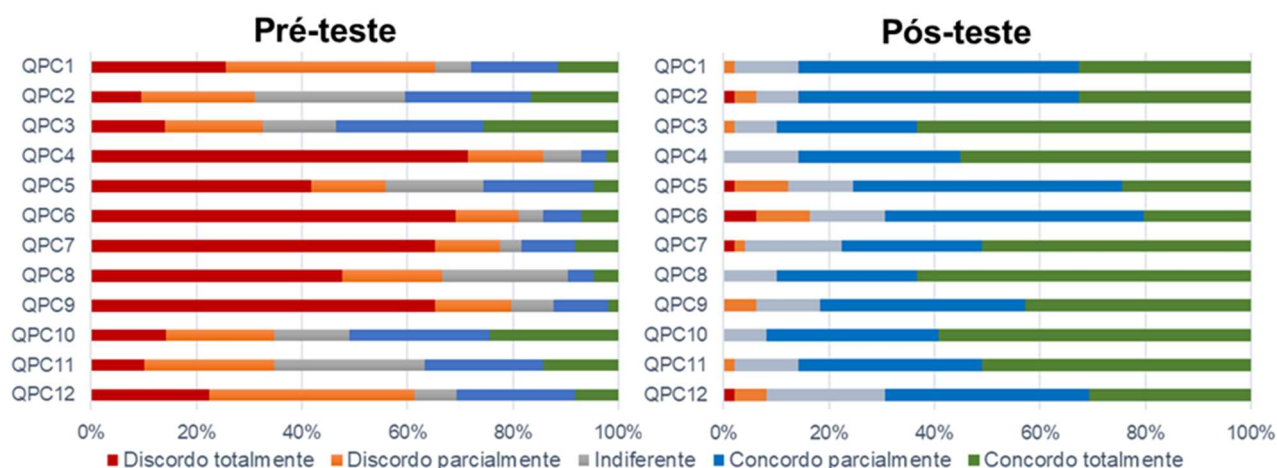


Figura 8.14 Distribuição das respostas das QPCs no grupo ET.

No grupo ES as avaliações positivas também estão presentes, como nas análises anteriores. No pré-teste, esse grupo apresentou piores avaliações que o grupo ET. Nesse grupo, as questões QPC3, QPC4, QPC7, QPC8, QPC9, QPC10 e QPC11 foram as que tiveram melhores avaliações no pós-teste, assim como foi verificado no grupo anterior. Em uma visão geral, é possível notar que no pós-teste esse grupo apresenta avaliações similares o grupo ET. A Figura 8.15 apresenta os gráficos de pré-teste e pós-teste do grupo ES.

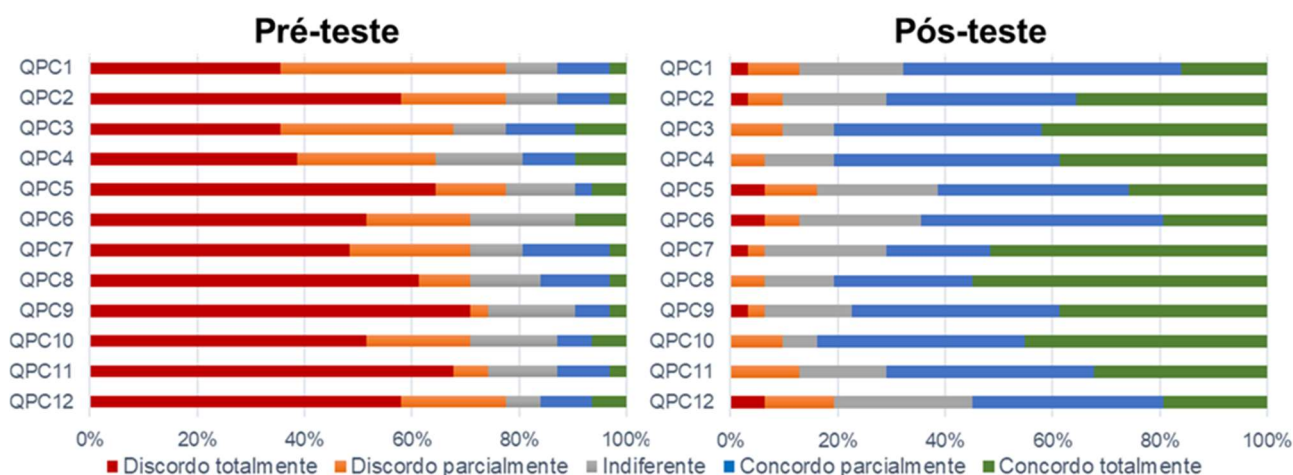


Figura 8.15 Distribuição das respostas das QPCs no grupo ES.

As diferenças entre os grupos ET e ES não demonstram que houve efeitos muito discrepantes entre os grupos, principalmente se observado os resultados no pós-teste. Esse fato pode ser visto intuitivamente por meios das representações gráficas, mas para uma comprovação, foi realizado *teste t* em todas as QPCs dos grupos ET em comparação às do grupo ES. Todos os testes com cada QPC resultaram em $p\text{-value} > .05$, demonstrando que não houve diferença significativa entre os grupos no pós-teste.

8.6.2 Análise das *Questões sobre a Percepção Emocional (QPE)*

Com relação análise sobre as QPEs, também será seguida a divisão entre grupo TODOS (todos os integrantes), ET (apenas os alunos do ensino técnico) e ES (apenas os alunos de ensino superior). As tabelas contendo a distribuição das respostas (valores da escala SAM) de cada grupo estão organizadas com o objetivo de distinguir cada um dos diferentes aspectos a serem analisados (*Dinâmica*, *Programar* e *Recursos*) e cada questão será sinalizada com uma identificação correspondente a (S)atisfação, (M)otivação ou (D)omínio. A última linha de cada tabela mostra a mediana de cada uma das QPEs, para indicar uma tendência central de cada questão.

A Tabela 8.4 mostra as respostas do grupo TODOS (total de 80 indivíduos) com relação ao questionário SAM atribuído. É exibido, além da quantidade de valores correspondente à escala SAM, a porcentagem correspondente ao total de respostas. Com relação ao aspecto da *Dinâmica*, é apresentada uma avaliação positiva em todos os aspectos de *Satisfação*, *Motivação* e *Domínio*. Nota-se que a QPE3 teve maiores incidências em de valores 5 e 6 com relação às demais questões. Sobre o questionamento da atividade de *Programar*, mesmo com uma mediana de mesmo valor nos grupos de resposta, é possível notar que as QPE4, QPE5 e QPE6 tiveram maior incidência de notas 5 e 6 que as questões relacionadas à *Dinâmica*. As questões sobre *Recursos* tiveram um sutil aumento de avaliações positivas (mostrado pelas medianas e porcentagens), com uma incidência maior de notas 8 e 9.

Tabela 8.4 Respostas do grupo TODOS às *Questões sobre a Percepção Emocional (QPE)*.

	Dinâmica			Programar			Recursos		
Escala SAM	QPE1 (S)	QPE2 (M)	QPE3 (D)	QPE4 (S)	QPE5 (M)	QPE6 (D)	QPE7 (S)	QPE8 (M)	QPE9 (D)
1	0 (0,0%)	0 (0,0%)	0 (0,0%)	1 (1,3%)	1 (1,3%)	1 (1,3%)	0 (0,0%)	0 (0%)	1 (1,3%)
2	0 (0,0%)	2 (2,5%)	2 (2,5%)	0 (0,0%)	0 (0,0%)	1 (1,3%)	0 (0,0%)	0 (0%)	1 (1,3%)
3	0 (0,0%)	1 (1,3%)	1 (1,3%)	0 (0,0%)	1 (1,3%)	3 (3,8%)	0 (0,0%)	1 (1,3%)	1 (1,3%)
4	1 (1,3%)	1 (1,3%)	1 (1,3%)	4 (5,0%)	1 (1,3%)	2 (2,5%)	0 (0,0%)	1 (1,3%)	1 (1,3%)
5	3 (3,8%)	3 (3,8%)	6 (7,5%)	7 (8,8%)	8 (10,0%)	10 (12,5%)	3 (3,8%)	5 (6,3%)	5 (6,3%)
6	1 (1,3%)	11 (13,8%)	16 (20,0%)	9 (11,3%)	12 (15,0%)	17 (21,3%)	5 (6,3%)	3 (3,8%)	8 (10,0%)
7	20 (25,0%)	11 (13,8%)	25 (31,3%)	15 (18,8%)	11 (13,8%)	19 (23,8%)	8 (10,0%)	17 (21,3%)	19 (23,8%)
8	18 (22,5%)	28 (35,0%)	19 (23,8%)	19 (23,8%)	24 (30,0%)	19 (23,8%)	15 (13,8%)	14 (17,5%)	26 (32,5%)
9	37(46,3%)	23 (28,8%)	10 (12,5%)	25 (31,3%)	22 (27,5%)	8 (10,0%)	49 (61,5%)	39 (48,8%)	18 (22,5%)
Mediana	8	8	7	8	8	7	9	8	8

A distribuição das respostas do grupo ET (total de 49 indivíduos) é apresentada na Tabela 8.5. Com relação aos valores das medianas obtidas em todas as QPEs, esse grupo possui valores iguais ao grupo TODOS. Em *Dinâmica*, mesmo a maior concentração em respostas 7

ou superior, a QPE3, relacionada a *Domínio* teve a pior avaliação, assim como no grupo TODOS. Com relação às questões sobre *Programar*, esse grupo teve uma maior dispersão entre as notas a partir de 5 quando comparado a TODOS. As avaliações sobre *Recursos* se mostraram mais positivas em relação a de *Dinâmica* e *Programar*, sendo a QPE7 sendo a mais bem avaliada, com a maior parte das respostas sendo o valor 9 (assim como em TODOS).

Tabela 8.5 Respostas do grupo ET às Questões sobre a Percepção Emocional (QPE).

Escala SAM	Dinâmica			Programar			Recursos		
	QPE1 (S)	QPE2 (M)	QPE3 (D)	QPE4 (S)	QPE5 (M)	QPE6 (D)	QPE7 (S)	QPE8 (M)	QPE9 (D)
1	0 (0,0%)	0 (0,0%)	0 (0,0%)	1 (2,0%)	1 (2,0%)	1 (2,0%)	0 (0,0%)	0 (0,0%)	1 (2,0%)
2	0 (0,0%)	2 (4,1%)	1 (2,0%)	0 (0,0%)	0 (0,0%)	0 (0,0%)	0 (0,0%)	0 (0,0%)	0 (0,0%)
3	0 (0,0%)	1 (2,0%)	0 (0,0%)	0 (0,0%)	1 (2,0%)	2 (4,1%)	0 (0,0%)	1 (2,0%)	1 (2,0%)
4	1 (2,0%)	1 (2,0%)	1 (2,0%)	3 (6,1%)	1 (2,0%)	2 (4,1%)	0 (0,0%)	1 (2,0%)	0 (0,0%)
5	2 (4,1%)	3 (6,1%)	5 (10,2%)	5 (10,2%)	6 (12,2%)	8 (16,3%)	3 (6,1%)	4 (8,2%)	3 (6,1%)
6	1 (2,0%)	8 (16,3%)	9 (18,4%)	5 (10,2%)	7 (14,3%)	7 (14,3%)	4 (8,2%)	3 (6,1%)	4 (8,2%)
7	11 (22,4%)	8 (16,3%)	16 (32,7%)	8 (16,3%)	8 (16,3%)	13 (26,5%)	3 (6,1%)	11 (22,4%)	12 (24,5%)
8	14 (28,6%)	16 (32,7%)	13 (26,5%)	13 (26,5%)	12 (24,5%)	12 (24,5%)	8 (16,3%)	8 (16,3%)	19 (38,8%)
9	20 (40,8%)	10 (20,4%)	4 (8,2%)	14 (28,6%)	13 (26,5%)	4 (8,2%)	31 (63,3%)	21 (42,9%)	9 (18,4%)
Mediana	8	8	7	8	8	7	9	8	8

Na Tabela 8.6, pode-se observar a distribuição das respostas do grupo ES (total de 31 indivíduos), que, no geral, apresenta uma melhor avaliação em *Dinâmica* e *Recursos* (como mostra as medianas), e em *Programar*, segue a tendência dos demais grupos. A QPE1 teve uma grande concentração de avaliações 9, sendo a mais bem avaliada com relação a todos os outros grupos. A QPE3, assim nos demais grupos, foi a que teve a pior avaliação em *Dinâmica*. Em *Programar* esse grupo segue o mesmo padrão dos demais tendo em vista suas medianas, sendo que a QPE6 foi a pior avaliada, com 32,3% de valores 6, a maior concentração abaixo de 7 encontrada nesta análise. Em *Recursos*, a questão QPE8 apresenta uma sutil melhora com relação aos demais grupos.

Tabela 8.6 Respostas do grupo ES às *Questões sobre a Percepção Emocional (QPE)*.

Escala SAM	Dinâmica			Programar			Recursos		
	QPE1 (S)	QPE2 (M)	QPE3 (D)	QPE4 (S)	QPE5 (M)	QPE6 (D)	QPE7 (S)	QPE8 (M)	QPE9 (D)
1	0 (0,0%)	0 (0,0%)	0 (0,0%)	0 (0,0%)	0 (0,0%)	0 (0,0%)	0 (0,0%)	0 (0,0%)	0 (0,0%)
2	0 (0,0%)	0 (0,0%)	1 (3,2%)	0 (0,0%)	0 (0,0%)	1 (3,2%)	0 (0,0%)	0 (0,0%)	1 (3,2%)
3	0 (0,0%)	0 (0,0%)	1 (3,2%)	0 (0,0%)	0 (0,0%)	1 (3,2%)	0 (0,0%)	0 (0,0%)	0 (0,0%)
4	0 (0,0%)	0 (0,0%)	0 (0,0%)	1 (3,2%)	0 (0,0%)	0 (0,0%)	0 (0,0%)	0 (0,0%)	1 (3,2%)
5	1 (3,2%)	0 (0,0%)	1 (3,2%)	2 (6,5%)	2 (6,5%)	2 (6,5%)	0 (0,0%)	1 (3,2%)	2 (6,5%)
6	0 (0,0%)	3 (9,7%)	7 (22,6%)	4 (12,9%)	5 (16,1%)	10 (32,3%)	1 (3,2%)	0 (0,0%)	4 (12,9%)
7	9 (29,0%)	3 (9,7%)	9 (29,0%)	7 (22,6%)	3 (9,7%)	6 (19,4%)	5 (16,1%)	6 (19,4%)	7 (22,6%)
8	4 (12,9%)	12 (38,7%)	6 (19,4%)	6 (19,4%)	12 (38,7%)	7 (22,6%)	7 (22,6%)	6 (19,4%)	7 (22,6%)
9	17 (54,8%)	13 (41,9%)	6 (19,4%)	11 (35,5%)	9 (29,0%)	4 (12,9%)	18 (58,1%)	18 (58,1%)	9 (29,0%)
Mediana	9	8	7	8	8	7	9	9	8

Em todas as análises dos grupos, é possível notar uma avaliação positiva por parte dos integrantes, com todas as QPEs em todos os grupos com a maior concentração de respostas com os valores 7 ou superior. A QPE3, questão com a pior avaliação em todos as análises, aponta que talvez possa haver melhorias a serem aplicadas à dinâmica adotada nas oficinas. A QPE6, que obteve esteve com a mediana mais baixa em todos os grupos (em especial no grupo ES), sinaliza que o ato de programar ainda pode ser complexo para o público participante, indicando que também pode ser revisto como apresentar os exemplos de código-fonte e a condução da programação durante as práticas.

Em uma visão geral, a avaliação se apresenta como positiva, em todos os agrupamentos e em todas as questões sobre *Dinâmica*, *Programar* e *Recursos*. Em *Recursos*, encontram-se as questões com a maior concentração de avaliações de valor 9 acima de 50%, indicando que a escolha pelos recursos didáticos do ComFAPOO pode ter sido o que mais agradou aos participantes.

8.6.3 Análise qualitativa sobre os relatos dos participantes

Ao final do questionário com as QPEs, foi aplicado um questionário com perguntas dissertativas, com o objetivo de capturar relatos que apontem aspectos que possam contribuir com a melhoria do método, tanto com apontamentos negativos, quanto positivos. Com a QA1 a intenção foi a de identificar, de maneira objetiva e direta, características que foram bem aceitas durante a prática ou que denotem aspectos positivos relacionados a componentes do ComFAPOO. Para análise das respostas dessa questão, abaixo seguem relatos que apontam aspectos sobre a dinâmica desenvolvida e os materiais disponibilizados:

“Muito prática, simples de entender e ilustrações que ajudam muito.”

“Interatividade, dinamismo e compreensão facilitada.”

“Fácil compreensão do assunto por meio de uma metodologia simples, dividindo o workshop em várias partes, com a explicação detalhada, imagens, exemplos de códigos e também a execução prática no arduino.”

“Os objetos concretos que auxiliaram no aprendizado, as páginas no GitHub com a explicação dos conceitos apresentados e a metodologia prática.”

“A aula foi bem dinâmica, acredito que é necessário ter essas experiências para o aprendizado.”

“Tivemos uma boa explicação sobre todo o assunto e utilizamos materiais físicos que deixaram a aula mais dinâmica porque podíamos ver o resultado na hora.”

“Entender de forma prática os conceitos de POO, Utilização de componentes eletrônicos, Didática bem simples e fácil entendimento”

Os relatos acima mostram que houve uma aceitação positiva com relação à dinâmica da oficina, assim como o material oferecido e os recursos utilizados. Um ponto que também surgiu nos relatos foi a percepção da apresentação dos conceitos de forma gradativa, como o esperado com a abordagem essencial da AS. Destacam-se os seguintes relatos sobre esse aspecto:

“A fácil introdução aos conteúdos, tudo explicado de maneira bem gradativa.”

“A apresentação gradativa dos conceitos.”

“A forma gradativa da exposição é bastante interessante.”

“Primeiramente fiquei assustado com os componentes, mas gradativamente fui entendendo como cada um funciona e por fim, compreendi cada função.”

Nessa QA1, a maior incidência de relatos ficou por conta do uso de objetos físicos (referenciados pelos alunos também como objetos “reais” e “concretos” nos relatos). Nota-se que esse foi um dos principais fatores favoreceram a aceitação positiva do método por parte dos participantes. A interação com o objeto físico programável e a resposta em tempo real que ele fornece, se destacou na maioria dos relatos. Entre os relatos, se destacam:

“Os principais pontos positivos da metodologia ComFAPOO são a aprendizagem com objetos [...]”

“O uso de objetos reais”

“O uso de objetos físicos mostrou mais claramente como a metodologia se aplica e acontece.”

“Unir objetos reais aos códigos tornou mais fácil o aprendizado.”

“Verificar os resultados do código fisicamente e a velocidade de aprendizado”

“A utilização de objetos reais para compreender a POO.”

“Fácil compreensão, o uso de objetos reais, a parte prática que era testar no próprio material, e imersão do assunto.”

“A presença de objetos concretos e a explicação dos conceitos abstratos foi bem simples de entender.”

“Conceitos aplicados com exemplos, a metodologia prática e a solução de um problema "real" (semáforo).”

“observar o objeto concreto juntamente a parte teórica da aplicação.”

A QA2, que teve como objetivo apresentar possíveis problemas com relação ao método e dinâmica das oficinas, tiveram a maior concentração de relatos negativos em dois aspectos. O primeiro está relacionado ao tempo dedicado à apresentação do conteúdo e condução das práticas didáticas. Nota-se uma insatisfação na questão do tempo dedicado à condução da oficina, como exemplifica os seguintes relatos:

“Achei que passamos muito rápido pela estrutura das classes, e o uso comum dos conceitos não ficou tão claro.”

“O conteúdo foi passado rápido.”

“Mais tempo para poder aplicar um projeto diferente dos propostos”

O outro aspecto negativo mais presente entre as respostas diz respeito da disposição dos códigos-fontes já prontos para teste e sua complexidade. No material disponível para os participantes, havia exemplos de códigos-fonte para que pudessem executar e verificar seu funcionamento junto aos componentes eletrônicos. Alguns relatos que destacam esse aspecto:

“os códigos ajudou bastante porem eram complicados.”

“Algumas partes do conteúdo não ficaram tão claras. Acredito que partes com código mais longos deveriam ser melhor explicada.”

“A utilização do "copiar e colar" atrapalhou um pouco na compreensão da linguagem em si, mas isso não impediu de aprender o conceito de POO. Acredito que essa medida foi tomada por conta da falta de tempo que tivemos.”

“a utilização do código fonte já pronto, poderíamos fazer a linha a linha”

Os relatos extraídos pela QA1 apontam que de maneira geral a estrutura do método, que privilegia a exposição gradativa de conceitos e o reforço de suas relações para compor a linha condutiva à aquisição de conhecimento, foi bem aceita. Também há evidências que a transição entre os modelos concretos, migrando para modelos mais abstratos foi notado e reconhecido como uma estratégia válida. Ainda com a QA1, a maior evidência identificada por

meio dos relatos foi a notória aceitação positiva pela adoção de objetos reais que possam ser controlados pelo código-fonte. A presença desses objetos reais, mesmo que alguns participantes tenham se sentido desafiados no início da prática, se mostrou como um recurso didático que agregou dinamismo e interativa para a prática, sendo um fator positivo para o método.

A QA2 mostrou apontou algumas deficiências do método e fatores que podem ser melhorados em trabalhos futuros. O tempo de cada aplicação e de toda a oficina poderia ser revisto, para contemplar mais conteúdo, exemplos e atividade, pois em alguns relatos fica explícito que esse é um fator que poderia agregar aspectos positivos se fosse revisto. Com relação ao problema sobre os códigos-fonte disponibilizados durante a oficina, os relatos mostram que também poderia haver um retrabalho, para talvez, alguma redefinição de exemplos mais simples para mais complexos, ou uma apresentação mais cadenciada desse código-fonte, iniciando com poucas linhas e comandos, e expandindo junto com os participantes.

8.7 Considerações finais

Esta seção apresenta as discussões finais sobre as contribuições, limitações e trabalhos futuros a partir desta pesquisa. Este trabalho teve como objetivo apresentar o método ComFAPOO e analisá-lo como um instrumento adequado ao seu propósito e que possa colaborar com o ensino de POO, por meio de sua aplicação em oficinas didáticas. Também apresenta um método de apresentação gradual de conceitos e que utiliza recursos potencialmente motivadores que podem ajudar os alunos no exercício de abstração.

8.7.1 Limitações da pesquisa

Apesar de avaliações positivas e perceptível evolução na aquisição de conhecimento por parte dos participantes, alguns aspectos devem ser levados em consideração sobre os resultados obtidos.

Como descrito anteriormente neste artigo, todos os participantes são alunos que estiveram sob o ensino à distância durante, pelo menos, 2 semestres do curso, devido à pandemia de Covid-19, nos semestres anteriores às oficinas conduzidas. Com isso, algumas deficiências na aprendizagem de conceitos fundamentais de programação e, consequentemente, de POO podem ter sido intensificadas devido à ausência de atividades laboratoriais (recorrente

desde o início de curso dessa área), devido à falta de experiência com esse modelo de ensino à distância ou até mesmo dificuldades de acesso à plataforma de estudo e equipamento para acompanhamento adequado das aulas. Esse cenário pode ter tido grande influência no questionário apresentado no início das oficinas, levando os participantes a darem avaliações mais baixas sobre o seu próprio conhecimento em POO. Essa baixa avaliação pode ter sido reflexo de não apenas deficiência na aprendizagem, mas também por falta de confiança em afirmar conhecimento em algo que pouco praticaram durante o curso. Com isso, a análise que mostra clara diferença encontrada entre pré-teste e pós-teste pode ser fragilizada e, consequentemente, questionada.

Outra limitação diz respeito ao perfil dos participantes das atividades de análise, todos alunos de cursos de formação profissional, técnico ou superior. A aceitação positiva do método ComFAPOO aponta para sua efetividade como instrumento didático, mas carece de novas aplicações, com grupos de perfis diferentes, para que se possa trazer novos resultados.

8.7.2 Trabalhos futuros

Entre os principais trabalhos futuros, pretende-se aplicar o método em novos grupos para testes e utilizar grupos de controle, para sanar fragilidades e limitações, supracitadas neste texto. Essas novas aplicações, além de poder validar os resultados positivos obtidos nesta pesquisa, poderão contribuir com o amadurecimento do método e dar base para análises mais criteriosas.

Outras premissas e ferramentas atreladas à AS podem ser incorporadas em novas versões do ComFAPOO. Por exemplo, a utilização de mapas conceituais para que possam ser usados recursos de composição e hierarquia da estrutura do conhecimento a ser apresentado junto à prática. A expansão de novas práticas também será cogitada, para que mais conceitos relacionados à POO possam ser incorporados, possivelmente cobrindo uma ementa completa de disciplina fundamental sobre esse paradigma de programação.

Devido às respostas positivas sobre os recursos utilizados, espera-se adotar novos construtos, mais complexos, que possam representar os objetos concretos a serem manipulados, como, por exemplo, robôs móveis e cenários compostos por vários componentes eletrônicos. A expansão desses construtos mais complexos, além de trazer uma demanda por práticas mais completas de programação, como o uso de mais objetos em um mesmo sistema ou princípios de práticas de projeto de software, podem favorecer mais o aspecto motivacional do método ComFAPOO.

8.7.3 Contribuições

O método ComFAPOO apresenta uma solução baseada em premissas fundamentais da AS e do modelo CoFa, comumente atrelados a métodos e práticas de ensino de ciências básicas e ao público de formação fundamental. Uma das contribuições que o método descrito neste trabalho apresenta é a construção de um material instrucional e dinâmica de apresentação do conteúdo que favorecem a construção do conhecimento do aluno no ensino de POO. Para o processo de apresentação gradual, que objetivasse a amparar o exercício de abstração, a adoção do modelo CoFa estrutura a forma de apresentação do conteúdo, por meio de modelos concretos, e migrando para modelos mais abstratos, como diagramas e código-fonte.

A adoção do ferramental da CF possibilitou que houvesse a interação desses objetos concretos com um modelo mais abstrato, a programação por meio do código-fonte. A utilização desses objetos concretos, no primeiro momento, ofereceu um recurso com o qual é possível verificar as características e ações inicialmente descritas, quando transportadas para um modelo mais abstrato (o código-fonte). Muitas vezes, em sala de aula, o professor tem dificuldade para estimular a capacidade de abstração de seu aluno, pela escassez de ferramentas que possam refletir na programação o comportamento de um objeto no mundo real. Do ponto de vista prático, umas das contribuições é o desenvolvimento de material didático como produto, podendo ser aplicado em oficinas com um público-alvo de mesmo perfil.

Com relação às análises feitas dos questionários apresentados durante a aplicação das oficinas, foi possível constatar o potencial do método ComFAPOO como uma ferramenta que pode contribuir com a aprendizagem de POO. As análises apontam que os aspectos relacionados aos recursos e dinâmica adotados pelo método foram bem recebidos pelos participantes de modo geral. Entretanto, as mesmas análises colaboram para a identificação de melhorias que podem ser avaliadas em novas versões do ComFAPOO. Dentre as melhorias, podem-se incluir: 1) inclusão de novos modelos concretos, mais complexos, que possam abordar mais conceitos de POO; 2) reestruturação do conteúdo programático para que, em vez de oficinas, o método possa ser implantado durante um semestre letivo; e 3) alteração do modelo de avaliação, para que seja adequado à aplicação durante uma disciplina.

