

CASSIANO PEREIRA VIANA

**PLATAFORMA DE AVALIAÇÃO DO PENSAMENTO
COMPUTACIONAL POR MEIO DE PUZZLES**

Itajaí (SC), dezembro de 2021



UNIVALI

UNIVERSIDADE DO VALE DO ITAJAÍ
CURSO DE MESTRADO ACADÊMICO EM
COMPUTAÇÃO APLICADA

PLATAFORMA DE AVALIAÇÃO DO PENSAMENTO
COMPUTACIONAL POR MEIO DE PUZZLES

por

Cassiano Pereira Viana

Dissertação apresentada como requisito parcial à
obtenção do grau de Mestre em Computação
Aplicada.

Orientador: André Luis Alice Raabe, Dr.

Itajaí, dezembro de 2021

Dedico este trabalho à minha família.

AGRADECIMENTOS

Sou infinitamente grato a todos que direta ou indiretamente contribuíram para a conclusão desse trabalho. Cito mais especificamente alguns:

Pai e mãe. Pela influência positiva, incentivo e entendimento sobre os momentos de ausência. Sem vocês, nem sei onde estaria.

Cesar, meu irmão. Por ter me auxiliado e ensinado, sempre, desde pequenos até agora, a encarar os problemas e enfrentar os desafios. Por ter testado e usado o projeto e participado ativamente dele.

Professor André Raabe. Pela paciência, incentivo, orientações, pelas mudanças de rumo da pesquisa, e por ter me ensinado uma nova visão sobre educação.

A todos os integrantes do GIE (Grupo de Informática na Educação), com quem nos encontramos todas as segundas-feiras para discutir nossos projetos e pesquisas. Isso foi de grande relevância.

A todos os professores da graduação e do mestrado. Principalmente o professor Aluísio H. Filho e o professor substituto Fábio, que nos ensinaram sobre escrita e que texto vem de têxtil, de tecer.

Aos mestres Leonardo e Aline (do GIE) por terem divulgado e convidado seus alunos a jogar um jogo de programação construído para esta pesquisa. Também a todos do grupo Pensamento Computacional, que compartilharam e divulgaram a pesquisa. E novamente ao professor André, por ter divulgado o experimento, inclusive durante suas palestras.

A todos aqueles que doaram uma parte do seu tempo, para participar, desafiando-se a resolver um puzzle de programação. Notadamente aos alunos dos mestres Leonardo e Aline. Meu sobrinho Pedro também deu uma enorme contribuição, testando e opinando sobre o jogo.

A todos os integrantes do LITE, com quem pude conviver pouco devido a correria do trabalho, mas com quem pude aprender muito, graças a paciência e atenção.

A Luna (nossa felina) pela companhia durante os períodos de desenvolvimento e escrita e a minha companheira Tatiana.

E por último, mas não menos importante, a CAPES e a UNIVALI, pela bolsa de estudos.

PLATAFORMA DE AVALIAÇÃO DO PENSAMENTO COMPUTACIONAL POR MEIO DE PUZZLES

Cassiano Pereira Viana

dezembro / 2021

Orientador: André Luis Alice Raabe, Dr

Área de Concentração: Computação Aplicada

Linha de Pesquisa: Informática na Educação

Palavras-chave: avaliação, Pensamento Computacional, puzzle

Número de páginas: 129

RESUMO

A adoção do Pensamento Computacional (PC) em conteúdos e disciplinas demanda formas de avaliação. Os testes com puzzles incentivam os jogadores a prestarem mais atenção aos erros, facilitando a autoavaliação. Ao analisar o processo seguido pelo aluno na resolução do item de puzzle, é possível medir o nível de habilidade do aluno em relação ao PC, mas essa tarefa não é prática sem uma solução de software específica. A proposta dessa pesquisa é investigar os requisitos para uma plataforma de criação e aplicação de testes com puzzles. A hipótese investigada é a de que a plataforma auxilia o trabalho de avaliação dos conceitos relacionados ao PC. A plataforma permite conectar puzzles com cenários parametrizáveis para a criação dos itens. Além disso explora o uso de Teoria da Resposta ao Item para calcular parâmetros de dificuldade e discriminação. A plataforma foi utilizada durante um experimento realizado com voluntários em 2021. A aplicação dos testes ocorreu pela resolução de puzzles de programação, sem supervisão de tempo, seguida de um questionário. A análise qualitativa das respostas levantou melhorias para o puzzle utilizado no teste. Os resultados demonstraram que a plataforma auxiliou a investigar hipóteses, mas não detalhou os resultados de acordo com os pilares do Pensamento Computacional. A plataforma permitiu integrar ambientes de puzzles com baixo acoplamento, sendo uma diferença em relação a soluções de testes fixos como CTt e soluções baseadas na análise de portfólio como Dr. Scratch. Estudos futuros sobre a plataforma e sua aplicação em ambientes escolares devem contribuir para análise de sua efetividade na geração de informações.

COMPUTATIONAL THINKING ASSESSMENT PLATFORM USING PUZZLES

Cassiano Pereira Viana

December / 2021

Advisor: André Luis Alice Raabe, PhD.

Area of Concentration: Applied Computer Science

Research Line: Informatics in Education

Keywords: evaluation, computational thinking, puzzle

Number of pages: 129

ABSTRACT

The adoption of Computational Thinking (CT) in contents and disciplines requires forms of assessment. Tests involving puzzles encourage players to pay more attention to mistakes, facilitating self-assessment. By analyzing the process followed by the student in solving the puzzle, it is possible to measure the student's skill level in relation to CT, but this task is not practical without a specific software solution. The purpose of this research is to investigate the requirements of a platform for creating and applying tests with puzzles. The hypothesis investigated is that the platform will help in the evaluation of concepts related to CT. The platform enables puzzles with customizable scenarios to be connected, for the creation of items. It also explores the use of Item Response Theory to calculate parameters of difficulty and discrimination. The platform was used during an experiment carried out with volunteers in 2021. The volunteers were asked to solve programming puzzles, without time supervision, followed by a questionnaire. The qualitative analysis of the answers raised improvements for the puzzle used in the test. The results of the tests showed that the platform was useful for investigating the hypotheses, but did not detail the results according to the pillars of Computational Thinking. The platform enabled puzzle environments to be integrated with loose coupling, distinguishing it from fixed test solutions such as CTt and solutions based on portfolio analysis such as Dr. Scratch. Future studies on the platform and its application in school environments should contribute to the analysis of its effectiveness in generating information.

LISTA DE ILUSTRAÇÕES

Figura 1: Mecânicas do CT Puzzle Test.....	16
Figura 2: Esquema da solução proposta.....	18
Figura 3: Popularidade do termo “Computational thinking” de 2004 a 2020 de acordo com pesquisas no Google. A escala é relativa ao momento de maior pesquisa (ponto em que o gráfico chega em 100).....	25
Figura 4: Quatro pilares do Pensamento Computacional.....	27
FFigura 5: Scrape mostrando (A) análise de projeto com muitos blocos utilizados e (B) projeto com poucos blocos utilizados.....	31
Figura 6: Mecânica de programação com fase simples e fase de <i>loop</i> (direita).....	37
Figura 7: Mecânica de ligação de pontos.....	38
Figura 8: Mecânica de encaixe de figuras na figura maior.....	39
Figura 9: Mecânica de Tangram.....	40
Figura 10: Mecânica de reconhecimento de padrões.....	41
Figura 11: Mecânica reconhecimento de padrões de cores, animais e objetos.....	42
Figura 12: Curva característica do item – CCI.....	47
Figura 13: CCI e função informação: comparação de CCI e Função de Informação do Item ao testar diferentes valores para os parâmetros.....	48
Figura 14: Cadastro de curso.....	52
Figura 15: Tela de acompanhamento do curso.....	53
Figura 16: Comparação de desempenho de grupos experimental e de controle em pré-teste e pós-teste.....	54
Figura 17: Bebras Task de detecção de padrões criada com Bebras Lodge. As peças podem ser arrastadas para a posição indicada com “?”.....	56
Figura 18: Questão de sequenciamento do CTt apresentada no cenário <i>maze</i>	58
Figura 19: Questão de depuração do CTt, apresentada no cenário <i>canvas</i>	58
Figura 20: Tela do Dr. Scratch mostrando os resultados da avaliação de um projeto.....	59
Figura 21: Painel de resultados de avaliação do CoTAS.....	60
Figura 22: Diagrama de casos de uso.....	67
Figura 23: Tela de configuração de tipo de puzzle.....	69
Figura 24: Tela de edição da função de escore.....	70
Figura 25: Tela de manutenção de item.....	72
Figura 26: Tela de manutenção de teste com questionário.....	73
Figura 27: Diálogo de início de aplicação.....	74
Figura 28: Resposta recebida pelo puzzle ao requisitar os dados da aplicação.....	75
Figura 29: Tela de acompanhamento da aplicação.....	77
Figura 30: Seletor de aplicação de teste.....	79
Figura 31: Tabela de dados gerais e números sumarizados.....	79
Figura 32: Índices gerais.....	80
Figura 33: Gráfico de categorias por faixa etária e gênero.....	81
Figura 34: Diagramas de caixa comparando escore obtido para itens em aplicação de teste e controle.....	82
Figura 35: Gráficos de TRI.....	83
Figura 36: Tela de entrada de código R e biblioteca de funções.....	84
Figura 37: Visualizador de respostas abertas.....	85
Figura 38: Comparação entre puzzles de programação.....	86

Figura 39: Áreas da interface do puzzle de programação com robô.....	87
Figura 40: Resultado da instanciação do item.....	90
Figura 41: Ambiente do RoPE AR com robô sobre mapa projetado e peças de programação.....	91
Figura 42: Mapas de programação e depuração criados pelo aplicativo com base em itens da plataforma.....	92
Figura 43: Diagrama de componentes.....	93
Figura 44: Visão geral do experimento.....	98
Figura 45: Diagrama de caixa comparando os usos do recurso de depuração entre aplicação de teste e aplicação de controle.....	102
Figura 46: Diagrama de caixa comparando os usos do recurso de lixeira entre aplicação de teste e aplicação de controle para o último item de teste.....	103
Figura 47: Diagrama de caixa comparando o tempo de resolução dos itens entre aplicação de teste e aplicação de controle.....	104
Figura 48: Curvas características dos itens do teste aplicado ao grupo de teste.....	108
Figura 49: Curvas características dos itens do teste aplicado ao grupo de controle.....	109

LISTA DE QUADROS

Quadro 1. Pilares do pensamento computacional com exemplos.....	23
Quadro 2. Processo interativo do pensamento computacional comparado ao processo de resolução de puzzle.....	30
Quadro 3. Análise comparativa do estado da arte das ferramentas de avaliação de PC.....	57
Quadro 4. Funcionalidades implementadas na fase de programação com RoPE.....	83
Quadro 5. Classe de da mecânica de programação.....	85
Quadro 6. Instanciação do item de programação.....	86
Quadro 7. Itens do teste.....	95
Quadro 8. Itens do questionário.....	96
Quadro 9. Ferramentas de desenvolvimento de jogos para web.....	115

LISTA DE ABREVIATURAS E SIGLAS

API	Application programming interface
CSTA	Computer Science Teachers Association
CSV	comma-separated-values
CTt	Computational Thinking Test
QI	Quociente de inteligência
ISTE	International Society for Technology in Education
JSON	JavaScript Object Notation
UNIVALI	Universidade do Vale do Itajaí

SUMÁRIO

1 INTRODUÇÃO.....	14
1.1 PROBLEMA DE PESQUISA.....	17
1.1.1 Solução proposta.....	18
1.1.2 Delimitação de Escopo.....	19
1.1.3 Justificativa.....	20
1.2 OBJETIVOS.....	20
1.2.1 Objetivo Geral.....	21
1.2.2 Objetivos Específicos.....	21
1.3 METODOLOGIA.....	21
1.3.1 Metodologia da Pesquisa.....	21
1.3.2 Procedimentos Metodológicos.....	22
1.4 ESTRUTURA DA DISSERTAÇÃO.....	23
2 FUNDAMENTAÇÃO TEÓRICA.....	25
2.1 PENSAMENTO COMPUTACIONAL.....	25
2.2 AVALIAÇÃO DO PENSAMENTO COMPUTACIONAL.....	29
2.3 PUZZLES E PENSAMENTO COMPUTACIONAL.....	33
2.4 PSICOMETRIA.....	42
2.4.1 Confiabilidade.....	43
2.4.2 Validade.....	44
2.4.3 Modelos da Psicometria.....	45
TRI e Pensamento Computacional.....	49
3 TRABALHOS RELACIONADOS.....	50
3.1 Abstractly.....	50
3.2 Bebras.....	53
3.3 CTt.....	55
3.4 Dr. Scratch.....	57
3.5 CoTAS.....	59
3.6 ANÁLISE COMPARATIVA.....	60
3.7 CONSIDERAÇÕES.....	62
4 DESENVOLVIMENTO.....	64
4.1 CASOS DE USO.....	66
4.1.1 Registro e login.....	68
4.1.2 Manutenção de mecânica.....	68
4.1.2.1 Função de cálculo de escore.....	70
4.1.3 Manutenção de item.....	71
4.1.4 Manutenção de testes.....	72
4.1.5 Aplicação de teste.....	74

4.1.6 Módulo estatístico.....	78
4.2 MECÂNICA DE PROGRAMAÇÃO COM ROBÔ.....	86
4.2.1 Configuração dos itens.....	88
4.3 MECÂNICA DE REALIDADE AUMENTADA.....	90
4.4 COMPONENTES DE SOFTWARE.....	92
4.5 CONSIDERAÇÕES.....	94
5 RESULTADOS.....	95
5.1 TESTE DE DEPURAÇÃO.....	95
5.1.1 Divulgação e contexto de aplicação.....	96
5.1.2 Participantes do teste.....	97
5.1.3 Conteúdo do teste.....	98
5.1.4 Plataforma como apoio à investigação de hipóteses.....	102
5.1.5 Plataforma como apoio de análise qualitativa.....	104
5.1.5.1 Análise qualitativa sobre dificuldade do teste.....	105
5.1.5.2 Análise qualitativa sobre experiência prévia.....	106
5.1.5.3 Análise qualitativa sobre pontos positivos do puzzle de programação...107	
5.1.5.4 Análise qualitativa sobre pontos negativos do puzzle de programação..108	
5.1.6 Análise por TRI: Curvas características dos itens (CCI).....	109
5.2 CONSIDERAÇÕES.....	110
6 CONCLUSÕES.....	112
6.1 CONTRIBUIÇÕES DA DISSERTAÇÃO.....	113
6.2 TRABALHOS FUTUROS.....	114

1 INTRODUÇÃO

A avaliação do Pensamento Computacional (PC), como conhecemos hoje, não era um tema discutido nos anos 60. Naquela década o PC foi fecundado em duas frentes de trabalho. A primeira ocorreu em 1964, com a invenção da linguagem BASIC (*Beginner's All-purpose Symbolic Instruction Code*) por John Kemeny e Thomas Kurtz, destinada ao aprendizado de programação por jovens iniciantes. A outra frente foi liderada por Seymour Papert, que desenvolveu a linguagem LOGO em 1968. Sobre essa linguagem, o pesquisador expressaria, em 1980, as ideias de “pensamento procedural” (KONG; ABELSON, 2019b).

As ideias de Papert viriam a ser renovadas por Wing em 2006. Wing definiu o PC como “uma habilidade fundamental para todos, não somente cientistas da computação”. Esse enfoque para o tema ocorreu num momento de expansão de acesso a tecnologia e internet, o que incentivou a inclusão do PC como disciplina em currículos escolares.

No entanto, as definições de Wing sobre o PC não indicavam dois aspectos essenciais para um currículo escolar: como ensinar e como avaliar o PC. Diante dessa demanda, a Computer Science Teachers Association (CSTA) e International Society for Technology in Education (ISTE) criaram uma definição operacional baseada em competências¹ e atitudes² inerentes do PC. Essa definição norteou a elaboração de recursos (CSTA; ISTE, 2011), mas as estratégias de avaliação do PC ainda permanecem pouco convergentes e derivadas da diversidade de definições e contextos de ensino.

-
- 1
- Organizar e analisar dados logicamente
 - Representar dados através de abstrações, como modelos e simulações
 - Automatizando soluções através do pensamento algorítmico (uma série de etapas ordenadas)
 - Identificar, analisar e implementar possíveis soluções com o objetivo de alcançar o máximo combinação eficiente e eficaz de etapas e recursos
 - Generalizar e transferir esse processo de solução de problemas para uma ampla variedade de problemas
- 2
- Confiança ao lidar com a complexidade
 - Persistência no trabalho com problemas difíceis
 - Tolerância para ambiguidades
 - A capacidade de lidar com problemas em aberto
 - A capacidade de se comunicar e trabalhar com outras pessoas para alcançar um objetivo ou solução comum

Diversos instrumentos de avaliação surgiram em contextos de ensino de PC com jogos e programação. Werner et al. (2012) e Román-González (2015), desenvolveram instrumentos de avaliação baseados em programação em blocos. Moreno-León, Robbles e Román-González (2015) criaram o Dr. Scratch, que avalia abstração e decomposição de problemas, pensamento lógico, sincronização, paralelismo, noções algorítmicas de controle de fluxo, interatividade e representação de dados.

De acordo com Selby et al (2010), o PC é um processo de resolução de problemas. A programação é um meio de expressar o PC durante a resolução. Mais do que aprender a comandar o computador, é importante saber quando é necessário fazê-lo (PAPERT, 1980). A partir dessa compreensão, pesquisas seguiram definições de PC que abordam aspectos além da programação. Esse movimento pode ser observado no trabalho de Brackmann (2017) que utiliza o CTt, um teste com elementos de algoritmos, que cita a definição dos quatro pilares da BBC (2015)³. Já Brennan e Resnick (2012) propuseram um *framework* de PC baseado em conceitos, práticas e perspectivas. Esse *framework* foi aplicado no contexto do Scratch, que é um ambiente de programação em blocos voltado para crianças.

Observou-se que nenhum dos instrumentos citados captura o processo cognitivo, somente a resposta final. O CTt, assim como o teste internacional Bebras (BEBRAS.ORG, 2020a), é composto por questões de múltipla escolha. Isso simplifica o processo de correção, mas não aborda o processo de resolução. Já o *framework* de Brennan e Resnick (2012) e ferramentas inspiradas como Dr. Scratch (MORENO-LEÓN et al, 2015b), ao avaliar projetos prontos, também não observam o raciocínio usado durante a construção dos projetos.

A análise do processo pode ser feita a partir da observação das decisões tomadas pelo respondente. Esses passos são anteriores à definição de uma alternativa correta ou finalização do projeto. Questões de respostas construídas fornecem mais informações psicométricas ao mesmo tempo que oferecem uma experiência mais rica ao examinado (VAN DER LINDEN; HAMBLETON, 1997).

3 A BBC (2015) divide as habilidades do PC em decomposição, abstração, reconhecimento de padrões e algoritmos

A partir dessa percepção, Gonçalves (2015) iniciou o desenvolvimento do CT Puzzle Test, um instrumento que se propõe a medir o PC por meio de puzzles. O instrumento foi posteriormente aprimorado e avaliado por Couto (2019) e Calbusch (2021), que coletou a avaliação de especialistas sobre sua adequação aos pilares do PC. O instrumento possui questões baseadas em atividades clássicas de resolução de problemas, questões do Programa de Enriquecimento Instrumental de Feuerstein (2006) e exercícios de testes de quociente de inteligência (QI) (Figura 1).

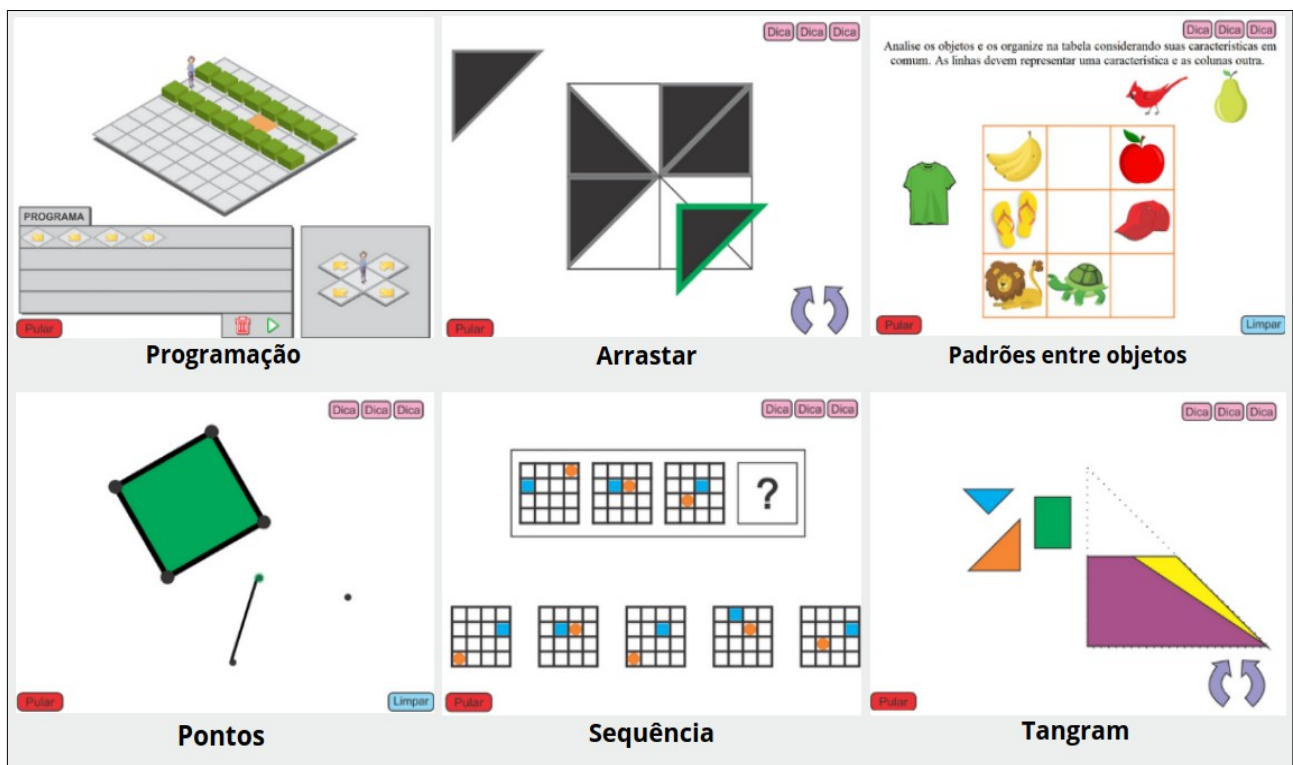


Figura 1: Mecânicas do CT Puzzle Test

Fonte: Autor

No CT Puzzle Test, o PC é mensurado a partir das interações do respondente com o instrumento. São contados (entre outras variáveis) o tempo, número tentativas, desistências, cliques e pedidos de dica. Assim, a avaliação por puzzles do CT Puzzle considera no cálculo dos escores, o processo seguido pelo examinado. Ao contrário de instrumentos baseados em itens dicotômicos⁴, essa abordagem requer a criação de fórmulas de cálculo de escore específicas para cada mecânica.