No ponto de vista teórico, o método ComFAPOO apresenta uma proposta metodológica para aplicar práticas de ensino de POO, que alinhem princípios da AS e ferramental da CF. Pode-se também ressaltar que a utilização do modelo CoFa colabora com a

condução de uma prática baseada em AS, pois fornece um processo bem estruturado para a apresentação gradual de conceitos, partindo de objetos concretos e chegando até objetos mais abstratos, auxiliando o aluno no exercício de abstração e transição gradual de conceitos relacionados à POO.

8.8 Referências

- Astolfi, G., & Junior, D. L. (2015). Investigação sobre conhecimentos prévios de alunos do curso Técnico em Informática a partir da aplicação de organizadores prévios. *Aprendizagem Significativa. Revista/Meaningful Learning Review*, 15-28. Disponível em: <http://www.if.ufrgs.br/asr/artigos/Artigo_ID85/v5_n3_a2015.pdf>.
- Ausubel, D. P. (2003). *Aquisição e retenção de conhecimentos: uma perspectiva cognitiva. Lisboa: Plátano, 1.*
- Berssanette, J. H., & Francisco, A. C. (2018). Proposta de abordagem prática para o ensino de Programação baseada em Ausubel. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)* (Vol. 29, No. 1, p. 398). DOI: 10.5753/cbie.sbie.2018.398 . Disponível em: <<http://ojs.sector3.com.br/index.php/sbie/article/view/7996>>.
- Bradley, M. M., & Lang, P. J. (1994). Measuring emotion: the self-assessment manikin and the semantic differential. *Journal of behavior therapy and experimental psychiatry*, 25(1), 49-59. DOI: 10.1016/0005-7916(94)90063-9 . Disponível em: <<https://www.sciencedirect.com/science/article/abs/pii/0005791694900639?via%3Dihub>>.
- Brazileiro, R. B. (2013). *tAMARINO: uma abordagem visual para prototipagem rápida em computação física/Ricardo Borges Brasileiro* (Tese de Mestrado, Universidade Federal de Pernambuco). Disponível em: <<https://attena.ufpe.br/handle/123456789/12362>>.
- Bruner, J. S. (1966). *Toward a theory of instruction* (Vol. 59). Harvard University Press.
- Buriticá, Ó. I. T. (2013). Estrategia metodológica para aproximar los paradigmas funcional, estructurado y orientado a objetos en ingeniería de sistemas a partir de aprendizaje significativo. *Avances Investigación en Ingeniería*, 10(2), 49-63. DOI: 10.26507/rei.v12n23.719 . Disponível em: <<https://revistas.unilibre.edu.co/index.php/avances/article/view/2748>>
- Buriticá, Ó. I. T. (2014). Relaciones de aprendizaje significativo entre dos paradigmas de programación a partir de dos lenguajes de programación. *Tecnura*, 18(41), 91-102. DOI:

- <https://doi.org/10.14483/udistrital.jour.tecnura.2014.3.a07> . Disponível em: <
<https://revistas.udistrital.edu.co/index.php/Tecnura/article/view/7025> >
- Corral, J. M. R., Balcells, A. C., Estévez, A. M., Moreno, G. J., & Ramos, M. J. F. (2014). A game-based approach to the teaching of object-oriented programming languages. *Computers & Education*, 73, 83-92. DOI: 10.1016/j.compedu.2013.12.013. Disponível em: <
<https://www.sciencedirect.com/science/article/abs/pii/S0360131513003370> >
- Decker, R., & Hirshfield, S. (1994). The top 10 reasons why object-oriented programming can't be taught in CS 1. *ACM SIGCSE Bulletin*, 26(1), 51-55. DOI:
<https://doi.org/10.1145/191033.191054> . Disponível em: <
<https://dl.acm.org/doi/10.1145/191033.191054>>.
- Desportes, K. S. (2018). *Physical computing education: Designing for student authorship of values-based learning experiences* (Tese de Doutorado, Georgia Institute of Technology). Disponível em: <
<https://smartech.gatech.edu/handle/1853/60292>>.
- Eckerdal, A., & Thuné, M. (2005). Novice Java programmers' conceptions of "object" and "class", and variation theory. *ACM SIGCSE Bulletin*, 37(3), 89-93. DOI:
10.1145/1151954.1067473 . Disponível em: <
<https://dl.acm.org/doi/10.1145/1151954.1067473> >.
- Fyfe, E. R., McNeil, N. M., Son, J. Y., e Goldstone, R. L. (2014). Concreteness fading in mathematics and science instruction: A systematic review. *Educational psychology review*, 26(1), 9-25
- Fyfe, E. R., & Nathan, M. J. (2019). Making “concreteness fading” more concrete as a theory of instruction for promoting transfer. *Educational Review*, 71(4), 403-422. DOI:
10.1080/00131911.2018.1424116 . Disponível em: <
<https://www.tandfonline.com/doi/abs/10.1080/00131911.2018.1424116?journalCode=cedr20> >.
- Georgantaki, S., & Retalis, S. (2007). Using educational tools for teaching object oriented design and programming. *Journal of Information Technology Impact*, 7(2), 111-130. Disponível em: <
<http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.134.2804&rep=rep1&type=pdf> >.
- Hartmann, B., Klemmer, S. R., Bernstein, M., & Mehta, N. (2005). d. tools: Visually prototyping physical UIs through statecharts. In *Extended Abstracts of UIST 2005*. Disponível em: <
<http://people.csail.mit.edu/msbernst/papers/dtools-uist05.pdf>>.

- Jaakkola, T., & Veermans, K. (2018). Exploring the effects of concreteness fading across grades in elementary school science education. *Instructional Science*, 46(2), 185-207. DOI: 10.1007/s11251-017-9428-y . Disponível em: < <https://link.springer.com/article/10.1007/s11251-017-9428-y>>.
- Jaakkola, T., & Veermans, K. (2020). Learning electric circuit principles in a simulation environment with a single representation versus “concreteness fading” through multiple representations. *Computers & Education*, 148, 103811. DOI: 10.1016/j.compedu.2020.103811 . Disponível em: < <https://www.sciencedirect.com/science/article/abs/pii/S0360131520300130> >.
- Jang, Y., Lee, W., & Kim, J. (2015). Assessing the usefulness of object-based programming education using Arduino. *Indian Journal of Science and Technology*, 8, 90. DOI: 10.17485/ijst/2015/v8is1/57701 . Disponível em: < <https://indjst.org/articles/assessing-the-usefulness-of-object-based-programming-education-using-arduino>>.
- Knudsen, J. L., & Madsen, O. L. (1988). Teaching object-oriented programming is more than teaching object-oriented programming languages. In *European Conference on Object-Oriented Programming* (pp. 21-40). Springer, Berlin, Heidelberg. DOI: 10.7146/dpb.v17i251.7607 . Disponível em: < <https://tidsskrift.dk/daimipb/article/view/7607>>.
- Kölling, M. (1999). The problem of teaching object-oriented programming, Part 1: Languages. *Journal of Object-oriented programming*, 11(8), 8-15. Disponível em: < https://kar.kent.ac.uk/21879/2/the_problem_of_teaching_object-oriented_kolling_1.pdf>
- Korson, T., & McGregor, J. D. (1990). Understanding object-oriented: A unifying paradigm. *Communications of the ACM*, 33(9), 40-60. DOI: 10.1145/83880.84459 . Disponível em: < <https://dl.acm.org/doi/10.1145/83880.84459>>.
- Lima, Á., Diniz, M., & Eliasquevici, M. (2019). Metodologia 7Cs: Uma Nova Proposta de Aprendizagem para a Disciplina Algoritmos. In *Anais do XXVII Workshop sobre Educação em Computação* (pp. 429-443). SBC. DOI: 10.5753/wei.2019.6648 . Disponível em: < <https://sol.sbc.org.br/index.php/wei/article/view/6648>>.
- Mellis, D., Banzi, M., Cuartielles, D., & Igoe, T. (2007). Arduino: An open electronic prototyping platform. In *Proc. Chi* (Vol. 2007, pp. 1-11). Disponível em: < <https://alumni.media.mit.edu/~mellis/arduino-chi2007-mellis-banzi-cuartielles-igoe.pdf> >.
- Montero, S., Díaz, P., Díez, D., & Aedo, I. (2010). Dual instructional support materials for introductory object-oriented programming: classes vs. objects. In *IEEE EDUCON 2010*

- Conference* (pp. 1929-1934). IEEE. DOI: 10.1109/educon.2010.5492438 . Disponível em: < <https://ieeexplore.ieee.org/document/5492438> >.
- Moreira, M. A., & Masini, E. F. S. (2001). *Aprendizagem significativa: a teoria de David Ausubel* (2ª ed.). Ed. Centauro.
- Moreira, M. A. (2003). Linguagem e aprendizagem significativa. In *Conferência de encerramento do IV Encontro Internacional sobre Aprendizagem Significativa, Maragogi, AL, Brasil* (Vol. 8). Disponível em: < <http://www.if.ufrgs.br/~moreira/linguagem.pdf> >.
- Moreira, M. A. (2006). Aprendizagem Significativa: da visão clássica à visão crítica (Meaningful learning: from the classical to the critical view). In *Conferência de encerramento do V Encontro Internacional sobre Aprendizagem Significativa, Madrid, Espanha, setembro de. sn.* Disponível em: < <https://www.if.ufrgs.br/~moreira/visaoclasicavisaocritica.pdf> >.
- Moreira, M. A. (2010). O que é afinal aprendizagem significativa? Aula Inaugural do Programa de Pós-Graduação em Ensino de Ciências Naturais, Instituto de Física, Universidade Federal do Mato Grosso, Cuiabá, MT. Disponível em: < <http://moreira.if.ufrgs.br/oqueefinal.pdf> >.
- Moström, J. E., Boustedt, J., Eckerdal, A., McCartney, R., Sanders, K., Thomas, L., & Zander, C. (2008). Concrete examples of abstraction as manifested in students' transformative experiences. In *Proceedings of the Fourth international Workshop on Computing Education Research* (pp. 125-136). DOI: 10.1145/1404520.1404533 . Disponível em: < <https://dl.acm.org/doi/10.1145/1404520.1404533> >.
- Novak, J. D. (2002). Meaningful learning: The essential factor for conceptual change in limited or inappropriate propositional hierarchies leading to empowerment of learners. *Science education*, 86(4), 548-571. DOI: 10.1002/sce.10032 . Disponível em: < <https://onlinelibrary.wiley.com/doi/10.1002/sce.10032> >.
- Or-Bach, R., & Lavy, I. (2004). Cognitive activities of abstraction in object orientation: an empirical study. *ACM SIGCSE Bulletin*, 36(2), 82-86. DOI: 10.1145/1024338.1024378 . Disponível em: < <https://dl.acm.org/doi/10.1145/1024338.1024378> >.
- O'Sullivan, D., & Igoe, T. (2004). *Physical computing: sensing and controlling the physical world with computers*. Course Technology Press.
- Piva Jr, D., & Freitas, R. L. (2010). Estratégias para melhorar os processos de abstração na disciplina de Algoritmos. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)* (Vol. 1, No. 1). DOI: 10.5753/cbie.sbie.2010.%25p. Disponível em: < <http://ojs.sector3.com.br/index.php/sbie/article/view/1464> >.

- Przybylla, M., & Romeike, R. (2017). The nature of physical computing in schools: Findings from three years of practical experience. In *Proceedings of the 17th Koli Calling International Conference on Computing Education Research* (pp. 98-107). DOI: 10.1145/3141880.3141889 . Disponível em: < <https://dl.acm.org/doi/10.1145/3141880.3141889>>.
- Rubio, M. A., Hierro, C. M., & Pablo, A. P. D. M. (2013). Using Arduino to enhance computer programming courses in science and engineering. In *Proceedings of EDULEARN13 conference* (pp. 1-3). IATED Barcelona, Spain. Disponível em: < <http://wpd.ugr.es/~marubio/wp-content/uploads/2012/03/arduino.pdf>>.
- Sentance, S., Waite, J., Yeomans, L., & MacLeod, E. (2017). Teaching with physical computing devices: the BBC micro: bit initiative. In *Proceedings of the 12th Workshop on Primary and Secondary Computing Education* (pp. 87-96). DOI: 10.1145/3137065.3137083 . Disponível em: < <https://dl.acm.org/doi/10.1145/3137065.3137083> >.
- Soares, A. P., Pinheiro, A. P., Costa, A., Frade, C. S., Comesaña, M., & Pureza, R. (2013). Affective auditory stimuli: Adaptation of the international affective digitized sounds (IADS-2) for European Portuguese. *Behavior research methods*, 45(4), 1168-1181. DOI: 10.3758/s13428-012-0310-1 . Disponível em: < <https://link.springer.com/article/10.3758/s13428-012-0310-1>>.
- Suh, S., Lee, M., & Law, E. (2020). How do we design for concreteness fading? survey, general framework, and design dimensions. In *Proceedings of the Interaction Design and Children Conference* (pp. 581-588). DOI: 10.1145/3392063.3394413. Disponível em: < <https://dl.acm.org/doi/abs/10.1145/3392063.3394413>>.
- Yan, L. (2009). Teaching object-oriented programming with games. In *2009 Sixth International Conference on Information Technology: New Generations* (pp. 969-974). IEEE. DOI: 10.1109/itng.2009.13 . Disponível em: <<https://ieeexplore.ieee.org/document/5070749>>.

Capítulo 9

DISCUSSÃO

Este capítulo é destinado a expor discussões com relação aos conteúdos dos artigos produzidos durante a pesquisa e apresentados neste trabalho. Tais discussões têm como objetivo mostrar as relações entre os artigos e como cada trabalho contribuiu com a totalidade da pesquisa. Essas discussões são orientadas pelas etapas de desenvolvimento da pesquisa (apresentadas na Seção 1.3).

A Seção 9.1 apresenta as discussões acerca das revisões sobre a literatura aplicada na pesquisa e a Seção 9.2 discute a investigação em torno da concepção e desenvolvimento do método ComFAPOO. Na Seção 9.3 são apresentadas discussões sobre as aplicações das oficinas com o método ComFAPOO e os resultados obtidos.

9.1 Sobre a Revisão da Literatura

A etapa de Revisão da Literatura teve como foco identificar oportunidades de pesquisa que pudessem trazer novas contribuições para a área de ensino de POO. O artigo que compõe o Capítulo 2 foi desenvolvido logo no início da trajetória do doutorado, tendo sido essencial para um primeiro mapeamento sobre possibilidades de pesquisa. Usando como foco trabalhos que abordassem PC apresentados em eventos e publicações no Brasil, o trabalho, mesmo não estando com a temática central que a pesquisa adotou, contribuiu para identificar questões que foram abordadas ao longo da pesquisa. No trabalho foi identificado que uma das principais habilidades praticadas em atividades que promovem PC era a de abstração, um dos focos desta pesquisa. Foi também observada a pequena quantidade de trabalhos que utilizassem recursos tangíveis e programação como os de RP e CF.

Os artigos dos Capítulos 3 e 4 tiveram como objetivo mapear trabalhos recentes para identificar como os recursos da CF e premissas da AS eram utilizados em trabalhos que envolviam programação. Os resultados desses mapeamentos auxiliaram nas respostas para o Problema de Pesquisa 1 (identificar quais recursos podem ajudar no exercício de abstração

necessária para o ensino de POO), guiando a pesquisa no aspecto teórico e determinando os principais elementos que compõem o ComFAPOO: recursos didáticos por meio da CF e alinhamento com a premissas da AS.

O artigo do Capítulo 3 conclui que as pesquisas buscam os recursos da CF para criar um ambiente que seja ao mesmo tempo motivador e eficaz no ensino de programação. Também constata que a plataforma Arduino é amplamente usada em trabalhos com CF. Essas conclusões foram relevantes para definir a CF e a plataforma Arduino como o ferramental adotado nesta pesquisa.

Por meio dos resultados do artigo presente no Capítulo 4, foi possível identificar o cenário atual de trabalhos que envolvem programação e AS. Um dos pontos mais importantes elucidados foi identificar quais eram as estratégias didáticas mais utilizadas nos trabalhos para promover a AS. A teoria da AS, como afirmam Moreira (2011) e Agra et al. (2019), é um conceito teórico, um modelo com premissas que devem ser seguidas, mas não apresenta nenhum método definido para institucionalizar o ensino. Por isso, para usar a AS como guia, devem ser adotados métodos que possam proporcionar a dinâmica da aula. Com isso, foi importante investigar quais estratégias e recursos são adotados na academia, para a compreensão do cenário vigente, assim como encontrar lacunas para novas contribuições. Outra relevante colaboração que o trabalho exposto no Capítulo 4 trouxe para esta pesquisa foi a identificação do público-alvo mais comum entre os trabalhos da área: em sua maior parte os trabalhos estão associados a formação profissional (técnico e superior). Ao final da etapa de mapeamento da literatura e definição de público-alvo, a escolha de conduzir o método para o ensino de POO está diretamente alinhado à demanda e lacunas de pesquisa que focalizam em alunos na formação profissional em cursos de Computação.

Além dos resultados, cada um dos artigos dos Capítulos 3 e 4 apresenta o referencial teórico que fundamentou toda a pesquisa. Com relação ao referencial teórico, os artigos presentes no Capítulo 5 (Seções 5.1 a 5.6), Capítulo 6 (Seções 6.1 e 6.2), Capítulo 7 (Seção 7.1) e Capítulo 8 (Seções 8.1 a 8.3) contribuem com descrições sobre toda as bases teóricas aplicadas na pesquisa, incluindo o referencial sobre CoFa.

9.2 Sobre o Desenvolvimento do Método

Os trabalhos apresentados nos Capítulos 5 e 6 estão diretamente relacionados com a etapa de Desenvolvimento do Método. Eles apresentam discussões e propostas sobre o desenvolvimento

de um método de ensino de POO que possa alinhar CF e AS. Nessa etapa de pesquisa, o objetivo foi definir qual seria a estrutura do método e as estratégias didáticas que iriam compô-lo. Um dos impactos para a pesquisa em geral desses trabalhos foi a base para resolução dos questionamentos dos Problema de Pesquisa 2 (definir estratégias didáticas que possam auxiliar na compreensão e correlação entre os conceitos de POO) e Problema de Pesquisa 3 (criar um método instrucional de apresentação do conteúdo didático), obtendo assim uma formalização do método ComFAPOO.

O Capítulo 5 apresenta um artigo de opinião, com uma análise crítica e embasada na literatura, sobre os potenciais usos de recursos da CF, juntamente com estratégias que sigam as diretrizes da AS. No texto, são colocados os pontos que levam a pesquisa a relacionar a CF e a AS com um modelo instrucional com uma dinâmica de aula baseada no modelo CoFa. Um modelo instrucional apresenta guias ou estratégias que podem orientar a aplicação das dinâmicas, estratégias e ferramentas didáticas em sala de aula (SALAM, IBRAHIM e SUKARDJO, 2019; OWSTON, YORK e MALHOTRA, 2019). Como resultado, foi possível delimitar qual seriam os principais componentes do ComFAPOO, orientados pela fundamentação teórica estudada até o momento.

O conceito de *subsunção* ou *conceito inclusor*, proposto pela AS, que auxilia o aluno a ingressar e emergir no processo de construção de conhecimento, é um ponto fundamental no processo de aprendizagem. Entretanto, para o ensino de programação, em especial em POO, são comumente utilizadas representações não-concretas, que não possibilitam uma interação com aluno ou uma elucidação sem muito uso do pensamento abstrato. Outra premissa essencial da AS é a intenção do aluno de aprender: para que isso ocorra, o engajamento deve ocorrer desde o começo do processo e perdurar ao longo de toda a prática didática.

A apresentação de objetos concretos visa proporcionar um ambiente em que o aluno possa ter domínio e compreensão maior dos objetivos da programação que ocorrerá, assim como associar conceitos de POO com situações em ambientes reais e verificar o resultado da programação em tempo real. A transição dessas representações concretas, até a representação mais abstrata (o código-fonte) está alinhada com duas outras premissas fundamentais da AS, a *diferenciação progressiva* e a *reconciliação integrativa*.

Contudo, a AS não traz um método instrucional para aplicação de práticas didáticas, havendo apenas diretrizes a serem seguidas. Para essa adequação instrucional, o método ComFAPOO adota o modelo CoFa, que se alinha com a demanda de encadeamento em que se baseiam *diferenciação progressiva* e a *reconciliação integrativa*, além de conduzir a prática

didática iniciando em objetos concretos e migrando por representações mais abstratas, até o conceito a ser aprendido.

O *framework* CoFa, como define Fyfe et al. (2014), tem como diretriz a utilização de objetos concretos (não abstratos) como o ponto de partida para a exposição de algum conceito e, gradativamente, por meio de representações intermediárias, apresentar o conceito em sua forma mais abstrata. Com isso, sua adoção possibilitou criar um meio sistemático para aplicação da prática didática. No Capítulo 6 é apresentada, de maneira breve, a formalização do método ComFAPOO, dando ênfase à adoção do CoFa, como sendo a estratégia que alinha CF e as premissas da AS. Nesse capítulo também são descritas todas as etapas principais do método e a estrutura básica do método ComFAPOO.

O Capítulo 7 (Seção 7.4) e o Capítulo 8 (Seção 8.4) apresentam maiores detalhes do desenvolvimento do método. No Capítulo 7 há uma descrição geral sobre o método, estendendo as discussões sobre a composição das práticas didáticas e avaliações. No Capítulo 8, são apresentados maiores detalhes das bases teóricas que compõem o método, assim como do desenvolvimento do material didático e das avaliações.

9.3 Sobre a Aplicação das Oficinas e Resultados

A etapa de Aplicação das Oficinas foi essencial para a validação do método, com a identificação de potenciais adequações e melhorias em versões futuras. Os Capítulos 7 e 8 discutem os resultados de duas aplicações do método.

O Capítulo 7 apresenta uma aplicação preliminar com um número reduzido de participantes, com uma primeira análise sobre a eficácia do método e dos meios de avaliações escolhidos. O formato atual do ComFAPOO é adequado para a condução de oficinas, fator este que foi determinante para a escolha dos procedimentos de avaliação adotados e conteúdo aplicado no material disponibilizado. Com relação ao desenvolvimento do material didático, um dos pontos fundamentais foi determinar quais conceitos compunham os fundamentos da POO (discussão apresentada na Seção 7.3).

O artigo apresentado no Capítulo 8 é o trabalho mais completo entre todos que formam esta tese. Além de apresentar um detalhamento mais abrangente sobre o referencial teórico (complementar aos pontos discutidos no Capítulo 5) e trabalhos relacionados, também mostra com maiores detalhes como foram compostas as propostas de práticas didáticas durante as oficinas e uma análise completa com todo os participantes envolvidos na pesquisa.

Com relação às avaliações aplicadas, as escolhas dos métodos avaliativos e questionários foram feitas para estar alinhadas com o formato de oficinas. A escolha pela não adoção de um grupo de controle deu-se devido à rotina de aulas que envolviam os participantes em um cenário de retorno pós pandemia de Covid-19¹¹. As unidades de ensino onde foram realizadas as oficinas, assim como grande parte das instituições de ensino, teve taxas de evasão altas no primeiro semestre letivo do ano de 2022. Portanto, para que não houvesse grupos com números pequenos de indivíduos, foi tomada a decisão de trabalhar com um delineamento experimental de pré e pós-teste, sem adotar grupos de controle e focal.

Segundo Sousa, Driessnack e Mendes (2007), a abordagem quase-experimental é útil para testar a efetividade de uma intervenção, mas sofre grandes ameaças à validade do experimento. Contudo, considerando-se que o método ComFAPOO é um método que ainda deve ser adequado para a aplicação em um contexto de currículo formal (semestral ou anual) dentro de disciplinas de POO, a adoção dessa estratégia experimental é satisfatória para um primeiro conjunto de avaliações do método.

Dentre as análises, houve também uma de caráter qualitativo, por meio de relatos (questões abertas) respondidas pelos participantes das oficinas. Neves (1996) aponta que a pesquisa qualitativa tem como objetivo traduzir e expressar o sentido de fenômenos em uma visão social, expresso por uma perspectiva vivida por uma experiência. Com isso, os resultados da análise qualitativa apresentam aspectos que impactam diretamente os alunos envolvidos, tanto positivamente quanto negativamente, fornecendo para a pesquisa pontos que podem ser revisitados, tanto para serem adaptados quanto melhorados.

Os resultados obtidos, como discutido no Capítulo 8, apontam a eficácia do método ComFAPOO dentro do cenário explorado pela pesquisa. Entretanto, há apontamentos de questões negativas feitos pelos participantes das oficinas e apresentado nas análises, que serão consideradas para trabalhos futuros.

¹¹ As oficinas foram realizadas entre abril e maio de 2022.

Capítulo 10

CONCLUSÃO

Esta pesquisa buscou investigar como utilizar recursos tangíveis e concretos para fornecer um método de ensino que auxilie o aluno iniciante em programação no exercício de abstração, apoiando o aprendizado de POO, de uma maneira que todo o processo de aquisição do conhecimento tenha significado e engajamento. O método desenvolvido apresenta uma solução por meio dinâmicas de ensino baseadas em premissas didáticas e ferramentas que conduzem a compreensão de conceitos fundamentais de POO, usando objetos físicos que possam interagir com programação.

A motivação deste trabalho veio da identificação de lacunas nas pesquisas recentes sobre o ensino de POO na formação profissional (superior e médio). O objetivo buscado foi eliminar essas lacunas com um método que use recursos que possam trazer maior dinamismo e interatividade durante as aulas. Tendo em vista a questão geral desta pesquisa e os problemas que a nortearam, foi possível verificar por meios dos resultados que o objetivo principal deste trabalho foi alcançado. O desenvolvimento do ComFAPOO proporcionou contribuições relevantes e atende a busca por um método que alinhe premissas da AS e recursos da CF para amparar o exercício de abstração do aluno no processo de ensino-aprendizagem de PO

A principal contribuição e maior fator de ineditismo deste trabalho foi a proposição do método ComFAPOO, utilizando recursos da CF alinhados a premissas definidas pela AS, para o ensino de POO em cursos profissionalizantes. O método apresenta uma solução que atende a abordagem dos conceitos fundamentais do ensino de POO, promovendo o exercício de abstração, interação com objetos tangíveis e fazendo associações com conceitos abstratos. A proposta atende um público-alvo abrangente, que compõe cursos de formação de nível técnico e superior, que demandam cada vez mais soluções que possam proporcionar uma experiência didática engajadora, e que, ao mesmo tempo, atenda os componentes curriculares de sua formação.

Os materiais desenvolvidos para as oficinas realizadas buscam oferecer ao aluno e professor-aplicador cenários de aprendizado interativos e de fácil compreensão e que apoie a capacidade de abstração do aluno. A transição gradual de conceitos e a facilidade de

compreensão são fatores que podem auxiliar o aluno no exercício de abstração, tornando a apresentação do código-fonte uma etapa mais natural e correlacionável com todos os conceitos e discussões anteriormente apresentadas. Através dos resultados obtidos, por meio das avaliações dos participantes das oficinas conduzidas, foi possível notar que o principal objetivo do método foi alcançado.

Além dos materiais instrucionais desenvolvidos, a dinâmica de apresentação dos conceitos foi concebida integrando bases teóricas amplamente utilizadas em pesquisas relacionadas ao ensino de programação, mas para as quais antes não havia uma convergência ou integração metodológica. A fim de integrar as premissas da AS e o ferramental da CF em um processo didático, o modelo instrucional CoFa foi aplicado, fornecendo uma estrutura de apresentações de conceitos que pode atender essa demanda. O CoFa apresentou uma solução para apresentação de conceitos que faz a transição de objetos concretos para mais abstratos, alinhando as premissas de *subsunções*, *diferenciação progressiva* e a *reconciliação integrativa* defendidas pela AS com artefatos existentes na CF. Essa integração se apresenta como uma das colaborações desta pesquisa, que pode ser utilizada como base para outras propostas didáticas envolvendo AS, CF e CoFa, inexistente na literatura e em pesquisas em torno da temática.

Com as análises dos resultados, é possível identificar aspectos positivos e negativos com relação aos questionários aplicados e ao público-alvo. Como a aceitação positiva de iniciativas, os resultados apontam que o envolvimento de objetos concretos e recurso utilizados, e a dinâmica das práticas ofertadas foram bem recebidos pelos participantes das oficinas.

Dentre os apontamentos de caráter negativo, relatados pelos participantes, destacam-se o tempo dedicado às práticas e a apresentação dos códigos-fontes. Com relação ao tempo, o modelo adotado (oficinas) tem o caráter de uma aplicação mais curta, se comparado a uma disciplina semestral.

Esta pesquisa contribui também para futuros trabalhos que visem explorar o uso de CF ou AS, de forma integrada ou não, expondo aspectos que podem ser adotados ou modificados em outras propostas de métodos de ensino. A apresentação de uma solução com o uso do modelo CoFa junto a objetos concretos que possam interagir diretamente com programação (modelos abstratos), proporciona uma base para que futuras pesquisas possam explorar esse cenário, apresentando novas soluções em conjunto com novas teorias ou modelos didáticos, diferentes da AS.

As limitações da pesquisa estão principalmente relacionadas ao contexto vivido pelos alunos participantes das oficinas durante o período da pandemia de Covid-19 e à estrutura atual do método, adequado para o formato de oficinas. Todos os participantes estavam em

regime à distância antes do semestre letivo no qual as oficinas foram realizadas, criando um impacto direto nas experiências e vivência com o conteúdo e práticas em laboratórios nos cursos em que estavam matriculados. Além disso, as instituições de ensino onde foram realizadas a pesquisa tiveram altos índices de evasão durante o período de ensino remoto que ocorreu devido à pandemia e em seu período de retomada às aulas presenciais. A presente versão do ComFAPOO foi desenvolvida para que fosse em um formato de oficina, para que pudesse atender tanto a proposta de investigação quanto à participação dos alunos e o contexto escolar em que estavam inseridos.

Tendo em vista as limitações desta pesquisa e os resultados obtidos, trabalhos futuros podem ser traçados. Com relação ao formato das práticas e conteúdo trabalhado, seria interessante a proposição de uma nova formatação do ComFAPOO, para a aplicação em uma disciplina no formato semestral. Com esse formato, além de maior possibilidade de experimentação, também seria possível compreender como o método pode apoiar uma disciplina dentro de um currículo de formação profissional. Essa adequação para o formato de disciplina também demandaria uma reformulação das ferramentas de avaliação, que deverão ser adequadas a uma rotina semestral em ensino formal.

Outro aspecto que pode ser explorado em um trabalho relacionado é a proposição de outros construtos, mais complexos, que possam exemplificar com maiores detalhes conceitos de POO, ou até mesmo expandi-los. Além dos circuitos simples propostos neste trabalho, criar cenários mais ricos, com detalhes e composições, poderá não só deixar a experiência mais atrativa aos alunos, mas também servirá para aplicar conceitos mais amplos, como boas práticas em POO e conceitos de projeto e otimização de software. Também é relevante expandir o conjunto de conceitos relacionados à POO discutidos no método e a utilização e adequação de novas ferramentas, como mapas mentais ou outros meios de relacionamentos de conceitos.

Esta pesquisa apresenta uma contribuição original sobre a temática de ensino de POO, em especial com o uso de CF e o alinhamento com a premissas da AS. Mesmo que o método ComFAPOO não esgote todos os questionamentos em torno do tema, ele apresentou resultados positivos, alcançando os objetivos propostos no início da pesquisa e apresentando uma solução viável e adequada ao público-alvo. Os resultados, obtidos e apresentados nesta tese, podem ser base de direcionamentos para futuras pesquisas, incluindo a continuidade do método ComFAPOO em novas versões, com novos materiais, práticas, apresentação de conceitos de POO e adequação para outros públicos-alvo.

REFERÊNCIAS

- ABBASI, S. et al. Investigating Student's Obstacles While Learning Object Orientation. **INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH**, v. 6, p. 7-11, 2021.
- AGRA, G. et al. Analysis of the concept of Meaningful Learning in light of the Ausubel's Theory. **Revista brasileira de enfermagem**, v. 72, p. 248-255, 2019.
- AURELIANO, V. C. O.; TEDESCO, P. C. A. R.; GIRAFFA, L. M. M. Desafios e oportunidades aos processos de ensino e de aprendizagem de programação para iniciantes. In: **WORKSHOP SOBRE EDUCAÇÃO EM COMPUTAÇÃO (WEI)**, 24., 2016, Porto Alegre. **Anais [...]**. Porto Alegre: Sociedade Brasileira de Computação, 2016. p. 2066-2075.
- BECK, K.; CUNNINGHAM, W. A laboratory for teaching object oriented thinking. **ACM Sigplan Notices**, v. 24, n. 10, p. 1-6, 1989.
- BENNEDSEN, J.; CASPERSEN, M. E. Abstraction ability as an indicator of success for learning object-oriented programming?. **ACM Sigcse Bulletin**, v. 38, n. 2, p. 39-43, 2006.
- BENNEDSEN, J.; CASPERSEN, M. E. Failure rates in introductory programming. **ACM SIGcSE Bulletin**, v. 39, n. 2, p. 32-36, 2007.
- FALCADE, A. et al. Design Instrucional: um comparativo de metodologias para definição de abordagem em mundo virtual. In: **Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)**. 2016. p. 80.
- FYFE, E. R. et al. Concreteness fading in mathematics and science instruction: A systematic review. **Educational psychology review**, v. 26, n. 1, p. 9-25, 2014.
- GOMES, A.; MENDES, A. Learning to program - difficulties and solutions. In: **ICEE Conference (FIE) Proceedings of the International Conference on Engineering Education, (Coimbra, Portugal)**. ICEE, 2007
- GOMES, A.; MENDES, A. A teacher's view about introductory programming teaching and learning: Difficulties, strategies and motivations. In: **2014 IEEE Frontiers in Education Conference (FIE) Proceedings**. IEEE, 2014. p. 1-8.

GUTIÉRREZ, L. E.; GUERRERO, C. A.; LÓPEZ-OSPINA, H. A. Ranking of problems and solutions in the teaching and learning of object-oriented programming. **Education and Information Technologies**, p. 1-35, 2022.

HADAR, I.. When intuition and logic clash: The case of the object-oriented paradigm. **Science of Computer Programming**, v. 78, n. 9, p. 1407-1426, 2013.

JAAKKOLA, T.; VEERMANS, K. Learning electric circuit principles in a simulation environment with a single representation versus “concreteness fading” through multiple representations. **Computers & Education**, v. 148, p. 103811, 2020.

KAILA, E. et al. Redesigning an object-oriented programming course. **ACM Transactions on Computing Education (TOCE)**, v. 16, n. 4, p. 1-21, 2016.

KEMPF, R.; STELZNER, M. Teaching object-oriented programming with the KEE system. **ACM SIGPLAN Notices**, v. 22, n. 12, p. 11-25, 1987.

KNUDSEN, J. L.; MADSEN, O. L. Teaching object-oriented programming is more than teaching object-oriented programming languages. In: **European Conference on Object-Oriented Programming**. Springer, Berlin, Heidelberg, 1988. p. 21-40.

LIAN, V.; VAROY, E.; GIACAMAN, N. Learning Object-Oriented Programming Concepts Through Visual Analogies. **IEEE Transactions on Learning Technologies**, 2022.

LIBERMAN, N.; BEERI, C.; KOLIKANT, B. Y. Difficulties in learning inheritance and polymorphism. **ACM Transactions on Computing Education (TOCE)**, v. 11, n. 1, p. 1-23, 2011.

MICHAELSON, G. Microworlds, objects first, computational thinking and programming. In: **Computational thinking in the STEM disciplines**. Springer, Cham, 2018. p. 31-48.

MOREIRA, M. A. O que é afinal, Aprendizagem Significativa? **Quriculum, Laguna**, v. 2, n. 3, p. 1-27, 2012. Disponível em: < <http://moreira.if.ufrgs.br/oqueeafinal.pdf> >.

MOSTRÖM, J. E. et al. Concrete examples of abstraction as manifested in students' transformative experiences. In: **Proceedings of the Fourth international Workshop on Computing Education Research**. 2008. p. 125-136.

NYGAARD, K.. Basic concepts in object oriented programming. In: **Proceedings of the 1986 SIGPLAN Workshop on Object-oriented Programming**. 1986. p. 128-132.

OR-BACH, R.; LAVY, I.. Cognitive activities of abstraction in object orientation: an empirical study. **ACM SIGCSE Bulletin**, v. 36, n. 2, p. 82-86, 2004.

OWSTON, R.; YORK, D. N.; MALHOTRA, T.. Blended learning in large enrolment courses: Student perceptions across four different instructional models. **Australasian Journal of Educational Technology**, v. 35, n. 5, p. 29-45, 2019.

RAGONIS, N.; BEN-ARI, M.. A long-term investigation of the comprehension of OOP concepts by novices. **Comp. Sci. Educ.** v. 15, n. 3, p. 203–221, 2005.

RENTSCH, T.. Object oriented programming. **ACM Sigplan Notices**, v. 17, n. 9, p. 51-57, 1982.

ROBINS, A., ROUNTREE, J., ROUNTREE, N. Learning and teaching programming: A review and discussion **Computer Science Education**, v. 13, n.2 p. 137-172, 2003.

SAJANIEMI, J.; BYCKLING, P.; GERDT, P. Animation metaphors for object-oriented concepts. **Electronic Notes in Theoretical Computer Science**, v. 178, p. 15-22, 2007

SALAM, M.; IBRAHIM, N.; SUKARDJO, M. Effects of Instructional Models and Spatial Intelligence on the Mathematics Learning Outcomes after Controlling for Students' Initial Competency. **International Journal of Instruction**, v. 12, n. 3, p. 699-716, 2019.

SOUSA, V. D.; DRIESSNACK, M.; MENDES, I. A. C.. Revisão dos desenhos de pesquisa relevantes para enfermagem: Parte 1: desenhos de pesquisa quantitativa. **Revista Latino-Americana de Enfermagem**, v. 15, p. 502-507, 2007.

STEFIK, M.; BOBROW, D. G. Object-oriented programming: Themes and variations. **AI magazine**, v. 6, n. 4, p. 40-40, 1985.

VILAÇA, M. L. C. Pesquisa e ensino: considerações e reflexões. **Revista escrita: Revista do Curso de Letras da UNIABEU**, v. 1, n. 2, p. 59-74, 2010.

WEBB, M. et al. Computer science in K-12 school curricula of the 21st century: Why, what and when?. **Education and Information Technologies**, v. 22, n. 2, p. 445-468, 2017.

XINOGALOS, S. Object-oriented design and programming: an investigation of novices' conceptions on objects and classes. **ACM Transactions on Computing Education (TOCE)**, v. 15, n. 3, p. 1-21, 2015.

XINOGALOS, S.; SATRATZEMI, M.; DAGDILELIS, V. An introduction to object-oriented programming with a didactic microworld: objectKarel. **Computers & Education**, v. 47, n. 2, p. 148-171, 2006.

ZANETTI, H. A. P.; BORGES, M. A. F. Ensino de Programação utilizando Computação Física: uma Revisão Sistemática da Literatura. **Comunicações em Informática**, [S. l.], v. 4, n. 1, p. 19–22, 2020. DOI: 10.22478/ufpb.2595-0622.2020v4n1.52065.

ZANETTI, H. A. P.; BORGES, M. A. F. Por que estimular a Aprendizagem Significativa no ensino de Programação Orientada a Objetos. In: **SIMPÓSIO BRASILEIRO DE EDUCAÇÃO EM COMPUTAÇÃO (EDUCOMP)**, 1., 2021, On-line. Anais [...]. Porto Alegre: Sociedade Brasileira de Computação, 2021. p. 290-295. DOI: <https://doi.org/10.5753/educomp.2021.14496>.

ZANETTI, H. A. P.; BORGES, M. A. F.; RICARTE, I. L. M. Pensamento computacional no ensino de programação: Uma revisão sistemática da literatura brasileira. In: **XXVII Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)**. 2016. p.21.

ZANETTI, H. A. P.; BORGES, M. A. F.; RICARTE, I. L. M. Método de Ensino de Programação Orientada a Objetos Baseada em Computação Física, Aprendizagem Significativa e Concreteness Fading. In: **Challenges 2021, desafios do digital: Livro de atas. Universidade do Minho. Centro de Competência**. ed. 1, p.101-107, 2021. Disponível em: <<https://www.nonio.uminho.pt/challenges/download/1344/>>

ZANETTI, H. A. P.; BORGES, M. A. F.; RICARTE, I. L. M. Aplicação de um Método para Ensino de Programação Orientada a Objetos por meio de Aprendizagem Significativa e Computação Física. In: **WORKSHOP SOBRE EDUCAÇÃO EM COMPUTAÇÃO (WEI)**, 30., 2022a, Niterói. **Anais [...]**. Porto Alegre: Sociedade Brasileira de Computação, 2022. p. 37-48. ISSN 2595-6175. DOI: <https://doi.org/10.5753/wei.2022.222953>.

ZANETTI, H. A. P.; BORGES, M. A. F.; RICARTE, I. L. M.. A Teoria de Aprendizagem Significativa no Ensino de Programação: um Mapeamento Sistemático da Literatura. In:

SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 33. , 2022, Manaus. **Anais** [...]. Porto Alegre: Sociedade Brasileira de Computação, 2022b . p. 01-14. DOI: <https://doi.org/10.5753/sbie.2022.224579>.

APÊNDICE A – Material didático das oficinas

Este apêndice apresenta o material didático utilizado nas oficinas com o método ComFAPOO. O material original que está disponível em <<https://github.com/humbertozanetti/comfapoo>>.

01/09/2022 17:20

humbertoanetti/comfapoo

humbertoanetti / comfapoo Public

☆ 2 stars 🍴 0 forks

☆ Star

👁 Unwatch ▾

Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

🔗 main ▾

...



humbertoanetti ...

4 minutes ago ⌚

[View code](#)

Workshop ComFAPOO (Computação Física para Aprendizagem de Programação)

☰ README.md



Workshop idealizado utilizando o [método ComFAPOO](#), criado por pesquisadores do [Laboratório de Informática, Aprendizagem e Gestão \(LIAG\)](#) da [Faculdade de Tecnologia \(FT\)](#) da [Universidade Estadual de Campinas \(UNICAMP\)](#).

Objetivo do método

Ensinar conceitos fundamentais de Programação Orientada a Objetos (POO) promovendo uma aprendizagem significativa através do uso de componentes físicos programáveis, auxiliando no processo abstração de conceitos complexos por meio de objetos concretos.

Algumas publicações sobre o método:

- ZANETTI, H. A. P.; BORGES, M. A. F.. Por que estimular a Aprendizagem Significativa no ensino de Programação Orientada a Objetos?. In: Anais do Simpósio Brasileiro de Educação em Computação. SBC, 2021. p. 290-295. Disponível [aqui](#).
- ZANETTI, H. A. P.; BORGES, M. A. F. ; RICARTE, I. L. M.. Método de ensino de programação Orientada a Objetos baseado em Computação Física, Aprendizagem Significativa e Concreteness Fading. In A. J. Osório, M. J. Gomes, A. Ramos, & A. L. Valente (Eds.), Challenges 2021, desafios do digital: Livro de atas (1.ª ed., pp. 23-32).

01/09/2022 17:20

humbertoanetli/compapoo

Universidade do Minho. Centro de Competência. Disponível [aqui](#) (baixar "Livro de Atas").

Questionário para ser respondido antes do Workshop

[Link para o questionário](#)

Etapas do workshop

Parte 1 - [Classe e Objeto](#)

Parte 2 - [Encapsulamento](#)

Parte 3 - [Herança](#)

Parte 4 - [Polimorfismo](#)

Parte 5 - [Sistema orientado a objetos](#)

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

01/09/2022 17:20

comfapoo/workshop_parte1.md at main · humbertozanetti/comfapoo

 humbertozanetti / comfapoo Public[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#) main ▾

...

[comfapoo](#) / [workshop_parte1.md](#)

humbertozanetti Update workshop_parte1.md

 1 contributor 193 lines (148 sloc) | 5.68 KB

...

WORKSHOP PARTE 1

Classe e Objeto

Objetivos da prática

- Entender o conceito de classe como uma identidade que descreve um componente real, o **objeto**.
- Descrever os principais componentes (propriedades e ações) do componente eletrônico, apresentando os conceitos de **atributos** e **métodos**.
- Mostrar a relação entre a classe e o objeto (**instanciação** e uso de método **construtor**).
- Fazer o exercício de **abstração**, ou seja, entender a **identidade** do objeto (sua função no sistema geral), suas **propriedades** e suas **ações**.

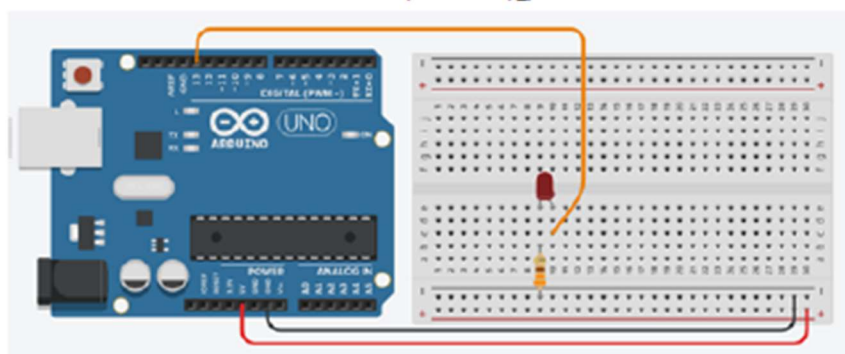
Tarefa 1 - Conhecendo o objeto LED

Nesta tarefa vamos conhecer o objeto LED real, entender quais são suas principais **propriedades** e **ações** através da demonstrações e alguns diagramas.

Circuito para a tarefa

01/09/2022 17:20

comfapoo/workshop_parte1.md at main · humberiozanetti/comfapoo



Lista de componentes para montagem

- 1 LED vermelho
- 1 resistor de 330 ohms
- 3 cabos de conexão

Objeto LED

ESTADOS



aceso



apagado



pino

Propriedades:

pino: pino ligado ao Arduino

estado: aceso ou apagado

Ações:

acender: acende o LED

apagar: apaga o LED

piscar: pisca o LED em um determinado intervalo

Classe LED

LED
pino : int
estado : int
acender() : void
apagar() : void
piscar(intervalo: int) : void

Representando em código-fonte

Para esta tarefa, vamos "traduzir" nossa classe LED em código-fonte, criando todas suas características (atributos) e ações (métodos)

```
class Led{
public:
    int pino;
```


01/09/2022 17:20

comfapoo/workshop_parte1.md at main · humbertozanetti/comfapoo

```

int estado;

public:
void acender(){
    digitalWrite(this->pino, HIGH);
}

void apagar(){
    digitalWrite(this->pino, LOW);
}

void piscar(int intervalo){
    acender();
    delay(intervalo);
    apagar();
    delay(intervalo);
}
};

```

Por enquanto, não iremos discutir o modo de acesso `public` presente no código (será discutido na Parte 2!). Usando a declaração `this` e o operador seta (`->`) para diferenciar nossos **atributos** das variáveis que são utilizadas dentro de nossos métodos. Uma classe descreve um objeto, mas para criar esse objeto, e essa criação chamamos de **instância**. Para criar ou "construir" esse objeto precisamos de um outro método, um método especial denominado **construtor**. Para construir um objeto LED temos que pensar o que seria interessante definir as características (**atributos**) principais do objeto real LED: qual é o pino em que ele está conectado? Esse método construtor, por regra da própria linguagem, deve ter o nome da classe e receber os parâmetros iniciais da sua criação (no caso, o pino o qual está conectado). Vamos também alterar os métodos `acender()` e `apagar()` para que eles alterem o atributo `estado`.

```

class Led{
public:
    int pino;
    int estado;

public:
    Led(int pino){
        this->pino = pino;
        pinMode(this->pino, OUTPUT);
    }
    void acender(){
        digitalWrite(this->pino, HIGH);
        this->estado = 1; // equivale a HIGH
    }
    void apagar(){
        digitalWrite(this->pino, LOW);
        this->estado = 0; // equivale a LOW
    }
};

```


01/09/2022 17:20

comfapoo/workshop_parte1.md at main · humbertozanetti/comfapoo

```
    }  
    void piscar(int intervalo){  
        acender();  
        delay(intervalo);  
        apagar();  
        delay(intervalo);  
    }  
};
```

Vamos agora criar e usar o objeto LED, que está conectado ao pino 13 e começara apagado. Também aproveitamos para usar o método `acender()`, para fazê-lo acender. Ainda não usaremos a função obrigatória `setup()` do Arduino, apenas a `loop()` a qual se localiza nossa lógica e o ciclo de repetição constante. Nossa configuração inicial do LED está inserido em nosso método construtor.

```
Led led = Led(13);  
  
void setup() {  
}  
  
void loop() {  
    led.acender();  
}
```

Depois podemos fazer o tradicional "Hello World!" do Arduino: fazer uma LED piscar, com o intervalo de 1 segundo (1000 ms)

```
void loop() {  
    led.acender();  
    delay(1000);  
    led.apagar();  
    delay(1000);  
}
```

Ou até mesmo, podemos usar o método `piscar()`, passando o intervalo desejado para o pisca-pisca.

```
void loop() {  
    led.piscar(1000);  
}
```

01/09/2022 17:20

comfapoo/workshop_parte1.md at main · humberiozanetti/comfapoo

Podemos notar que acessamos todas as características do objeto através de suas ações ou métodos. Também podemos acessar suas propriedades ou atributos, como no exemplo abaixo, que podemos ver o estado do LED no *Monitor Serial* do Arduino, sendo 0 (equivalente a LOW ou apagado) ou 1 (equivalente a HIGH ou aceso):

```
Led led = Led(13);

void setup() {
  Serial.begin(9600);
}

void loop() {
  led.acender();
  Serial.print("Estado do LED:");
  Serial.println(led.estado);
  delay(1000);
  led.apagar();
  Serial.print("Estado do LED:");
  Serial.println(led.estado);
  delay(1000);
}
```

Portanto, acessamos todas as características de um objeto por meios de seus atributos e métodos.

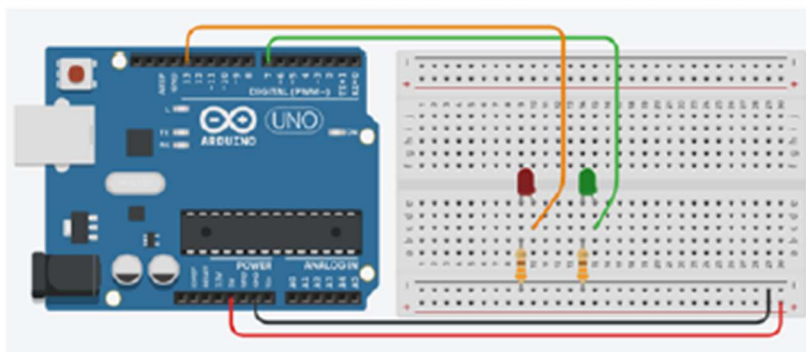


Tarefa 2

Para esta tarefa, vamos ver uma das maiores vantagens da Programação Orientada a Objetos, que é o reuso de código da classe para criar mais de um objeto. Vamos adicionar mais um LED (verde) e vamos criar um objeto utilizando a classe existente. Vamos fazer uma pisca-pisca alternando entre os 2 LEDs. O LED vermelho continua no pino 13 e irá iniciar aceso, enquanto o LED verde está conectado ao pino 7 e se iniciará apagado, e depois alternamos entre eles.

01/09/2022 17:20

comfapoo/workshop_parte1.md at main · humbertozanetti/comfapoo



Circuito da

Tarefa 2

Lista de componentes para montagem

- 2 LEDs (1 vermelho e 1 verde)
- 2 resistores de 330 ohms
- 4 cabos de conexão

Código-fonte

```
Led ledVermelho = Led(13);  
Led ledVerde = Led(7);  
  
void setup() {  
}  
  
void loop()  
{  
  ledVermelho.acender();  
  ledVerde.apagar();  
  delay(1000);  
  ledVermelho.apagar();  
  ledVerde.acender();  
  delay(1000);  
}
```

Parte 2 - Encapsulamento

01/09/2022 17:20

comfapoo/workshop_parte2.md at main · humbertozanetti/comfapoo

 humbertozanetti / comfapoo Public[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#) main ▾

...

[comfapoo](#) / [workshop_parte2.md](#)

humbertoanetti Update workshop_parte2.md

 1 contributor 204 lines (149 sloc) | 6.15 KB

...

WORKSHOP PARTE 2

Encapsulamento

Objetivos da prática

- Entender melhor o conceito de encapsulamento.
- Compreender os modificadores público e privado.
- Aprender sobre os *tipos de acesso* à atributos e métodos dos objetos.

Tarefa 1 - Melhorando nossa classe LED

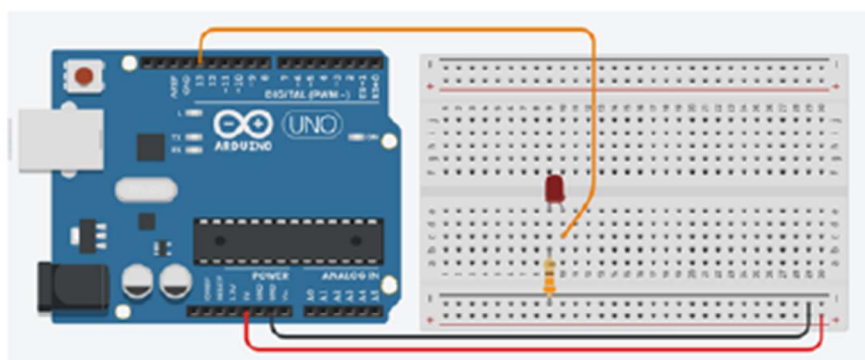
Nesta tarefa vamos melhorar a nossa classe LED usando o conceito de Encapsulamento. Esse conceito corresponde a "esconder" o que é essencial do objeto, permitindo que outros objetos interajam através de "interfaces" (através de métodos). Simplificando: os objetos possuem internamente os dados que podem sofrer alterações (no caso do LED, o estado seria um exemplo) mas não devem ser acessados por outros objetos ou demais trechos de código. Quem tem que definir essa alteração é o próprio objeto, é a **responsabilidade dele!** Através de métodos podemos dar acesso a alterações ou verificar os estados desses dados. Nesse momento vamos criar uma nova classe LED, com os métodos `acender()` e `apagar()`, mas acrescentando mais 2:

01/09/2022 17:20

comfapoo/workshop_parte2.md at main · humbertozanetti/comfapoo

- `trocaEstado` : esse método será o acesso para que seja trocado o estado do LED, de aceso para apagado, ou vice-versa.
- `mostraEstado` : esse método irá mostrar qual é o estado do LED (aceso ou apagado)

Circuito para a tarefa



Lista de componentes para montagem

- 1 LED vermelho
- 1 resistor de 330 ohms
- 3 cabos de conexão

Objeto LED

ESTADOS aceso apagado	pino	Propriedades: pino: pino ligado ao Arduino estado: aceso ou apagado Ações: acender: acende o LED apagar: apaga o LED trocaEstado: muda o estado do LED mostraEstado: informe qual o estado atual
---	-------------	--

Acesso aos atributos e métodos

Precisamos definir o que será "encapsulado", "escondido" e o que será acessível:

- Escondido:
 - os atributos `pino` e `estado`, pois só podem ser alterados pelo próprio objeto
 - os métodos `acender()` e `apagar()`, pois agora eles irão alterar o atributo `estado`
- Acessível:

01/09/2022 17:20

comfapoo/workshop_parte2.md at main · humbertozanetti/comfapoo

- o o método construtor, para que se possa criar o objeto
- o os métodos `trocaEstado()` e `mostraEstado()`, que serão como vamos pedir para mudar o estado do LED e como vemos se seu estado mudou. Com isso, podemos definir seus modos de acessos. O que será escondido será demarcado como *privado* (*private*) e o que será acessível será demarcado como *público* (*public*).