A ideia de criar um conjunto de itens com uma pontuação máxima para o teste é aderente a Teoria Clássica dos Testes (TCT). Alternativamente, a Teoria da Resposta ao Item (TRI) permite a construção de escalas de habilidades calibradas para itens de respostas. Como vantagem, a TRI

⁴ Itens em que há somente duas opções de resultado: certo ou errado.

considera as propriedades dos itens permitindo elaboração de testes equivalentes e sua comparação ao longo do tempo (ANDRADE; TAVARES, 2000; NERING, 2010).

Nesse contexto, este trabalho apresenta uma solução de apoio para a criação de testes psicométricos baseados em puzzles. Além disso, explora o uso da TRI no contexto de avaliação de PC baseada em puzzles. Para isso foi desenvolvida uma plataforma para formulação de testes e banco de itens, que fornece informações que apoiam a evolução de testes.

1.1 PROBLEMA DE PESQUISA

O CT Puzzle Test é uma ferramenta de avaliação do PC, que tem um conjunto fixo de itens. Isso é útil para o estabelecimento de um teste bem definido. Mas, ao mesmo tempo, o teste não é adaptável a diferentes contextos de aplicação:

- avaliação de indivíduos com diferentes faixas etárias
- avaliações focadas em diferentes conteúdos
- avaliações para tempo de aplicação limitado

Assim, o teste pode ser inadequado para cenários não previstos, subestimando ou superestimando a habilidade do respondente. Também não é possível instanciar duas versões simultâneas do teste. Portanto, o principal problema é que o teste é fixo e não aplicável para diferentes contextos.

A partir disso, elaborou-se a seguinte pergunta de pesquisa:

Como criar uma plataforma para auxiliar a avaliar o PC utilizando puzzles?

1.1.1 Solução proposta

A solução consiste de uma plataforma para criação, aplicação e análise de testes de PC baseados em *puzzles* (jogos de resolução de problemas). A plataforma permite criar testes distintos

dispensando a necessidade de alteração direta de código fonte dos puzzles. Os testes podem então ser aplicados e reproduzidos em diferentes momentos e amostras, facilitando a reprodução dos experimentos. Por fim, durante a aplicação dos testes, a interação dos participantes com os puzzles é armazenada. A partir desses registros, é possível gerar estatísticas de desempenho dos participantes e qualidade dos itens (Figura 2).

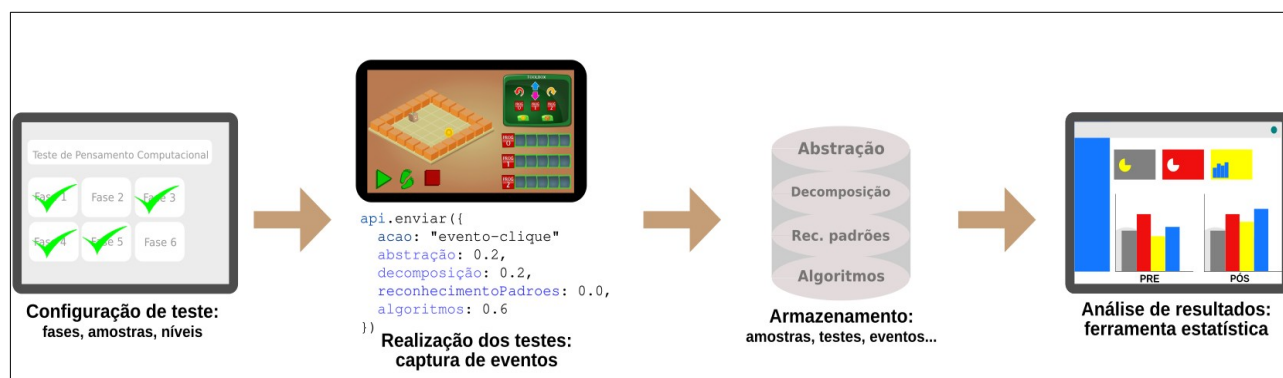


Figura 2: Esquema da solução proposta

Fonte: Autor

Para criar os testes, são utilizados puzzles parametrizáveis⁵. A partir deles, é possível configurar diferentes itens e armazená-los em uma base de itens. Para a definição de puzzles parametrizáveis, foi analisado O CT Puzzle Test, que durante seu histórico de desenvolvimento, teve seu código aprimorado. Contudo, foi observado um alto acoplamento entre as partes do código, dificultando a refatoração para a criação dos itens separados. Por isso, surgiu a necessidade de criar uma mecânica separada, escolhendo-se para isso, a mecânica de programação⁶.

A aplicação dos testes pode ser feita mais que uma vez, permitindo a análise de pré e pós testes e a comparação entre grupos de teste e controle. O acesso às aplicações é feito através de um endereço único, que leva o participante diretamente ao teste, sem necessidade de identificação. Ao

5 Cada puzzle possui um modelo de configuração específico registrado na plataforma. Essa configuração, tem uma estrutura de dados livre, definida pelo usuário. Assim, o criador da mecânica deve definir os atributos apropriados ao puzzle. Ao receber diferentes valores, esses atributos permitem configurar diferentes cenários (itens de teste). Já a tarefa de mostrar o cenário para o participante, fica a cargo do ambiente de puzzle, separado da plataforma.

6 A nova mecânica de programação, teve diversas alterações em relação à mecânica original do CT Puzzle Test. A fase original é apresentada no capítulo 2.3.1.1 e a nova mecânica é apresentada no capítulo 4.2. Essas alterações foram embasadas na análise de mecânicas de programação diversas, coletadas por alunos de graduação em computação.

final de cada teste, um questionário é apresentado. Dados pessoais não são requisitados pela plataforma. Mas isso pode ser feito através do questionário se o usuário criador do teste desejar.

Para tornar possível análise dos dados de interação pela plataforma, cada puzzle deve respeitar regras definidas por meio de API (Application Programming Interface). Ao receber os dados formatados, a plataforma permite visualizar os dados das respostas em uma tabela, gerar gráficos, índices de confiança dos testes e informações sobre a dificuldade dos itens. Para outros cálculos, pode-se exportar os dados e analisar em ferramenta externa.

Portanto, a solução engloba o fluxo de criação de testes, aplicação e análise, dependendo, para isso, da integração com puzzles parametrizáveis, que são desenvolvidos separadamente. Essa flexibilidade elimina a necessidade de alteração direta no código fonte dos puzzles, o que seria necessário para criação de testes diferenciados. Além disso, a definição da comunicação por meio de uma API permite coletar dados em formato próprio para análises estatísticas e para o cálculo do score do participante.

1.1.2 Delimitação de Escopo

O foco do trabalho é desenvolver a plataforma de criação de testes via itens parametrizáveis. Contudo não foram adaptados ou desenvolvidos puzzles para operacionalizar todos os quatro pilares do PC. Ao invés disso, um tipo de puzzle de programação – inspirado no que existente nas versões de Gonçalves, Couto e Calbusch – foi desenvolvido para validação da proposta da plataforma: permitir a criação de testes a partir de itens parametrizáveis.

A aplicação dos testes em grandes amostras de indivíduos não ocorreu. Foi realizada uma aplicação em amostra de conveniência e para a ilustração das aplicações da TRI, foram gerados dados simulados de resposta.

1.1.3 Justificativa

Avaliar do PC de forma consistente é difícil, pois não há um consenso sobre o tema na comunidade científica (KURSHAN, 2016). Para Grover e Pea (2013), sem a atenção para a

avaliação, existe pouca esperança em realizar com sucesso a incorporação do PC nos currículos da educação básica. Neste sentido, existe a necessidade de desenvolvimento e aprimoramento de ferramentas de avaliação.

A avaliação por puzzles se apresenta como uma alternativa aos métodos de avaliação convencionais. Os puzzles do CT Puzzle Test atraem a atenção do aluno e, ao mesmo tempo, operacionalizam as habilidades do PC. No caso da plataforma de avaliação, as vantagens são relacionadas ao ajuste do teste, aplicação e análise dos resultados:

1. Flexibilidade de configuração das fases: com a possibilidade de configurar diferentes fases, será possível construir testes maiores mais rapidamente, eliminando itens pouco informativos. Assim, será mais fácil criar um teste confiável, uma vez que é possível gerar e avaliar itens.
2. Visibilidade dos dados: o CT Puzzle Test atual não traz dados da eficiência dos itens, e desempenho dos alunos. A adição dessa funcionalidade visível em um painel da ferramenta deve facilitar o acompanhamento da qualidade dos testes e do desempenho dos estudantes.

Por fim, diferentes autores pesquisaram a utilização de TRI para itens dicotômicos sobre instrumentos conhecidos como CTt e Bebras (ARAUJO et al., 2018; WIEBE et al., 2019) Mas não foram encontrados trabalhos com uso de TRI para itens politômicos, que é o caso dos itens do CT Puzzle Test.

1.2 OBJETIVOS

A seguir são descritos o objetivo geral e os objetivos específicos desta pesquisa.

1.2.1 Objetivo Geral

Construir uma plataforma para a avaliação do PC a partir da criação de testes baseados em puzzles interativos favorecendo a evolução dos itens a partir da análise de dados.

1.2.2 Objetivos Específicos

1. Criar uma mecânica de jogo de programar parametrizável para o CT Puzzle Test.
2. Implementar a plataforma de parametrização de testes baseados em puzzle.
3. Implementar um módulo para avaliação dos testes usando funções de psicometria.
4. Especificar e aplicar um teste de programação a partir da plataforma.

1.3 METODOLOGIA

A presente seção apresenta o método utilizado na pesquisa e também os procedimentos metodológicos utilizados para desenvolver a solução proposta.

1.3.1 Metodologia da Pesquisa

A pesquisa levanta a hipótese sobre a adequação da abordagem por puzzles aos modelos da TRI. Partindo desse pressuposto pode-se classificar a pesquisa como hipotético-dedutiva.

Uma vez que foi desenvolvida uma plataforma para criar testes de (PC), percebe-se, portanto, que esta pesquisa é de natureza aplicada. Mas também é exploratória, ao propor explorar o uso da TRI sobre itens de puzzles.

Sob o ponto de vista da abordagem do problema, a pesquisa é qualitativa e quantitativa: qualitativa pois foram analisados qualitativamente os elementos de software da plataforma, do ponto de vista de pesquisadores; e quantitativa pois, os dados gerados pela plataforma são de natureza quantitativa.

1.3.2 Procedimentos Metodológicos

Esta seção apresenta como o trabalho foi desenvolvido para atingir os objetivos. Os itens demonstram o caminho percorrido para a definição e construção da solução proposta, desde a base teórica até o desenvolvimento e aplicação.

1. Leitura dos trabalhos anteriores: nesta etapa foram lidos os trabalhos de Gonçalves (2015) e Couto (2018).
2. Pesquisa bibliográfica: pesquisou-se em bases de artigos com as palavras de busca “computational thinking assessment”. Os seguintes resultados foram obtidos para a busca nas bases: Google Scholar (235 resultados), ACM (23 resultados), IEEE *Xplore* (2 resultados), ERIC (2 resultados) e ScienceDirect (4 resultados).
3. Levantamento de requisitos: três pesquisadores testaram uma versão do CT Puzzle Test, sugerindo melhorias em cada tipo de puzzle. Essa análise produziu um quadro, que guiou melhorias no CT Puzzle Test e a criação de uma nova versão do puzzle de programação.
4. Análise de jogos de programar: analisou-se uma lista de jogos de programar coletada por alunos das disciplinas de informática na educação. Características dos jogos - em particular, o jogo Cargo Bot (APÊNDICE C – CARGO BOT) - influenciaram o desenvolvimento do puzzle de programação apresentado no capítulo 4.2.
5. Escolha de ferramenta de desenvolvimento: para implementar o jogo de programação, buscou-se motores, frameworks e bibliotecas de desenvolvimento de jogos para web (APÊNDICE B – Ferramentas analisadas para desenvolvimento do puzzle). A opção pela ferramenta Phaser 3⁷ justificou-se pela popularidade no GitHub (30.9k) e por ser de código aberto e poder ser utilizada para desenvolvimento em sistemas operacionais e ambientes de desenvolvimento comuns.
6. Desenvolvimento do jogo de programar: em posse da ferramenta de desenvolvimento, iniciou-se a escrita do código do jogo. O jogo foi construído iterativamente com coleta de opiniões de usuários a cada versão. Essa etapa atendeu o objetivo específico 1.
7. Implementação da plataforma: essa etapa atendeu ao objetivo específico 2. Com base na análise de outras ferramentas, a plataforma priorizou o baixo acoplamento, permitindo a comunicação de outros de jogos com a plataforma.
8. Criação e aplicação de teste: a fim de testar a plataforma, foram criados testes e enviados para participantes. A coleta de dados ocorreu em duas etapas: uma para homologação do

7 <https://www.phaser.io/phaser3>

teste e outra para aplicação real. Os testes seguiram o modelo de grupo de teste e controle. Essa etapa atendeu o objetivo específico 4.

9. Desenvolvimento de módulo de análises estatísticas: em posse dos dados coletados, a tela de análise estatística da plataforma foi desenvolvida, procurando disponibilizar um conjunto de funções estatísticas para análise dos testes. Essa tarefa permitiu concluir o objetivo específico 3.

1.4 ESTRUTURA DA DISSERTAÇÃO

O trabalho está organizado em quatro capítulos. O capítulo 1 (introdução), apresentou por meio de sua contextualização, o tema proposto neste trabalho. Da mesma forma, estão estabelecidos os resultados esperados por meio da definição de seus objetivos. Depois, definiu os limites do trabalho permitindo uma visão clara do escopo. Por fim, a metodologia e os passos utilizados na pesquisa foram apresentados.

O capítulo 2 (fundamentação teórica) apresenta a fundamentação teórica. Traz os seguintes temas: Pensamento Computacional, avaliação do PC e métricas de avaliação e uma introdução à TRI. Com isso, busca-se apresentar algumas definições e dados sobre o PC bem como as métricas e teorias sobre a avaliação.

O capítulo 3 (trabalhos relacionados) apresenta os seguintes trabalhos: Abstractly, CTt, Bebras e Dr. Scratch e CoTAS. Analisa os tipos de questões que apresentam, classificação etária e quais aspectos do PC cada um avalia. Depois compara os trabalhos entre si e com a solução proposta.

O capítulo 4 (desenvolvimento) apresenta o desenvolvimento da plataforma de testes e do puzzle parametrizável. Traz uma lista de requisitos em forma de caso de uso, protótipos de tela e diagramas que ilustram a estrutura da solução.

O capítulo 5 (resultados) apresenta um teste criado e aplicado a partir da plataforma como meio de validação do seu funcionamento. Descreve a estrutura, contexto de aplicação e dados sobre

a amostra. Por fim, apresenta os resultados dos testes e a análise qualitativa realizada sobre o instrumento de coleta.

2 FUNDAMENTAÇÃO TEÓRICA

Esse capítulo apresenta as bases teóricas para o desenvolvimento do trabalho. Ele está dividido em três sessões. A primeira seção (2.1) apresenta o termo *Pensamento Computacional*, seu histórico e definições. A segunda seção (2.2) discorre sobre o tema da avaliação do Pensamento Computacional, sua importância e as principais abordagens. A terceira seção (2.3) apresenta brevemente as relações entre puzzles e Pensamento Computacional, com casos de uso. Por fim, a quarta seção (2.4) traz dois temas. Primeiramente apresenta as métricas e conceitos utilizados para desenvolvimento de testes válidos e confiáveis. Depois introduz alguns conceitos relacionados à Teoria de Resposta ao Item.

2.1 PENSAMENTO COMPUTACIONAL

O Pensamento Computacional é um tema que desperta cada vez mais interesse (Figura 3). Antes de 2006, havia pouca consistência no número de pesquisa pelo termo. Nesse mesmo ano, após a publicação de Wing, as buscas aumentaram e mantiveram uma tendência média até 2013. Neste ano, após o discurso de apoio do presidente americano Barack Obama⁸ ao aprendizado de Ciência da Computação por jovens na educação básica, as pesquisas tomaram uma tendência de alta que continua até hoje.

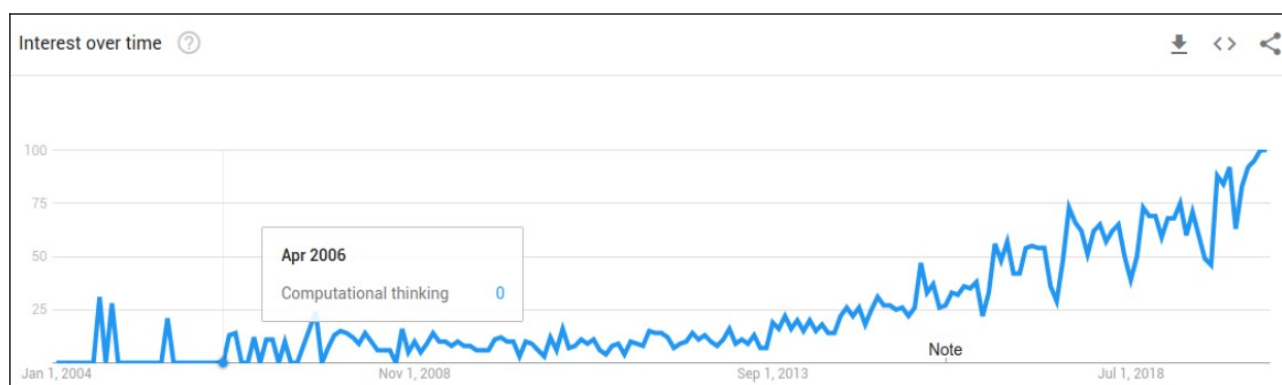


Figura 3: Popularidade do termo “Computational thinking” de 2004 a 2020 de acordo com pesquisas no Google. A escala é relativa ao momento de maior pesquisa (ponto em que o gráfico chega em 100).

Fonte: (GOOGLE, 2020)

⁸ Vídeo do discurso do presidente Barack Obama: <https://www.youtube.com/watch?v=6XvmhE1J9PY>

No entanto, o termo PC ainda não recebeu uma definição comum amplamente aceita pela comunidade científica. Papert (1980) argumenta sobre “pensamento procedural”, como uma “ferramenta intelectual”, uma estratégia para resolver problemas de forma análoga aos computadores. Wing (2006) diz que o PC é “uma habilidade fundamental, para todos, não só para cientistas da Computação. Como a leitura, escrita e matemática”, envolve “coragem para resolver problemas”, que é um “conjunto de ferramentas mentais”, presentes na ciência da computação.

Já, a International Society for Technology in Education (ISTE) e a Computer Science Teachers Association (CSTA) (2011) buscaram uma definição operacional, que pudesse ser incorporada em sala de aula. Segundo essas instituições, o PC é uma processo de resolução de problemas que inclui (mas não se limita a) capacidade de formular problemas para serem resolvidos com auxílio do computador; coletar, analisar e representar de dados; automatizar tarefas usando algoritmos; pensar e criar soluções considerando o uso eficiente de recursos e; generalizar soluções para problemas variados.

Outras definições trazem mais a ideia do uso do computador e dos fundamentos da Ciência da Computação. Kurshan (2016) argumenta que o PC envolve “utilizar os fundamentos da Computação [...] com a finalidade de identificar e resolver problemas [...] através de passos claros, de tal forma que uma pessoa ou máquina possam executá-los”. Blikstein (2018) falava em “saber usar o computador como um instrumento de aumento do poder cognitivo e operacional humano” identificando os problemas e criando soluções automáticas para realizar as tarefas cognitivas identificadas. A definição de Wing em 2011 traz no conceito de PC a ideia de “agente” ao definir PC como “formular problemas e suas soluções representando-as em uma forma que possam ser efetivamente realizadas por um agente de processamento de informações”.

Mas a inclusão da figura “computador” nas definições como uma ferramenta indispensável para o PC, é cada vez menos comum. Ao mesmo tempo, o PC é cada vez mais entendido como um processo de uso de um conjunto de ferramentas para a resolução de questões não só da Ciência da Computação. A BBC (2015), cita que embora computadores possam ser usados para nos ajudar a resolver problemas, antes de abordar como usar essas ferramentas, deve-se buscar como resolver o problema em si. Este processo de busca e organização das ideias é o PC: ele “nos permite pegar um

problema complexo, entender o que é e desenvolver possíveis soluções”. Então nós podemos apresentar as soluções em uma forma que um computador, humano ou ambos possam entender.

Para isso, o PC é dividido em quatro pilares interdependentes. São eles: decomposição, reconhecimento de padrões, abstração e algoritmos (Figura 4).

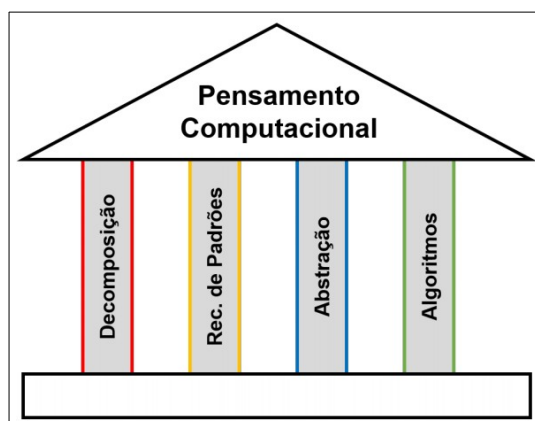


Figura 4: Quatro pilares do Pensamento Computacional

Fonte: (BRACKMANN, 2017)

Ainda segundo a BBC (2015), o Pensamento Computacional envolve pegar um problema complexo e quebrar ele em partes menores, mais controláveis (decomposição). Cada uma dessas partes pode então ser olhada individualmente, considerando problemas similares que foram resolvidos previamente (reconhecimento de padrões) e focando somente nos detalhes mais importantes (abstração) ignorando informações irrelevantes. Depois, passos simples ou regras para resolver cada parte pequena podem ser desenvolvidas (algoritmos). Por último, esses passos simples podem então ser concretizados na forma de um programa de computador para ajudar a resolver o problema complexo automaticamente. O Quadro 1 traz exemplos de cada dimensão.