Classe LED

Nosso novo diagrama da Classe LED, com as marcações privado (-) e público (+).

LED
-pino : int -estado : int
-acender() : void -apagar() : void +trocaEstado() : void +mostraEstado() : int

Representando em código-fonte

Para essa nova classe, nosso método construtor também iremos indicar qual é o nosso estado inicial do LED.

```
class Led{
    int pino;
    int estado;

    public:
    Led(int pino, int estado){
        this->pino = pino;
        this->estado = estado;
        pinMode(this->pino, OUTPUT);
        digitalWrite(this->pino, this->estado);
    }

    void trocaEstado(){
        if(this->estado == LOW)
            acender();
        else
            apagar();
    }

    int mostraEstado(){
        return this->estado;
    }

    private:
    void acender(){
        digitalWrite(this->pino, HIGH);
```

01/09/2022 17:20

comfapoo/workshop_parte2.md at main · humbertozanetti/comfapoo

```

        this->estado = HIGH;
    }

    void apagar(){
        digitalWrite(this->pino, LOW);
        this->estado = LOW;
    }

};

```

Podemos notar que os atributos são por padrão definidos como **privado** (*private*), mas os métodos precisam da demarcação **private**. A partir da demarcação **public**: podemos ver nossos métodos **públicos** (*public*). Chamaos esses coqueiros de acesso de **visibilidade**. Para comprovar que não acessamos o que é privado, vamos tentar criar um objeto LED apagado e tentar acender através do acesso direto ao método `acender()`.

```

Led led = Led(13,LOW);

void setup()
{
}

void loop()
{
    led.acender();
}

```

Esse trecho de código trará um erro de acesso.

Vamos agora fazer um pisca-pisca usando o método `trocaEstado()` e mostrar no "monitor Serial" (interface para mostrar mensagem do Arduino) o estado do LED, com o método `mostraEstado()`.

```

Led led = Led(13,LOW);

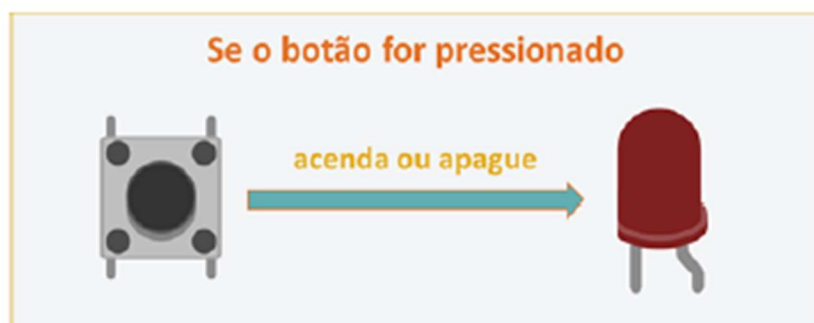
void setup()
{
    Serial.begin(9600);
}

void loop()
{
    led.trocaEstado();
    delay(1000);
    Serial.print("Estado do LED: ");
    Serial.println(led.mostraEstado());
}

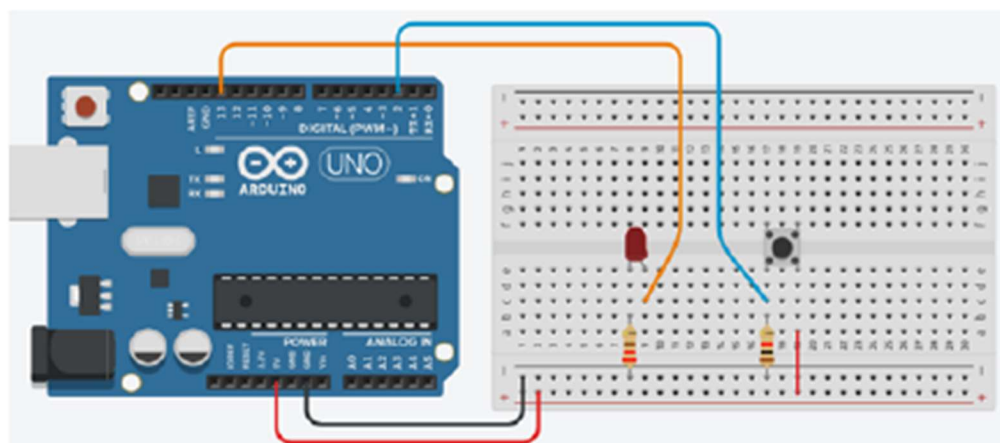
```

Tarefa 2 - Interação com botão

O encapsulamento tem uma grande vantagem que delegar as responsabilidades de cada objeto, deixando o sistema mais modular e independente dos demais objetos (desacoplamento). Para que outro objeto possa interagir com outro, apenas usaremos esses métodos "interfaces", ou seja, que provêm acesso a informações e dados. Nessa tarefa, usando nossa classe LED, vamos construir um sistema onde temos um botão (*push button*) que aciona o acender ou apagar de um LED, através de um método que verifica se o botão está ou não pressionado (`verificaEstado()`). Como nosso LED pode iniciar aceso ou apagado, pois este não tem acesso aos métodos `acender()` e `apagar()` . Nesse cenário, o LED tem a responsabilidade de verificar seu estado e decidir se acende ou apaga, enquanto o botão apenas de verificar se houve ou não o pressionar do botão.



Circuito da Tarefa



Lista de componentes para montagem

- 1 LED
- 1 resistor de 330 ohms
- 1 resistor de 1 kohms

01/09/2022 17:20

comfapoo/workshop_parte2.md at main · humbertozanetti/comfapoo

- 1 push button
- 5 cabos de conexão

Objeto Botão



Propriedades:

pino: pino ligado ao Arduino

estado: pressionado ou não

Ações:

verificaEstado: verifica o estado do botão

Classe Botão

BOTÃO
-pino : int -estado : int
+verificaEstado() : int

Código-fonte

Abaixo temos a representação em código da classe Botão.

```
class Botao{
    int pinoBotao;
    int estado;

    public:
    Botao(int pino){
        pinoBotao = pino;
        pinMode(pinoBotao, INPUT);
    }

    int verificaEstado(){
        estado = digitalRead(pinoBotao);
        delay(10);
        return estado;
    }

};
```

01/09/2022 17:20

comfapoo/workshop_parte2.md at main · humbertozeiteil/comfapoo

E depois, podemos utilizar os objetos LED e Botão para fazer uma interação de acender e apagar.

```
Led led = Led(13,LOW);
Botao botao = Botao(2);

void setup() {
  Serial.begin(9600);
}

void loop() {
  int est;
  est = botao.verificaEstado();
  if (est == HIGH){
    led.trocaEstado();
  }
  Serial.print("Estado do LED: ");
  Serial.println(led.mostraEstado());
  delay(100);
}
```

Parte 3 - [Herança](#)

01/09/2022 17:21

comfapoo/workshop_parte3.md at main · humbertoanetti/comfapoo

humbertoanetti / comfapoo Public

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

main ▾

...

comfapoo / workshop_parte3.md



humbertoanetti Update workshop_parte3.md



1 contributor

249 lines (179 sloc) | 7.29 KB

...

WORKSHOP PARTE 3

Herança

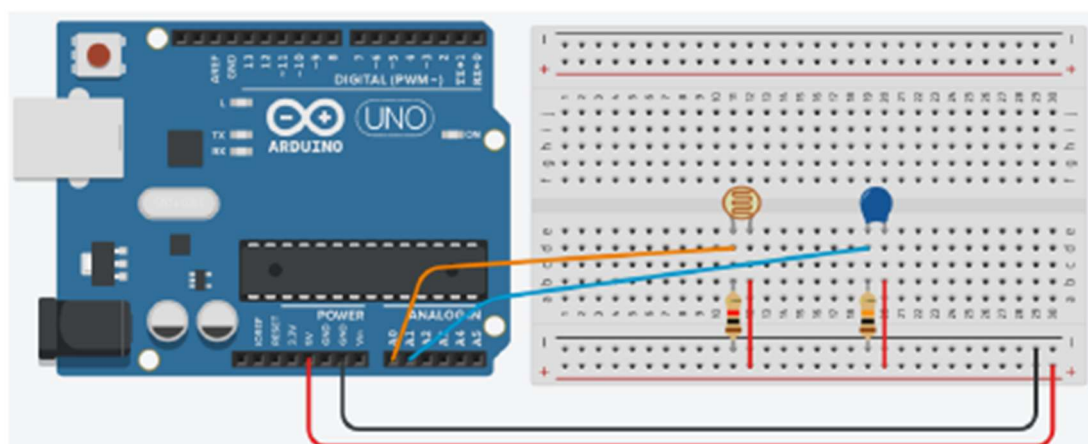
Objetivos da prática

- Entender o conceito de herança.
- Ampliar os conhecimentos sobre o reuso de código.
- Aprender sobre como estruturar um sistema orientado a objetos.

Tarefa 1 - Entendendo como funciona um sensor de luminosidade

Nesta tarefa vamos definir quais são as características principais de um sensor. Sensores são dispositivos que percebem alterações ambientes, e como resultado, nos fornece um valor correspondente ao fenômeno ambiental o qual ele está "sentindo". Por exemplo, um sensor de luminosidade, como o LDR (*Light Dependent Resistor*), nos fornecerá um valor que varia conforme a incidência de luz sobre ele. Em Arduino, esse valor pode variar de um intervalo de 0 a 1023. Esse valor refere-se desde a ausência de luz (0) até a maior concentração de luz que o sensor consegue captar (1023). Esse valor não é nenhuma medida em específico, mas sim um valor que representa a variação de resistência do sistema. Para que esse valor seja uma medida de luz (lumens ou lux), temos que fazer uma conversão.

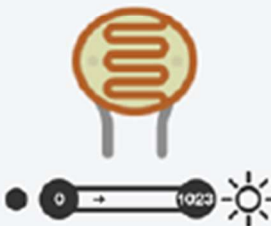
Circuito para a tarefa



Lista de componentes para montagem

- 1 LDR (sensor de luminosidade)
- 1 termistor NTC (sensor de temperatura)
- 1 resistor de 1 kohms
- 1 resistor de 10 koms
- 6 cabos de conexão

Objeto LDR



Propriedades:

pino: pino ligado ao Arduino

valor: valor gerado pelo sensor (0 - 1023)

lux: valor convertido em lumens

Ações:

leValor: faz a leitura do valor

mostraLux: mostra o valor lido em lumens

Classe LED

LDR
+pino : int
+valor : int
-lux : int
+leValor() : int
+mostraLux() : float

Representando em código-fonte

```

class LDR{
  int lux;
  int valor;
  int pino;

  public:
  LDR(int pino){
    this->pino = pino;
  }

  int leValor(){
    this->valor = analogRead(this->pino);
    return this->valor;
  }

  float mostraLux (){
    float voltagem = leValor()*0.0048828125;
    this-> lux = 500 /(10*((5 - voltagem)/voltagem));
    return this->lux;
  }
};

```

Objeto Termistor



Propriedades:

pino: pino ligado ao Arduino

valor: valor gerado pelo sensor (0 - 1023)

tempK: valor convertido em Kelvin

tempC: valor convertido em Celsius

Ações:

leValor: faz a leitura do valor

mostraKelvin: mostra o valor convertido Kelvin

mostraCelsius: mostra o valor convertido em Celsius



Classe Termistor

Termistor
+pino : int
+valor : int
-tempK: float
-tempC : float
+leValor() : int
+mostraKelvin() : float
+mostraCelsius() : float

Representando em código-fonte

```
class Termistor{
  int valor;
  int pino;
  float tempK;
  float tempC;

  public:
  Termistor(int pino){
    this->pino = pino;
  }

  int leValor(){
    this->valor = analogRead(this->pino);
    return this->valor;
  }

  float mostraKelvin(){
    this->tempK = log(10000.0 * (1024.0 / leValor() - 1));
    this->tempK = 1 / (0.001129148 + (0.000234125 + (0.0000000876741 * this->tempK) * this->tempK)) * this->tempK;
    return this->tempK;
  }

  float mostraCelsius() {
    this->tempC = mostraKelvin() - 273.15;
    return this->tempC;
  }
};
```

Podemo fazer um teste rápido com essas classes.

```
Termistor termistor = Termistor(A1);
LDR ldr = LDR(A0);

void setup(){
  Serial.begin(9600);
}

void loop(){
  Serial.print("Temperatura em Celsius: ");
  Serial.println(termistor.mostraCelsius());
  Serial.print("Temperatura em Kelvin: ");
  Serial.println(termistor.mostraKelvin());
  Serial.print("Luminosidade em lux: ");
  Serial.println(ldr.mostraLux());
  delay(1000);
}
```


Similaridades entre os objetos

É possível perceber que ambos os objetos compartilham características semelhantes, tanto nos atributos quanto nos métodos. Inclusive, se repararmos, até suas montagens na protoboard são semelhantes! Quando temos classes com esse comportamento, podemos utilizar o conceito de herança. Podemos perceber que há similaridades entre os sensores. Isso é comum entre componentes sensores, e podemos otimizar nosso código nos utilizando dessas características similares. Quando temos objetos que possuem atributos e métodos similares, podemos criar uma classe que tenha essas similaridades e, a partir dela, criar classes mais especializadas.

Tarefa 2 - Separando as características e usando herança

A herança é um recurso que permite que as características comuns a mais de uma classe podem ser colocadas em uma classe base. Também podemos chamar de uma classe "mãe" e suas classes "filhas", reforçando o conceito de herança. Cada classe derivada ou subclasse terá toda a estrutura da classe base e mais os novos recursos que eles precisam suas funções particulares. As classes LDR e Termistor tem várias características similares, como vemos abaixo:

LDR
+pino : int +valor : int -lux : int
+leValor() : int +mostraLux() : float

Termistor
+pino : int +valor : int -tempK: float -tempC : float
+leValor() : int +mostraKelvin() : float +mostraCelsius() : float

Superclasse Sensor

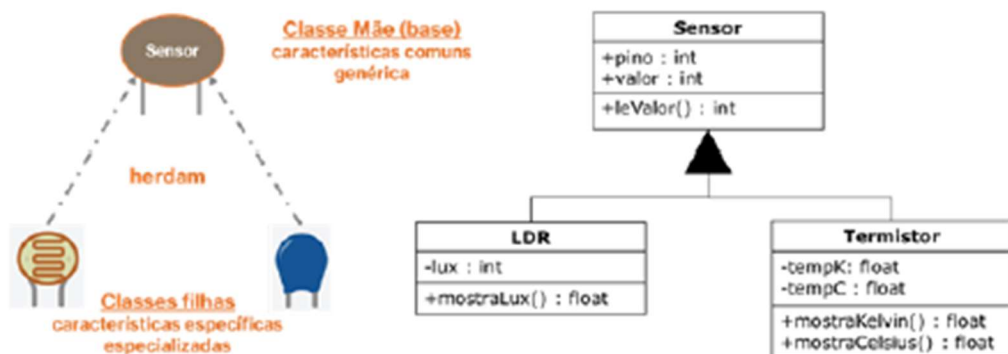
Nesse caso, podemos criar uma classe Sensor, que possui todas as características em comuns, e criar classes complementares, que só possua as especialidades (atributos e métodos especializados). Com isso, podemos definir que as classes LDR e Termistor poderiam ser essas classes especializadas. A partir dessas semelhanças, podemos criar uma classe base++, com essas características comuns. Essa classe "Sensor" genérico, como vemos abaixo:

Sensor
+pino : int +valor : int
+leValor() : int

01/09/2022 17:21

comfapoo/workshop_parte3.md at main · humbertoazane/comfapoo

Com isso, as classes LDR e Termistor podem herdar dessa classe Sensor essas características comuns a elas, e definindo apenas suas especialidades. Essa organização com as novas classes seria:



Representando em código-fonte a classe Sensor

```
class Sensor{
    public:
        int valor;
        int pino;

    public:
        int leValor(){
            this->valor = analogRead(this->pino);
            return this->valor;
        }
};
```

Representando em código-fonte a classe LDR

```
class LDR: public Sensor{
    int lux;

    public:
        LDR(int pino){
            this->pino = pino;
        }

        float mostraLux (){
            float voltagem = leValor()*0.0048828125;
            this-> lux = 500 /(10*((5 - voltagem)/voltagem));
            return this->lux;
        }
};
```


Representando em código-fonte a classe Termistor

```
class Termistor: public Sensor{
    float tempK;
    float tempC;

    public:
    Termistor(int pino){
        this->pino = pino;
    }
    float mostraKelvin(){
        this->tempK = log(10000.0 * (1024.0 / leValor() - 1));
        this->tempK = 1 / (0.001129148 + (0.000234125 + (0.0000000876741 * this->tempK * this->tempK)) * this->tempK);
        return this->tempK;
    }

    float mostraCelsius() {
        this->tempC = this->tempK - 273.15;
        return this->tempC;
    }
};
```

Na construção da classe `Sensor`, vemos que não há um construtor, pois não há necessidade, já que vamos utilizar as classes mais especializadas (as derivadas). Também notamos que na definição na classe, logo a frente de seu nome, utilizamos : (dois pontos), que indica de quem a classe irá herdar as características, como vemos em:

```
class LDR: public Sensor{
```

Agora, podemos testar a criação e uso dessas classes derivadas usando o mecanismo de herança.

```
LDR ldr = LDR(A0);
Termistor termistor = Termistor(A1);

void setup(){
    Serial.begin(9600);
}

void loop(){
    Serial.print("Valor do LDR: ");
    Serial.println(ldr.leValor());
    Serial.print("Lux: ");
    Serial.println(ldr.mostraLux());
    Serial.print("Valor do Termistor: ");
    Serial.println(termistor.leValor());
```

01/09/2022 17:21

comfapoo/workshop_parte3.md at main · humberzanetti/comfapoo

```
Serial.print("Kelvin: ");  
Serial.println(termistor.mostraKelvin());  
Serial.print("Celsius: ");  
Serial.println(termistor.mostraCelsius());  
delay(1000);  
}
```

Parte 4 - Polimorfismo

01/09/2022 17:21

comfapoo/workshop_parte4.md at main · humbertoanetti/comfapoo

 humbertoanetti / comfapoo Public[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#) main ▾

...

[comfapoo](#) / [workshop_parte4.md](#)

humbertoanetti Update workshop_parte4.md

 1 contributor 233 lines (170 sloc) | 6.52 KB

...

WORKSHOP PARTE 4

Polimorfismo

Objetivos da prática

- Entender o conceito de **polimorfismo**.
- Aplicar o polimorfismo em um contexto prático.

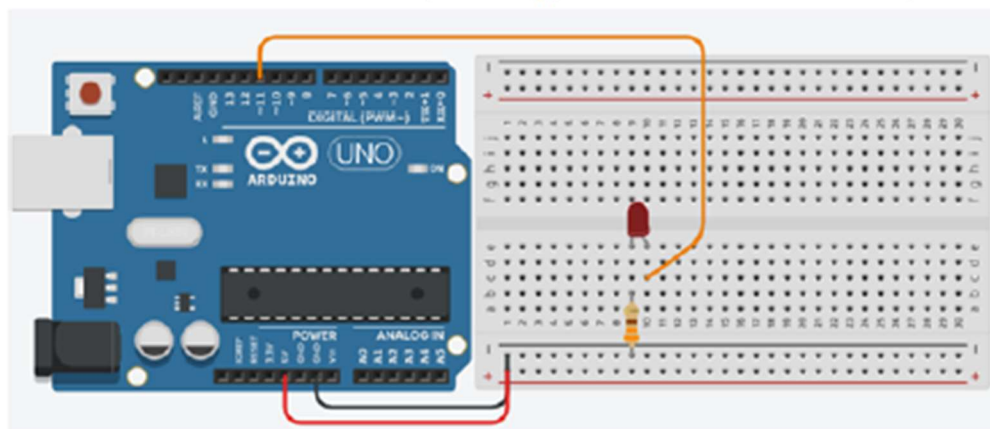
Tarefa 1 - Revendo o conceito de LED

Nessa tarefa vamos ver como podemos programar e utilizar objetos que podem se comportar de formas diferentes. O conceito de **Polimorfismo** (*poli* = muitas, *morfo* = forma). Essa é umas das características mais poderosas da Programação Orientada a Objetos, pois podemos através de um mesmo objeto usar seus métodos de formas diferentes, dependendo do contexto. Para isso, vamos voltar a usar o conceito de **classe base** (visto em Herança). No Polimorfismo, duas ou mais classes derivadas de uma superclasse podem usar métodos de **mesma assinatura** (*mesmo nome e parâmetros*), mas que podem se comportar de maneiras diferentes ou formas diferentes.

Circuito para a tarefa

01/09/2022 17:21

comfapoo/workshop_parte4.md at main · humbertoanetili/comfapoo



Lista de componentes para montagem

- 1 LED
- 1 resistor de 330 ohms
- 3 cabos de conexão

Objeto LED

Para essa tarefa, vamos utilizar um objeto LED mais simples.



Propriedades:

pino: pino ligado ao Arduino
estado: aceso ou apagado

Ações:

acender: acende o LED
apagar: apaga o LED

Classe LED

Led
+pino : int
+estado : int
+acender() : void
+apagar() : void

Representando em código-fonte

```
class Led{
  public:
  int pino;
  int estado;

  public:
  Led(int pino){
    this->pino = pino;
    pinMode(this->pino, OUTPUT);
  }

  void acender(){
    digitalWrite(this->pino,HIGH);
  }

  void apagar(){
    digitalWrite(this->pino,LOW);
  }
};
```

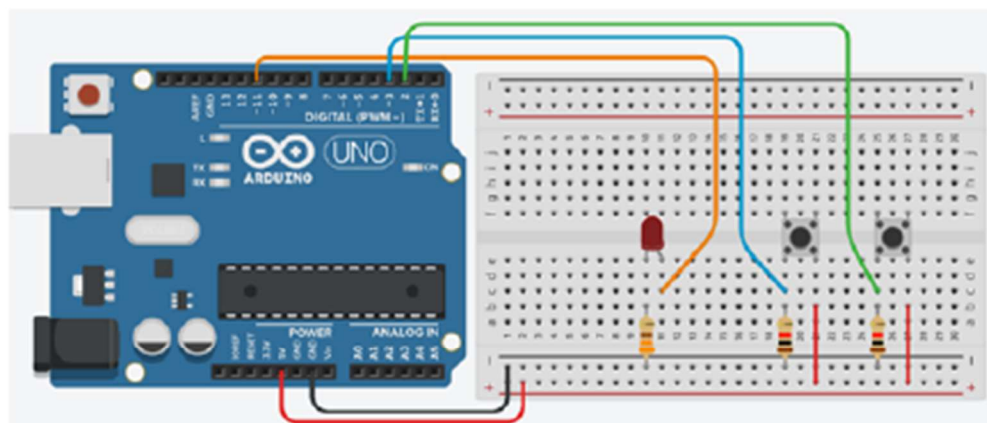
A partir dessa classe LED simples, que possui o método `acender()` como a ação de acender totalmente o LED, vamos imaginar que ele não poderia ser aplicada diretamente para uma aplicação de pisca-pisca ou de um sinal oscilante (com a iluminação aumentando gradativamente), comum em sistema de sinalização de emergência.

Com isso, vamos implementar 2 formas novas de acender o LED: piscando e oscilando.

Tarefa 2 - Criando uma classe base e suas novas formas

Nesta tarefa, vamos criar 2 novas formas de acender um LED

Circuito para a tarefa



Lista de componentes para montagem

- 1 LED
- 2 push button
- 1 resistor de 330 ohms
- 2 resistores 2 kohms
- 8 cabos de conexão

Para criar nossa classe base, devemos sinalizar qual ou quais métodos terão comportamentos diferentes. Esse método será um **método virtual**, ou seja, na verdade será uma regra na classe base e terá sua implementação, um diferente da outra, nas classes derivadas. Como não usaremos essa classe para um objeto, ela não terá um construtor. Para isso, colocamos o termo **virtual** antes da definição do método. Com isso, esse método poderá ser **sobrescrito** nas classes derivadas, ou seja, poderá assumir formas diferentes de implementação.

Representando em código-fonte

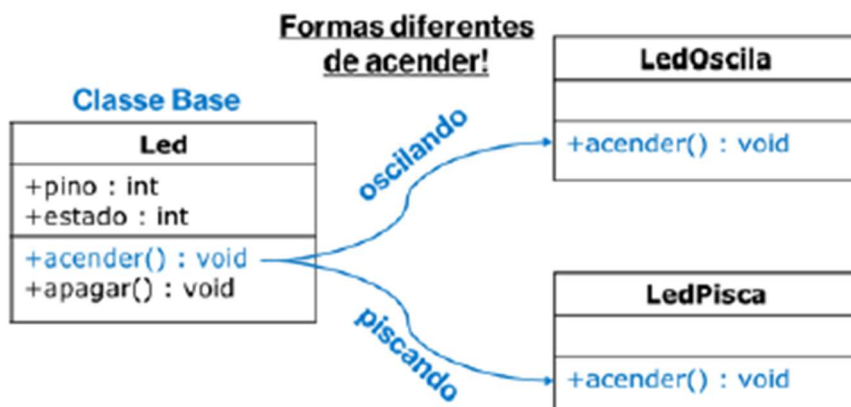
```
class Led{
    public:
    int pino;
    int estado;

    public:
    virtual void acender(){
        digitalWrite(this->pino,HIGH);
    }

    void apagar(){
        digitalWrite(this->pino,LOW);
    }
};
```

Vemos que o método `acender()` se mantém igual, apenas com precedência do termo **virtual**, indicando que esse pode ser um método que poderá ser sobrescrito nas classes derivadas. Se a classe derivada não sobrescrever o método, prevalecerá o que está definido na classe base. O método `apagar()` e os atributos serão herdados integralmente e sem mudança pelas subclasses. Vamos criar 2 classes derivadas para redefinir o método `acender()` em cada uma delas. Em uma teremos os métodos `acender` de forma piscante, na outra de forma oscilante.