Quadro 1. Pilares do Pensamento Computacional com exemplos

Pilar	Definição	Exemplo
Decomposição	A decomposição é a divisão de problemas complexos em partes menores, manejáveis e fáceis de entender.	Resolução de um crime: Pode ser decomposto em: - Qual crime foi cometido? - Onde aconteceu? - Quando aconteceu?

		<ul style="list-style-type: none"> - Houve testemunhas? - Foi encontrada alguma arma? - A cena do crime foi alterada? - Há evidências? - Ocorreram crimes parecidos recentemente?
Reconhecimento de padrões	Padrões são características que se repetem. Reconhecer padrões em problemas ajuda a entender como soluções podem ser aplicadas de forma comum, mais eficientemente.	<p>Fazer bolos: Os padrões para fazer bolos (2 diferentes por exemplo) podem ser, por exemplo:</p> <ul style="list-style-type: none"> - <i>O forno deve ser preaquecido</i> (um em 180° e o outro em 190°) - <i>Os ingredientes devem ser misturados</i> (em um vai morango, e em outro vai chocolate) - <i>Assar por um determinado tempo</i> (um por 30 minutos e outro por 20)
Abstração	Abstrair significa ignorar os detalhes dos padrões encontrados. Isso ajuda a criar uma ideia do problema. Essa ideia é chamada de modelo. Sem a abstração, podemos acabar confusos com os detalhes irrelevantes para resolver o problema real.	<p>Desenhar gatos: os gatos podem ter, por exemplo, pelo <i>preto</i>, olhos <i>verdes</i>, rabo <i>longo</i>, e tamanho <i>grande</i>. Ou pelo <i>branco</i>, olhos <i>azuis</i>, rabo <i>longo</i>, tamanho <i>pequeno</i>. Desconsiderar os detalhes (verde, azul, pequeno, grande etc) e se concentrar em desenhar o rabo, olhos, patas, cabeça etc (tudo o que for padrão), é o processo de abstração.</p>
Algoritmos	Algoritmo é um passo a passo para resolução de um problema. Cada passo deve ser identificado e ordenado. Um algoritmo deve ter um ponto inicial, um (ou mais) pontos finais e um conjunto de instruções claras entre eles.	<p>Escovar os dentes:</p> <ol style="list-style-type: none"> 1 – Pegar a escova 2 – Pegar a pasta 3 – Tentar passar pasta na escova 3.1 – Se não saiu pasta: <ol style="list-style-type: none"> 3.1.1 – Pegar outro tubo 3.1.2 – Voltar ao passo 3 4 – Escovar os dentes 5 – Enxaguar a boca 6 – Lavar a escova 7 – Guardar a escova e pasta <p>Esses passos poderiam ser ainda mais detalhados: <i>abrir o tubo da pasta, escovar os dentes superiores, inferiores e língua</i>. Mas o processo de abstração eliminou a necessidade de focar nos detalhes irrelevantes para o entendimento do problema geral.</p>

Fonte: adaptado de (BBC, 2015)

Selby et al (2010) argumentam que uma definição única pode ser menos importante do que entender como o PC é ensinado e aprendido. Contudo, uma definição é

necessária para facilitar o currículo e avaliações apropriadas. Por isso, ao pesquisar a frequência com que termos utilizados para definição de PC aparecem, foram levantadas as seguintes expressões: processo de pensamento, abstração, decomposição, pensamento algorítmico, avaliação, depuração e generalização (SELBY et al., 2010; SHUTE; SUN; ASBELL-CLARKE, 2017).

Propondo uma definição convergente e que é a adotada por este trabalho, Raabe et al (2020) fundamentam a capacidade de resolução de problemas sobre em quatro pilares destacando as práticas de construção, uso e avaliação. Assim, definem o PC como sendo “uma extensão da capacidade humana de resolução de problemas, que se fundamenta nos conceitos e práticas relacionados à construção, uso e avaliação de algoritmos, no reconhecimento de padrões, na decomposição de problemas e na abstração”.

2.2 AVALIAÇÃO DO PENSAMENTO COMPUTACIONAL

Com a contribuição de vários autores, o PC emergiu como um termo guarda-chuva sobre um conjunto de habilidades para resolução de problemas. O entendimento de que essas habilidades são importantes em um mundo influenciado pela tecnologia incentivou a adoção do PC nos currículos escolares (KONG; ABELSON, 2019a). Nesse momento surgiu a necessidade de medir o PC para posicioná-lo ao lado dos outros conteúdos. Isso incentivou o desenvolvimento de instrumentos de avaliação que, dada a variedade de definições de PC, adotam diferentes estratégias.

Em geral, as abordagens de avaliação podem ser separadas em quatro grupos: (1) avaliação por métodos tradicionais, (2) análise de portfólio, (3) entrevistas e (4) pesquisas. Os métodos tradicionais usam questões de múltipla escolha e questões descritivas sobre as competências de PC. A análise de portfólio avalia os constructos produzidos pelos alunos (programas do Scratch e Alice, por exemplo) procurando pelas aplicações de conceitos do PC utilizando checklists e rubricas de classificação⁹ (Dr. Scratch, de Moreno-León et al (2015) é um exemplo). As entrevistas seguem protocolos que buscam verificar o entendimento dos alunos sobre conceitos de PC. E as pesquisas são análises exploratórias que procuram identificar como o PC é aplicado em diferentes contextos (TANG et al., 2020).

9 As rubricas de classificação são semelhantes às checklists, mas no contexto de avaliação de PC, além de indicar se o aluno atingiu determinada tarefa, marca qual o nível de proficiência naquele item.

Quase um quarto dos trabalhos encontrados na revisão de Tang et al. (2020), traz a programação e Ciência da Computação como elementos chave, indicando uma relação forte entre PC e essas duas áreas. Contudo, essa intersecção forte não é compatível com as posições de Papert (1980) e Wing (2006), uma vez que esses autores citam o computador e a programação como meios através dos quais o PC se manifesta e não como sendo PC em si. Ainda assim, a maioria dos estudos não demonstrou como o uso de programação pode apoiar a avaliação de PC em outros contextos. Por isso, “mais pesquisas precisam estudar como a avaliação da PC pode ser aplicada a uma variedade maior de conteúdos” (TANG et al., 2020, p. 12).

Um trabalho de avaliação de PC relacionado com programação é o framework de Brennan e Resnick (2012). Os autores utilizaram projetos do Scratch¹⁰, para avaliar o PC sobre aspectos de conceitos, práticas e perspectivas. Os conceitos são as funções básicas de como funciona a linguagem de programação: sequências, repetições, eventos paralelismo, condicionais, operadores, dados etc. As práticas são as estratégias de desenvolvimento: desenvolvimento incremental; teste e depuração; reutilização e remodelagem; e abstração e modularização. As perspectivas são mais abrangentes e se relacionam com valores, motivações e atitudes: expressão; conexão com outras pessoas; e questionamento sobre a solução e o mundo a partir da tecnologia.

A partir dessas três visões, os autores testaram três abordagens de avaliação: análise de portfólio, entrevista a partir de artefatos e design de cenários. Nenhuma abordagem é completa para avaliação ampla de PC. Por isso o autor argumenta que devem ser utilizadas em conjunto:

- Análise de portfólio: É uma avaliação automática dos projetos Scratch do aluno. Um programa chamado Scrape varre todos os projetos contando a frequência de uso de cada tipo de bloco (Figura 5). Segundo os autores isso pode indicar quais conceitos o aluno domina ou não e também dar uma visão geral do seu progresso. A vantagem dessa abordagem é a rapidez da avaliação. Uma desvantagem é a falta de garantia de que o aluno realmente entendeu o que programou, mesmo utilizando muitos blocos diferentes.

10 O Scratch é um ambiente de programação em blocos que permite criar histórias, games, e animações e compartilhá-las online. <https://scratch.mit.edu>

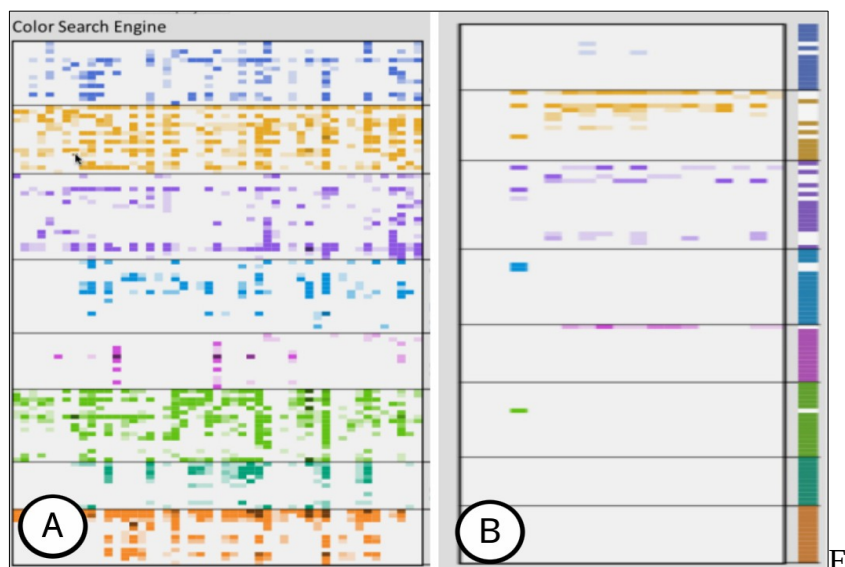


Figura 5: Scrape mostrando (A) análise de projeto com muitos blocos utilizados e (B) projeto com poucos blocos utilizados

Fonte: (BRENNAN; RESNICK, 2012)

- Entrevista baseada em artefatos: Nessa abordagem, os autores selecionaram aleatoriamente participantes e perguntaram sobre a forma como seus projetos foram feitos. As perguntas são gerais procurando entender como o aluno iniciou no Scratch, como criou seus projetos (*o que você faz quando está travado em um erro?*) como participa da comunidade, o gosta não gosta sobre a ferramenta. É possível observar aqui o foco na ferramenta e não propriamente no PC. Uma vantagem da abordagem de entrevista direta é verificar se o aluno realmente entendeu os conceitos que aplicou em seus próprios projetos. Como pontos negativos, (1) esse tipo de avaliação demanda muito tempo e (2) captura somente aquilo que o aluno expressa.
- Design de cenários: Nessa abordagem os pesquisadores apresentaram aos alunos, cenários iniciados no formato de programas Scratch. A tarefa envolvia (1) escolher um cenário (2) descrever seu funcionamento (3) identificar e corrigir erros e (4) estender o projeto com alguma característica nova. Como ponto positivo, essa abordagem captura (em tempo real) as estratégias de resolução dos problemas. Contudo, consome mais tempo que processos de avaliação automatizados.

Por fim, os autores argumentaram que uma combinação de diferentes abordagens de avaliação seria o mais adequado para avaliar todos os aspectos do PC (conceitos, práticas e perspectivas). Mas não deixam claro como isso poderia ser feito em larga escala, ou para outros contextos que não envolvessem o ambiente de programação Scratch.

Independente do uso ou não de programação para avaliar o PC, existem diferentes processos e abordagens para coleta e análise dos dados. Román-González et al (2019) classificam as ferramentas de avaliação do PC, de acordo com seis tipos de abordagens:

1. Ferramentas de diagnóstico de PC: medem a amplitude de conhecimento sobre certa habilidade de PC. Podem ser utilizadas em pré-testes e pós-testes sem a necessidade de experiência prévia. O CTt, de Román-González (2015), citado no capítulo 3.3, é um exemplo dessa abordagem.
2. Ferramentas de PC somativas: verificam se o aluno atingiu um determinado grau de conhecimento ou se consegue realizar uma tarefa após receber instruções com base nas habilidades do PC. O Fairy Assessment, de Werner et al (2012), é um exemplo desse tipo de ferramenta.
3. Ferramentas formativo iterativas: o objetivo dessas ferramentas é prover retorno imediato da performance para o aluno. Geralmente a avaliação é feita sobre os projetos. Um exemplo de ferramenta que usa essa abordagem é o Dr. Scratch (MORENO-LEÓN et al, 2015), citado no capítulo 3.4.
4. Ferramentas de mineração de dados de PC: essas ferramentas são focadas no processo de aprendizado. Enquanto as ferramentas formativo iterativas analisam estaticamente o código-fonte, as ferramentas de mineração de dados recuperam e armazenam a atividade do aprendiz em tempo real. A partir desses dados o processo cognitivo pode ser inferido e conceitos mal compreendidos podem ser detectados.

5. Ferramentas de transferência de habilidade do PC: Avaliam o quanto os alunos são capazes de transferir habilidades do PC para diferentes contextos. Um exemplo é o Bebras Task (BEBRAS.ORG, 2020) (capítulo 3.2), um teste que avalia como as habilidades do PC são aplicadas em problemas reais.
6. Escalas de percepção do PC: são autoavaliações dos alunos frente às habilidades de PC. São administráveis antes e depois de intervenções para ensino. Computational Thinking Scale (CTS), de é um exemplo (KORKMAZ et al, 2017).

De maneira mais geral, é possível dividir os trabalhos de avaliação de PC entre aqueles que usam métricas específicas, baseadas nos próprios contextos (ambientes de programação específicos) e aqueles que tentam criar modelos genéricos (escalas e testes). Os primeiros são mais restritos por obrigar o teste utilizar um instrumento único, que pode tornar-se defasado ao longo do tempo e não ser aplicável em diferentes contextos. Os últimos são mais recentes e revelam a preocupação em desenvolver testes confiáveis e reproduzíveis. Essa abordagem permite identificar a efetividade das intervenções para ensino de PC e auxilia na geração de informações sobre o progresso do aluno ao longo do tempo (SHUTE; SUN; ASBELL-CLARKE, 2017, p. 150).

2.3 PUZZLES E PENSAMENTO COMPUTACIONAL

Puzzles são definidos como “jogos ou atividades de juntar peças e responder questões utilizando habilidades” ou “jogos em que é necessário pensar cuidadosamente para responder” (CAMBRIDGE, 2020; OXFORD, 2020). *Pensar e habilidade* são palavras recorrentes nas definições de Pensamento Computacional.

Rowe et al. (2017) mapearam as dimensões cognitivas do Pensamento Computacional à progressão interativa operacionalizada em puzzles. Da mesma forma, Law (2016) argumenta que, diante de puzzles, os alunos consideram diferentes opções de solução, decompondo o desafio, identificando padrões, abstraindo informações irrelevantes e desenvolvendo algoritmos adequados.

Quadro 2. Processo interativo do Pensamento Computacional comparado ao processo de resolução de puzzle

Decomposição → Rec. Padrões → Abstração → Algoritmos
Tentativa e erro → Teste sistemático → Solução funcional → Solução completa → Solução geral

Fonte: (ROWE et al., 2017)

Os puzzles são problemas de escopo limitado e auto contido, ou seja, contém todas as informações necessárias para resolvê-lo. Por isso, são mais distantes de situações de escopo aberto (mundo real) e mais próximos de situações abstratas (mundo abstrato). Ainda assim, o aprendizado baseado em puzzles, propicia uma base sólida sobre a qual construir habilidades de resolução de problemas de mais alto nível (LAW, 2016).

Os puzzles têm ao menos dois componentes principais: (i) momentos de descoberta e (ii) fatores de entretenimento. O momento de descoberta é quando a solução é alcançada. Os fatores de entretenimento estão relacionados à missão, enredo e elementos sensoriais. Sem esses elementos, o interesse pelo jogo pode ser perdido.

Puzzles de Pensamento Computacional não precisam envolver elementos interativos, mostrando blocos de código de arrastar e soltar ou games. A revista *cs4fn*¹¹ tem uma vasta de puzzles de PC que abordam diversos aspectos do Pensamento Computacional como recursos para professores¹². Por outro lado, puzzles como os da Hora do Código¹³ e *LightBot*¹⁴ utilizam diversos recursos de mídia para tornar os puzzles mais atrativos.

Por fim, Hong et al., (2012) define que jogos de gênero puzzle incentivam os jogadores a prestarem atenção aos erros, facilitando a autoavaliação mais rapidamente do que em outras abordagens. Ainda argumenta que jogos são métodos de estudo preferidos, ao que lista algumas vantagens dos puzzles: são imaginativos e desafiadores; estimulam curiosidade; encoraja o aprendizado por meio de *feedback*; diferentes cenários apresentados permitem explorar; tomar decisões e aplicar habilidades relacionadas ao domínio do conteúdo; estimulam a competição, como um fator que pode ser motivador.

11 <http://www.cs4fn.org/>

12 <https://teachinglondoncomputing.org/puzzles/>

13 <https://hourofcode.com/br/learn>

14 <https://lightbot.com>

2.3.1 CT Puzzle Test

O trabalho de Gonçalves (2015) iniciou desenvolvimento do CT Puzzle Test com o objetivo de avaliar o Pensamento Computacional por meio de puzzles. Para isso, foram criadas diferentes mecânicas de puzzles que demandam a aplicação dos conceitos relacionados ao PC. Além disso, os puzzles não exigiam conhecimento prévio sobre conteúdos específicos para resolução. Os registros da interação dos usuários com os puzzles, são armazenados e servem de base o cálculo dos escores do PC.

A ferramenta foi aprimorada e avaliada por Couto (2019) e por Calbusch (2021), que confirmaram a confiabilidade e a validade. A confiabilidade é a capacidade de reproduzir resultados consistentemente ao longo do tempo e no espaço. Para confiabilidade, o CT Puzzle Test alcançou um valor de 0.67 para alpha de Cronbach, valor próximo ao valor 0.72 alcançado pelo teste de Román-González (COUTO, 2019).

A busca por confiabilidade e validade permeiam o trabalho de Couto e Cabulsch. Couto aplicou o teste em turmas de diversos níveis, em disciplinas de Pensamento Computacional. Durante as aplicações percebeu a necessidade de ajustes em fórmulas e eliminação de itens muito difíceis. Cabulsch aprimorou o instrumento, corrigiu falhas e ampliou mecânicas. Depois submeteu o teste a um processo de análise por especialistas e aplicou a um número reduzido de participantes.

A seguir, são apresentadas as mecânicas da última versão do CT Puzzle Test, aprimorada por Calbusch (2020). As mecânicas originais podem ser encontradas no trabalho de Gonçalves (2015, p. 65).

2.3.1.1 Mecânica de programação

Na mecânica de programação (Figura 6), o jogador deve programar o personagem para ele chegar aos objetivos, indicados em laranja no mapa. A mecânica permite ao personagem mover-se nas quatro direções. Não há movimento de giro em torno próprio eixo (que é comum em outros puzzles semelhantes [*LightBot*, *LOGO*]). Ainda, há obstáculos, que dificultam a tarefa dependendo do nível. Algumas fases são compostas por mais de um objetivo a ser alcançado.

O mapa é um “tabuleiro” de 8 x 8 casas. Cada movimento do personagem corresponde a andar uma casa. Os movimentos do personagem são descritos na área de PROGRAMA, na medida que o examinado arrasta as instruções disponíveis da área inferior direita para a área de PROGRAMA. Ao passar o mouse sobre a instrução disponível, é exibida uma prévia do movimento unitário a ser executado pelo personagem, de acordo com a peça indicada pelo mouse. Quando o botão *play* é clicado, o personagem começa a se movimentar e os movimentos são destacados na área de PROGRAMA por uma borda verde na peça durante cada movimento.

Por último, há fases que permitem a repetição de movimentos, por meio de um comando de *loop*. Esse comando funciona permitindo adicionar as instruções de direção dentro do comando. Depois, o usuário pode clicar em “+” e “-”, para aumentar e diminuir o número de vezes que as instruções inseridas na instrução maior devem repetir. Os contadores de repetição (indicados na peça de *loop*), também são ativos durante a execução, e vão diminuindo cada repetição finalizada até chegar em zero. As ações possíveis na mecânica são:

- Arrastar e clicar nas setas de direções para adicionar ao PROGRAMA
- Clicar em *executar*, para que o personagem comece a andar
- Remover um comando
- Remover todos os comandos
- Aumentar e diminuir número de repetições do *loop*
- Pular a fase

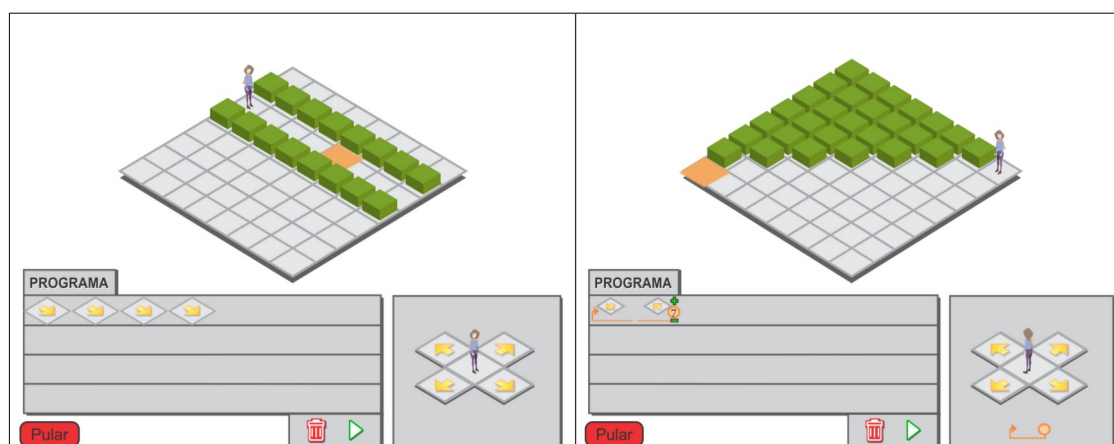


Figura 6: Mecânica de programação com fase simples e fase de *loop* (direita).

Fonte: Autor

Algoritmos é o principal pilar exercitado nesta mecânica. Mas a decomposição também é praticada, pois as soluções podem ser obtidas em partes. Assim, a fórmula de cálculo considera o tempo despendido, a quantidade de comandos utilizados, a quantidade de utilizações da opção “remover um comando”, quantidade de utilizações da opção “remover todos os comandos” e a quantidade de execuções do programa.

2.3.1.2 Mecânica de ligação de pontos

Na mecânica de ligação de pontos (Figura 7) é baseado nas atividades do Programa de Enriquecimento Instrumental de Feuerstein, que é uma série de tarefas, focadas no desenvolvimento de habilidades cognitivas. “Os instrumentos de papel e lápis são projetados para fornecer aos alunos oportunidades para desenvolver estratégias cognitivas e hábitos de trabalho que podem ser aplicados a situações de resolução de problemas” (FEUERSTEIN, 2000).

A tarefa implementada no CT Puzzle Test consiste em conectar pontos para formar figuras geométricas mostradas previamente. Por exemplo: mostra-se um quadrado e um triângulo ao examinado. Depois são apresentados um conjunto de pontos que, quando ligados corretamente, formam o quadrado e o triângulo. A dificuldade está em encontrar a combinação correta entre os diversos pontos. Nesta fase, é possível pedir dicas para que parte da solução seja resolvida. Assim fica mais fácil ligar o restante dos pontos. As ações disponíveis na mecânica são:

- Ligar ponto
- Clicar em ponto para desmarcar o ponto ligado
- Limpar todos os pontos
- Pedir dica
- Pular fase

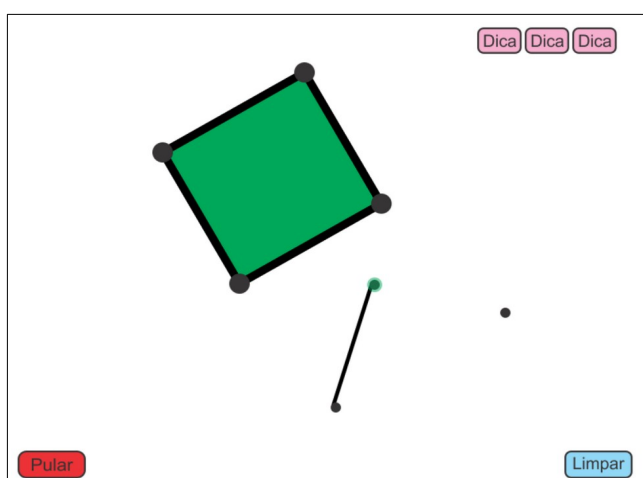


Figura 7: Mecânica de ligação de pontos

Fonte: Autor

Na mecânica de pontos, as interações avaliadas são: tempo despendido; quantidade de cliques nos pontos da tela; quantidade de utilizações da opção “limpar” e quantidade de dicas utilizadas. Assim como nas outras mecânicas, há a opção de pular a fase.

2.3.1.3 Mecânica de encaixe de figuras

A mecânica de encaixe de figuras (Figura 8) envolve encaixar uma série de figuras menores para formar uma figura maior. Além de arrastar as peças, o examinado deve rotacioná-las para que se ajustem nos espaços da figura maior. Assim como na fase de pontos, é possível pedir dicas, ao que parte da solução é resolvida. Durante o processo de resolução, a decomposição é a principal habilidade do Pensamento Computacional medida. As ações disponíveis são:

- Arrastar figura menor

- Rotacionar figura menor
- Pedir dica
- Pular fase

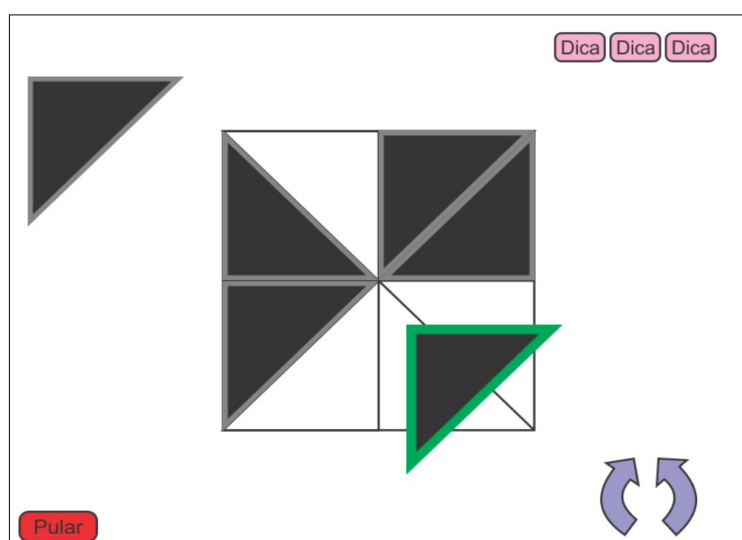


Figura 8: Mecânica de encaixe de figuras na figura maior.

Fonte: Autor

Nesta mecânica, são considerados no cálculo do escore as variáveis de: tempo despendido, quantidade de cliques nas peças, quantidade de utilizações da opção “girar peça” e quantidade de dicas utilizadas.

2.3.1.4 Mecânica de Tangram

O Tangram é um antigo jogo chinês, cujo objetivo é formar figuras a partir de 7 peças (5 triângulos, 1 quadrado e 1 paralelogramo). Segundo uma das lendas, o jogo surgiu quando um imperador chinês quebrou um espelho, e percebeu que poderia montar diversas figuras com os cacos (DANTAS, [s.d.]).

A mecânica de Tangram do CT Puzzle Test (Figura 9) não traz todas as peças presentes na versão original do Tangram. Mas o princípio é o mesmo: juntar as peças para formar figuras. No CT Puzzle Test, as figuras alvo são indicadas por linhas pontilhadas na tela. A partir do Tangram, o CT

Puzzle Test trabalha as dimensões de decomposição e abstração. As opções de interação são parecidas com a mecânica de encaixe de figuras:

- Arrastar figura do Tangram
- Rotacionar figura menor
- Pedir dica
- Pular fase

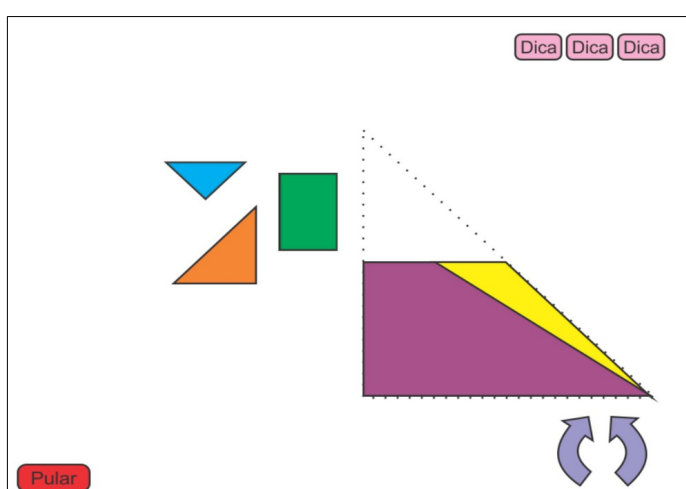


Figura 9: Mecânica de Tangram

Fonte: Autor

As interações consideradas para o cálculo do escore são tempo despendido; quantidade de cliques nas peças, quantidade de utilizações da opção “girar peça” e quantidade de dicas utilizadas.

2.3.1.5 Mecânica de padrões de sequência

Na elaboração dessa mecânica, foram utilizados os conceitos de análise de padrões. O participante deve analisar uma sequência de figuras e selecionar, entre opções disponíveis, qual é a próxima figura da sequência. As interações disponíveis são:

- Arrastar figura da sequência
- Pedir dica

- Pular fase

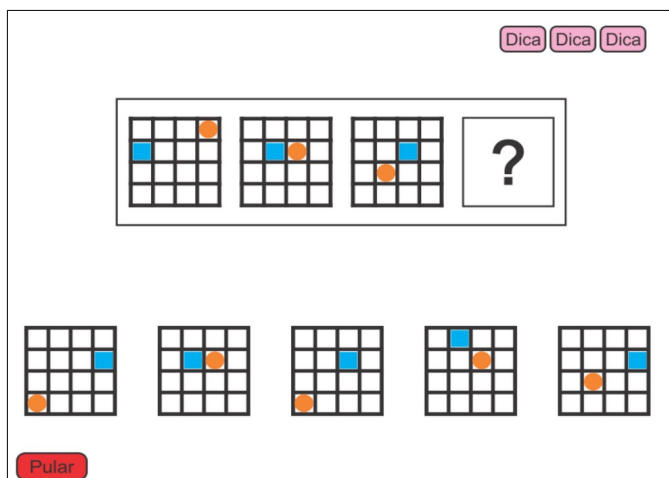


Figura 10: Mecânica de reconhecimento de padrões.

Fonte: Autor

Para resolver o puzzle, o respondente deve arrastar uma das opções disponíveis para o quadrado com o símbolo de “?”. Durante essa interação, são capturados o tempo despendido, a quantidade de tentativas e quantidade de dicas utilizadas, para depois calcular o escore.

2.3.1.6 Mecânica de padrões de objetos

Na mecânica de padrões de objetos o respondente é desafiado a organizar objetos uma matriz - de tamanho 3 x 3 - considerando duas características: cor e tipo de elemento. Há três cores: verde, vermelho e amarelo. E também há três tipos de elementos: animais, frutas, ou peças de vestuário. Cada objeto da cena tem essas duas características (cor, e tipo). O desafio é solucionado quando todas as características estão organizadas corretamente na matriz. As interações disponíveis são:

- Arrastar objeto para a matriz
- Pedir dica
- Pular fase

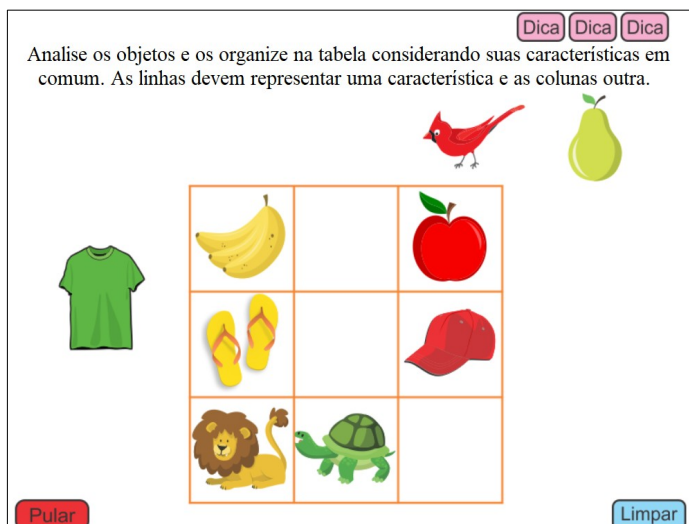


Figura 11: Mecânica reconhecimento de padrões de cores, animais e objetos.

Fonte: Autor

As interações avaliadas são o tempo despendido, a quantidade de vezes que o estudante completou a tabela (quantidade de tentativas); quantidade de utilizações da opção “limpar” e quantidade de dicas utilizadas.

2.4 PSICOMETRIA

Um teste de qualidade mede exatamente o que se propõe a medir, com consistência ao longo do espaço e do tempo. Isso significa que o teste deve medir a(s) característica(s) para a qual foi desenvolvido e não outra qualquer. Além disso, ao ser aplicado em momentos diferentes (tempo) e para diferentes tamanhos de amostra (espaço), os resultados mantêm-se os mesmos. Para testes cujas questões mudam entre diferentes versões, também são necessárias métricas de comparabilidade. Esta seção descreve os conceitos de consistência e validade e traz uma breve introdução sobre os conceitos da Teoria de Resposta ao Item (TRI). O objetivo não é trazer uma revisão completa do tema, mas sim apresentar alguns conceitos base, que são explicados (em português) principalmente nos trabalhos de Souza et al., (2017), Pasquali (2009b), Andrade e Tavares (2000).

2.4.1 Confiabilidade

A confiabilidade ou fidedignidade é o grau em que o teste pode ter seu resultado reproduzido consistentemente ao longo do tempo e espaço e também a partir de diferentes observadores. A coerência, equivalência e homogeneidade fazem parte dessa característica, uma das principais para qualidade do teste (SOUZA et al., 2017, p. 650).

Souza et al (2017) esclarecem que a consistência do instrumento não é constante, pois é suscetível a interferências de fatores como população, circunstâncias, contextos e modos de aplicação. Assim, a interpretação dos resultados de uma pesquisa a partir de instrumentos de medida devem estar associados à descrição clara desses fatores.

Para a pesquisa, três critérios de confiabilidade são essenciais: estabilidade, consistência interna e equivalência:

- Estabilidade: refere-se à semelhança entre os resultados de um teste aplicado em dois momentos diferentes. Pode ser verificada por meio de teste-reteste. O coeficiente de correlação intraclass é uma métrica utilizada para estimar a estabilidade.
- Consistência interna: indica se todas as partes do instrumento medem a mesma característica. Existe uma discussão relacionada a quanto uma característica pode ser considerada pura, ou seja, unidimensional. Nesse caso, pode-se considerar a característica mais aparente, ou dominante (ANDRADE; TAVARES, 2000, p. 16). O coeficiente mais indicado para calcular a consistência interna é o alfa de Cronbach, que reflete o grau de covariância entre os itens de uma escala.
- Equivalência: refere-se ao grau em que dois ou mais especialistas treinados concordam em relação ao escore gerado pelo instrumento. Nesse caso, é essencial que os especialistas tenham as mesmas condições de avaliação, ou seja, deve haver uma padronização na aplicação do teste.

2.4.2 Validade

A validade é um dos aspectos da qualidade do teste. A validade diz respeito ao aspecto da medida ser congruente com a propriedade da medida (PASQUALI, 2009b, p. 995). Em outras palavras: se o teste mensura o que se propõe a medir. Um exemplo, seria um teste de Pensamento Computacional, que não mede outros tipos de pensamento (matemático, por exemplo). Isso é diferente de dizer que o teste é preciso, estável ou consistente, e sim que o traço medido é o buscado.

Da mesma forma que a consistência, a validade é subdividida em diversos tipos. Dentre esses, citamos as validades de conteúdo, de constructo e de critério (concorrente ou preditiva), convergente e discriminante:

- Validade de conteúdo: é o grau em que o conteúdo apresentado no instrumento (descrição dos itens do teste), reflete o constructo que está sendo medido. Ou seja, se os itens do teste constituem uma amostra representativa de um universo finito de comportamentos (domínios) alvo da medição (PASQUALI, 2009b, p. 998).
- Validade de construto: é o grau em que inferências podem ser feitas a partir das operacionalizações que evidenciam o construto teórico que baseia tais operacionalizações (TROCHIM; DONNELLY, 2006, p. 64). Constitui a maneira direta de verificar se o comportamento evidencia a demonstração da habilidade avaliada. Conceitos ou construtos são pesquisáveis apenas se forem, ao menos, passíveis de representação comportamental (PASQUALI, 2009b, p. 996).
- Validade de critério: a validade de critério refere-se ao uso de um critério de comparação externo. Medidas de outros testes amplamente aceitos são considerados critérios. Utiliza-se coeficientes de correlação para constatar a validade de critério. (SOUZA et al., 2017, p. 653). Costuma-se separar a validade de critério entre concorrente e preditiva, cuja diferença é o momento de coleta das informações. Na validade concorrente, a obtenção das informações comparativas deve ser feita de forma paralela ao teste principal (grupos de teste e controle, por exemplo). E a validade preditiva implica que os dados obtidos pelo

instrumento terão correlação forte com o critério de comparação obtido depois (PASQUALI, 2009b, p. 996).

- Validade convergente: utiliza outros instrumentos para comparação. A diferença, em relação à validade de critério é, o instrumento de comparação pode medir outro conceito relacionado, mas não o mesmo que o instrumento em questão.
- Validade discriminante: ao contrário da validade convergente, compara-se com outro instrumento feito para medir algo não relacionado com o objetivo do instrumento em questão. Nesse caso, para obter validade discriminante, o coeficiente de correlação deve ser negativo.

2.4.3 Modelos da Psicometria

A Psicometria procura explicar o significado das respostas dadas pelos respondentes a uma série de tarefas (itens). No caso da Teoria Clássica dos Testes (TCT), o foco reside em entender o resultado do escore total. Por exemplo, em uma prova com 30 questões, o que significa o examinado ter acertado 10, 20 ou 30 questões? Já a Teoria da Resposta ao Item (TRI) foca nos itens, tentando entender quais habilidades são demandadas e quais as probabilidades de o examinando responder corretamente, dado tal nível de habilidade. Assim, enquanto a TCT tenta produzir *testes* de qualidade, a TRI se interessa por produzir *itens* de qualidade, que ao serem armazenados formam bancos de itens válidos para avaliar os traços latentes (PASQUALI, 2009a, p. 63).

2.4.3.1 Teoria Clássica dos Testes

A Teoria Clássica dos Testes, também chamada de Modelo da Psicometria Clássica, foi criada em cima de dados de testes de inteligência por psicólogos de preocupação estatística. A TCT trabalha com uma situação onde a soma da pontuação dos itens (tarefas) gera, ao final, um *escore total*. O valor desse escore representaria a magnitude do que o teste se propõe a medir no sujeito. Contudo, o teste é suscetível a erros de medida, que devem ser considerados no cálculo para se obter o escore verdadeiro:

$$T = V + E \quad (1)$$

V é escore verdadeiro, T é o escore empírico e E é o erro. Assim, o escore empírico é a soma do escore verdadeiro mais o erro. O erro tem diversos fatores não controláveis. Porém, o erro pode ser para mais ou para menos em relação ao escore verdadeiro. Por isso, na suposição de aplicar o teste infinitas vezes, o escore empírico tende a ser igual ao escore verdadeiro. Mas isso é válido, apenas considerando as aplicações para um mesmo sujeito e que as respostas dadas nas entre as aplicações não se afetem (PASQUALI, 2009a, p. 72).

2.4.3.2 Teoria da Resposta ao Item

Enquanto a TCT avalia o comportamento representado no teste, a TRI trata o *item* como conceito fundamental. Mais especificamente, a TRI é um conjunto de modelos que representam a probabilidade de o examinando responder corretamente ao item, como função dos parâmetros do item e da habilidade (ou habilidades) do respondente. Assim, quanto maior a habilidade, maior é a probabilidade de acerto (ANDRADE; TAVARES, 2000, p. 7).

A habilidade, comumente chamada de traço latente, não é observável de forma empírica, mas é demonstrável na operacionalização do item, ou seja, nas operações executadas pelo sujeito. (PASQUALI, 2009a, p. 58). Já os parâmetros dos itens são obtidos por meio de técnicas estatísticas após a aplicação dos itens em uma dada amostra. Entre essas técnicas há o método por máxima verossimilhança marginal e métodos bayesianos (ANDRADE; TAVARES, 2000, p. 5).

A partir das estimações dos parâmetros dos itens, é possível formar um banco de itens calibrado, que podem ser utilizados para formar testes equivalentes. Com isso é possível, por exemplo, avaliar o desempenho da turma em diferentes momentos. Além disso, os parâmetros aparecem na chamada Curva Característica do Item (CCI), facilitando a comparação entre itens.

Há três parâmetros: discriminação a , dificuldade b e probabilidade de acerto por chute c . Os modelos que utilizam esses parâmetros são chamados de modelos logísticos (ML). Há modelos que

utilizam um, dois ou três parâmetros (ML1, ML2 e ML3), sendo que o parâmetro da dificuldade é sempre utilizado. O modelo logístico de três parâmetros (ML3) é dado pela equação 2:

$$P(U_{ij}=1|\theta_j) = c_i + (1 - c_i) \frac{1}{1 + e^{-a_i(\theta_j - b_i)}} \quad (2)$$

$P(U_{ij}=1|\theta_j)$ é a probabilidade do indivíduo j , com habilidade θ_j , responder corretamente ao item i , ou seja, a variável U_{ij} recebe o valor 1.

a_i é o parâmetro de discriminação, que significa o grau em que o item é capaz de diferenciar indivíduos com maior e menor habilidade.

b_i é a dificuldade do item i .

c_i é a probabilidade de acertar por acaso (por chute) o item i .

Na Figura 12, pode-se observar o efeito dos parâmetros, sobre a curva característica do item. Itens com dificuldade b muito próxima de -3 ou +3 (em torno de ± 2.95) devem ser descartados do teste. Da mesma forma, itens com probabilidade de acerto ao acaso (c) acima de 0.4 não são adequados (ARAUJO et al., 2018, p. 13).

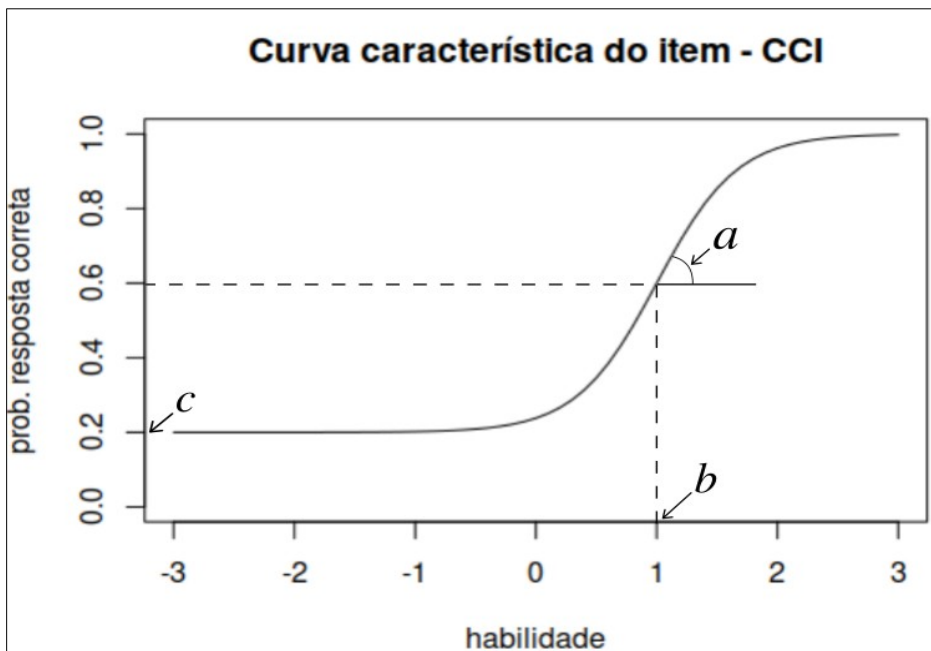


Figura 12: Curva característica do item – CCI

Fonte: Adaptado de Andrade e Tavares (2000, p. 11)

Além da função característica, a Função Informação do Item ajuda a entender quanta informação o item fornece em cada nível de habilidade. No caso do modelo logístico de 3 parâmetros, a equação é escrita como na equação 3 (ANDRADE; TAVARES, 2000, p. 12):

$$I_i(\theta) = D^2 a_i^2 \frac{1 - P_i(\theta)}{P_i(\theta)} \left[\frac{P_i(\theta) - c_i}{1 - c_i} \right]^2 \quad (3)$$

$I_i(\theta)$ é a informação fornecida pelo item i , para habilidade θ .

$$P_i(\theta) = P(U_{ij} = 1 | \theta)$$

Intuitivamente, a informação é maior quando a discriminação é maior, já que o item terá mais capacidade de discernir entre diferentes níveis de habilidade. Da mesma forma, quanto mais a possibilidade de acerto ao acaso se aproximar de zero, mais informativo o item é, pois não diminui a chance de falsos acertos. Essa relação pode ser observada na Figura 13. No primeiro gráfico utilizou-se os valores 0.8 para a , 0.2 para b e 0.2 para c , sendo a informação menor que a apresentada no segundo gráfico, onde utilizou-se os valores 1.2 para a , 0.4 para b e 0.1 para c .