Classe LED base e suas derivadas



Representando em código-fonte (classe LedOscila)

```
class LedOscila: public Led{

    public:
    LedOscila(int pino){
        this->pino = pino;
        pinMode(this->pino, OUTPUT);
    }

    void acender(){
        int lum;
        for(lum = 0; lum < 255; lum += 5){
            analogWrite(this->pino, lum);
            delay(50);
        }
        lum = 0;
        this->apagar();
    }
};
```

Representando em código-fonte (classe LedPisca)

```
class LedPisca: public Led{

    public:
    LedPisca(int pino){
        this->pino = pino;
        pinMode(this->pino, OUTPUT);
    }

    void acender(){
        digitalWrite(this->pino,HIGH);
        delay(100);
        digitalWrite(this->pino,LOW);
    }
};
```

01/09/2022 17:21

comfapoo/workshop_parte4.md at main · humbertoanetli/comfapoo

```
    delay(100);  
  }  
};
```

Vamos agora adicionar a classe Botão, já conhecida por nós, para fazermos alguns testes.

```
class Botao{  
  int pinoBotao;  
  int estado;  
  
  public:  
  Botao(int pino){  
    pinoBotao = pino;  
    pinMode(pinoBotao, INPUT);  
  }  
  
  int verificaEstado(){  
    estado = digitalRead(pinoBotao);  
    delay(10);  
    return estado;  
  }  
  
};  
  
Led *led;  
Botao botao1 = Botao(2);  
Botao botao2 = Botao(3);  
  
void setup()  
{  
  Serial.begin(9600);  
}  
  
void loop()  
{  
  if(botao1.verificaEstado() == HIGH){  
    led = &LedPisca(11);  
    led->acender();  
  }else if(botao2.verificaEstado() == HIGH){  
    led = &LedOscila(11);  
    led->acender();  
  }  
}
```


01/09/2022 17:21

comfapoo/workshop_parte4.md at main · humbertozanetti/comfapoo

Vemos que ao criar o objeto `led`, há um asterisco (*). Isso é um recurso da linguagem do Arduino para que esse objeto não seja construído no momento, mas que possa ser construído posteriormente a partir de uma de suas classes derivadas, e assim assumir seu comportamento (forma). Usamos o `&` logo a frente do construtor `LedPisca` e `LedOscila`, para a referência do objeto `led` conseguir assumir sua forma. Como fazemos isso através de referência, acessamos o método através do conhecido operador seta (`->`). Com isso, podemos dizer que o objeto `led` pode assumir várias formas (polimorfismo). Para provar que a forma original do método `acender()` ainda pode ser usada, pode-se, por exemplo, não sobrescrever esse método em uma de suas derivadas e usá-lo. Abaixo um exemplo de exclusão do método na classe `LedPisca`, fazendo que o método `acender()` seja o original da classe base.

```
...
class LedPisca: public Led{
public:
    LedPisca(int pino){
        this->pino = pino;
        pinMode(this->pino, OUTPUT);
    }
};

...

void loop()
{
    if(botao1.verificaEstado() == HIGH){
        led = &LedPisca(11);
        led->acender();
    }else if(botao2.verificaEstado() == HIGH){
        led = &LedOscila(11);
        led->acender();
    }
}
```

Parte 5 - Sistema orientado a objetos

01/09/2022 17:21

comfapoo/workshop_parte5.md at main · humbertoanetti/comfapoo

 humbertoanetti / comfapoo Public[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#) main ▾

...

[comfapoo](#) / [workshop_parte5.md](#)

humbertoanetti Update workshop_parte5.md

 1 contributor 168 lines (128 sloc) | 4.99 KB

...

WORKSHOP PARTE 5

Sistema orientado a objetos

Objetivos da prática

- Entender o relacionamento e troca de mensagens entre objetos.
- Compreender o uso de objetos em um sistema organizado por objetos.

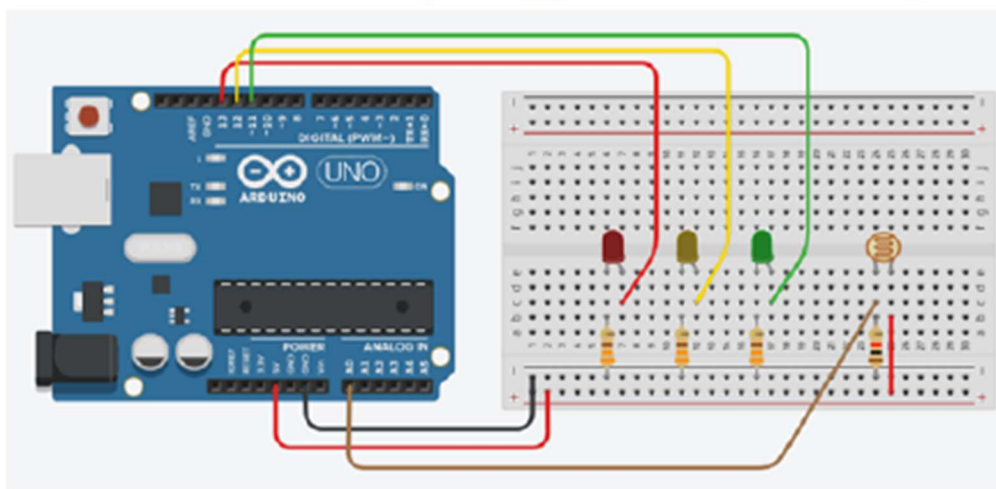
Tarefa 1 - Usando vários objetos

Nesta tarefa vamos utilizar um conjunto de objetos, a fim de criar um sistema mais organizado e utilizando o conceito de troca de mensagens entre esses objetos. Até o momento, mesmo utilizando algumas vezes mais de um objeto, ainda não vimos a organização básica de um sistema orientado a objetos. Vamos criar um pequeno sistema que gerencia um semáforo, que além de suas luzes, possui um sensor de luminosidade para controlar o seu funcionamento durante o dia e a noite.

Circuito para a tarefa

01/09/2022 17:21

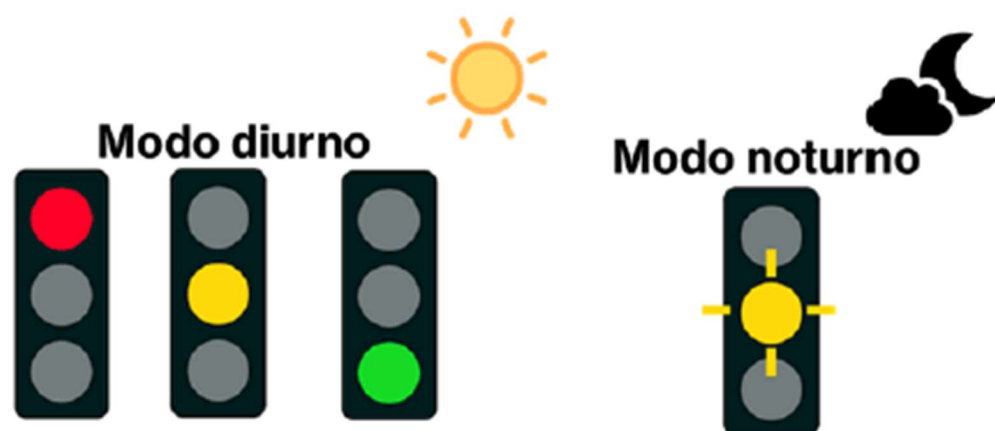
comfapoo/workshop_parte5.md at main · humbertoanettil/comfapoo



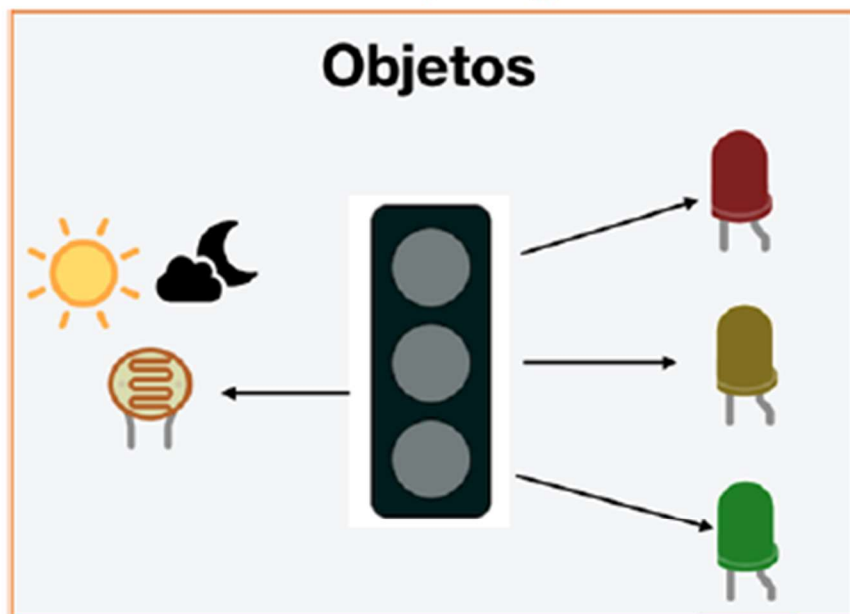
Lista de componentes para montagem

- 3 LEDs
- 1 LDR
- 3 resistores de 330 ohms
- 1 resistor 1 kohms
- 7 cabos de conexão

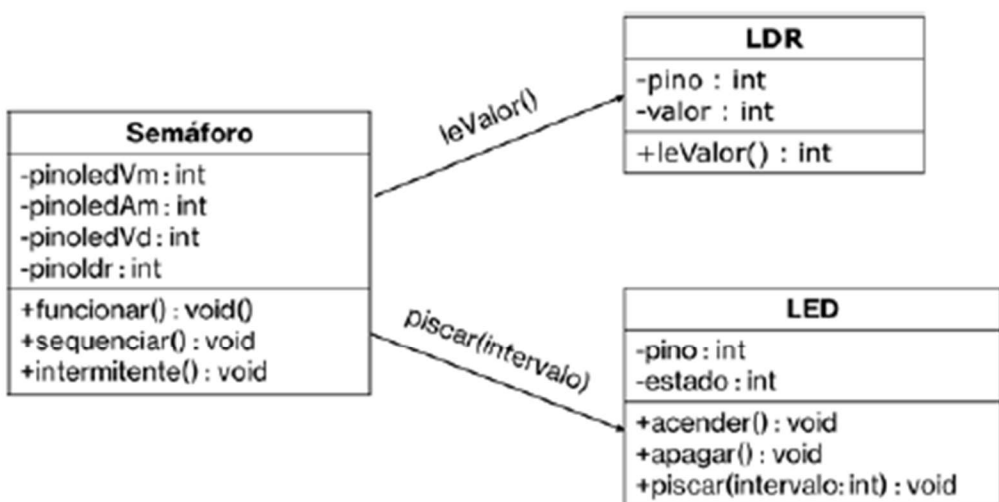
Nosso objeto Semáforo será um objeto que irá interagir com um grupo de 3 objetos LEDs e 1 objeto LDR. A construção do nosso semáforo será por meio do objeto objeto Semáforo, que irá instanciar 3 objetos Leds ou apenas 1 objeto LED, dependendo de seu modo de uso. Sempre será utilizado um objeto LDR, para verificar a luminosidade, determinando se é dia ou noite. No **modo diurno**, o semáforo terá a sequência convencional de um semáforo. Já no **modo noturno**, teremos apenas o LED amarelo piscando, indicando um alerta de proximidade de cruzamento.



Objetos envolvidos



Classes do Semáforo



Em um sistema orientado a objetos esses objetos interagem entre si, através de *troca de mensagens*. Uma troca de mensagem, na prática, indica a invocação de um método de um objeto por outro objeto. Acima vemos que um objeto Semáforo irá chamar o método `leValor()` da classe LDR e o `piscar()` da classe LED, por exemplo. Para fazer essa invocação ou chamada, temos que criar um objeto, e através dele utilizar o método e, algumas vezes, enviar também alguma informação (parâmetro para o método), como no método `piscar()`.

led.piscar(500)

objeto mensagem informação

Tarefa 2 - Utilizando o sistema de semáforo

Nesta tarefa vamos criar as classes e ver como o objeto Semáforo interage com os demais objetos. Vemos que na classe Semáforo encontramos a utilização de objetos LED e LDR, e consequentemente o uso de seus métodos. Por isso, colocaremos primeiro as essas classes

Representando em código-fonte (classe LED e LDR)

```
class Led{
    int pino;
    int estado;

    public:
    Led(int pino){
        this->pino = pino;
        pinMode(this->pino, OUTPUT);
    }
    void acender(){
        digitalWrite(this->pino, HIGH);
    }
    void apagar(){
        digitalWrite(this->pino, LOW);
    }
    void piscar(int intervalo){
        acender();
        delay(intervalo);
        apagar();
        delay(intervalo);
    }
};

class LDR{
    int valor;
    int pino;
    float lux;

    public:
    LDR(int pino){
        this->pino = pino;
    }

    int leValor(){
        this->valor = analogRead(this->pino);
        return this->valor;
    }
};
```

Agora, podemos criar a classe Semáforo com a inclusão das classes LED e LDR.

Representando em código-fonte (Semáforo)

```
class Semaforo{
    int pinoledVm;
    int pinoledAm;
    int pinoledVd;
    int pinoldr;

public:
    Semaforo(int pledVm, int pledAm, int pledVd, int pldr){
        this->pinoledVm = pledVm;
        this->pinoledAm = pledAm;
        this->pinoledVd = pledVd;
        this->pinoldr = pldr;
    }

    void sequenciar(){
        Led ledVm = Led(this->pinoledVm);
        Led ledAm = Led(this->pinoledAm);
        Led ledVd = Led(this->pinoledVd);
        ledVm.acender();
        delay(5000);
        ledVm.apagar();
        ledVd.acender();
        delay(5000);
        ledVd.apagar();
        ledAm.acender();
        delay(1000);
        ledAm.apagar();
    }

    void intermitente(){
        Led ledAm = Led(this->pinoledAm);
        ledAm.piscar(500);
    }

    void funcionar(){
        LDR ldr = LDR(this->pinoldr);
        if(ldr.leValor() > 200){
            this->sequenciar();
        }else{
            this->intermitente();
        }
    }
};
```

01/09/2022 17:21

comfapoo/workshop_parte5.md at main · humbertozanetti/comfapoo

Agora vamos utilizar um objeto Semáforo, que irá controlar too o sistema de semáforo, juntamente com os demais objetos. É interessante notar que apenas utilizamos o método `funcionar()` _ do objeto Semáforo, pois está a lógica que determina o comportamento da sinalização.

Colocando o Semáforo para funcionar

```
Semaforo semaforo = Semaforo(13, 12, 11, A0);

void setup(){
  Serial.begin(9600);
}

void loop(){
  semaforo.funcionar();
  delay(100);
}
```

Questionário para ser respondido depois do Workshop

[Link para o questionário](#)

APÊNDICE B – Planos de aula

Este apêndice apresenta os planos de aula desenvolvidos para serem utilizados como guia de aplicação. Eles foram criados durante o desenvolvimento do material didático que foi disposto aos participantes das oficinas. Esses guias poderão ser utilizados por educadores que queiram utilizar o método ComFAPOO juntamente com o material didático.

Parte 1 – Classe e Objeto

PARTE 1 – Classe e Objeto

ASSUNTO	Abstração, classe e objeto
FORMATO	Individual ou em grupos (2 a 3 alunos)
DURAÇÃO DA ATIVIDADE	45 – 50 minutos
TEMPO DE PREPARAÇÃO	1 hora a 1 hora e 30 minutos de preparação

OBJETIVOS

Este plano de aula serve como guia de aplicação da Parte 1 do Workshop do Método ComFAPOO (<https://github.com/humbertoanetti/comfapoo>, Parte 1 – Classe e Objeto). Os objetivos a serem alcançados desta parte da aplicação são:

- Estimular o exercício de **abstração** a partir de objetos reais.
- Entender o conceito de **classe** como uma identidade que descreve um componente real, o **objeto**.
- Descrever os principais componentes (propriedades e ações) do componente eletrônico, apresentando os conceitos de **atributos** e **métodos**.
- Mostrar a relação entre a classe e o objeto (**Instanciação** e uso de **método construtor**).

MATERIAIS UTILIZADOS

Sala ou laboratório com projetor ou televisão.

Para cada grupos de alunos:

- 1 computador.
- 1 placa Arduino ou similar.

Componentes para os circuitos:

- 1 protoboard.
- 4 cabos de conexão.
- 2 LEDs (1 vermelho e 1 verde).
- 2 resistores de 330 ohms (podem ser de 220 ohms a 470 ohms).

TAREFA 1 - Guia de aplicação

Nesta tarefa o objetivo principal é mostrar como o objeto real LED se comporta, levando o aluno a compreender suas características e ações, para que posteriormente ele associe esses fatores com **atributos** e **métodos** de uma classe.

Comece apresentando a montagem, explicando de maneira bem clara, mas objetiva as ligações da protoboard e como o objeto LED se comporta. Abaixo temos a montagem na Figura 1.

Parte 1 – Classe e Objeto

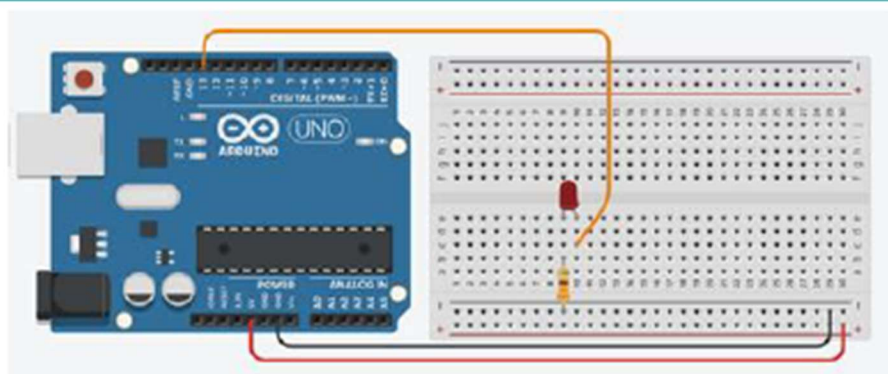


Figura 1 – Montagem (presente no material disponível para o aluno)

O próximo passo é apresentar as características que serão levadas como atributos (pino de conexão com o Arduino e o estado do LED). Para isso, utilize a Figura 2 (presente no material disponível para o aluno como Objeto LED).



Figura 2 – Imagem de apoio sobre o LED (presente no material disponível para o aluno)

Explane sobre as propriedades e ações que o LED tem:

- **Propriedades:** associar com o conceito de **atributos**. Se necessário, associar com o conceito de variáveis;
- **Ações:** associar com o conceito de **métodos**. Se necessários, associar com os conceitos de funções ou modularização.

Como um recurso auxiliar, faça com que o LED acenda e apague. Com isso, já pode-se discutir sobre essas ações do LED, remetendo à métodos. Abaixo segue sugestões de códigos que podem ser executados para acender e apagar o LED (códigos não disponíveis para no material do aluno).

{CÓDIGOS}

Ainda sobre métodos, também apresente a ação de "piscar". Sugestão de código abaixo (códigos não disponíveis para no material do aluno).

{CÓDIGOS}

Com a apresentação das características e ações apresentadas, apresente o diagrama de classe abaixo na Figura 3 (presente no material disponível para o aluno como **Classe LED**), fazendo a relações do que foi visto no objeto real. Também pode dar atenção no método "piscar" e seu parâmetro intervalo.

Parte 1 – Classe e Objeto

LED
pino : int estado : int
acender() : void apagar() : void piscar(intervalo: int) : void

Figura 3 – Classe LED (presente no material disponível para o aluno)

A partir desse momento, será apresentado o código-fonte que representa a classe LED (presente no material disponível para o aluno como **Representando em código-fonte**). Nesse momento, explique a formação da classe em código, apontando como está organizado os atributos e métodos.

Alguns pontos relevantes nesse momento da aplicação e trabalho com os código-fonte presentes no material do aluno (item **Representando em código-fonte**):

- Ainda não é necessário discutir os modos de acesso, como o **public** e **private**;
- Apresentar a declaração **this** e operador seta (**->**), como está descrito no material do aluno;
- Dar destaque a o que é **Instância do objeto**, ajudando o aluno entender a **relação da classe e o objeto**.
- Enfatizar a função do **método construtor** e sua utilização na criação de um objeto.
- A partir do objeto criado, mostrar como o **acesso aos métodos e atributos** ocorrem, como mostra a Figura 4.



Figura 4 – Acesso a atributos e métodos (presente no material disponível para o aluno)

TAREFA 2 – Guia de aplicação

Nesta tarefa o objetivo principal é mostrar umas das vantagens da Programação Orientada a Objetos, que é o reuso de código, através da instância de mais de um objeto por meio de uma única classe.

Apresente a montagem da Tarefa 2, como mostra a Figura 5. Após isso, mostre que, usando a mesma classe, é possível criar um novo objeto (usando o código no item **Código-fonte** da Tarefa 2). Dê destaque que o novo LED adicionado ao circuito será controlado pelo novo objeto em código-fonte.

Parte 1 – Classe e Objeto

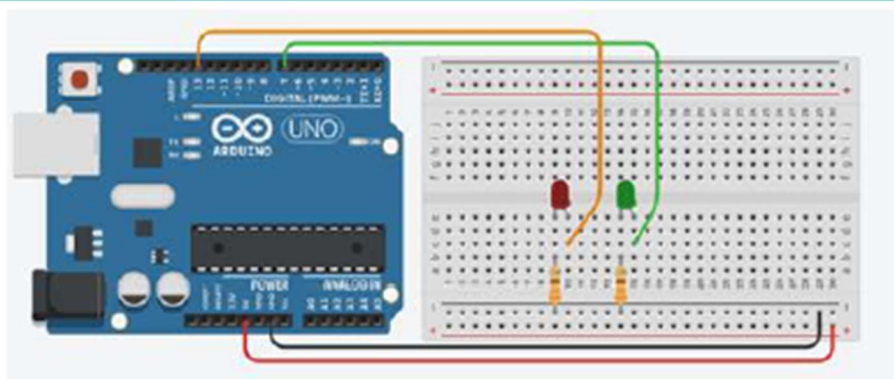


Figura 5 – Montagem (presente no material disponível para o aluno)

DISCUSSÃO SOBRE A APLICAÇÃO

Na Tarefa 1 o foco de aprendizado pode ser separado nos seguintes pontos e pode ser exposto e discutidos com os alunos:

- **Exercício de abstração:** nessa Parte 1 a apresentação do objeto real é o ponto inicial (que se repete em todas as outras partes do Workshop) e serve como recurso para que sejam apresentadas as características do objeto e como o podemos representar em modelo mais abstratos, até o código-fonte. É importante enfatizar a relação de como a classe é definida a partir da análise sobre o objeto real e como o objeto em código se associa com o objeto real.
- **Relação classe-objeto:** deixar claro qual é a função da classe e como, por meio dela, criamos e usamos o objeto. Evidenciar que o mapeamento feito com o objeto real conduz a representação classe (em diagrama e código-fonte), e como a partir dela criamos (a instância) o objeto em "código".
- **Atributos e métodos:** enfatizar como a partir do mapeamento das características do objeto real, chegamos na definição dos atributos e métodos que farão parte da classe.
- **Objeto real x objeto em código:** verificar a relação entre o objeto real e o objeto em código, fazendo com que seja perceptível que ambos são representações em níveis de abstração diferentes.

Na Tarefa 2 o foco da aprendizagem pode ser separado nos seguintes pontos e pode ser exposto e discutidos com os alunos:

- **Fortalecer a relação classe x objeto:** com a inclusão de um outro objeto real e a utilização de outro em código-fonte com a mesma classe, fortalecendo a percepção do aluno sobre a relação entre classe e objeto, assim como instanciação e método construtor.
- **Reuso de código:** mostrar ao aluno umas das vantagens de projeto orientado à objetos, podendo utilizar a mesma solução de software (classe) em um projeto.

RESULTADOS ESPERADOS

O esperado por parte dos alunos ao final da aplicação:

- Identificar a importância da **Abstração** na POO;
- Compreender a **relação entre Classe e Objeto**;
- A utilização e **Instância do objeto** na programação;
- O **reuso de código**, como algo vantajoso na programação;
- Percepção da **transição** do objeto real para a sua representação e uso em código-fonte.

Parte 2 – Encapsulamento

PARTE 2 - Encapsulamento

ASSUNTO	Encapsulamento e tipos de acesso
FORMATO	Individual ou em grupos (2 a 3 alunos)
DURAÇÃO DA ATIVIDADE	45 – 50 minutos
TEMPO DE PREPARAÇÃO	1 hora a 1 hora e 30 minutos de preparação

OBJETIVOS

Este plano de aula serve como guia de aplicação da Parte 2 do Workshop do Método ComFAPOO (<https://github.com/humbertozeanetti/comfapoo>, Parte 2 – Encapsulamento). Os objetivos a serem alcançados desta parte da aplicação são:

- Compreender o conceito de **encapsulamento** e seu uso.
- Diferenciar os tipos de acesso **público** e **privado**.
- Aplicar esses tipos de acesso em métodos e atributos.

MATERIAIS UTILIZADOS

Sala ou laboratório com projetor ou televisão.

Para cada grupo de alunos:

- 1 computador.
- 1 placa Arduino ou similar.

Componentes para os circuitos:

- 1 LED
- 1 resistor de 330 ohms
- 1 resistor de 1 kohms
- 1 *push button*
- 5 cabos de conexão

TAREFA 1 - Guia de aplicação

Nesta tarefa o objetivo principal é compreender o conceito de encapsulamento e alterar as funções da classe LED e interagindo com outro objeto. Para iniciar, apresente que haverá novas funções atreladas à classe LED, como a de trocar de estado, entre ligado e desligado, e a de mostrar o estado do LED.

Nesse momento é importante explicar sobre o conceito de Encapsulamento, como está descrito no texto apresentado no material para os alunos (início do item **Tarefa 1 - Melhorando nossa classe LED**). Após isso, apresentar a montagem, como na Figura 1, e discutir sobre as novas funções, mostrando a Figura 2.

Parte 2 – Encapsulamento

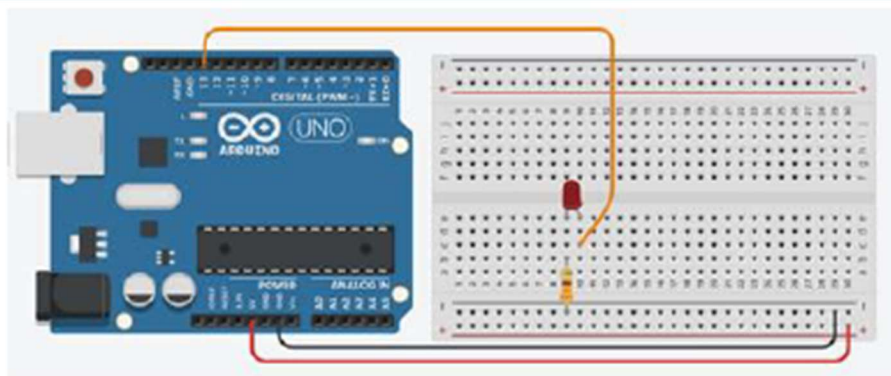


Figura 1 – Montagem (presente no material disponível para o aluno)



Figura 2 – Imagem de apoio sobre o LED (presente no material disponível para o aluno)

Reforçando o conceito de Encapsulamento, enfatizar que terão métodos "escondidos" e outros "acessíveis", apoiando ainda mais o conceito de "responsabilidade". Dar destaque para esse trecho de discussão:

- Escondido:
 - os atributos pino e estado, pois só podem ser alterados pelo próprio objeto
 - os métodos `acender()` e `apagar()`, pois agora eles irão alterar o atributo estado
- Acessível:
 - o método construtor, para que se possa criar o objeto
 - os métodos `trocaEstado()` e `mostraEstado()`, que serão como vamos pedir para mudar o estado do LED e como vemos se seu estado mudou. Com isso, podemos definir seus modos de acessos. O que será escondido será demarcado como privado (*private*) e o que será acessível será demarcado como público (*public*).

Apresentar a nova classe (Figura 3), dando destaque aos símbolos privado (-) e público (+) e depois o código-fonte (disponível no material do aluno no item Representando em código-fonte).

LED
-pino : int
-estado : int
-acender() : void
-apagar() : void
+trocaEstado() : void
+mostraEstado() : int

Figura 3 – Classe LED (presente no material disponível para o aluno)

Parte 2 – Encapsulamento

Uma discussão válida nesse momento é relatar que os métodos `acender()` e `apagar()` são "encapsulados" pois são responsabilidade que o próprio objeto irá definir o momento da ação. Caso outro componente, "peça" para que o objeto LED acenda, deverá pedir por meio de uma "interface" (o método `trocaEstado()`).

Faça os testes de acesso com os códigos-fontes presentes no material do aluno e mostre as mensagens de erro. Depois mostrar o código-fonte que realiza o pisca-pisca (mostrando que o `trocaEstado()` funciona) e o estado no console de saída serial (mostrando que o `mostraEstado()` funciona)

TAREFA 2 – Guia de aplicação

Nesta tarefa o objetivo principal é aplicar no conceito de Encapsulamento em um cenário real, onde um objeto Botão (representado pelo *push button*) com a interação com esses 2 componentes. Nesse contexto, ao ser pressionado o Botão, o LED irá trocar seu estado (ligado ou desligado). Esse exemplo ilustra que o objeto LED terá a responsabilidade de decidir se o LED acende ou não, por meio do método `trocaEstado()`.

Apresentar a montagem da Figura 4, podendo mostrar o que deverá ocorrer quando o Botão for pressionado antes de apresentar de fato o código-fonte.

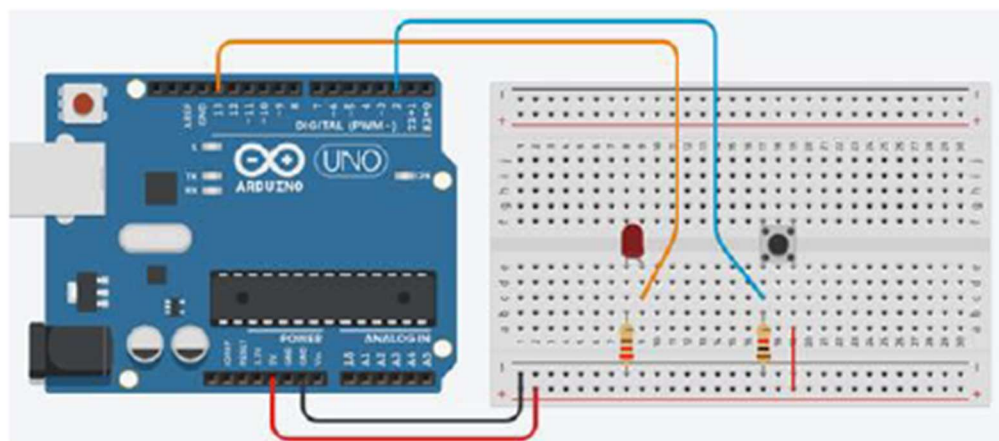


Figura 4 – Montagem (presente no material disponível para o aluno)

Mostrar também como ficará as funções do Botão (Figura 5) e como será a classe (Figura 6).



Figura 5 – Objeto Botão (presente no material disponível para o aluno)

Parte 2 – Encapsulamento

BOTÃO
-pino : int -estado : int
+verificaEstado() : int

Figura 6 – Classe Botão (presente no material disponível para o aluno)

Após a apresentação da classe, mostrar o código-fonte da mesma (item **Código-fonte** da Tarefa 2) e, por fim, o código-fonte completo, verificando a interação entre os componentes no circuito.

DISCUSSÃO SOBRE A APLICAÇÃO

Na Tarefa 1 o foco de aprendizado pode ser separado nos seguintes pontos e pode ser exposto e discutidos com os alunos:

- **O conceito de Encapsulamento:** mostrar o que o conceito de Encapsulamento é importante para o projeto de sistema orientado a objetos, e que cada objeto tem suas "responsabilidades" dentro de um cenário maior.
- **Tipos de acesso:** a construção do objeto e suas definições de Encapsulamento dependem da compreensão dos tipos de acesso público e privado.

Na Tarefa 2 o foco da aprendizagem pode ser separado nos seguintes pontos e pode ser exposto e discutidos com os alunos:

- **Uso do Encapsulamento:** definir que em um contexto de sistemas, com diferentes objetos a importância de ocultamento de funções e acessos por meios de "interfaces".

RESULTADOS ESPERADOS

O esperado por parte dos alunos ao final da aplicação:

- Compreender e aplicação o conceito do Encapsulamento em objetos.
- Entender o contexto dos tipos de acesso e suas aplicações.
- Contextualizar a aplicação do Encapsulamento em um projeto orientado a objetos.

Parte 3 – Herança

PARTE 3 - Herança

ASSUNTO	Herança
FORMATO	Individual ou em grupos (2 a 3 alunos)
DURAÇÃO DA ATIVIDADE	45 – 50 minutos
TEMPO DE PREPARAÇÃO	1 hora a 1 hora e 30 minutos de preparação

OBJETIVOS

Este plano de aula serve como guia de aplicação da Parte 3 do Workshop do Método ComFAPOO (<https://github.com/humbertozezanetti/comfapoo>, Parte 3 – Herança). Os objetivos a serem alcançados desta parte da aplicação são:

- Entender e aplicar o conceito de **Herança**.
- Estender a compreensão sobre **reuso de código**.
- Conhecer práticas básicas de estruturação de um sistema orientado a objetos.

MATERIAIS UTILIZADOS

Sala ou laboratório com projetor ou televisão.

Para cada aluno ou grupo de alunos:

- 1 computador.
- 1 placa Arduino ou similar.

Componentes para os circuitos:

- 1 LDR (sensor de luminosidade)
- 1 termistor NTC (sensor de temperatura)
- 1 resistor de 330 ohms
- 1 resistor de 1 kohms
- 1 *push button*
- 5 cabos de conexão

TAREFA 1 - Guia de aplicação

Nesta tarefa o objetivo principal é compreender e aplicar o conceito de Herança utilizando 2 sensores. Tais sensores compartilham características em comum, que ajudará o aluno a entender a necessidade e vantagens da Herança na programação. Inicie a apresentação dos sensores, explicando de maneira sucinta quais são as principais funções de ambos, além de enfatizar suas características e ações. Mostrar a montagem da Figura 1.

Parte 3 – Herança

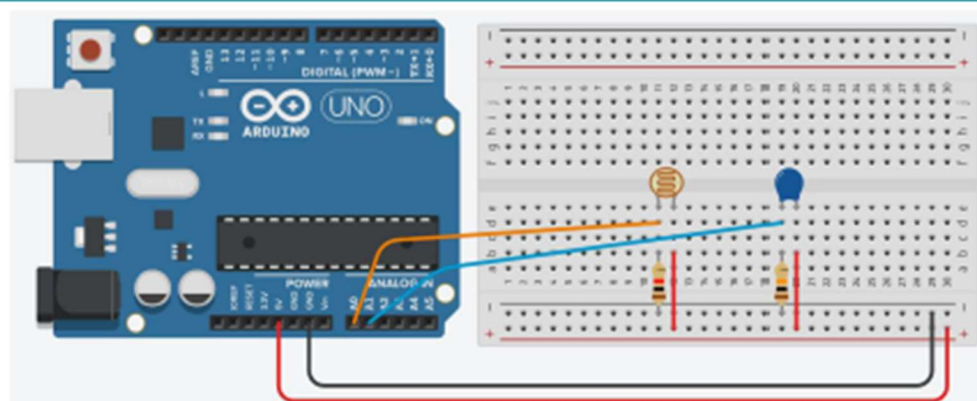


Figura 1 – Montagem (presente no material disponível para o aluno)

Após a apresentação da montagem, enfatize quais são as características (atributos e métodos) em comum entre os sensores, dando indícios que eles poderão ser “modelados” (representações em classe) similares, levando a compreensão da classe-mãe ou superclasse. Também destaque o que há de diferente entre eles, assim dando sentido para as classes-filhas ou subclasses. Use as Figura 2 e 3 para auxiliar na apresentação, e também execute os códigos-fontes (presentes no material do aluno no item **Representando em código-fonte**) de leitura desses sensores, para mostrar o funcionamento dos sensores



Figura 2 – Imagem de apoio sobre o LDR (presente no material disponível para o aluno)



Figura 3 – Imagem de apoio sobre o termistor (presente no material disponível para o aluno)

Juntamente com as imagens ilustrativas, apresentar os diagramas de classe se cada um dos sensores (Figuras 4 e 5). Discuta sobre os métodos que cada classe tem, principalmente sobre a necessidade de

Parte 3 – Herança

fazer as conversões para cada uma das unidades de medidas necessárias. Nesse momento é importante enfatizar que nos diagramas de classe é possível perceber que há similaridades, como apresentado com os objetos reais e diagramas.

LDR
+pino : int +valor : int -lux : int
+leValor() : int +mostraLux() : float

Figura 4 – Classe LDR (presente no material disponível para o aluno)

Termistor
+pino : int +valor : int -tempK: float -tempC : float
+leValor() : int +mostraKelvin() : float +mostraCelsius() : float

Figura 5 - Classe Termistor (presente no material disponível para o aluno)

TAREFA 2 – Guia de aplicação

Nesta tarefa o objetivo ainda é utilizado a mesma montagem da tarefa anterior e o objetivo principal é destacar que as representações das similaridades e diferenças dos objetos (e suas classes), irão orientar a estrutura do projeto de software. Novamente é necessário apontar as similaridades e diferenças, para a construção das classes. A Figura 6 aponta as similaridades, que conduzem a criar uma superclasse, que leve essas características.

LDR	Termistor
+pino : int +valor : int -lux : int	+pino : int +valor : int -tempK: float -tempC : float
+leValor() : int +mostraLux() : float	+leValor() : int +mostraKelvin() : float +mostraCelsius() : float

Figura 6 - Comparações das classes (presente no material disponível para o aluno)

A parti disso, pode-se apresentar a classe Sensor, relacionando as discussões anteriores e já levando a distinção das classes-filhas ou subclasses, como mostra a Figura 7.

Parte 3 – Herança

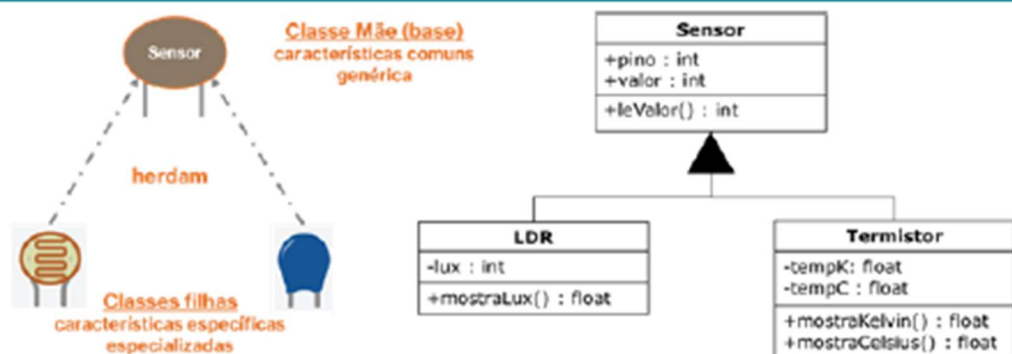


Figura 7 – Estrutura entre as classes (presente no material disponível para o aluno)

Após a apresentação dos diagramas, apresentar os códigos-fontes (presentes no material do aluno no item **Representando em código-fonte**) e executar, dando ênfase em mostrar que a superclasse **Sensor** deve ser criada antes, pois será a base que fornecerá os recursos para as classes **LDR** e **Termistor**. Deixar claro que nas classes-filhas são implementadas apenas as suas diferenças, anteriormente mapeadas.

Como experimentação adicional, mostrar que mesmo **Sensor** sendo uma classe, não faz sentido criar um objeto por meio dela (tanto que não há método construtor).

DISCUSSÃO SOBRE A APLICAÇÃO

Na Tarefa 1 o foco de aprendizado pode ser separado nos seguintes pontos e pode ser exposto e discutidos com os alunos:

- **O conceito de Herança:** mostrar o que o conceito de Herança é importante para o projeto de sistema orientado a objetos, sendo uma propriedade que é possível identificar nos objetos reais e replicado no código-fonte.
- **Complementar o exercício de Abstração:** o mapeamento necessário das características e comuns entre os objetos se apresenta como um exercício de abstração comum e necessário em projetos de software orientado a objetos.

Na Tarefa 2 o foco da aprendizagem pode ser separado nos seguintes pontos e pode ser exposto e discutidos com os alunos:

- **Aplicação de Herança:** experimentação e prática do conceito de Herança em código-fonte, que espelha todas as discussões apresentadas na Tarefa 1.
- **Diferentes tipos de classes:** identificar que a classe-mãe e suas classes-filhas possuem funções diferentes dentro do projeto.

RESULTADOS ESPERADOS

O esperado por parte dos alunos ao final da aplicação:

- Compreender a aplicação e as vantagens do uso de Herança em projetos.
- Estender o conhecimento sobre reuso de código
- Conhecer estratégias de estruturação de um projeto de software.

Parte 4 – Polimorfismo

PARTE 4 - Polimorfismo

ASSUNTO	Polimorfismo
FORMATO	Individual ou em grupos (2 a 3 alunos)
DURAÇÃO DA ATIVIDADE	45 – 50 minutos
TEMPO DE PREPARAÇÃO	1 hora a 1 hora e 30 minutos de preparação

OBJETIVOS

Este plano de aula serve como guia de aplicação da Parte 4 do Workshop do Método ComFAPOO (<https://github.com/humbertoanetti/comfapoo>, Parte 4 – Polimorfismo). Os objetivos a serem alcançados desta parte da aplicação são:

- Entender e aplicar o conceito de **Polimorfismo**.
- Conhecer estratégias básicas para o desenvolvimento de um sistema orientado a objetos.

MATERIAIS UTILIZADOS

Sala ou laboratório com projetor ou televisão.

Para cada aluno ou grupo de alunos:

- 1 computador.
- 1 placa Arduino ou similar.

Componentes para os circuitos:

- 1 LED
- 2 botões (push button)
- 1 resistor de 330 ohms
- 2 resistores 2 kohms
- 8 cabos de conexão

TAREFA 1 - Guia de aplicação

Nesta tarefa o objetivo principal é compreender as vantagens e aplicar o conceito fundamental de Polimorfismo. Para isso será utilizado uma nova versão da classe LED, sendo mais simples, para atender o propósito da tarefa. Comece mostrando apenas o LED (Figura 1) e sua nova estrutura de classe (Figura 2 e 3).

Parte 4 – Polimorfismo

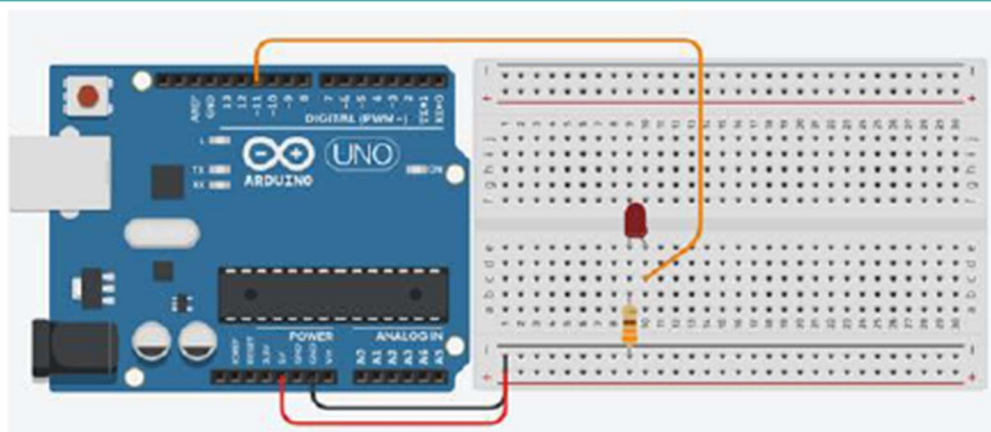


Figura 1 – Montagem (presente no material disponível para o aluno)



Figura 2 – Imagem sobre o LED (presente no material disponível para o aluno)

Led
+pino : int
+estado : int
+acender() : void
+apagar() : void

Figura 3 – Classe LED (presente no material disponível para o aluno)

Nessa tarefa o mais importante é discutir sobre como podemos “enxergar” 2 tipos de LEDs diferentes, ou como podemos entender 2 “formas” distintas de **acender** o LED (acender normalmente, piscando, gradualmente, etc.). Comece apresentando o conceito de Polimorfismo e que se utilizado uma classe base (como em Herança), que terá características que poderão assumir funções ou formas diferentes (no caso, o método `acender()`). Também se faz necessário apontar a necessidade da assinatura virtual no método `acender()`, denotando que ele será o que terá formas diferentes de uso (presente no item **Representando em código-fonte** no material disponível ao aluno).

TAREFA 2 – Guia de aplicação

Nesta tarefa o objetivo é aplicar a estratégia do Polimorfismo, criando as classes e utilizando os objetos em “formas” diferentes. A montagem (Figura 4) será ampliada, agora com a adição de 2 botões. Cada um deles irá acender o LED de uma forma diferente.

Parte 4 – Polimorfismo

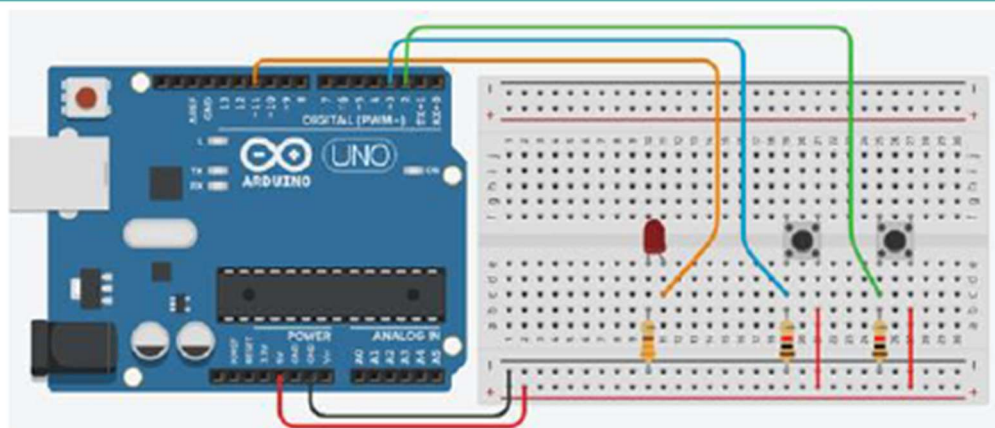


Figura 4 – Montagem (presente no material disponível para o aluno)

Nesse contexto teremos 2 maneiras, ou formas, de acender o LED: piscando e oscilando. Essas duas maneiras são diferentes da convencional, deixando claro as diferentes formas de atuação do LED. Na classe base **Led** o método `acender()` é o convencional, mas que poderá ter duas outras formas, implementadas nas classes **LedOscila** e **LedPisca**. A Figura 5 mostra essas formas de comportamento em destaque.

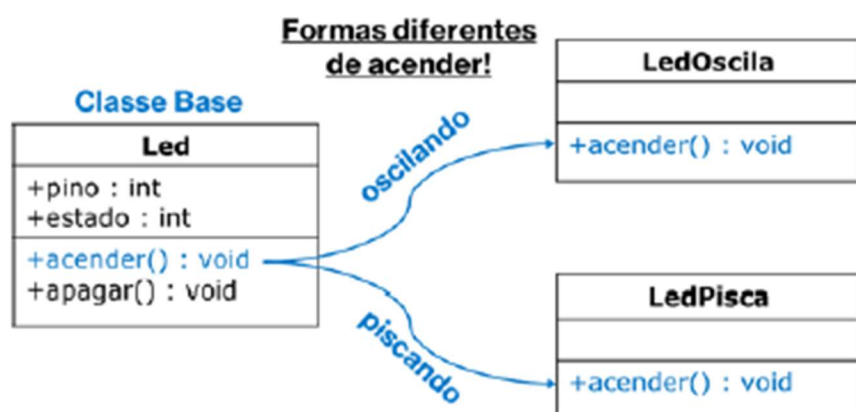


Figura 5 – Formas de diferentes e suas classes (presente no material disponível para o aluno)

É importante ressaltar nesse momento que o definirá a forma em que o LED acenderá será a escolha entre as classes para criar o objeto. Nesse momento apresente os códigos-fontes das classes que redefinem as formas, nos itens **Representando em código-fonte (classe LedOscila)** e **Representando em código-fonte (classe LedPisca)**. Também apresentar o código fonte da classe Botão, que será usada no projeto.

Nesse ponto é importante enfatizar a instância do objeto **led** e suas particularidades. Notar que há um asterisco (*), denotando um ponteiro. Essa estratégia, por regras da linguagem C++, possibilita podermos associar a este objeto características que sejam baseadas nele, como as classes **LedOscila** e **LedPisca** que tem como base a classe **Led**. Destacar também que antes de usar o objeto, temos que associar uma forma para ele (entre **LedOscila** ou **LedPisca**) usando a nota **&** (e comercial), que referencia a memória da classe logo a frente. Nessa dinâmica, caso seja pressionada um botão, será criado o **objeto de uma forma**, caso seja o outro botão, será de outra forma.

DISCUSSÃO SOBRE A APLICAÇÃO

Na Tarefa 1 o foco de aprendizado pode ser separado nos seguintes pontos e pode ser exposto e discutidos com os alunos:

Parte 4 – Polimorfismo

- **O conceito de Polimorfismo:** mostrar como aplicar o conceito de Polimorfismo no desenvolvimento de sistemas.
- **Complementar a estruturação de projeto:** a adoção do Polimorfismo amplia os conhecimentos aplicados à estruturação de software orientado a objetos.

Na Tarefa 2 o foco da aprendizagem pode ser separado nos seguintes pontos e pode ser exposto e discutidos com os alunos:

- **Aplicação de Polimorfismo:** experimentação e prática do conceito de Polimorfismo em código-fonte, que espelha todas as discussões apresentadas na Tarefa 1.
- **Diferentes tipos de classes:** identificar a relação da definição da classe base e as classes que definem os fatores polimórficos.

RESULTADOS ESPERADOS

O esperado por parte dos alunos ao final da aplicação:

- Compreender a aplicação e as vantagens do uso de Polimorfismo em projetos.
- Estender o conhecimento sobre reuso de código e relações entre classes e objetos.
- Conhecer estratégias de estruturação de um projeto de software.

Parte 5 – Sistema orientado a objetos

PARTE 5 – Sistema orientado a objetos

ASSUNTO	Sistema orientado a objetos
FORMATO	Individual ou em grupos (2 a 3 alunos)
DURAÇÃO DA ATIVIDADE	45 – 50 minutos
TEMPO DE PREPARAÇÃO	1 hora a 1 hora e 30 minutos de preparação

OBJETIVOS

Este plano de aula serve como guia de aplicação da Parte 5 do Workshop do Método ComFAPOO (<https://github.com/humbertozezanetti/comfapoo>, Parte 5 – Sistema orientado a objetos). Os objetivos a serem alcançados desta parte da aplicação são:

- Entender e aplicar o **relacionamento e troca de mensagens** entre objetos.
- Conhecer conceitos básicos de organização de um sistema orientado a objetos.

MATERIAIS UTILIZADOS

Sala ou laboratório com projetor ou televisão.

Para cada aluno ou grupo de alunos:

- 1 computador.
- 1 placa Arduino ou similar.

Componentes para os circuitos:

- 3 LEDs
- 1 LDR
- 3 resistores de 330 ohms
- 1 resistor 1 kohms
- 7 cabos de conexão

TAREFA 1 - Guia de aplicação

Nesta tarefa o objetivo principal é como estruturar um sistema orientado a objetos que utilize a troca de mensagens entre os objetos que o compõe. Esse conceito direciona o aluno a compreender a organização básica de um sistema mais complexo e como os objetos se relacionam. Para essa prática será utilizado a seguinte montagem (Figura 1), simulando um semáforo com controle de modo diurno (funcionamento normal) e noturno (sinal amarelo piscante).

Parte 5 – Sistema orientado a objetos

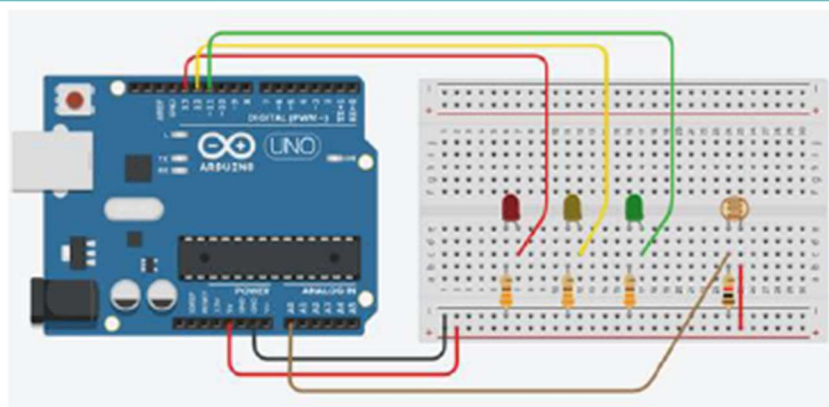


Figura 1 - Montagem (presente no material disponível para o aluno)

O conceito de troca de mensagens se caracteriza como a invocação de um objeto por outro. Esse aspecto é importante ao apresentar a proposta do sistema da tarefa, o semáforo. Apresente a proposta do projeto, utilizando as imagens ilustrativas Figura 2 e Figura 3. Enfatize que o "sistema" será o semáforo e que ele é composto pelos objetos LEDs (para as luzes), o LDR (para identificação se é dia ou noite) e o semáforo, que irá definir quais objetos usar. Para determinado modo.