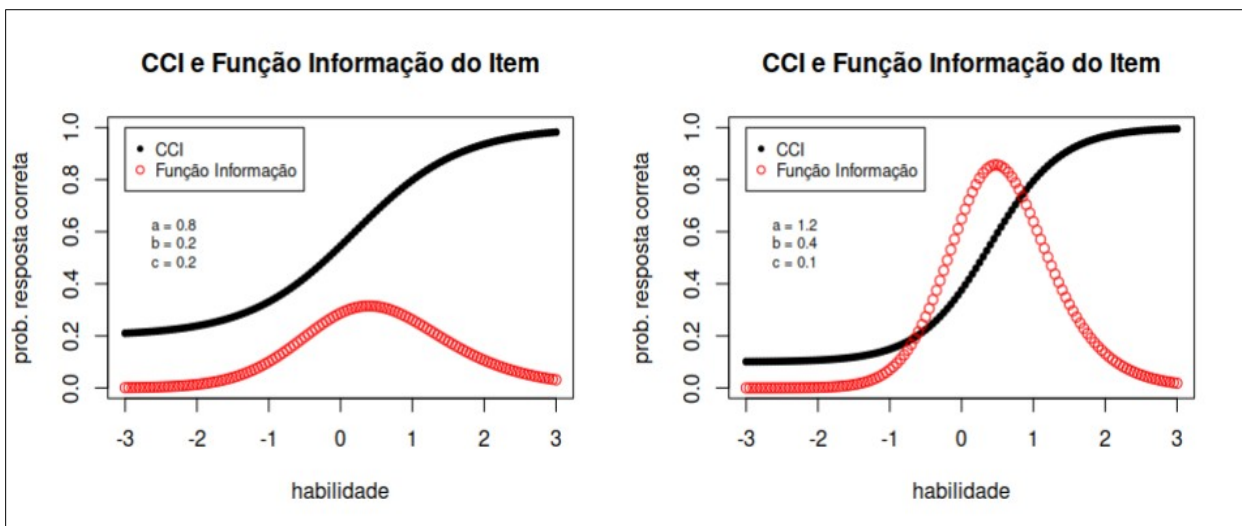


Figura 13: CCI e função informação: comparação de CCI e Função de Informação do Item ao testar diferentes valores para os parâmetros.

Fonte: Autor

TRI e Pensamento Computacional

Alguns trabalhos investigaram a relação entre TRI e o Pensamento Computacional. Araújo et al (2018), por exemplo, investigaram a adequação das questões do Bebras, e descobriram que as estimativas de dificuldade originais foram diferentes das calculadas a partir da análise dos itens com TRI. Já Wiebe et al (2019) construíram um banco de itens com 25 itens a partir de questões do Bebras (6) e também questões do CTt, de Román-González (19). Eles filtraram itens por adequação ao modelo Rasch (ML1) e ML2 e, ao final, obtiveram um conjunto de itens preliminar calibrado, para ser usado em pré e pós-testes.

3 TRABALHOS RELACIONADOS

Este capítulo tem por objetivo demonstrar o estudo sobre trabalhos que possuem um grau de similaridade com o tema de avaliação do PC. Por isso, identifica as contribuições de cada trabalho e depois compara com a solução proposta. Os critérios utilizados para escolher os trabalhos relacionados foram a preferência por trabalhos validados, amplamente aplicados e sistemas de avaliação de PC. Assim, foram analisados cinco trabalhos:

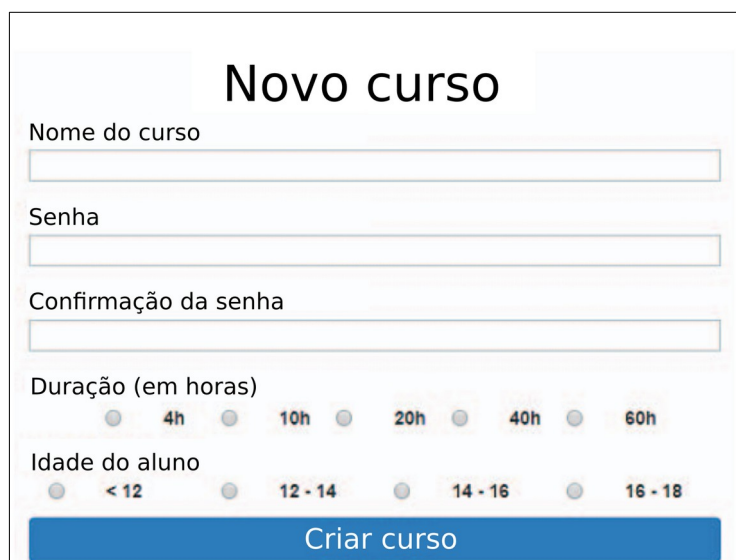
1. Abstractly: uma aplicação online para a elaboração de conteúdos de PC e avaliação de alunos a partir de realização das tarefas e pré e pós testes com questões do Bebras;
2. Bebras: um teste internacional com tarefas de PC curtas, divertidas e divididas por faixas etárias;
3. CTt (*Computational Thinking Test*): um teste de múltipla escolha amplamente validado;
4. Dr. Scratch: uma aplicação online de avaliação de projetos Scratch;
5. CoTAS (*Computational Thinking Assessment System*): uma proposta de ambiente de avaliação de PC, que adota múltiplas estratégias baseando-se no framework de Brennan e Resnick (2012).

3.1 Abstractly

O Abstractly, de Perez e Valladares (2018), é uma plataforma de avaliação do PC, cujo público-alvo são alunos do ensino médio. O uso do ambiente de avaliação segue três etapas: (1) um pré teste, (2) realização de cursos sobre PC e (3) participação em pós teste.

O instrumento não possui um teste próprio. Por isso, os pré e pós testes são formados por questões do Bebras. São 18 questões (6 questões de cada nível: fácil, médio e difícil) para serem resolvidas em 40 minutos. Essas questões não são acessíveis pelos professores para evitar que os alunos recebam conteúdo antecipado, influenciando no resultado do teste.

Ao se registrar na ferramenta, o usuário escolhe entre perfil de aluno ou professor. No caso dos professores, é possível cadastrar cursos para os alunos realizarem. No caso dos alunos, podem realizar as atividades cadastradas pelos professores e acompanhar o próprio progresso. Para cadastrar um curso, o professor define as informações básicas, uma senha de matrícula que cada aluno vai usar para acessar, a duração do curso e a faixa etária (Figura 14).



O formulário, intitulado "Novo curso", contém os seguintes campos e opções:

- Nome do curso: campo de texto.
- Senha: campo de texto.
- Confirmação da senha: campo de texto.
- Duração (em horas): opções de radio buttons para 4h, 10h, 20h, 40h e 60h.
- Idade do aluno: opções de radio buttons para < 12, 12 - 14, 14 - 16 e 16 - 18.
- Botão "Criar curso" em azul na base.

Figura 14: Cadastro de curso

Fonte: Adaptado de (PEREZ; VALLADARES, 2018)

A ideia da ferramenta é criar dois tipos de curso: um cujo objetivo seja estimular o PC e outros com tarefas diversas. No caso do curso relacionado com PC, o professor pode propor tarefas externas como Code.org, desafios do Scratch, ou mesmo atividades desplugadas. Já as atividades do curso de controle podem abordar outros assuntos como xadrez, esporte, dança, atividades que não tenham sido (ao menos propositalmente) desenvolvidas para estimular o PC. Enquanto o grupo experimental realiza o curso de PC, o grupo de controle faz as tarefas do segundo curso. O professor atribui uma situação de publicação para cada atividade, que pode ser de espera, aberta ou completada (Figura 15).

The screenshot displays the 'Abstractly' course management interface. At the top, there is a navigation bar with the logo and the text 'MEUS CURSOS CONTA'. Below this, there are four buttons: 'Cambiar vista', 'Ver informes', 'Gestionar alumnos', and 'Añadir actividades'. The main section is titled 'Progreso do curso' and features a green progress bar indicating 56.25% completion. Below the bar, it states 'Foram realizadas 2.25 horas de 4 horas do curso'. The interface is divided into three main sections: 'Medição [pré-teste]' containing a 'PRE-TEST' activity; 'Atividades [grupo experimental]' containing a list of activities with various control buttons; and 'Medição [pós-teste]' containing a 'POST-TEST' activity.

Medição [pré-teste]	
PRE-TEST	

Atividades [grupo experimental]	
PUZZLE: APRENDER DRAG & DROP	Qualificar atividade
NUMEROS BINARIOS I	Qualificar atividade
ACTIVIDAD PROPIA	Propor Qualificar atividade
MUÉVELO, MUÉVELO	Encerrar atividade
MAPAS FELICES	Ativar Remover do curso
LABERINTO: SECUENCIA	Ativar Remover do curso

Medição [pós-teste]	
POST-TEST	Ativar

Figura 15: Tela de acompanhamento do curso

Fonte: Adaptado de (PEREZ; VALLADARES, 2018)

Depois que os grupos completam as atividades, o professor aplica o pós-teste, novamente com questões do Bebras. Os resultados são disponibilizados em relatórios que comparam o desempenho entre o grupo experimental e grupo de controle (Figura 16). Também é possível comparar o desempenho por gênero. Além dos relatórios de desempenho, o professor pode acessar

relatórios de progresso da turma para cada curso cadastrado bem como relatórios de nível de satisfação para cada atividade.

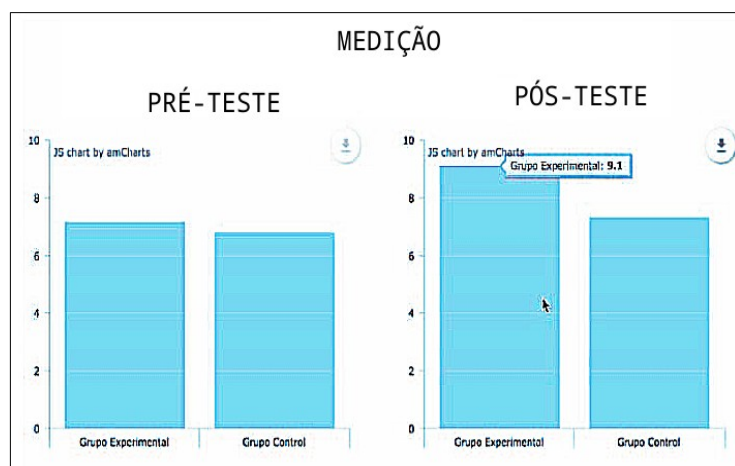


Figura 16: Comparação de desempenho de grupos experimental e de controle em pré-teste e pós-teste

Fonte: Adaptado de (PEREZ; VALLADARES, 2018)

3.2 Bebras

O Bebras é uma iniciativa que busca promover a Ciência da Computação e PC através de um desafio internacional. Nesse desafio, cada país participante aplica questões do desafio integrando-os nas atividades escolares. Essas questões são chamadas de Bebras Tasks, e são divididas por níveis de dificuldades separados por idade. No entanto, essa divisão não é rígida, e cada país pode aplicar só níveis em faixas etárias diferentes (BEBRAS.ORG, 2020a).

O principal ponto do Bebras é o formato das questões. As tarefas são pensadas para promover as habilidades como decomposição, algoritmos, generalização e abstração. Ao mesmo tempo, as questões são divertidas e curtas. Elas simulam situações da vida real aplicando conceitos de computação que estimulam o aluno a pensar sobre a tecnologia.

Os temas abordam algoritmos e programas, diferença entre sequencial e concorrente, estruturas (pilha, filas, etc), estados, interação humano-computador entre outros. Os requisitos para as Bebras Tasks são: apresentar um conceito de informática, fácil entendimento, curtas (para serem

resolvidas em até 3 minutos), independentes de plataforma e serem possíveis de resolver com computador.

O desafio padrão é composto por 18 questões. 6 questões fáceis, que valem 3 pontos, 6 questões médias que valem 4 pontos e 6 questões difíceis, que valem 5 pontos. O tempo do teste é de 45 minutos. No entanto, o Bebras não se propõe a ser um teste rígido de Pensamento Computacional no estilo do CTt de Román-González (2017), mas sim um desafio de estímulo e avaliação de conceitos de computação. Por esse motivo, as questões são diferentes a cada ano.

3.2.1 Bebras Lodge

Embora as tarefas do Bebras sejam realizadas normalmente em computador, a maioria não é interativa. Para adicionar interatividade, foi criada uma ferramenta chamada Bebras Lodge. É uma ferramenta gráfica online, que permite gerar questões interativas por meio de configuração de arquivos JSON. O Bebras.org contém um repositório de questões desenvolvidas e testadas que servem como modelo para novas questões. É possível criar uma tarefa onde o aluno tem que, por exemplo, clicar em posições corretas da imagem, ou arrastar peças para determinadas posições (Figura 17).

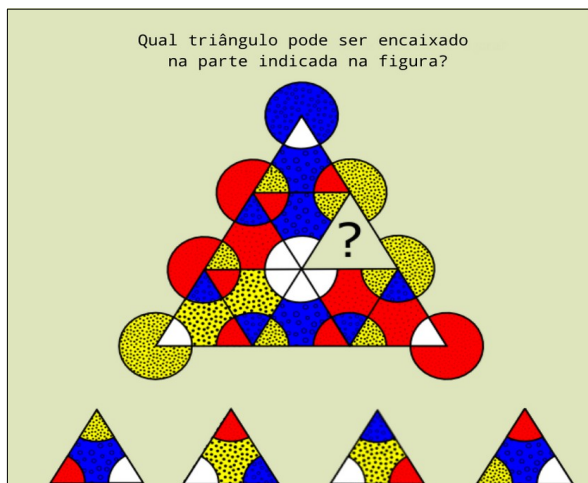


Figura 17: Bebras Task de detecção de padrões criada com Bebras Lodge. As peças podem ser arrastadas para a posição indicada com “?”.
 Fonte: Adaptado de (BEBRAS.ORG, 2020b)

3.3 CTt

O Computational Thinking Test (CTt), de Román-González et al (2017), é um instrumento de diagnóstico que objetiva medir o nível de desenvolvimento do PC. A definição de construto que embasa o teste é de “formular e resolver problemas utilizando lógica sintática de linguagens de programação: sequência, iterações, condicionais e variáveis”. Assim, embora seja utilizado para avaliar os quatro pilares do PC (BRACKMANN et al., 2017), o teste é majoritariamente focado em algoritmos.

O teste, que pode ser aplicado como avaliação de pré e pós testes, tem como público-alvo, alunos de 12 a 14 anos. Ao aplicá-lo em alunos de 5º a 9º anos, Román-González (2017) observou uma melhora da pontuação de acordo com a idade dos alunos. Segundo o autor, este é um resultado condizente com a suposição de que o PC é uma habilidade de resolução de problemas que, portanto, acompanha a evolução cognitiva do indivíduo.

O teste deve ser finalizado em 45 minutos e possui 28 questões de múltipla escolha. Cada questão tem quatro opções, apenas uma correta. Os conceitos abordados são relacionados a sequenciamento (14 questões), completar (9 questões) e depuração (5 questões).

As questões são apresentadas em dois cenários: o maze e o canvas. O maze é uma matriz de posições por onde o personagem é programado para se movimentar. Já no canvas, a programação do personagem faz com que ele controle uma caneta para desenhar linhas na tela. As questões tem dificuldade crescente e teste total tem dificuldade média, com pontuação de 0.59. Já em relação à consistência interna do teste, foi alcançada uma Alpha de Cronbach de 0.793, considerada boa.

Por utilizar esses dois cenários (labirinto e quadro), o conteúdo geral do teste fica basicamente formado por problemas de movimentação, geometria, ângulos e situações de desvio do personagem. Não aborda outros conceitos como recursividade, paralelismo e eventos. Outro ponto é que tem alinhamento com práticas (testar, reusar, abstrair e modularizar), sobre situações já formadas. Por isso o teste não tem alinhamento com as perspectivas de: expressar, conectar e questionar.

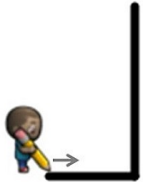
Mesmo assim, por ter passado por um processo rigoroso de validação e testes, o CTt é um dos testes mais robustos. Por esse motivo, pode ser utilizado como um referencial para o desenvolvimento de novos testes ao se buscar uma correlação.

Qual sequência leva o "Pac-Man" até o fantasma pelo caminho indicado?	
	Alternativa A
	Alternativa B
	Alternativa C
	Alternativa D

Figura 18: Questão de sequenciamento do CTt apresentada no cenário *maze*

Fonte: (BRACKMANN, 2017, p. 120)

Qual sequência o artista deve seguir para desenhar a figura abaixo? O lado menor mede 50 pixels e o maior mede 100 pixels.



<p>Alternativa A</p> <p>avance por 50 pixels</p> <p>vire à esquerda por 90 graus</p> <p>avance por 100 pixels</p>	<p>Alternativa B</p> <p>avance por 50 pixels</p> <p>vire à direita por 90 graus</p> <p>avance por 100 pixels</p>
<p>Alternativa C</p> <p>avance por 100 pixels</p> <p>vire à esquerda por 90 graus</p> <p>avance por 50 pixels</p>	<p>Alternativa D</p> <p>avance por 100 pixels</p> <p>vire à direita por 90 graus</p> <p>avance por 50 pixels</p>

Figura 19: Questão de depuração do CTt, apresentada no cenário *canvas*

Fonte: (BRACKMANN, 2017, p. 121)

3.4 Dr. Scratch

O Dr. Scratch, é um software livre e de código aberto para avaliação de projetos Scratch. A partir dela, os alunos podem obter informações sobre a qualidade de seus projetos. Também é útil para ajudar professores e instrutores no processo de ensino e avaliação. Ao analisar o projeto, o Dr. Scratch procura por más práticas de programação e, caso encontre, dá dicas de como resolvê-las. Por fim, calcula uma pontuação para diferentes conceitos de PC (MORENO-LEÓN; ROBLES; ROMÁN-GONZÁLEZ, 2015c). Sete conceitos são calculados, considerando aspectos:

- Paralelismo: uso de eventos simultâneos e mensagens assíncronas
- Pensamento lógico: uso de estruturas condicionais
- Sincronização: uso de blocos de espera, início e movimentação de objetos
- Controle de fluxo: uso de diferentes blocos de repetição: (repita, sempre, repita até)
- Interatividade: uso de interatividade por cliques, entradas do teclado, áudio e vídeo
- Representação de dados: alteração de propriedades dos objetos (tamanho, cor etc), variáveis e operações em listas

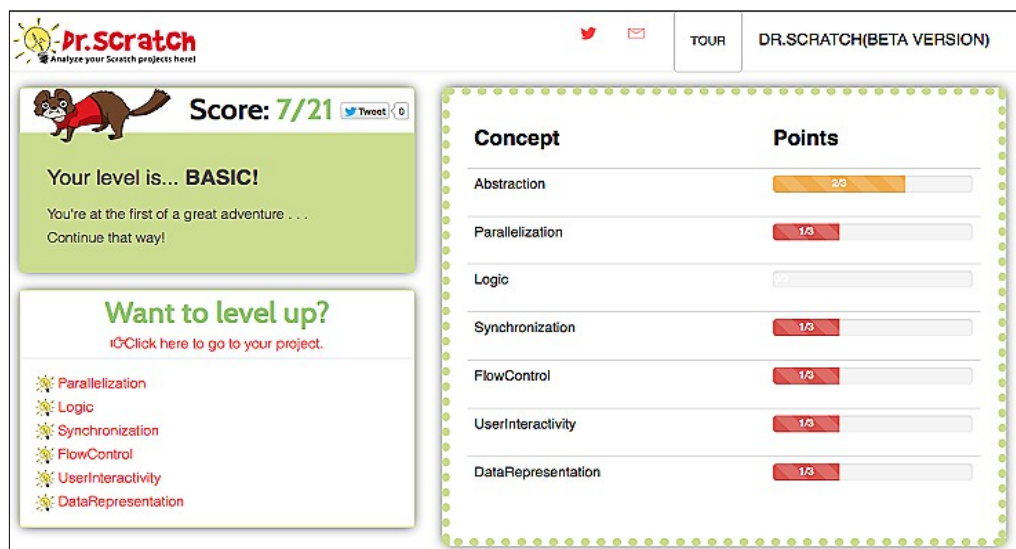


Figura 20: Tela do Dr. Scratch mostrando os resultados da avaliação de um projeto

Fonte: (MORENO-LEÓN e al, 2015a)

Cada conceito recebe uma nota de zero a três, sendo a nota máxima 21 pontos. A partir do cálculo da nota, o Dr. Scratch infere o nível do aluno, e indica recursos da linguagem que podem ser utilizados para alcançar o próximo nível (Figura 20). Os níveis de classificação são básico, desenvolvimento e mestre. O número de dicas apresentadas é maior de acordo com o nível alcançado.

3.5 CoTAS

O Computational Thinking Assessment System (CoTAS) é uma proposta de Yeny e Hermans (2019) para um sistema de avaliação do PC. O objetivo é avaliar alunos do ensino médio de cursos de programação através de múltiplas estratégias. Na prática, são utilizados desde avaliação automática de código Python até entrevistas para obter dados de avaliação. Assim, a ferramenta avalia o aluno sobre conceitos, práticas e perspectivas (BRENNAN; RESNICK, 2012).

Para avaliar os conceitos, códigos fontes em Python são analisados automaticamente. O nível de proficiência do projeto é classificado com base na qualidade do código fonte, podendo ser básico, em desenvolvimento ou proficiente (Figura 21, área 1). Depois é considerada a frequência de utilização de conceitos do PC. Na prática, são contabilizados quantas vezes o aluno utilizou

recursos como estruturas de controle, operadores, funções, estruturas de dados, sequências e condicionais (Figura 21, área 2).

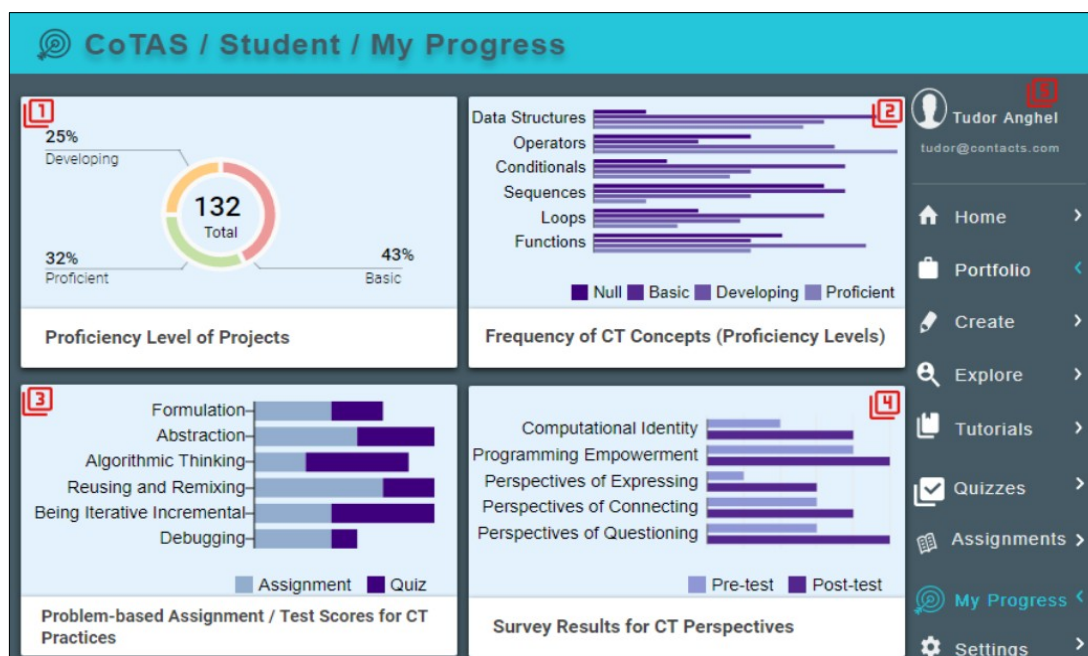


Figura 21: Painel de resultados de avaliação do CoTAS

Fonte: (YENI; HERMANS, 2019)

Já para medir as práticas, os autores utilizam dois recursos: cenários de programação e questões de múltipla escolha. A avaliação é feita a partir da contagem do número de tentativas, tempo de resolução e da análise de semelhança entre a saída gerada e a saída esperada. As questões de múltipla escolha abordam os níveis de avaliação, síntese e análise da Taxonomia de Bloom. Dessa forma, é possível obter dados da capacidade de formulação, abstração, pensamento algorítmico, reutilização e reformulação, decomposição e depuração por meio de prática e questionário (Figura 21, área 3).

Por fim, as perspectivas são avaliadas a partir de questionários e entrevistas. Os questionários abordam o grau de concordância com afirmações relacionadas à identificação com computação, empoderamento através da programação, perspectivas de expressão, conexão e questionamento, sendo aplicados em pré e pós-testes (Figura 21, área 4). Já as entrevistas são feitas pelo sistema e analisadas manualmente, o que demanda tempo e esforço.

3.6 ANÁLISE COMPARATIVA

Esta seção apresenta uma análise comparativa dos trabalhos selecionados. As características comparadas foram: abordagem de avaliação, os ambientes/formatos possíveis de aplicação, vantagens, desvantagens e o suporte à evolução dos testes (Quadro 3). Depois os trabalhos são posicionados em relação à solução proposta.

Quadro 3. Análise comparativa do estado da arte das ferramentas de avaliação de PC

Característica	Abstractly	CTt	Dr. Scratch	Bebras e Bebras Lodge	CoTAS	CT Puzzle Test Platform
Abordagem	Somativa	Diagnóstico	Formativo-iterativas	Diagnóstico	Somativa e Formativa	Diagnóstico
Ambiente de aplicação	Computador	Computador	Computador	Computador	Computador	Computador/Celular
Vantagens	Separação facilitada de grupos experimental e de controle	Teste amplamente validado	Avaliação automática que serve para direcionar o aluno	Questões simples e divertidas	Avalia o PC sob diferentes conceitos, práticas e perspectivas	Visualização de estatísticas, criação de amostras
Desvantagens	Depende de questões externas. Pouca flexibilidade de consultas	Avalia mais fortemente a programação que outros pilares	Avalia somente projetos do Scratch	Múltipla escolha. Não avalia dificuldades de processo de resolução	Algumas etapas necessitam de análise manual	Não avalia projetos criados pelo aluno. É uma plataforma de testes
Apoia evolução dos testes	Não	Não	Não	Não	Não	Sim
Referência	(PEREZ; VALLADARES, 2018)	(GONZÁLEZ, 2015a)	(MORENO-LEÓN; ROBLES; ROMÁN-GONZÁLEZ, 2015c)	(BEBRAS.ORG, 2020b)	(YENI; HERMANS, 2019)	

Fonte: Autor

Em relação às abordagens o Abstractly é uma ferramenta somativa pois o objetivo é avaliar o desempenho dos alunos antes e depois de uma série de exercícios de PC (isto para o caso do grupo experimental). O CTt e o Bebras são ferramentas de diagnóstico. A diferença entre elas é que o CTt é um teste confiável validado enquanto o Bebras apresenta tarefas que não são sempre as mesmas ou que não passaram pelo mesmo processo de validação. O Dr. Scratch é uma ferramenta do tipo formativo-iterativa pois auxilia o aluno a perceber onde pode melhorar nos próprios projetos. Já o CoTAS tem características somativas e formativas, pois apresenta informações aos

professores e aos alunos¹⁵. Por fim, a CT Puzzle Test Platform é uma ferramenta de diagnóstico, pois se propõe a avaliar o nível de PC do indivíduo podendo ser aplicada em condição de pré-teste ou pós-teste.

Quanto ao ambiente de aplicação, todas as ferramentas são possíveis de aplicar com o uso de computador. A diferença é que as ferramentas CTt e Bebras são mais facilmente transportadas para papel e caneta caso necessário. Já o Abstractly, Dr. Scratch, CTt e CoTAS demandam o uso de computador. A ideia do CT Puzzle Test Platform é criar uma API que informe para a plataforma dados de PC. Nesse caso pode-se executar experimentos de PC para outros tipos de ambientes conectados à internet (interfaces tangíveis, robôs por exemplo). Nesse caso não seria obrigatório o uso de computador por parte do participante.

Sobre as vantagens de cada ferramenta: o Abstractly facilita a separação de grupos experimental e de controle. A interface intuitiva permite a divisão por faixas etárias, cadastro de tarefas e visualização do desempenho dos dois grupos. O CTt é amplamente validado. Pelo alto número de participantes envolvidos e etapas de validação, alcançou um bom coeficiente de confiabilidade. O Dr. Scratch tem como vantagem permitir a análise rápida de projetos Scratch pelo próprio aluno, sem a necessidade de avaliação pelo professor. O Bebras e Bebras Lodge tem como vantagens, o formato das questões. O protocolo de questões simples e divertidas permite a criação de avaliações diferentes. O CoTAS se destaca por ser o mais completo ao permitir a análise do PC sob diferentes perspectivas. Já o CT Puzzle Test Platform tem como vantagem a possibilidade de configuração dos testes e análise estatística dos resultados por meio da TRI.

Já em relação às desvantagens, o Abstractly é uma plataforma dependente de questões externas e por isso não permite avaliar o progresso do aluno nas questões. Além disso, o formato dos resultados estatísticos é fixo na aplicação, limitado a um número de gráficos. O CTt tem como desvantagem apresentar questões fortemente ligadas à algoritmos, embora se diga avalie também os outros pilares do PC (BRACKMANN, 2017, p. 118). Já o Dr. Scratch avalia somente projetos do Scratch, embora a mesma lógica possa, com alguma modificação, ser transportada para outros ambientes de programação. O Bebras não analisa o processo de resolução das questões por ser um teste de múltipla escolha. Também não é amplamente validado como o CTt. O CoTAS tem como

15 A dedução de que o CoTAS apresenta informações aos alunos se deu a partir do menu “my progress” da Figura 21.

desvantagem a necessidade de análise manual na etapa de entrevista. Por fim, o CT Puzzle Test Platform tem como desvantagem, não analisar o constructo do aluno, limitando-se a avaliar eventos da interação do usuário com um ambiente de testes.

Por fim, o suporte à evolução dos testes é relacionado a geração de estatísticas sobre a qualidade dos testes. Por isso, considera-se que apenas a proposta da plataforma de testes desse trabalho traz índices desse tipo. Abstractly e CoTAS apresentam dados estatísticos. Contudo, esses dados referem-se ao desempenho de participantes. A proposta da plataforma é, além de permitir avaliar os participantes, oferecer dados sobre a qualidade do teste.

3.7 CONSIDERAÇÕES

Finalizando o capítulo de Trabalhos Relacionados, podemos perceber que as ferramentas para avaliação de PC apresentam características muito relacionadas a cada abordagem adotada. Por isso, as vantagens e desvantagens de cada ferramenta são próprias de cada proposta de avaliação. Enquanto algumas ferramentas procuram por confiabilidade e validade de dados, outras propõem a criação de questões divertidas, aplicáveis e instigantes.

Há ferramentas que são fortemente acopladas a ambientes de programação (Dr. Scratch) e outras que tentam tornar-se independentes e utilizar recursos externos (Abstractly). A urgência por ferramentas para avaliação do PC para aplicação em currículos é mais recente. Nesse contexto, a proposta atual procura desenvolver uma plataforma de testes de PC. Essa comparação foi feita ao final deste capítulo, como uma prévia do que será apresentado no próximo capítulo, que trata do desenvolvimento.

4 DESENVOLVIMENTO

A CT Puzzle Platform é um ambiente para criação, aplicação e análise de testes de PC baseados em puzzles. Os puzzles desafiam o aluno a resolver problemas relacionados aos pilares do PC. Os puzzles não são contidos na plataforma. Ao invés disso são desacoplados, desenvolvidos separadamente para comunicar-se via requisições, a fim de receber a configuração do teste, apresentar o cenário ao aluno e enviar as respostas. Assim, a plataforma é o ambiente de configuração do teste pelo pesquisador e os puzzles são o ambiente de interação com o aluno.

A plataforma permite executar quatro tipos de tarefas: cadastro de mecânicas e itens, configuração e aplicação de testes, manutenção de usuários e análise dos resultados. O cadastro de mecânicas e itens é feito via linguagem de programação. As mecânicas definem um conjunto de atributos gerais, que são utilizados nos itens para criar cenários específicos. Após cadastrar um conjunto de itens, um teste pode ser formulado e aplicado.

Durante as aplicações de testes, a plataforma recebe os eventos das interações dos usuários com os puzzles. Assim como os itens, as respostas também tem um conjunto de atributos declarados no cadastro da mecânica. A partir da especificação dos atributos dos itens e dos atributos das respostas é definida uma função de cálculo de score, pois se conhece as variáveis de ambos. O desenvolvedor de cada puzzle, deve respeitar atributos para apresentar o item e para enviar as respostas durante a aplicação do teste. Somente assim, a comunicação entre o puzzle e a plataforma funciona.

Os testes podem ser aplicados diversas vezes, de forma pública ou privada. Os testes aplicados publicamente aparecem no site da plataforma. Já as aplicações privadas devem ser respondidas a partir de um link de participação, gerado pela plataforma. Ao acessar o link, o aluno é direcionado para o puzzle, que passa a enviar para a plataforma os eventos de interação.

No início da interação, a plataforma gera um identificador único aleatório para o aluno. Assim, o aluno não tem dados pessoais (ou que permitam identificá-lo) registrados, e responde ao teste anonimamente.

Ao acessar o teste, o aluno é direcionado para o primeiro item do teste. Durante a tentativa de resolução, eventos de resposta são gerados e transmitidos para a plataforma. Número de erros, número de acertos, desistências, quantidade de dicas solicitadas e lista de comandos são alguns exemplos de dados úteis para o cálculo do score. Ao fim de cada item, o puzzle direciona o aluno para o item seguinte, até o fim do teste ou desistência.

Ao fim do teste, o pesquisador pode apresentar um questionário com perguntas configuráveis via plataforma. As respostas às perguntas do questionário são registradas na mesma base em que são armazenadas as respostas aos itens.

Dessa forma, o conjunto de respostas aos itens e ao questionário forma uma base de dados estruturada. A base é explorada a partir de um módulo de estatísticas da plataforma, que permite gerar índices de consistência, finalização, dificuldade e gráficos interativos e consultar as respostas ao questionário. Por fim, os dados podem ser exportados em formato de texto para uso em outras ferramentas.

As seções seguintes detalham o sistema resumido nos parágrafos anteriores. Para isso, utiliza diagramas de UML (*Unified Modeling Language* – Linguagem de Modelagem Unificada), capturas de tela, a apresentação do puzzle de programação e um diagrama de componentes.

4.1 CASOS DE USO

O objetivo do diagrama de casos de uso é dar uma visão externa geral das funcionalidades do sistema, sem especificar como essas funcionalidades serão implementadas. A definição dos casos de uso ajuda a conhecer os atores que se comunicarão com as funcionalidades. O diagrama especifica as relações de associação, inclusão e dependência entre os casos. Como utiliza uma linguagem informal, o diagrama simplifica o entendimento das partes e interações entre atores e funcionalidades, facilitando a compreensão e identificação de falhas de especificação (GUEDES, 2011).

No caso da CT Puzzle Platform (Figura 22), dois tipos de usuário interagem com o sistema: o pesquisador e o participante. O pesquisador é responsável por manter as entidades do sistema e

consultar os resultados dos testes. O participante é o usuário que responde aos testes e questionários, gerando registros de participação e respostas. Como já citado, apenas os usuários pesquisadores devem registrar-se com dados pessoais para entrar no sistema.

Após logar no sistema, o pesquisador pode cadastrar mecânicas e itens, criar e aplicar os testes e consultar os resultados. No cadastro do teste, o pesquisador monta uma lista de itens e cria um questionário de fim de teste para coletar dados demográficos. Cada item é um cenário de puzzle caracterizado pelas restrições definidas na mecânica do item. Ao finalizar a construção do teste, o pesquisador pode aplicá-lo, tornando-o disponível para o participante.

O participante pode realizar três casos de uso: responder teste, responder questionário e consultar o resultado. A participação no teste é iniciada por meio do convite feito pelo pesquisador ou de forma voluntária para as aplicações disponibilizadas publicamente. Para poder consultar os resultados, o participante deve finalizar o teste (responder todas as questões) e responder o questionário. O questionário, pode conter perguntas gerais (opinião do participante sobre o teste por exemplo). As respostas do questionário não são utilizadas no cálculo do escore.

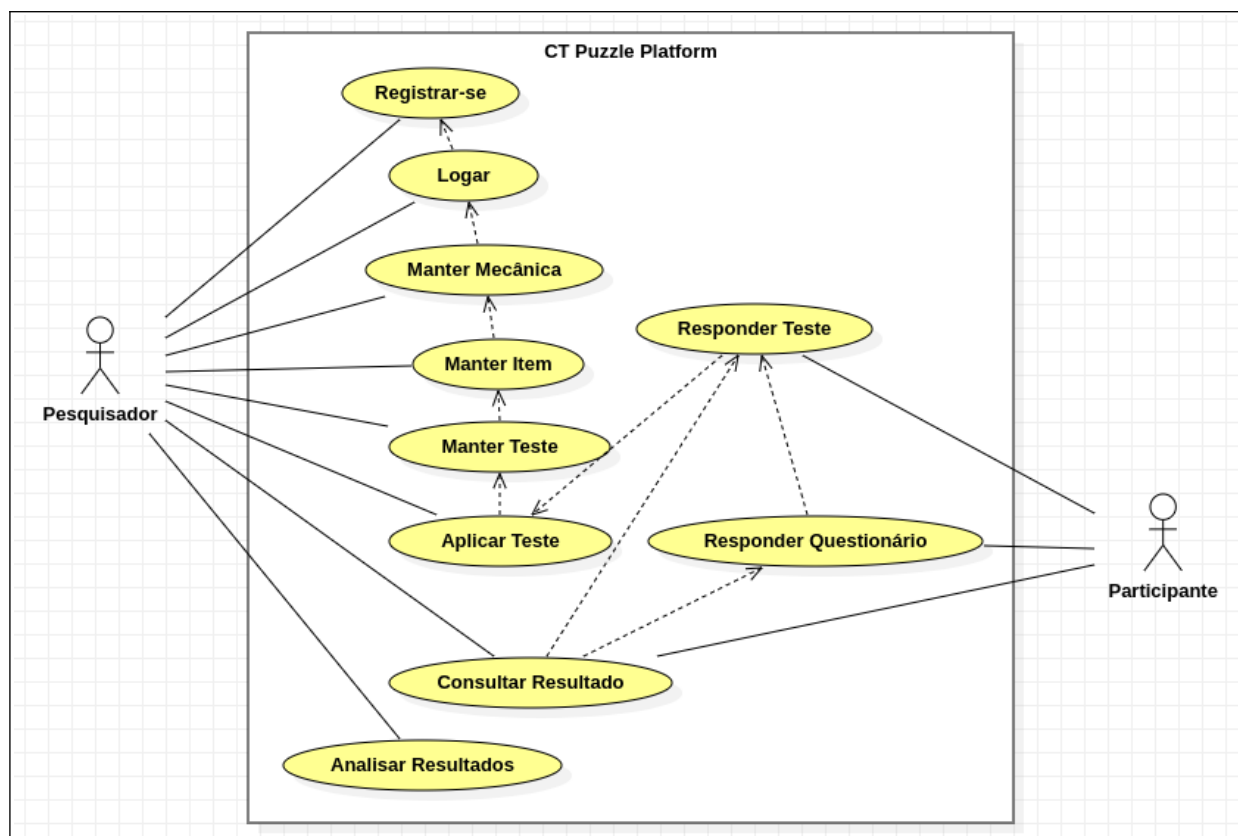


Figura 22: Diagrama de casos de uso

Fonte: Autor

A “consulta aos resultados” não é a mesma para o pesquisador e para o participante. Enquanto o participante tem acesso somente aos resultados de sua própria participação de forma resumida, o pesquisador tem acesso aos dados de todos os participantes da aplicação. Em posse das respostas, o pesquisador pode fazer análises estatísticas através da plataforma, que permite gerar gráficos a partir das variáveis das respostas aos itens e ao questionário de fim de teste.

4.1.1 Registro e login

Para acessar a plataforma, o pesquisador precisa passar pelo registro e login. Não há grandes peculiaridades nessa parte. Os dados solicitados são básicos: nome, data de nascimento, gênero e e-mail e senha. Após o cadastro, estando todos os dados válidos, o sistema envia um e-mail com um link e código de confirmação. Ao clicar no link ou informar o código de verificação, a plataforma redireciona o pesquisador para o ambiente interno.

4.1.2 Manutenção de mecânica

Ao acessar a plataforma, a primeira coisa que o pesquisador deve fazer é criar uma *mecânica*, caso ainda não possua. Essa estrutura refere-se ao tipo de puzzle: são as propriedades que um determinado tipo de puzzle tem. Exemplo: a mecânica de programação tem um ponto de partida, um tamanho de mapa, um melhor caminho a ser percorrido, obstáculos e objetivos. A mecânica para um jogo de ligação de pontos tem uma lista de coordenadas (x,y) e arestas. Há também a possibilidade de adicionar dicas e outras configurações. Em suma, a mecânica define atributos que os itens de tal tipo devem especificar: na fase de programação 1, o personagem pode começar na posição (3,4), na fase 4 começar na posição (1,2) por exemplo.

Em termos de modelo de dados, a mecânica contém três elementos principais: uma classe de especificação, uma classe de resposta e uma função de cálculo de escore. A classe de especificação é um modelo geral que contém os atributos do puzzle. A classe de resposta é parecida, mas contém os atributos de resposta a ser retornada durante a resolução do item (o caminho percorrido, as peças utilizadas, os pontos ligados, tempo de resolução por exemplo). O cálculo de escore é uma função que recebe dois parâmetros: um item e uma a resposta. Com isso compara-se os dados da resposta com a melhor solução esperada para o item, e retorna-se o escore ao lado do escore máximo alcançável. A especificação desses três elementos ajuda transferir a responsabilidade pelo cálculo do escore do instrumento de coleta (puzzle) para a plataforma.

Seguindo o fluxo da tela de cadastro de mecânicas (Figura 23), o pesquisador deve informar o nome, uma descrição e uma imagem de identificação. Em seguida, deve codificar a classe da mecânica, a classe da resposta e a função de cálculo de escore. Toda essa codificação usa linguagem de programação TypeScript¹⁶.

16 <https://www.typescriptlang.org>

CT Puzzle Test Platform Cassiano ▾


Plataforma / Mecânicas / Edição

Edição de mecânica de puzzle

* Nome
Tangram

* Endereço do aplicativo (URL) que apresenta o puzzle
https://exemplo.dominio.jogo.edu.br

* Descrição
Nesta mecânica, o participante será desafiado a organizar corretamente as partes do tangram para formar a figura esperada.

Imagem de ilustração

Dica: CTRL+V cola imagem

⚠ A especificação da mecânica é uma classe `TypeScript` com as características do puzzle. Posição de objetos, tamanho de mapa, dificuldade, tempo esperado de resolução, número de vidas são exemplos de atributos.

* Especificação da mecânica

```
class ItemTangram {
  pontosFiguras: {x:number, y:number}[] = [];
}
```

typescript

⚠ A classe de resposta contém os dados que são utilizados para o cálculo de escore. Tempo de resposta, número de erros, solução informada, número de tentativas são exemplos de atributos.

* Especificação da resposta

```
class RespostaItem {
  figurasInformadas: {x:number, y:number}[] = []
  tempo!:number
  dicas!:number
}
```

typescript

* Função de cálculo do escore
[Editar função de cálculo de escore](#)

[✓ Salvar](#) [< Voltar](#)

Figura 23: Tela de configuração de tipo de puzzle

Fonte: Autor

4.1.2.1 Função de cálculo de escore

A tarefa de edição da função de cálculo de escore tem uma tela dedicada (Figura 24). A edição é feita em duas partes: (1) codificação da função de escore e (2) criação de casos de teste. Na primeira parte, o usuário deve criar uma função para calcular o escore com base em um item e uma resposta. A partir desses parâmetros, o pesquisador desenvolve a lógica para pontuar o desempenho do participante na resposta em relação ao valor máximo, retornando o resultado em formato pré-

definido. Um retorno igual a $\{“score”: 8, “max”: 10\}$, significa que o participante atingiu 80% da nota máxima.