Figura 2 – Simulação de um semáforo (presente no material disponível para o aluno)

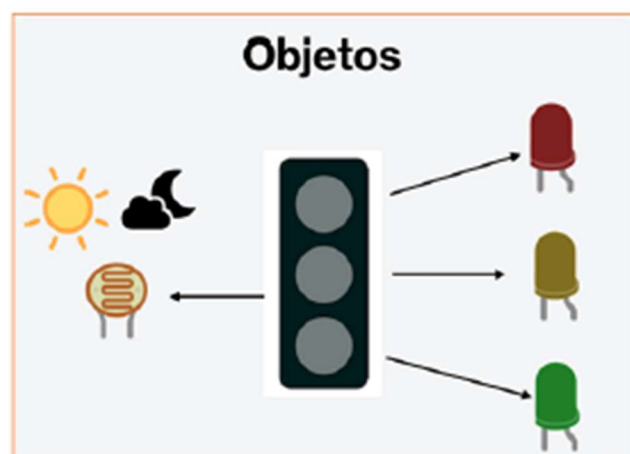


Figura 3 – Objetos que compõem o semáforo (presente no material disponível para o aluno)

A partir dessa apresentação, pode-se apresentar o diagrama de classe do sistema (Figura 4).

Parte 5 – Sistema orientado a objetos

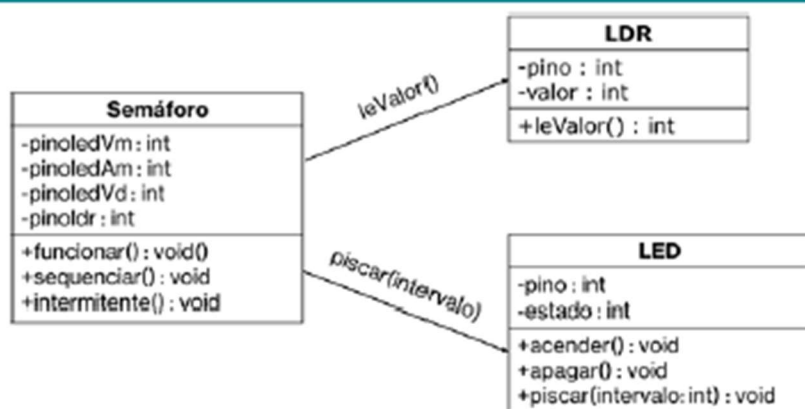


Figura 4 – Diagrama de classes do sistema (presente no material disponível para o aluno)

Também é adequado já trazer a discussão do que se trata uma mensagem. Um exemplo, é o objeto semáforo invocando um objeto LED e “pedindo” por meio de mensagem que ele pisque e passando por meio dessa mensagem a informação “500” (intervalo de 500 milissegundos), como mostra a Figura 5.

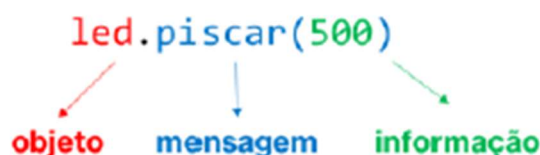


Figura 5 – Exemplo de troca de mensagens (presente no material disponível para o aluno)

TAREFA 2 – Guia de aplicação

Nesta tarefa o objetivo é implementar o sistema de semáforo, apresentado na Tarefa 1. Comece apresentando os códigos-fontes do sistema (presentes no itens **Representando em código-fonte (classe LED e LDR)** e **Representando em código-fonte (Semáforo)**). Importante enfatizar que deve-se primeiro definir as classes **Led** e **LDR**, pois a classe **Semáforo** faz instâncias de objetos dessa classe.

Ao criar o objeto da classe **Semáforo**, o método construtor recebe quais pinos que todos os componentes estão conectados e seus métodos se encarregam de criar os objetos necessários para cada situação do sistema (modos diurno e noturno).

DISCUSSÃO SOBRE A APLICAÇÃO

Na Tarefa 1 o foco de aprendizado pode ser separado nos seguintes pontos e pode ser exposto e discutidos com os alunos:

- **Organização de um sistema orientado a objetos:** mostrar funciona a organização básica e o relacionamento entre objetos em um sistema.
- **Exercício de Abstração:** nesse de um novo cenário, como objetos relacionados entre si, cria-se a necessidade de novamente exercitar a abstração, para compreender como os objetos podem ser compostos por outros objetos.

Na Tarefa 2 o foco da aprendizagem que pode ser exposto e discutido com os alunos:

- **Aplicação da organização:** experimentação e prática da organização de um sistema orientado a objetos, a partir das discussões apresentadas na Tarefa 1.

Parte 5 – Sistema orientado a objetos

RESULTADOS ESPERADOS

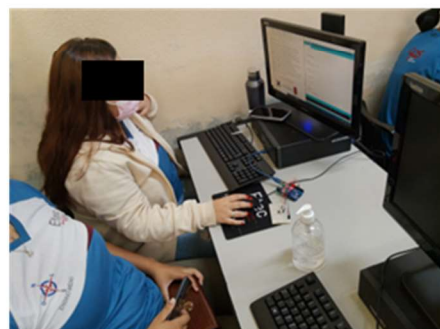
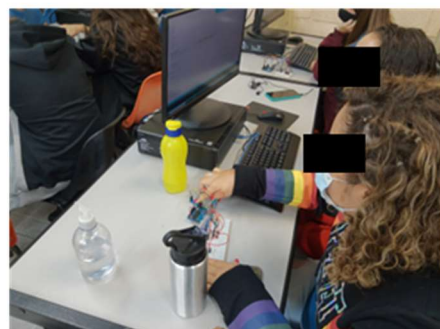
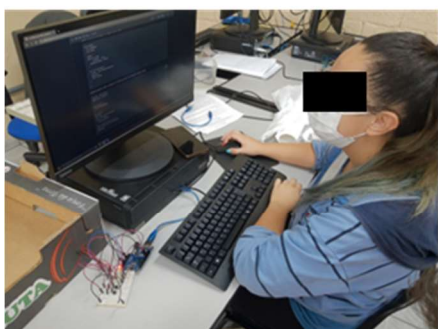
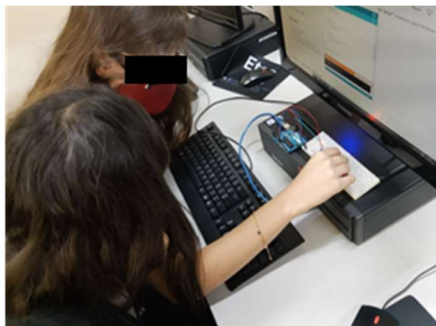
O esperado por parte dos alunos ao final da aplicação:

- Compreender e aplicar a organização básica de um sistema orientado a objetos.
- Entender a utilização de troca de mensagens e a relação entre objetos.
- Exercitar a abstração de objetos que são formados por outros objetos.

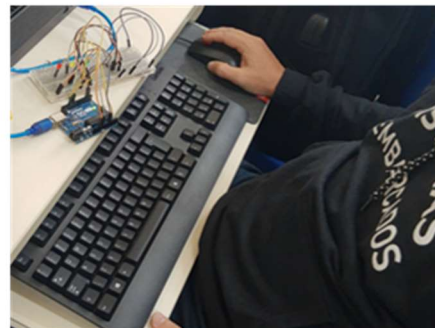
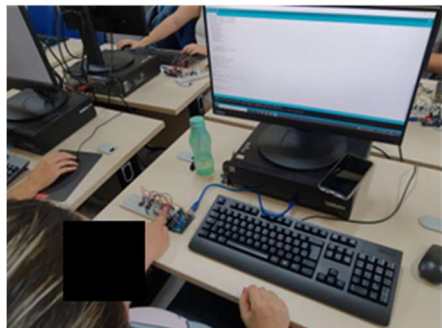
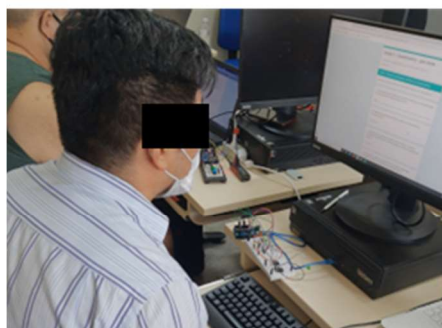
APÊNDICE C – Imagens das oficinas

Este apêndice apresenta algumas imagens das aplicações das oficinas na Escola Técnica (ETEC) Rosa Perrone Scavone, na cidade de Itatiba/SP, e na Faculdade de Tecnologia (FATEC) Deputado Ary Fossen, na cidade de Jundiaí/SP. Essas oficinas foram realizadas entre os meses de março e maio de 2022.

Oficinas realizadas na ETEC Rosa Perrone Scavone



Oficinas realizadas na FATEC Deputado Ary Fossen



ANEXO A – Parecer do Comitê de Ética

Este anexo apresenta o parecer de ética aprovado para esta pesquisa. A seguir é exibido a última versão do parecer consubstanciado do Comitê Ético de Pesquisa (CEP) com o Certificado de Apresentação de Apreciação Ética (CAAE) de número 29622720.8.0000.5404.



UNICAMP - CAMPUS
CAMPINAS



PARECER CONSUBSTANCIADO DO CEP

DADOS DA EMENDA

Título da Pesquisa: Uma metodologia de ensino de Programação Orientada a Objetos utilizando Computação Física e Aprendizagem Significativa.

Pesquisador: HUMBERTO AUGUSTO PIOVESANA ZANETTI

Área Temática:

Versão: 5

CAAE: 29622720.8.0000.5404

Instituição Proponente: Faculdade de Tecnologia

Patrocinador Principal: Financiamento Próprio

DADOS DO PARECER

Número do Parecer: 5.175.247

Apresentação do Projeto:

Trata-se da apresentação de uma Emenda ao projeto CAAE 29622720.8.0000.5404

Objetivo da Pesquisa:

Apresentar uma Emenda (alteração do cronograma para coleta de dados)

Avaliação dos Riscos e Benefícios:

Os riscos e benefícios não foram alterados em relação ao projeto original

Comentários e Considerações sobre a Pesquisa:

Data da aprovação do projeto por este CEP:

- 22/04/2020 (parecer número 3.983.923, em 'PB_PARECER_CONSUBSTANCIADO_CEP_3983923.pdf', de 22/04/2020 10:41:25)

- 17/11/2020 (parecer número 4.402.951, em 'PB_PARECER_CONSUBSTANCIADO_CEP_4402951_E1.pdf', de 17/11/2020 09:30:51)

- 19/03/2021 (parecer número 4.601.064, em 'PB_PARECER_CONSUBSTANCIADO_CEP_4601064_E2.pdf', de 19/03/2021 11:15:05)

Através da Emenda apresentada, o pesquisador informa que (item 'Justificativa da Emenda' do arquivo 'PB_INFORMAÇÕES_BÁSICAS_1860718_E3.pdf', de 17/11/2021 14:40:14, e 'CartaEmenda_Nov21.docx', de 17/11/2021 14:38:35):

Endereço: Rua Tessália Vieira de Camargo, 126, 1º andar do Prédio I da Faculdade de Ciências Médicas
Bairro: Barão Geraldo **CEP:** 13.083-887
UF: SP **Município:** CAMPINAS
Telefone: (19)3521-8936 **Fax:** (19)3521-7187 **E-mail:** cep@unicamp.br



Continuação do Parecer: 5.175.247

"Devido à pandemia de COVID-19, as instituições de ensino paralisaram suas atividades presenciais por tempo indeterminado.

Até a presente data, muitas ainda optam pelo ensino à distância ou remoto, ou parcialmente presencial (sem a carga completa de aulas no formato presencial).

O projeto previa a aplicação de práticas pedagógicas (workshops) presenciais, ação que não pôde ocorrer no prazo estipulado anteriormente.

Esta emenda propõe um novo cronograma para o ano de 2022, com todas as atividades que foram propostas no cronograma anterior".

Informa, também, que "Na nova descrição do projeto (documento 'Projeto_Pesquisa_Modelo_Unicamp_v8.docx'), o texto do item 8 (Cronograma) sofreu modificações apenas nas datas, que estão tarjadas de amarelo no texto.

Os demais itens do documento não sofreram alterações desde a última versão submetida".

- Dessa forma, a coleta de dados, conforme apresentado no arquivo com o projeto completo, se encontra prevista para o período de MARÇO DE 2022 a MAIO DE 2022: "2. Aplicação dos workshops (práticas): período previsto para a aplicação dos workshops (práticas pedagógicas) com a metodologia proposta. Os 6 encontros poderão ocorrer entre março de 2022 a maio de 2022" (arquivo 'Projeto_Pesquisa_Modelo_Unicamp_v8.docx', de 17/11/2021 14:37:08)

- Conforme apresentado (arquivo 'AtestadoMatricula_nov2021.pdf', de 17/11/2021 14:39:10), o 'Prazo para Integralização' do doutoramento proposto se entende até 12/2022.

Considerações sobre os Termos de apresentação obrigatória:

Para avaliação desta Emenda foi analisado o documento intitulado 'PB_INFORMAÇÕES_BÁSICAS_1860718_E3.pdf', de 17/11/2021 14:40:14.

Recomendações:

RECOMENDAÇÃO 1 - Desde a Emenda aprovada em novembro/20 (arquivo 'PB_PARECER_CONSUBSTANCIADO_CEP_4402951_E1.pdf', de 17/11/2020 09:30:51), se encontra previsto a aumento do número de voluntários, dado que houve adesão de uma nova escola ao

Endereço: Rua Tessália Vieira de Camargo, 126, 1º andar do Prédio I da Faculdade de Ciências Médicas
 Bairro: Barão Geraldo CEP: 13.083-887
 UF: SP Município: CAMPINAS
 Telefone: (19)3521-8936 Fax: (19)3521-7187 E-mail: cep@unicamp.br



UNICAMP - CAMPUS
CAMPINAS



Continuação do Parecer: 5.175.247

projeto proposto.

No entanto, ainda nessa versão da Emenda, a informação 'Grupos em que serão divididos os participantes da pesquisa neste centro' ainda prevê o número original de voluntários, assim como um único grupo de estudos.

Dentro do possível, readequar - caso seja submetida uma nova emenda.

RECOMENDAÇÃO 2 - O novo prazo solicitado para a coleta de dados aparece (conforme apontado pelo pesquisador) no cronograma do 'projeto completo' (arquivo 'Projeto_Pesquisa_Modelo_Unicamp_v6.docx', de 17/11/2021 14:37:08), mas no atual arquivo da Emenda é apresentado, no item 'Cronograma de Execução', o período previsto para coleta de dados aprovada na Emenda anterior ('de 01/09/2021 a 31/12/2021' - arquivo 'PB_PARECER_CONSUBSTANCIADO_CEP_4801064_E2.pdf', de 19/03/2021 11:15:05).

Dentro do possível, readequar - caso seja submetida uma nova emenda.

A Comissão Nacional de Ética em Pesquisa (Conep), do Conselho Nacional de Saúde (CNS) orienta a adoção das diretrizes do Ministério da Saúde (MS) decorrentes da pandemia causada pelo Coronavírus SARS-CoV-2 (Covid-19), com o objetivo de minimizar os potenciais riscos à saúde e a integridade dos participantes de pesquisas e pesquisadores.

De acordo com carta circular da CONEP intitulada "ORIENTAÇÕES PARA CONDUÇÃO DE PESQUISAS E ATIVIDADE DOS CEP DURANTE A PANDEMIA PROVOCADA PELO CORONAVÍRUS SARS-COV-2 (COVID-19)" publicada em 09/05/2020, referente ao item II. "Orientações para Pesquisadores":

- Aconselha-se a adoção de medidas para a prevenção e gerenciamento de todas as atividades de pesquisa, garantindo-se as ações primordiais à saúde, minimizando prejuízos e potenciais riscos, além de prover cuidado e preservar a integridade e assistência dos participantes e da equipe de pesquisa.
- Em observância às dificuldades operacionais decorrentes de todas as medidas impostas pela pandemia do SARS-CoV-2 (COVID- 19), é necessário zelar pelo melhor interesse do participante da pesquisa, mantendo-o informado sobre as modificações do protocolo de pesquisa que possam afetá-lo, principalmente se houver ajuste na condução do estudo, cronograma ou plano de trabalho.
- Caso sejam necessários a suspensão, interrupção ou o cancelamento da pesquisa, em decorrência dos riscos imprevisíveis aos participantes da pesquisa, por causas diretas ou indiretas, caberá aos investigadores a submissão de notificação para apreciação do Sistema

Endereço: Rua Tessália Vieira de Camargo, 126, 1º andar do Prédio I da Faculdade de Ciências Médicas
 Bairro: Barão Geraldo CEP: 13.083-887
 UF: SP Município: CAMPINAS
 Telefone: (19)3521-8935 Fax: (19)3521-7187 E-mail: cep@unicamp.br



UNICAMP - CAMPUS
CAMPINAS



Continuação do Parecer: S.175.247

CEP/Conep.

- Nos casos de ensaios clínicos, é permitida, excepcionalmente, a tramitação de emendas concomitantes à implementação de modificações/alterações no protocolo de pesquisa, visando à segurança do participante da pesquisa, assim como dos demais envolvidos no contexto da pesquisa, evitando-se, ainda, quando aplicável, a interrupção no tratamento dos participantes da pesquisa. Eventualmente, na necessidade de modificar o Termo de Consentimento Livre e Esclarecido (TCLE), o pesquisador deverá proceder com o novo consentimento, o mais breve possível.

Conclusões ou Pendências e Lista de Inadequações:

Emenda aprovada

Considerações Finais a critério do CEP:

- O participante da pesquisa deve receber uma via do Termo de Consentimento Livre e Esclarecido, na íntegra, por ele assinado (quando aplicável).
- O participante da pesquisa tem a liberdade de recusar-se a participar ou de retirar seu consentimento em qualquer fase da pesquisa, sem penalização alguma e sem prejuízo ao seu cuidado (quando aplicável).
- O pesquisador deve desenvolver a pesquisa conforme delineada no protocolo aprovado. Se o pesquisador considerar a descontinuação do estudo, esta deve ser justificada e somente ser realizada após análise das razões da descontinuidade pelo CEP que o aprovou. O pesquisador deve aguardar o parecer do CEP quanto à descontinuação, exceto quando perceber risco ou dano não previsto ao participante ou quando constatar a superioridade de uma estratégia diagnóstica ou terapêutica oferecida a um dos grupos da pesquisa, isto é, somente em caso de necessidade de ação imediata com intuito de proteger os participantes.
- O CEP deve ser informado de todos os efeitos adversos ou fatos relevantes que alterem o curso normal do estudo. É papel do pesquisador assegurar medidas imediatas adequadas frente a evento adverso grave ocorrido (mesmo que tenha sido em outro centro) e enviar notificação ao CEP e à Agência Nacional de Vigilância Sanitária – ANVISA – junto com seu posicionamento.

Endereço: Rua Tessália Vieira de Camargo, 126, 1º andar do Prédio I da Faculdade de Ciências Médicas
 Bairro: Barão Geraldo CEP: 13.083-887
 UF: SP Município: CAMPINAS
 Telefone: (19)3521-8936 Fax: (19)3521-7187 E-mail: cep@unicamp.br



UNICAMP - CAMPUS CAMPINAS



Continuação do Parecer: 5.175.247

- Eventuais modificações ou emendas ao protocolo devem ser apresentadas ao CEP de forma clara e sucinta, identificando a parte do protocolo a ser modificada e suas justificativas e aguardando a aprovação do CEP para continuidade da pesquisa. Em caso de projetos do Grupo I ou II apresentados anteriormente à ANVISA, o pesquisador ou patrocinador deve enviá-las também à mesma, junto com o parecer aprovatório do CEP, para serem juntadas ao protocolo inicial.

- Relatórios parciais e final devem ser apresentados ao CEP, inicialmente seis meses após a data deste parecer de aprovação e ao término do estudo.

- Lembramos que segundo a Resolução 466/2012, item XI.2 letra e, "cabe ao pesquisador apresentar dados solicitados pelo CEP ou pela CONEP a qualquer momento".

- O pesquisador deve manter os dados da pesquisa em arquivo, físico ou digital, sob sua guarda e responsabilidade, por um período de 5 anos após o término da pesquisa.

Este parecer foi elaborado baseado nos documentos abaixo relacionados:

Tipo Documento	Arquivo	Postagem	Autor	Situação
Informações Básicas do Projeto	PB_INFORMAÇÕES_BASICAS_1860718_E3.pdf	17/11/2021 14:40:14		Aceito
Outros	AtestadoMatricula_nov2021.pdf	17/11/2021 14:39:10	HUMBERTO AUGUSTO PIOVESANA ZANETTI	Aceito
Outros	CartaEmenda_Nov21.docx	17/11/2021 14:38:35	HUMBERTO AUGUSTO PIOVESANA ZANETTI	Aceito
Projeto Detalhado / Brochura Investigador	Projeto_Pesquisa_Modelo_Unicamp_v6.docx	17/11/2021 14:37:08	HUMBERTO AUGUSTO PIOVESANA ZANETTI	Aceito
Folha de Rosto	FolhadeRostoDigitalizada.pdf	06/11/2020 17:16:32	HUMBERTO AUGUSTO PIOVESANA ZANETTI	Aceito

Endereço: Rua Tessália Vieira de Camargo, 126, 1º andar do Prédio I da Faculdade de Ciências Médicas
 Bairro: Barão Geraldo CEP: 13.083-887
 UF: SP Município: CAMPINAS
 Telefone: (19)3521-8936 Fax: (19)3521-7187 E-mail: cep@unicamp.br



CEPUNICAMP
CURSO DE GRADUAÇÃO EM EDUCAÇÃO

UNICAMP - CAMPUS CAMPINAS



Continuação do Parecer: 5.175.247

TCLE / Termos de Assentimento / Justificativa de Ausência	TCLE_Responsaveis_v1.docx	06/04/2020 21:29:39	HUMBERTO AUGUSTO PIOVESANA ZANETTI	Aceito
TCLE / Termos de Assentimento / Justificativa de Ausência	TCLE_Maiores_v2.docx	06/04/2020 21:28:58	HUMBERTO AUGUSTO PIOVESANA ZANETTI	Aceito
TCLE / Termos de Assentimento / Justificativa de Ausência	TALE_Menores_v1.docx	06/04/2020 21:28:31	HUMBERTO AUGUSTO PIOVESANA ZANETTI	Aceito
TCLE / Termos de Assentimento / Justificativa de Ausência	TCLE_Menores_v1.pdf	06/02/2020 15:42:19	HUMBERTO AUGUSTO PIOVESANA ZANETTI	Aceito
TCLE / Termos de Assentimento / Justificativa de Ausência	TCLE_Maiores_v1.pdf	06/02/2020 15:42:02	HUMBERTO AUGUSTO PIOVESANA ZANETTI	Aceito

Situação do Parecer:

Aprovado

Necessita Apreciação da CONEP:

Não

CAMPINAS, 17 de Dezembro de 2021

Assinado por:

Renata Maria dos Santos Celeghini
(Coordenador(a))

Endereço: Rua Tessália Vieira de Camargo, 126, 1º andar do Prédio I da Faculdade de Ciências Médicas
 Bairro: Barão Geraldo CEP: 13.083-887
 UF: SP Município: CAMPINAS
 Telefone: (19)3521-8936 Fax: (19)3521-7187 E-mail: cep@unicamp.br

ANEXO B – Autorizações do uso dos artigos

Este anexo apresenta conversas por e-mail, contratos de uso de imagem e direitos autorais dos artigos que compõem os capítulos 2, 3, 5, 6 e 7.

21/08/2022 22:50

Solicitação para uso de artigo - hzanetti@gmail.com - Gmail

Solicitação para uso de artigo

Caixa de entrada



Humberto Zanetti <hzanetti@gmail.com>
para publicacoes

Bom dia,

Logo defenderei meu Doutorado em Tecnologia pela Faculdade de Tecnologia da Universidade Estadual de Campinas (FT-UNICAMP).

Envio este e-mail com o intuito de solicitar autorização da organização da SBC para incorporar o artigo que foi publicado no evento "Simpósio Brasileiro de Informática na Exatidão" (SBIE) em 2016. A tese será disponibilizada online na biblioteca digital da UNICAMP em 2023.

O artigo em questão é: ZANETTI, H. A. P.; BORGES, M. A. P.; RICARTE, I. L. M. Pensamento computacional no ensino de programação: Uma revisão sistemática da literatura (SBIE). 2016, p. 21.

Agradeço a atenção!

Humberto A. P. Zanetti



Publicações <publicacoes@sbc.org.br>
para mim

Prezado Humberto, bom dia!

Os autores que publicam seus trabalhos na SBC retêm os direitos autorais e não necessitam de nossa autorização para utilizá-los. Entretanto, é importante que seja mencionada nossa política de direitos autorais que está disponível na página "Expediente" do evento EDUCOMP: <https://ol.sbc.org.br/index.php/educomp/about/editorialTeam>

Direitos Autorais

Os autores dos artigos publicados nos Anais do EduComp retêm os direitos autorais de suas obras e autorizam a SBC a publicá-las de acordo com os termos da licença. Fica permitido aos autores ou a terceiros a reprodução ou distribuição, em parte ou no todo, de material extraído dessas obras, de forma verbatim, adaptada ou não, desde que sejam atribuídos os devidos créditos às criações originais. Cópias das obras não devem ser utilizadas de nenhum modo que implique o encobrimento de autoria.

Um abraço,



Jessica Paiva Macedo Muller
Assistente de Publicações
publicacoes@sbc.org.br
Sociedade Brasileira de Computação
Caixa Postal 15012
91.501-970 - Porto Alegre - RS
Tel.: +55 51 3308-7740 Fax: +55 51 3308-7142



Publicações <publicacoes@sbc.org.br>
para mim

Prezado Humberto, bom dia!

Os autores que publicam seus trabalhos na SBC retêm os direitos autorais e não necessitam de nossa autorização para utilizá-los. Entretanto, é importante que seja mencionada nossa política de direitos autorais que está disponível na página "Expediente" do evento SBIE: <https://ol.sbc.org.br/index.php/sbie/about/editorialTeam>

Direitos Autorais

Os autores dos artigos publicados nos Anais do SBIE retêm os direitos autorais de suas obras e autorizam a SBC a publicá-las de acordo com os termos da licença. Fica permitido aos autores ou a terceiros a reprodução ou distribuição, em parte ou no todo, de material extraído dessas obras, de forma verbatim, adaptada ou não, desde que sejam atribuídos os devidos créditos às criações originais. Cópias das obras não devem ser utilizadas de nenhum modo que implique o encobrimento de autoria.

Um abraço,



Jessica Paiva Macedo Muller
Assistente de Publicações
publicacoes@sbc.org.br
Sociedade Brasileira de Computação
Caixa Postal 15012
91.501-970 - Porto Alegre - RS
Tel.: +55 51 3308-7740 Fax: +55 51 3308-7142

Em 15/02/2022 08:39, Humberto Zanetti escreveu:

21/08/2022 22:50

Solicitação para uso de artigo - hzanetti@gmail.com - Gmail

V Congresso Brasileiro de Informática na Educação (CBIE 2016)
XXVII Simpósio Brasileiro de Informática na Educação (SBIE 2016)

Este volume contém os artigos apresentados no 27º Simpósio Brasileiro de Informática na Educação (SBIE 2016). A inclusão nesta publicação não necessariamente constitui endosso pelos editores e/ou organizadores.

A fonte e os direitos da SBC devem ser devidamente referenciados. As cópias não devem ser utilizadas de nenhum modo que implique o endosso da SBC. Cópias da obra não podem ser colocadas à venda sem a autorização expressa da SBC.

Permissão para fazer cópias impressas ou digitais de todo ou parte deste trabalho para uso pessoal ou acadêmico é concedido sem taxas desde que cópias não sejam feitas ou distribuídas para renda ou vantagem comercial e que cópias contenham esta observação e citação completa na primeira página.