Playground para teste de função de escore

Função de escore

```
function calculaScore(item: ItemProgramacao, resposta: RespostaItem) {
  let nota = 0;

  let ganhoPorSimilaridadeDeComandos = 0
  let ganhoPorDesempenhoDeTempo = 0;

  let comandos: Comando[] = ["UP", "DOWN", "LEFT", "RIGHT", "PROG_0", "PROG_1", "PROG_2"]
  if (item.comandosEsperados && resposta.comandosUtilizados) {
```

+ linhas - linhas typescript

Caso de teste 1 **Passou nos testes** Encolher

Função de item 1 **Remove**

```
// Sem tempo esperado mas acertou quase tudo
// Acertou 75% dos comandos
(function criarItem(): ItemProgramacao {
  let item = new ItemProgramacao();
  item.comandosEsperados = ['UP', 'UP', 'UP', 'UP']
  item.mapa = [
    ['tile', 'null', 'tile', 'tile']
  ]
  //item.tempoEsperado = 30
  return item;
})
```

typescript

Função de resposta 1 **Remove**

```
//Sem tempo esperado mas acertou quase tudo
(function criarResposta(): RespostaItem {
  let resposta = new RespostaItem();
  // Acertou 3 de 4
  resposta.comandosUtilizados = ['DOWN', 'UP']
  //resposta.tempoEmSegundos = 20
  return resposta;
})
```

typescript

Valor esperado: 8

Valor obtido: 8/10

Passou no teste?

+ Adicionar resposta

Caso de teste 2 **Passou nos testes** Expandir

Caso de teste 3 **Passou nos testes** Expandir

+ Adicionar caso de teste

4 testes passaram 0 testes falharam

Rodar casos de teste

Figura 24: Tela de edição da função de escore

Fonte: Autor

Os casos de teste servem para validar o funcionamento da função de escore diante de diferentes cenários. Cada caso é composto por um item e uma ou mais respostas. Cada resposta recebe um valor esperado de escore. Ao executar os testes (clicando no botão azul na parte inferior direita da tela), a plataforma compara os resultados obtidos com os escores esperados, e informa o número de testes que ocorreram com sucesso ou com falha. Também fornece as causas de erro e registros de execução. Assim, ao criar diferentes testes, é possível ir aprimorando a função de escore ao mesmo tempo que se assegura seu funcionamento para cada cenário.

4.1.3 Manutenção de item

A manutenção dos itens de teste é o processo de criar, editar e remover cenários de puzzles. Os itens são criados respeitando as especificações de classe da mecânica. Caso não respeitem a especificação, o editor mostra erros de sintaxe para alertar o pesquisador. Na tela (Figura 25), o segundo editor de código é dedicado para a configuração do item. O primeiro editor serve apenas para a consulta da classe de especificação. No exemplo, o pesquisador escolheu o tipo de mecânica “Programação RoPE”. Em seguida descreveu o cenário para o item especificando suas variáveis: comandos esperados, ganho de energia ao capturar pilha, falas do personagem, direção inicial da face, mapa, obstáculos e posição inicial.

Além de criar o item, essa tela permite testá-lo, acessando o ambiente do puzzle com os parâmetros de demonstração. No exemplo, ao clicar para testar o item, o navegador de internet abre uma nova aba no endereço:

```
https://ct.playerweb.com.br/?op=playground&urlToInstantiateItem=https://  
api.ctplatform.playerweb.com.br/items/public/instantiate/70
```

No endereço, o parâmetro “op=playground” indica que é uma operação de teste do item. E o parâmetro *urlToInstantiateItem* é o endereço que o puzzle deve requisitar à plataforma para obter os dados do cenário a ser testado (o item de número 70).

Além de testar o item, é possível copiá-lo e marcá-lo como item de tutorial. A cópia é útil para criar itens a partir do item copiado. Já os itens de tutorial servem para explicar a mecânica ou o desafio do item para o participante e não devem ter pontuação considerada. Assim, a informação é útil para filtragem das respostas durante a análise estatística.

Plataforma / Itens de teste / Edição


Edição de item de testes

* Nome: [CAÇA ÀS MOEDAS] Curva para quebrar bloco!

* Tipo de mecânica: Programação RoPE [Acessar](#)

* Descrição: Esta fase assegura que o RoPE consegue quebrar blocos!

Tutorial Este é um item de tutorial

Ilustração do item 

Dica: CTRL+V cola imagem

⚠ A configuração do item deve obedecer as especificações definidas na mecânica.
O objeto json/TypeScript configurado deve ser utilizado pelo desenvolvedor no momento de instanciar o item do teste.

* Especificação do item

Classe da mecânica (apenas leitura)

```

type Obstaculo = "coin" | "block" | "null"
type Mapa = "tile" | "null"
type Face = "up" | "down" | "right" | "left"
type Comando = "UP" | "DOWN" | "RIGHT" | "L"

class ItemProgramacao {
  mapa!: Mapa[][];
  obstaculos!: Obstaculo[][];
  face!: Face;
  x: number = 0;
  y: number = 0;
  comandosEsperados!: Comando[];
  tempoEsperado!: number;
  tentativasEsperadas!: number;
  nivelBateria: number = 10;
  nivelMaximoBateria: number = 10;
  custoBateriaEmCadaMovimento: number = 1;
  ganhoBateriaAoCapturarPilha: number = 2;
  acoesTutorial: AcaoTutorial[] = [];
  falasAntesDeIniciar: string[] = [];
  mensagemAoPularFase: string = 'Tem certeza q';
  mensagemAoSairDoJogo: string = 'Tem certeza q';
}

type Acao = "click" | "drag"
type Elemento = "btn-step" | "arrow-up" | "L"
class AcaoTutorial {
  acao!: Acao;
  elemento!: Elemento;
  arrastarSobre?: Elemento;
  frase?: string;
}

```

Instanciação do item [Testar](#)

```

function createItem() {
  let item = new ItemProgramacao();
  item.comandosEsperados = ['UP', 'LEFT', 'PROG_1', 'UP', 'PROG_1']
  item.ganhoBateriaAoCapturarPilha = 2
  item.falasAntesDeIniciar = [
    'Quebre o bloco!',
    'Vamos pegar a moeda!',
  ]
  item.face = "down"
  item.mapa = [
    ['tile', 'null', 'null', 'null'],
    ['tile', 'tile', 'tile', 'tile'],
  ]
  item.obstaculos = [
    ['null', 'null', 'null', 'null'],
    ['null', 'null', 'block', 'coin'],
  ]
  item.x = 0
  item.y = 0
  return item;
}

```

typescript

✓ Salvar [Voltar](#) [Fazer uma cópia](#)

Figura 25: Tela de manutenção de item

Fonte: Autor

4.1.4 Manutenção de testes

Dois componentes fazem parte do teste: um conjunto de itens seguidos e um questionário de fim de teste. Os itens formam uma lista ordenada escolhida da base de itens: é o que o participante deve responder no ambiente de puzzle. Já o questionário de fim de teste visa extrair informações complementares (como idade do participante ou fazer perguntas gerais). O pesquisador deve criar os itens previamente ao teste. Mas deve elaborar as perguntas do questionário durante o cadastro do

teste. As perguntas do questionário podem ser obrigatórias ou opcionais. Os tipos de respostas podem ser: numéricas, descritivas curtas, descritivas longas, de seleção de opções ou de “sim ou não” (Figura 26).

CT Puzzle Test Platform Cassiano ▾


Plataforma / Testes / Edição


Edição de teste


* Nome do teste

Pesquisar itens

Itens adicionados (arraste para ordenar)

 **Fase 1: Estrela com curva**
É necessário fazer uma curva e coletar a moeda

 **Fase 2: Curva de estrelas**
Nesta fase o participante deve coletar 3 estrelas fazendo curva

 **Fase 3: Estrela na frente**
Basta andar para frente

Questionário de fim de teste

Pergunta	Tipo da resposta	Obrigatório	Remover
<input type="text" value="Nome"/>	Texto ▾	<input type="checkbox"/>	<input type="button" value="Remover"/>
<input type="text" value="Idade"/>	Número ▾	<input type="checkbox"/>	<input type="button" value="Remover"/>
<input type="text" value="Já programou alguma vez?"/>	Sim ou não ▾	<input type="checkbox"/>	<input type="button" value="Remover"/>
<input type="text" value="O que achou do teste?"/>	Texto ▾	<input type="checkbox"/>	<input type="button" value="Remover"/>

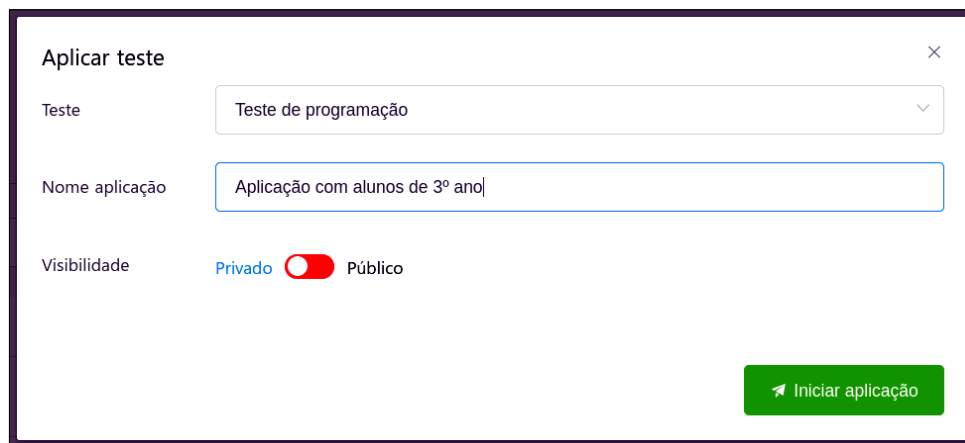
Figura 26: Tela de manutenção de teste com questionário

Fonte: Autor

Ao salvar o teste, o sistema oferece ao pesquisador a opção de criar uma aplicação de teste. Se decidir aplicar o teste, o sistema abre a página de aplicações. Se preferir não aplicar, o pesquisador pode fazê-lo posteriormente através da lista de testes ou do próprio cadastro de teste.

4.1.5 Aplicação de teste

A aplicação é a disponibilização de um teste para o recebimento das respostas dos participantes. Para criá-la, o pesquisador escolhe um teste, um nome e a visibilidade (pública ou privada) (Figura 27). Depois, para receber as respostas, o pesquisador envia aos participantes um link gerado pela plataforma, que permite acessar o puzzle e responder aos itens. Adicionalmente, pode-se optar por criar um grupo de controle para direcionar participantes aleatoriamente para outra aplicação existente. Seja na aplicação de teste ou na aplicação de controle, conforme novas respostas chegam, o sistema atualiza a tabela de participações (Figura 29). Esta tabela, por sua vez, dá acesso aos detalhes das respostas aos itens e aos questionários de fim de teste de cada participante.



Aplicar teste

Teste: Teste de programação

Nome aplicação: Aplicação com alunos de 3º ano

Visibilidade: Privado Público

Iniciar aplicação

Figura 27: Diálogo de início de aplicação

Fonte: Autor

As aplicações públicas são apresentadas no site da plataforma. Qualquer pessoa pode respondê-las. Já para acessar as aplicações privadas é necessário utilizar um link de participação como este, gerado pela plataforma:

```
https://ct.playerweb.com.br/?op=application&dataUrl=http://
api.ctplatform.playerweb.com.br/test-applications/public/data/04ecaf7f-3cb3-
4411-becc-8d688d13e962/<user_uuid>
```

Além do protocolo, o link é formado por mais duas partes: o domínio e os parâmetros. O domínio aponta para o endereço onde o puzzle está disponível na web. Os parâmetros *op* e *dataUrl*, dizem respeito à comunicação entre o puzzle e a plataforma. O puzzle deve interpretá-los. O parâmetro *op* com valor *application* indica que o puzzle deve ser aberto para uma aplicação real de teste. Já o parâmetro *dataUrl* contém o endereço de dados que o ambiente de puzzle deve requisitar à plataforma.

Ao requisitar os dados da aplicação, o puzzle deve substituir o parâmetro *<user_uuid>* do endereço de dados por um identificador aleatório e guardar esse identificador em algum meio de armazenamento local (cache por exemplo). Assim, em caso interrupção (atualização da página do navegador por exemplo), para continuar respondendo o teste com o mesmo usuário basta repetir a requisição de dados usando o identificador armazenado. E caso o desenvolvedor opte por iniciar uma nova participação (para caso de troca de participantes em um mesmo aparelho por exemplo) basta gerar um novo identificador. Esses identificadores devem seguir o formato SHA-256, por ser praticamente impossível de repetir e não depender de uma sequência gerada pela plataforma.

A resposta obtida da requisição ao endereço de dados (Figura 28) contém o teste e um conjunto de endereços de comunicação entre o puzzle e a plataforma. Os endereços de comunicação permitem realizar cinco tarefas: (1) direcionar o participante para o questionário ao fim de teste, (2) atualizar o progresso no teste, (3) enviar as respostas para a aplicação, (4) enviar a fonte onde o usuário partiu para chegar na aplicação e (5) armazenar dados adicionais do usuário.

```

▼ {participationId: 1024, lastVisitedItemId: 163,...}
  lastVisitedItemId: 163
  participationId: 1024
  ▶ test: {id: 36, name: "Teste COM aprendizado de depuração", items: [{id: 163,...}, {id: 164,...},...]}
  ▶ urlToEndOfTestQuiz: {url: "https://ctplatform.playerweb.com.br/end-of-test-quiz/1024"}
  ▶ urlToSendProgress: {method: "PUT", url: "https://api.ctplatform.playerweb.com.br/participations/public/save-progress",...}
  ▶ urlToSendResponses: {method: "POST",...}
  ▶ urlToSendSource: {method: "PUT", url: "https://api.ctplatform.playerweb.com.br/participations/public/save-source/1024",...}
  ▶ urlToSendUserData: {method: "POST",...}

```

Figura 28: Resposta recebida pelo puzzle ao requisitar os dados da aplicação

Fonte: Autor

Ao responder o puzzle, entregando os dados da aplicação, a plataforma pode direcionar o participante para um grupo de controle. Essa opção é útil caso o pesquisador queira investigar o efeito de diferenças nos testes sobre o desempenho dos participantes. Para utilizar essa funcionalidade, o pesquisador deve criar dois testes e duas aplicações (uma para teste e outra para controle). Em seguida deve indicar no cadastro de aplicação de teste qual é a aplicação de controle. Assim, ao acessar o link de participação, o usuário tem 50% de chance de ser direcionado para qualquer uma das duas aplicações.

CT Puzzle Test Platform Cassiano ▾

Plataforma / Aplicações / Aplicação de teste

Aplicação de teste

Nome:

Descrição e observações:

Teste: [Caça às moedas \(PROD\) ↗](#)

Grupo de controle:

Visibilidade: Privado Público

Link: [Copiar](#) [Acessar](#)

[Informações sobre o link de aplicação](#)

Participações (4)

Última resposta: **748d249c-1c3a-48dc-9c74-823cc1c01d18**, há 24 dias

[Atualizar](#) [Recalcular escores de todas as participações](#) [Baixar respostas](#)

Quando	Participante	Respostas	Ações
há 25 dias	a842eb8d-0d5e-4079-9b98-633792c66b36	9 respostas	Observar... 🔗 Remover
há um mês	710d357b-5296-4dce-bdb7-a1b32295fc4b	5 respostas	Observar... 🔗 Remover
há um mês	c7d7ee46-a2dd-488c-8123-df49a3401f2e	3 respostas	Observar... 🔗 Remover
há um mês	56f29de3-9997-48f0-8c90-18a05c5c493d	6 respostas	Observar... 🔗 Remover

[✓ Salvar](#) [< Voltar](#)

Figura 29: Tela de acompanhamento da aplicação

Fonte: Autor

Após o processo de entrada (eventual definição do grupo, recebimento dos dados e apresentação dos itens), se o usuário responde aos itens a comunicação entre o puzzle funciona corretamente, os escores são registrados na plataforma. Ainda, pode ser necessário ajustar a função de cálculo de escore posteriormente ao teste (para corrigir alguma discrepância de peso, por exemplo).

Para isso, a plataforma permite recalculiar o escore de todas as respostas dos participantes da aplicação.

Por fim, durante ou após a finalização do recebimento das respostas, o pesquisador pode querer analisá-las detalhadamente. Por isso, a plataforma permite exportar as respostas em um arquivo no formato CSV (*comma-separated-values* - valores separados por vírgulas) e importá-lo em ferramenta estatística de sua preferência. Cada linha do arquivo, contém uma resposta a um item, identificadores de participação, data de criação e as respostas aos questionários de fim de teste.

4.1.6 Módulo estatístico

O módulo estatístico oferece ao pesquisador diferentes representações dos dados das aplicações dos testes. Por meio de, indicadores, gráficos¹⁷ e tabelas, traz uma visão geral acompanhada de índices de dificuldade e a consistência e visões sumarizadas por tipo de dado. Ao mesmo tempo, exportar dados seletivamente, realizar cálculos avançados. As sessões seguintes explicam os componentes do módulo.

4.1.6.1 Seletor de aplicações

O seletor de aplicações lista todas as sessões de aplicação de teste disponíveis. O componente permite filtrar as aplicações por nome. As estatísticas posteriores são computadas sobre os dados da aplicação de teste escolhida.

¹⁷ Todos os gráficos apresentados foram criados com a biblioteca Plotly, em sua versão na linguagem JavaScript: <https://plotly.com/javascript/>

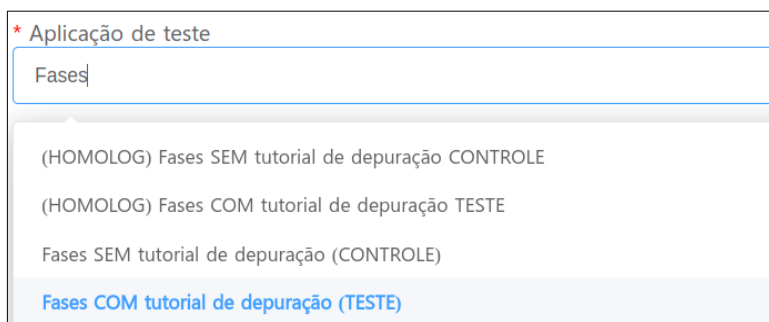


Figura 30: Seletor de aplicação de teste

Fonte: Autor

Ao entrar na tela pela primeira vez, o sistema pede ao usuário que selecione uma aplicação. Após selecionar, o sistema guarda o identificador da aplicação selecionada, de forma que na próxima abertura da tela de estatísticas, carrega a mesma aplicação automaticamente.

4.1.6.2 Tabela geral

A tabela de dados gerais lista os dados de usuário, as respostas obtidas durante o teste, as respostas às perguntas do questionário, e o score obtido. Além disso, os dados numéricos são sumarizados na parte inferior da tabela, com dados de média, variância e desvio padrão.

Respostas (100/1173)													Baixar	
data	usuario	Quantos anos você tem?	Qual é seu gênero?	Qual sua opinião sobre a dificuldade do jogo?	Qual sua opinião sobre o número de fases do jogo?	Qual	score_obt...							
6/3/2021, 2:03:51 PM	978e48b9-9c59-4632-b623-8ca5566f5b2e						10.00							
6/3/2021, 1:43:43 PM	ef207a2c-7855-4539-b27c-9a9fc2ba2eb6	12 a 18	FEMININO	MUITO DIFÍCIL	ADEQUADA	NORMA	7.50							
6/3/2021, 1:43:43 PM	ef207a2c-7855-4539-b27c-9a9fc2ba2eb6	12 a 18	FEMININO	MUITO DIFÍCIL	ADEQUADA	NORMA	10.00							
6/3/2021, 1:42:12 PM	88a98527-0dbd-4a67-95fe-e2f8a58c18b0	12 a 18	MASCULINO	MUITO FÁCIL	ADEQUADA	ATENÇÃO	10.00							
6/3/2021, 1:38:10 PM	cbe71a33-a0b7-4f84-8fc4-3f1deb0316a7	12 a 18	MASCULINO	MÉDIO	ADEQUADA	NORMA	10.00							
	participacao	item	order	contadorUsoLixeira	contadorUsoDe...	contadorUsoP...	contadorUsoS...	contadorReinicioFase	contadorTentativas	tempoEmSeg...	tempoInicio	score_max	score_obt...	
Média	1196.05	77.32	5.06	2.38	0.83	1.69	0.1	0.01	1.64	44.17	1623472066.55	10	9.38	
Variância	14414.89	111.63	11.56	43.46	34.53	6.69	0.29	0.02	4.33	11365.36	100159243164.103	0	1.5	
Desvio padrão	120.06	10.57	3.4	6.59	5.88	2.59	0.54	0.13	2.08	106.61	1000795.9	0	1.23	

Figura 31: Tabela de dados gerais e números sumarizados

Fonte: Autor

Ao abrir a tela, a tabela carrega poucas linhas, pois caso contrário a página do navegador pode se tornar lenta. Ao rolar para baixo, a tabela carrega mais linhas. Para obter todos os dados, basta acessar a tela de exportação. Antes de baixar os dados, o sistema permite alterar a formatação do cabeçalho e escolher as colunas que devem ser contidas no arquivo final.

4.1.6.3 Índices gerais

Os índices gerais (Figura 32) dão uma visão global sobre o teste e o desempenho dos participantes na sessão de aplicação. Quatro índices estão disponíveis: alpha de Cronbach, escore médio, avanço no teste e taxa de finalização.

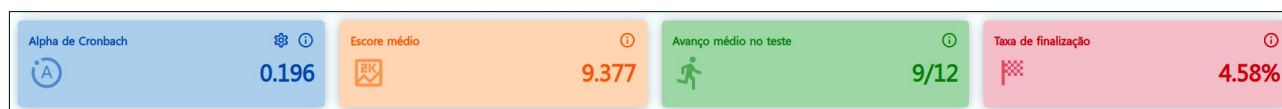


Figura 32: Índices gerais

Fonte: Autor

- O alpha de Cronbach: é um coeficiente de consistência interna do teste. Indica o quão relacionados os itens do teste são. O sistema permite adicionar e remover itens para testar a alteração do alpha.
- Escore médio: é a média aritmética dos escores obtidos por todos os participantes durante a aplicação do teste.
- Avanço médio no teste: apresenta até que ponto (item) do teste, os participantes chegaram em média. De acordo com o valor exemplificado na figura, a maioria dos participantes chegou até o item 9 de 12.
- Taxa de finalização: mostra o percentual de participantes que conseguiram realizar todos os itens do teste (finalizar). A comparação da taxa de finalização com o valor de avanço médio, permite ter uma ideia sobre a dificuldade dos itens não solucionados pela maioria. No exemplo, a maioria dos participantes chegou até o 9º item, mas menos de 5% finalizou o teste.