Sociedade Brasileira de Computação
CNPJ no. 29.532.264/0001-78
Inscrição Estadual isenta
CCM nº 18115128
<http://www.sbc.org.br>

Av. Bento Gonçalves, 9500 – Setor 4 – Sala 116 – Prédio 43424 – Agronomia
CEP 91501-970 – Porto Alegre – RS, Brasil

Produzido em Uberlândia, MG, Brasil.

CONTRATO DE CESSÃO DE DIREITOS AUTORAIS

DAS PARTES

CESSIONÁRIO: Sociedade Brasileira de Computação, doravante denominada SBC, inscrita no CNPJ sob o nº 29.532.264/0001-78, Inscrição Estadual isenta e CCM sob o nº 18115128, com sede à Av. Bento Gonçalves, 9500 – Setor 4 – Sala 116 – Prédio 43424 – Agronomia – CEP 91501-970 – Porto Alegre – RS, neste ato representada por André Raabe, Coordenador do comitê de programa do evento 27º Simpósio Brasileiro de Informática na Educação (SBIE 2016), 2016. **CEDENTE(S):** Humberto Augusto Piovesana Zanetti, professor, Marcos Augusto Francisco Borges, professor, e Ivan Luiz Marques Ricarte, professor, doravante denominado(s) autor(es).

DO OBJETO

TÍTULO DO ARTIGO (doravante denominado “a obra”): **Pensamento Computacional no Ensino de Programação: Uma Revisão Sistemática da Literatura Brasileira**

NOME DA PUBLICAÇÃO: Anais do XXVII Simpósio Brasileiro de Informática na Educação (Anais do SBIE 2016)

O(s) autor(es) da obra transfere(m) os direitos autorais para a Sociedade Brasileira de Computação (SBC). O(s) autor(es) representa(m) e garante(m) que a obra é original e de sua (vossa) autoria, com a exceção de algumas passagens de texto, figuras e dados cuja a fonte esteja claramente referenciada, e acompanhadas da permissão de reprodução do material emitida pelos detentores dos direitos autorais quando necessário. O(s) autor(es) afirma(m) que ele(s) ou ela(s) têm o poder e autoridade para executar esta tarefa.

A SBC reconhece e retém os direitos descritos nos itens 1 e 2 e permite ao(s) autor(es) a utilização do material segundo os itens 3 e 4.

1. O(s) autor(es) retém todos os direitos proprietários sobre qualquer processo, procedimento ou artigo descrito na obra.
2. Em todos os casos não cobertos pelos itens 3 e 4, o(s) autor(es) devem solicitar permissão para reprodução da obra, incluindo figuras e tabelas, diretamente a SBC.
3. O(s) autor(es) podem reproduzir ou autorizar terceiros a reprodução de parte do material extraído desta obra, material extraído de forma *verbatim* ou derivados para o uso do(s) autor(es), uma vez que a fonte e os direitos da SBC sejam devidamente referenciados. As cópias não devem ser utilizadas de nenhum modo que implique o endosso da SBC. Cópias da obra não podem ser colocadas à venda sem a autorização expressa da SBC.
4. O(s) autor(es) podem distribuir de forma limitada todo ou partes da obra antes de sua publicação, desde que informem a SBC da natureza da utilização, extensão e propósito da distribuição.

DO FORO

Para as questões resultantes deste contrato, será competente o Foro da Cidade de Porto Alegre - RS, renunciando as partes contratantes, desde logo, a qualquer outro, seja qual for a sua sede.

Porto Alegre, 12 de agosto de 2016.

Assinatura Autorizada
Sociedade Brasileira de Computação

Humberto Augusto Piovesana Zanetti

Empregador para o qual a obra foi criada (se aplicável)

AUTORIA CONJUNTA

21/08/2022 22:54

Solicitação de uso de artigo - hzanetti@gmail.com - Gmail

Solicitação de uso de artigo

Caixa de entrada



Humberto Zanetti <hzanetti@gmail.com>

para Lillane

Bom dia,

Logo defenderei meu Doutorado em Tecnologia pela Faculdade de Tecnologia da Universidade Estadual de Campinas (FT-UNICAMP).

Envio este e-mail com o intuito de solicitar autorização da organização da Revista Comunicações em Informática para incorporar o artigo que foi publicado na edição Volume A tese será disponibilizada online na biblioteca digital da UNICAMP em 2023.

O artigo em questão é: ZANETTI, H.A.P.; BORGES, M.A.F.. Ensino de Programação utilizando Computação Física: uma Revisão Sistemática da Literatura. Comunicações e

Agradeço a atenção!

Humberto A. P. Zanetti



Lillane@di.ufpb.br

para mim

Bom dia Humberto,

Sim, o artigo pode ser utilizado e integrado sem problemas na sua tese.

Sucesso na sua defesa!

Lillane S. Machado

Department of Informatics/ Center of Informatics

Coordination of VR and Games area at LabTEVE

Federal University of Paraíba - Brazil

Contact: +55 (83) 3216-7785 ext. 219

<http://www.de.ufpb.br/~labeve>

<http://lattes.cnpq.br/0240551533292579>

De: "Humberto Zanetti" <hzanetti@gmail.com>

Para: "Lillane" <lillane@di.ufpb.br>

Enviadas: Quarta-feira, 9 de fevereiro de 2022 7:48:30

Assunto: Solicitação de uso de artigo

Sobre o Portal

Um pouco da nossa história.

O **Portal de Periódicos da UFPB** objetiva dar visibilidade à produção científica por meio de revistas eletrônicas científicas elaboradas ou gerenciadas por pesquisadores vinculados à UFPB, possibilitando o acesso e uso de informações científicas e tecnológicas, oferecendo suporte aos editores de periódicos da Instituição, em atenção às determinações da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) no que tange a gestão de periódicos científicos.



O Portal de Periódicos Científicos Eletrônicos da UFPB foi inaugurado em 18 de maio de 2006, sendo um dos primeiros do gênero em Universidades Federais brasileiras. Foi viabilizado por meio de uma parceria entre a Pró-Reitoria de Pesquisa e Pós-Graduação da UFPB e o Programa de Pós-Graduação em Ciência da Informação da UFPB (PPGCI/UFPB).

O Portal está vinculado à Editora da UFPB e utiliza o Open Journal Systems (OJS), desenvolvido pelo Public Knowledge Project (PKP), que adota as iniciativas de acesso aberto ao conhecimento.

Serviços

Serviços ofertados pela equipe.

O Portal de Periódicos da UFPB tem como princípio orientar e prestar auxílio aos editores nos seguintes serviços:

- Orientação aos editores quanto ao credenciamento no Portal;
- Auxílio e orientação aos editores quanto à indexação em bases de dados;
- Auxílio e orientação aos editores quanto à utilização do sistema de gerenciamento do Portal na inserção de números retrospectivos;
- Capacitação e prestação de suporte para treinamentos referente a plataforma Open Journal Systems (OJS) para os editores e equipe técnica;
- Orientação aos editores quanto às ética, boas práticas e normas editoriais para publicações periódicas;
- Orientação sobre o surgimento de fontes e recursos de financiamento (editais);
- Orientação e validação do Digital Object Identifier (DOI);
- Orientação sobre utilização e abertura do registro ORCID (Open Researcher and Contributor ID);
- Orientação aos editores da obtenção da numeração do ISSN junto ao IBICT;
- Inclusão do periódico na Rede de Serviços de Preservação Digital (CARINIANA) do IBICT;
- Incubação de periódicos emergentes que precisam obter os requisitos para inclusão no Portal;

Equipe

Conheça a equipe do Portal de Periódicos

25/03/2022 19:47

Portal de Periódicos UFFB



Cassandra Campos

Coordenadora



Fabiana França

Bibliotecária -
Documentalista

Gilvanilson Bernardino

Estágio - Ciência da
Computação

Menu

[Início](#)[Quem Somos](#)[Dúvidas](#)[Periódicos](#)[Contato](#)[Acesso](#)Perguntas
Frequentes[Como me cadastrar?](#)[Como acessar?](#)[Meu artigo foi avaliado?](#)[Como entrar em contato?](#)[Quem é a equipe?](#)

 O conteúdo deste site está licenciado sob uma Licença Internacional Creative Commons Atribuição 4.0.

21/08/2022 22:43

Solicitação para uso de artigo - hzanetti@gmail.com - Gmail

Solicitação para uso de artigo

Caixa de entrada



Humberto Zanetti <hzanetti@gmail.com>
para publicacoes

Bom dia,

Logo defenderei meu Doutorado em Tecnologia pela Faculdade de Tecnologia da Universidade Estadual de Campinas (FT-UNICAMP).

Envio este e-mail com o intuito de solicitar autorização da organização da SBC para incorporar o artigo que foi publicado no evento "Simpósio Brasileiro de Educação em Co A tese será disponibilizada online na biblioteca digital da UNICAMP em 2023.

O artigo em questão é: ZANETTI, H.A.P.; BORGES, M.A.F.. Por que estimular a Aprendizagem Significativa no ensino de Programação Orientada a Objetos?. In: Anais do S

Agradeço a atenção!

Humberto A. P. Zanetti



Publicações <publicacoes@sbc.org.br>
para mim

Prezado Humberto, boa tarde!

Os autores que publicam seus trabalhos na SBC retêm os direitos autorais e não necessitam de nossa autorização para utilizá-los. Entretanto, é importante que seja mencionada nossa política de direitos autorais:

Direitos Autorais

Os autores dos artigos publicados nos Anais Principais do SSB retêm os direitos autorais de suas obras e autorizam a SBC a publicá-las de acordo com os termos. Dessa forma, fica permitido aos autores ou a terceiros a reprodução ou distribuição, em parte ou no todo, de material extraído dessas obras, de forma verbatim, e comerciais, desde que sejam atribuídos os devidos créditos às criações originais. Cópias das obras não devem ser utilizadas de nenhum modo que implique o enc

Um abraço,



Jéssica Paula Macedo Müller
Assistente de Publicações
publicacoes@sbc.org.br
Sociedade Brasileira de Computação
Caixa Postal 15012
91.501-970 - Porto Alegre - RS
Tel.: +55 51 3306-7740 Fax: +55 51 3306-7142



ANAIS DO SIMPÓSIO BRASILEIRO DE EDUCAÇÃO EM COMPUTAÇÃO (EDUCOMP)

SOL ▾ TODAS AS EDIÇÕES SOBRE O EVENTO EXPEDIENTE

Buscar

INÍCIO / **Expediente**

Comissão organizadora do evento

O Simpósio Brasileiro de Educação em Computação é promovido pelo Grupo de Interesse em Educação em Computação (GIEC) da SBC. Esse Grupo de Interesse, que é o responsável geral pela organização do evento e pela publicação dos Anais do EduComp, é constituído por:

- Roberto Almeida Bittencourt (UEFS) - Coordenador
- Esdras Lins Bispo Júnior (UFJ) - Vice-Coordenador
- André Luís Alice Raabe (UNIVALI)
- Ellen Francine Barbosa (ICMC-USP)
- Leandro Silva Galvão de Carvalho (UFAM)
- Rodrigo Silva Duran (IFMS)
- Taciana Pontual da Rocha Falcão (UFRPE)

Para ver os Coordenadores de cada edição do evento e os respectivos integrantes dos Comitês de Programa, que participam do processo de revisão por pares, visite o site do evento.

Periodicidade

Os Anais do EduComp são publicados anualmente.

Idiomas

São aceitos artigos em português ou inglês para publicação nos Anais do EduComp.

Direitos Autorais

Os autores dos artigos publicados nos Anais do EduComp retêm os direitos autorais de suas obras e autorizam a SBC a publicá-las de acordo com os termos da licença Creative Commons Attribution-NonComercial 4.0 International Public License (CC BY-NC 4.0). Dessa forma, fica permitido aos autores ou a terceiros a reprodução ou distribuição, em parte ou no todo, de material extraído dessas obras, de forma verbatim, adaptada ou remixada, bem como a criação ou produção a partir do conteúdo dessas obras, para fins não comerciais, desde que sejam atribuídos os devidos créditos às criações originais. Cópias das obras não devem ser utilizadas de nenhum modo que implique o endosso da SBC.

Responsável pela publicação

José Viterbo Filho | Diretor de Publicações
Sociedade Brasileira de Computação
Av. Bento Gonçalves, 9500 | Setor 4 | Prédio 43.412 | Sala 219
Bairro Agronomia | Porto Alegre - RS | CEP 91501-970
(51) 3308-6835
publicacoes@sbc.org.br

IDIOMA

Português (Brasil)

English

**O conteúdo publicado neste portal representa exclusivamente a opinião de seus autores e não necessariamente a posição da Sociedade Brasileira de Computação – SBC, seus colaboradores e associados. A SBC poderá adotar a qualquer tempo, e sem a necessidade de prévio aviso, a cobrança de uso e disponibilização da plataforma e seu conteúdo para não associados.*



Sociedade Brasileira
de Computação

Av. Bento Gonçalves, 9500 | Setor 4 | Prédio 43.412 | Sala 219 | Bairro Agronomia
Caixa Postal 15012 | CEP 91501-970
Porto Alegre - RS
CNPJ: 29.532.264/0001-78
Fone: (51) 99252-6018
sbc@sbc.org.br





TERMO DE AUTORIZAÇÃO DE PUBLICAÇÃO

DAS PARTES

CESSIONÁRIO: Sociedade Brasileira de Computação, doravante denominada SBC, inscrita no CNPJ sob o no. 29.532.264/0001-78, Inscrição Estadual isenta e Inscrição Municipal nº 181.151.2.8, com sede à Av. Bento Gonçalves, 9500 – Setor 4 – Prédio 43412, salas 217 e 219, bairro Agronomia – CEP 91509-900 – Porto Alegre – RS, neste ato representada por José Viterbo Filho, Diretor de Publicações da SBC.

CEDENTE(S): Humberto Augusto Piovesana Zanetti e Marcos Augusto Francisco Borges, doravante denominado(s) “autor(es)”.

DO OBJETO

TÍTULO: Por que estimular a Aprendizagem Significativa no ensino de Programação Orientada a Objetos?, doravante denominado “a obra”, a ser publicado em

VEÍCULO: Anais do Simpósio Brasileiro de Educação em Computação (EduComp 2021).

DA AUTORIA E ORIGINALIDADE DA OBRA

O(s) autor(es) representa(m) e garante(m) que a obra é original e de sua autoria, com a exceção de algumas passagens de texto, figuras e dados, cujas fontes estejam claramente referenciadas e, quando necessário, acompanhadas da permissão de reprodução do material emitida pelos detentores dos direitos autorais. O(s) autor(es) afirma(m) ainda que a obra não foi formalmente publicada em qualquer outro veículo, ou seja, que não realizaram a cessão dos direitos autorais para outrem, e que a obra não contém nada que seja ilegal, difamatório, ou que, se publicado, constitua-se em violação de compromisso de sigilo.

DA AUTORIZAÇÃO DE PUBLICAÇÃO

O(s) autor(es) da obra autoriza(m) a SBC a publicá-la de acordo com os termos da licença Creative Commons Attribution-NonCommercial 4.0 International Public License (CC BY-NC 4.0). Dessa forma, fica permitido ao(s) autor(es) ou a terceiros a reprodução ou distribuição, em parte ou no todo, do material extraído dessa obra, de forma verbatim, adaptada ou remixada, bem como a criação ou produção a partir do conteúdo dessa obra, para fins não comerciais, desde que sejam atribuídos os devidos créditos à criação original. Cópias da obra não devem ser utilizadas de nenhum modo que implique o endosso da SBC.

DA AUTORIZAÇÃO PARA A COLETA E USO DE METADADOS

O(s) autor(es) da obra autoriza(m) a SBC a coletar os metadados que descrevem a obra e dispor destes com a finalidade de permitir o acesso automático a informações básicas sobre a obra, promover a sua indexação em bases de referência, facilitar a sua localização em mecanismos de busca e compor datasets de acesso público que descrevem o conteúdo geral da biblioteca digital da SBC.

DO FORO

Para as questões resultantes deste contrato, será competente o Foro da Cidade de Porto Alegre - RS, renunciando as partes contratantes, desde logo, a qualquer outro, seja qual for a sua sede.

O autor que firma este contrato garante que é o agente autorizado pelos demais coautores para executar essa tarefa.

Porto Alegre, 22 de janeiro de 2021.

Agente autorizado:

Humberto Augusto Piovesana Zanetti

Diretor de Publicações da SBC:

José Viterbo Filho

21/08/2022 22:52

Solicitação para uso de artigo - hzanetti@gmail.com - Gmail

Solicitação para uso de artigo

Caixa de entrada



Humberto Zanetti <hzanetti@gmail.com>

para Luis

Boa tarde prof. Valente,

Logo defenderei meu Doutorado em Tecnologia pela Faculdade de Tecnologia da Universidade Estadual de Campinas (FT-UNICAMP).

Envio este e-mail com o intuito de solicitar autorização da organização da Challenges 2021 para incorporar o artigo que foi publicado no evento "XII Conferência Internacional de Teoria e Prática da Computação". A tese será disponibilizada online na biblioteca digital da UNICAMP em 2023.

O artigo em questão é: *Método de Ensino de Programação Orientada a Objetos Baseada em Computação Física, Aprendizagem Significativa e Concreticidade* Fadi Marques Ricarte.

Agradeço a atenção!

Humberto A. P. Zanetti



valente@fe.uminho.pt

para mim

Viva, Humberto!

Desculpe a demora na resposta, mas, por lapso, respondi a outro autor 😊.

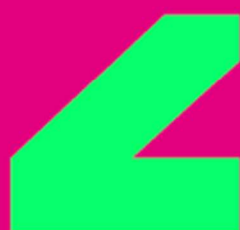
Claro que pode utilizar/publicar o seu artigo. A Challenges não tem nenhum direito exclusivo de publicação.

Desejamos-lhe, todos, muito êxito no seu doutoramento (doutorado).

Tudo de bom!

Luís Valente

XIII Conferência Internacional
de Tecnologias de Informação e Comunicação
na Educação



1999 - 2021

Challenges

2021

Challenges 2021 — Desafios do Digital

Livro de Atas

António José Osório

Maria João Gomes

Alina Ramos

António Luís Valente

Universidade do Minho — Centro de Competência

XII Conferência Internacional de Tecnologias de Informação e Comunicação na Educação
Challenges 2021: Desafios do Digital
(Online), 10 a 17 de setembro de 2021, Universidade do Minho, Braga, Portugal

TÍTULO

Challenges 2021, Desafios do Digital: Livro de Atas

ORGANIZADORES

António José Osório
Maria João Gomes
Altina Ramos
António Luís Valente

EDITORA

Universidade do Minho. Centro de Competência

CAPA

Pedro Miranda

COMPOSIÇÃO

CCTIC-IEUM

LOCAL

Braga, Portugal

ANO

2021



Este trabalho está publicado com uma licença Creative Commons Attribution-ShareAlike 4.0 International.

21/08/2022 22:45

Autorização de uso de artigo - hzanetti@gmail.com - Gmail

Autorização de uso de artigo

Caixa de entrada



Humberto Zanetti <hzanetti@gmail.com>

para Publicações

Boa tarde,

Logo defenderei meu Doutorado em Tecnologia pela Faculdade de Tecnologia da Universidade Estadual de Campinas (FT-UNICAMP).

Envio este e-mail com o intuito de solicitar autorização da organização da SBC para incorporar o artigo que foi publicado no evento "XXX Workshop sobre Educação em Cor". A tese será disponibilizada online na biblioteca digital da UNICAMP em 2023.

O artigo em questão é:

ZANETTI, Humberto A. P.; BORGES, Marcos A. F.; RICARTE, Ivan L. M. Aplicação de um Método para Ensino de Programação Orientada a Objetos por meio de Aprendizagem 2022, Niterói. Anais [...], Porto Alegre: Sociedade Brasileira de Computação, 2022. p. 37-48. ISSN 2595-6175. DOI: <https://doi.org/10.5753/wel.2022.222953>.

Agradeço a atenção!

Humberto A. P. Zanetti



Publicações

para José, mim

Prezado Humberto, boa tarde!

Os autores que publicam seus trabalhos na SBC relem os direitos autorais e não necessitam de nossa autorização para utilizá-los. Entretanto, é importante que seja mencionados os direitos autorais:

Direitos Autorais

Os autores dos artigos publicados nos Anais Principais do SBC relem os direitos autorais de suas obras e autorizam a SBC a publicá-las de acordo com os termos da licença permitida aos autores ou a terceiros a reprodução ou distribuição, em parte ou no todo, de material extraído dessas obras, de forma verbal, adaptada ou remixada, bem como os devidos créditos às criações originais. Cópias das obras não devem ser utilizadas de nenhum modo que implique o endosso da SBC.

Deste modo, confirmo que você pode utilizá-lo com essa finalidade.

Um abraço,



Jessica Paula Macedo Muller
Assistente de Publicações
publicacoes@sbc.org.br
Sociedade Brasileira de Computação
Caixa Postal 15012
91.501-970 - Porto Alegre - RS
Tel.: +55 51 3308-7740 Fax: +55 51 3308-7142



ANAIS DO WORKSHOP SOBRE EDUCAÇÃO EM COMPUTAÇÃO (WEI)

SOL ▾ TODAS AS EDIÇÕES SOBRE O EVENTO EXPEDIENTE

Buscar

INÍCIO / **Expediente**

ISSN: 2595-6175

Comissão organizadora do evento

O Workshop sobre Educação em Computação é organizado pela Diretoria de Educação da SBC. Essa Diretoria, que é a responsável geral pela organização do evento e pela publicação dos Anais do WEI, é constituída por:

- Itana Maria de Souza Gimenes (UEM) - Diretora de Educação
- Vitor Estêvão Silva Souza (UFES)
- Ronaldo Celso Messias Correia (UNESP)
- Jair Cavalcante Leite (UFRN)
- Daltro José Nunes (UFRGS)
- Ig Ibert Bittencourt (UFAL)
- Sílvia Amélia Bim (UTFPR)
- Tayana Conte (UFAM)
- Rodrigo Duran (IFMS)

Para ver os Coordenadores de cada edição do evento e os respectivos integrantes dos Comitês de Programa, que participam do processo de revisão por pares, visite o [site do evento](#).

Periodicidade

Os Anais do WEI são publicados anualmente.

Idiomas

São aceitos artigos em português e inglês para publicação nos Anais do WEI.

Direitos Autorais

Os autores dos artigos publicados nos Anais do WEI retêm os direitos autorais de suas obras e autorizam a SBC a publicá-las de acordo com os termos da licença Creative Commons Attribution-NonCommercial 4.0 International Public License (CC BY-NC 4.0). Dessa forma, fica permitido aos autores ou a terceiros a reprodução ou distribuição, em parte ou no todo, de material extraído dessas obras, de forma verbatim, adaptada ou remixada, bem como a criação ou produção a partir do conteúdo dessas obras, para fins não comerciais, desde que sejam atribuídos os devidos créditos às criações originais. Cópias das obras não devem ser utilizadas de nenhum modo que implique o endosso da SBC.

Responsável pela publicação

Sociedade Brasileira de Computação

Av. Bento Gonçalves, 9500 | Setor 4 | Prédio 43.412 | Sala 219

Bairro Agronomia | Porto Alegre - RS | CEP 91501-970

(51) 3308-6835

publicacoes@sbc.org.br

IDIOMA

Português (Brasil)

English

**O conteúdo publicado neste portal representa exclusivamente a opinião de seus autores e não necessariamente a posição da Sociedade Brasileira de Computação – SBC, seus colaboradores e associados. A SBC poderá adotar a qualquer tempo, e sem a necessidade de prévio aviso, a cobrança de uso e disponibilização da plataforma e seu conteúdo para não associados.*



Sociedade Brasileira
de Computação

Av. Bento Gonçalves, 9500 | Setor 4 | Prédio 43.412 | Sala 219 | Bairro Agronomia

Caixa Postal 15012 | CEP 91501-970

Porto Alegre - RS

CNPJ: 29.532.264/0001-78

24/08/2022 22:25

Expediente | Anais do Workshop sobre Educação em Computação (WEI)

Fone: (51) 99252-6018
sbc@sbc.org.br

PKP | PUBLIC KNOWLEDGE PROJECT powered by OJS | Open Journal Systems



TERMO DE AUTORIZAÇÃO DE PUBLICAÇÃO

DAS PARTES

CESSIONÁRIO: Sociedade Brasileira de Computação, doravante denominada SBC, inscrita no CNPJ sob o nº 29.532.264/0001-78, inscrição Estadual isenta e inscrição Municipal nº 181.151.2.8, com sede à Av. Bento Gonçalves, 9500, Setor 4, Prédio 43412, salas 217 e 219, bairro Agronomia, CEP 91509-900, Porto Alegre – RS, neste ato representada por José Viterbo Filho, Diretor de Publicações da SBC.

CEDENTE(S): Humberto Augusto Piovesana Zanetti, Marcos Augusto Francisco Borges e Ivan Luiz Marques Ricarte
 doravante denominado(s) "autor(es)".

DO OBJETO

TÍTULO: Aplicação de um Método para Ensino de Programação Orientada a Objetos por meio de Aprendizagem Significativa e Computação Física
 doravante denominada "a obra", a ser publicado em
VEÍCULO: Anais do XXX Workshop sobre Educação em Computação (WEI 2022)

DA AUTORIA E ORIGINALIDADE DA OBRA

O(s) autor(es) representa(m) e garante(m) que a obra é original e de sua autoria, com a exceção de algumas passagens de texto, figuras e dados, cujas fontes estejam claramente referenciadas e, quando necessário, acompanhadas da permissão de reprodução do material emitida pelos detentores dos direitos autorais. O(s) autor(es) afirma(m) ainda que a obra não foi formalmente publicada em qualquer outro veículo, ou seja, que não realizaram a cessão dos direitos autorais para outrem, e que a obra não contém nada que seja ilegal, difamatório, ou que, se publicado, constitua-se em violação de compromisso de sigilo.

DA AUTORIZAÇÃO PARA A PUBLICAÇÃO DA OBRA

O(s) autor(es) da obra autoriza(m) a SBC a publicá-la de acordo com os termos da licença Creative Commons Attribution 4.0 International Public License (CC BY 4.0). Dessa forma, fica permitido ao(s) autor(es) ou a terceiros a reprodução ou distribuição, em parte ou no todo, do material extraído dessa obra, de forma verbatim, adaptada ou remixada, bem como a criação ou produção a partir do conteúdo dessa obra, desde que sejam atribuídos os devidos créditos à criação original. Cópias da obra não devem ser utilizadas de nenhum modo que implique o endosso da SBC.

DA AUTORIZAÇÃO PARA A COLETA E USO DE METADADOS

O(s) autor(es) da obra autoriza(m) a SBC a coletar os metadados que descrevem a obra e dispor destes com a finalidade de permitir o acesso automático a informações básicas sobre a obra, promover a sua indexação em bases de referência, facilitar a sua localização em mecanismos de busca e compor datasets de acesso público que descrevem o conteúdo geral da biblioteca digital da SBC.

DO FORO

Para as questões resultantes deste contrato, será competente o Foro da Cidade de Porto Alegre - RS, renunciando as partes contratantes, desde logo, a qualquer outro, seja qual for a sua sede.

O autor que firma este contrato garante que é o agente autorizado pelos demais coautores para executar essa tarefa.

Porto Alegre, 14 de maio de 2022

Agente autorizado:

Diretor de Publicações da SBC:

Nome: Humberto A.P. Zanetti

José Viterbo Filho