4.1.6.4 Análise de variáveis categóricas

As variáveis categóricas são geradas principalmente a partir das perguntas de questionário do tipo alternativas. Faixa etária, nível de dificuldade, gênero, são perguntas típicas. O gráfico permite comparar o número de usuários que responderam uma determinada categoria. Ao selecionar uma segunda categoria o gráfico relaciona as duas informações. No exemplo aplicado (Figura 33), o gráfico mostra que a maioria dos participantes tem entre 12 e 18 anos e que, dentre esses, há mais participantes do sexo feminino.

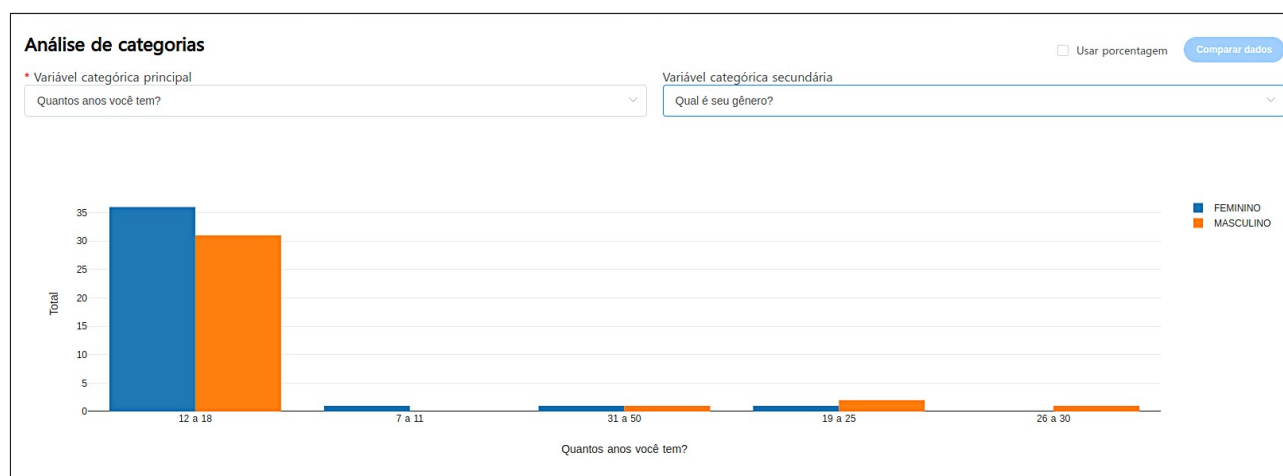


Figura 33: Gráfico de categorias por faixa etária e gênero

Fonte: Autor

Além de permitir comparar variáveis categóricas de uma mesma aplicação de teste, o sistema permite comparar dados categóricos de aplicações diferentes. Isso pode ser útil para comparar dados categóricos entre aplicações de teste e controle.

4.1.6.5 Diagrama de caixa

Na plataforma, os diagramas de caixa permitem analisar, quartis e extremos de dados numéricos da aplicação. O tratamento dos dados pelo pesquisador pode ser feito de diferentes formas:

1. Obter extremos e quartis de dados numéricos: comparar valores numéricos gerais: score, tempo de resolução, número de cliques, etc.

2. Comparar extremos e quartis entre de itens: é possível comparar, por exemplo, o escore obtido em cada item de teste.
3. Comparar dados numéricos relacionando-os com dados categóricos: qual é o escore médio dos participantes de uma determinada faixa etária? Quanto tempo demoraram para responder o teste, os participantes de acordo com o nível de dificuldade atribuído por eles ao teste? A divisão dos extremos e quartis por dados categóricos permite responder a esse tipo de pergunta.
4. Comparar dados entre aplicação de teste e controle (Figura 34): todas as operações anteriores podem ser realizadas comparando-se os dados duas aplicações de teste. O componente de seleção da aplicação comparativa sugere uma aplicação de controle configurada previamente.

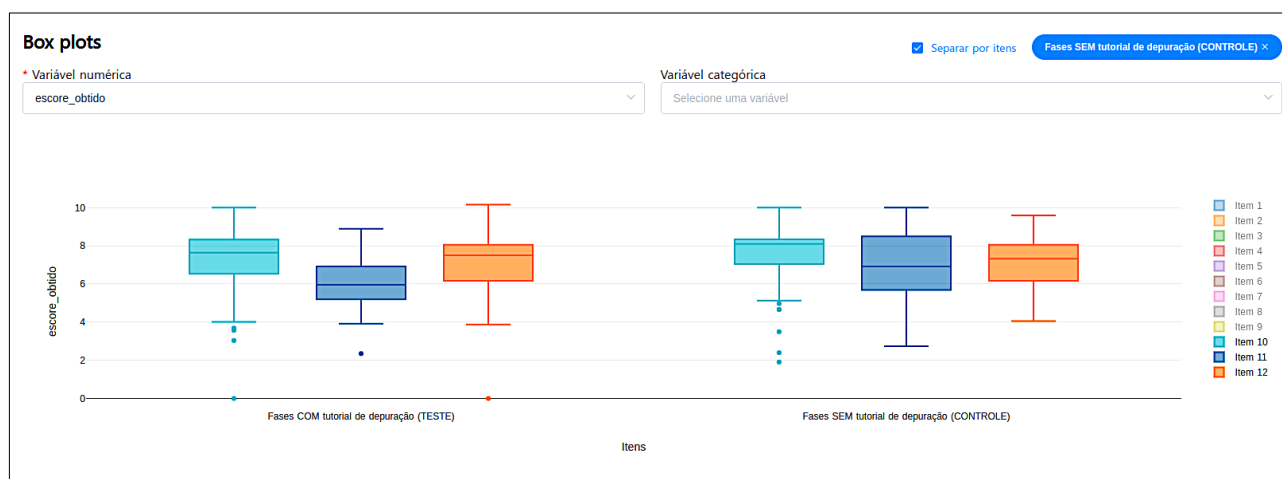


Figura 34: Diagramas de caixa comparando escore obtido para itens em aplicação de teste e controle

Fonte: Autor

4.1.6.6 Gráfico de Curva Característica do Item (CCI)

Os gráficos de teoria da resposta ao item relacionam a probabilidade de um determinado indivíduo acertar uma questão dado o nível de habilidade deste indivíduo. Essa relação pode ser expressa seguindo diferentes modelos. O modelo utilizado foi o modelo de 2 parâmetros, que considera a dificuldade e discriminação dos itens. Este modelo é geralmente utilizado para itens dicotômicos (em que há resposta certa ou errada). Contudo, no caso, as respostas tem característica

contínuas (consideram acerto parcial), considerou-se a proximidade das respostas em relação a resposta ideal (100% correta).

Com isso, gerou-se, para cada item, uma estimativa de probabilidade de acerto total a cada nível de habilidade. Para estimar esse nível de habilidade, utilizou-se o escore obtido nos itens mais difíceis. A partir dessas estimativas (Figura 35, gráfico da esquerda), encontrou-se por aproximação a curva característica de probabilidade mais adaptada aos dados. Dessa forma, embora os parâmetros sejam aproximados, as curvas dos itens (Figura 35, gráfico da direita) ilustraram a diferença de dificuldade e capacidade de discriminação entre os itens.

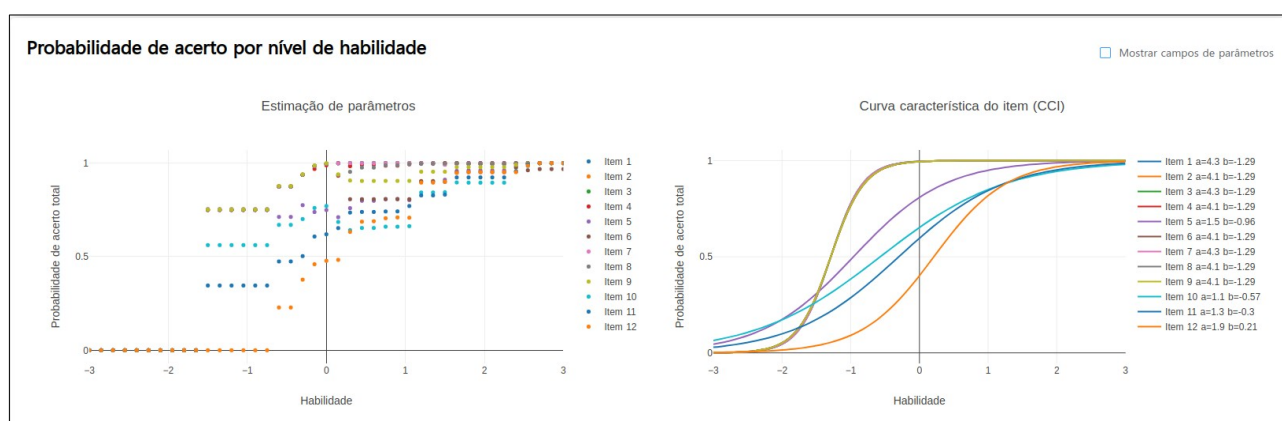


Figura 35: Gráficos de TRI

Fonte: Autor

O componente de gráfico de estimação e curva característica dos itens possui campos que permitem alterar manualmente os parâmetros de entrada da expressão. Como essa função serve somente para testes, não é mostrada por padrão. Para mostrar os campos de parâmetros, o pesquisador deve marcar o campo presente na parte superior direita do componente.

4.1.6.7 Interpretador R

O interpretador R (Figura 36) é uma funcionalidade útil para casos em que o pesquisador conhece a linguagem R, precisa realizar cálculos complexos e não tem acesso a (ou não deseja utilizar) ferramentas dedicadas para a codificação na linguagem R. Através do editor, o pesquisador pode utilizar funções da linguagem não disponíveis de forma fácil na plataforma. Assim, essa

função é de uso mais complicado, porém possível caso o pesquisador não queira utilizar uma ferramenta indicada como RStudio¹⁸.

Ao abrir a tela, o pesquisador pode escolher entre as variáveis numéricas a serem interpretadas. Essa seleção é feita por meio de um componente seletor, que ao ser alterado, adiciona o valor no editor de código. Nesse momento o código é automaticamente interpretado, revelando o conteúdo da variável.

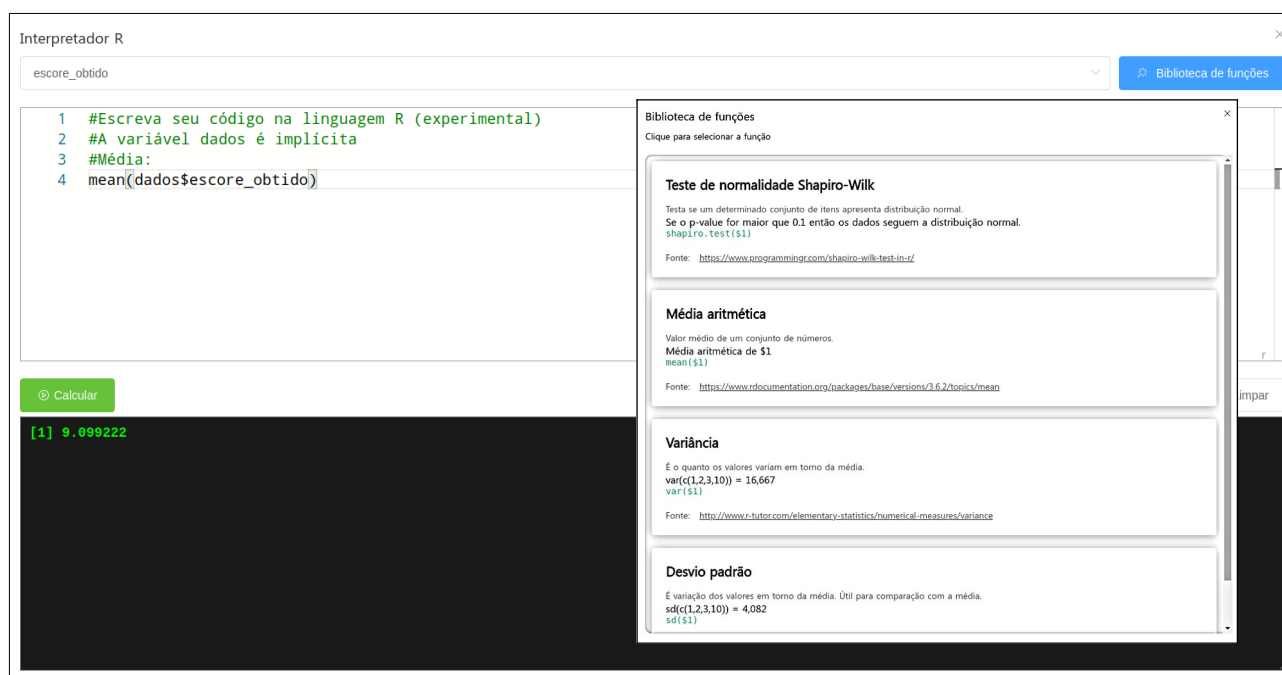


Figura 36: Tela de entrada de código R e biblioteca de funções

Fonte: Autor

Na verdade, a tela não interpreta o código R propriamente. Ao invés disso, ocorre o envio do texto do código para o servidor da plataforma. Na plataforma, o código é executado em um interpretador R. O resultado volta e é apresentado ao pesquisador.

Funções auxiliares de média, desvio padrão, variância e teste de normalidade estão disponíveis em uma biblioteca de funções. Ao inserir uma função, comentários no código auxiliam o pesquisador sobre a interpretação do resultado. Contudo, essa biblioteca é inicial e cabe ao

18 O RStudio é o editor de código especializado na linguagem R: <https://www.rstudio.com/>

pesquisador pesquisar sobre funções avançadas ou baixar os dados para utilizar em outra ferramenta que domine.

4.1.6.8 Visualizador de respostas abertas

O visualizador de respostas abertas (Figura 37) é dedicado à análise do conteúdo de variáveis textuais. A principal fonte de variáveis textuais é o questionário aplicado aos participantes que finalizam o teste.

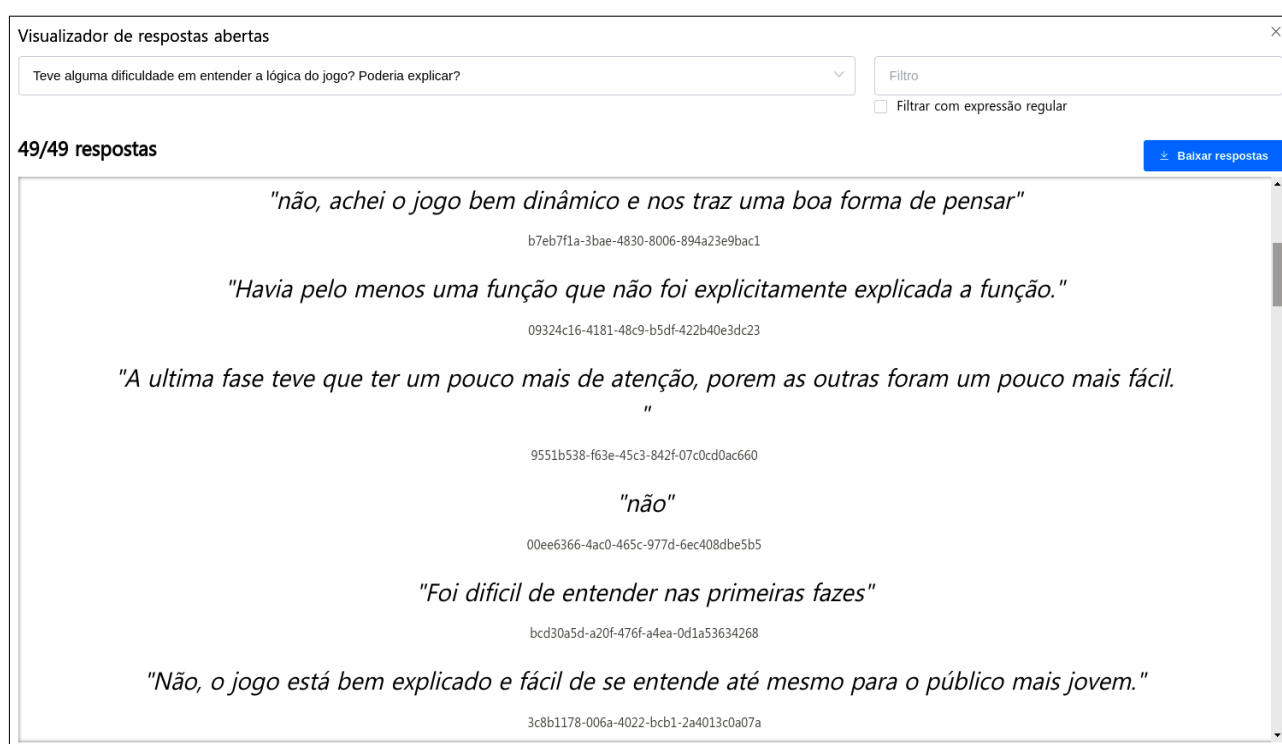


Figura 37: Visualizador de respostas abertas

Fonte: Autor

O visualizador de respostas abertas contém um seletor de variáveis. Ao selecionar a variável de interesse, as respostas são apresentadas acompanhadas do identificador único do usuário. Além do seletor, a tela contém um filtro de respostas que suporta a entrada de expressões regulares e um botão de download, para baixar as respostas.

4.2 MECÂNICA DE PROGRAMAÇÃO COM ROBÔ

Para elaborar testes, o aplicador precisa de um conjunto de itens. A criação de um banco de itens demanda que os puzzles possam ser configurados. Essa flexibilidade deve permitir apresentar diferentes cenários: mais ou menos obstáculos; uma figura de Tangram mais complexa; diferentes pontos em um puzzle de ligação de pontos para formar figuras, etc.

Para atender esse requisito, foi implementada uma mecânica de programação. Essa mecânica foi inspirada na mecânica de programação do CT Puzzle Test, possuindo semelhanças mas também diferenças notáveis em relação aquela versão (Figura 38). Similarmente, possui um mapa de obstáculos e objetivos. Mas a movimentação do personagem, tratamento de acertos, tutoriais, estruturas de repetição e possibilidade de edição do programa são diferentes.

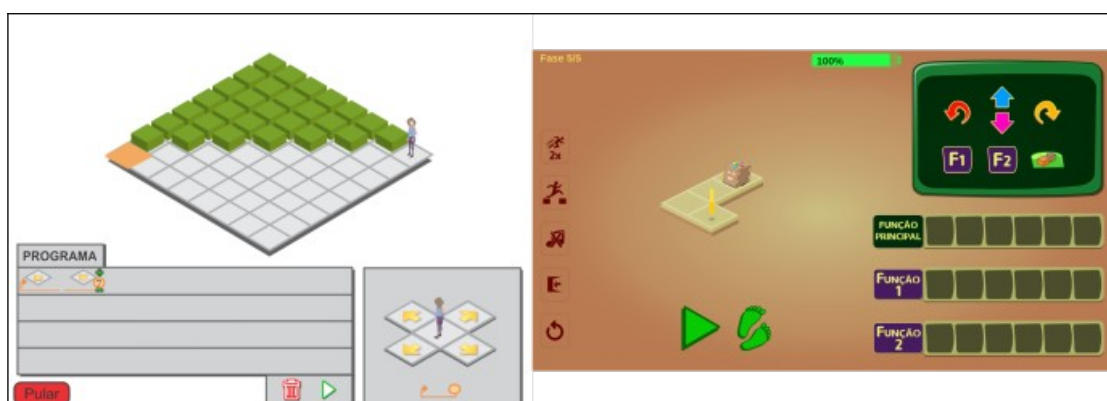


Figura 38: Comparação entre puzzles de programação

Fonte: Autor

O puzzle tem quatro área de interação: um menu lateral, uma área de botões, uma caixa de comandos e uma área de programa (Figura 39):

1. Menu lateral: permite mudar a velocidade (1x e 2x), pausar a música de fundo, reiniciar a fase, sair do jogo e pular fase.
2. Área de botões: tem botões de início de movimento (executar), de passo a passo (depurar) e de interrupção do movimento (parar).

3. Caixa de comandos: tem comandos direcionais (frente, ré, giro para esquerda e giro para direita), funções (F1 e F2) e condicional (condicional para bloco à frente).
4. Área de PROGRAMA: exhibe três funções que recebem os comandos arrastados da caixa, sendo que a primeira função é a principal.



Figura 39: Áreas da interface do puzzle de programação com robô




Fonte: Autor

O Quadro 4 ilustra as funcionalidades presentes no jogo, com imagem de exemplo e descrição de cada funcionalidade.

Quadro 4. Funcionalidades implementadas na fase de programação com RoPE

Imagem	Funcionalidade
	<p>Movimentação RoPE</p> <p>Por utilizar um personagem de robô inspirado no RoPE¹⁹, o personagem tem os seguintes movimentos: para frente, para trás, giro em 90° para esquerda ou para direita em torno do próprio eixo.</p>
	<p>Tutorial Interativo</p> <p>Para explicar o funcionamento da mecânica ao usuário, esta mecânica apresenta tutoriais interativos: os movimentos são indicados e o usuário deve repeti-los para completar o tutorial.</p>

19 <http://lite.acad.univali.br/pt/rope-brinquedo-de-programar/>

	<p>Funções Esta mecânica permite o uso de funções. Há a função principal, e as funções F1 e F2. São úteis para quando não há mais espaço no programa principal ou para isolar lógicas.</p>
	<p>Recursividade A recursividade é uma consequência da funcionalidade anterior. Ao permitir a chamada de funções dentro das próprias funções, é possível resolver com poucas peças, desafios onde o personagem precisa andar muito.</p>
	<p>Desvio condicional É possível adicionar comportamentos condicionais ao personagem usando blocos de condição sobre os blocos do programa. Por exemplo: para virar para direita, somente se tiver um bloco na frente, utiliza-se o desvio condicional de bloco (ver figura). Isso permite desviar de obstáculos.</p>
	<p>Edição pontual de código Esta mecânica permite arrastar um comando de uma posição para outra do programa. Isso facilita a edição do código, sem que seja necessário apagar e reinserir peças, facilitando o uso.</p>
	<p>Depuração: passo a passo A depuração é uma função que auxilia na descoberta e correção de erros. No caso desta mecânica, um botão de “passo a passo” permite que os comandos do programa sejam executados e pausados em sequência. Ao completar cada movimento, o botão de passo e a peça referente ao próximo movimento piscam indicando o que irá ocorrer ao clicar para continuar. É possível pausar um programa em andamento e continuar um programa pausado (em modo de depuração).</p>

Fonte: Autor

O jogo pode ser acessado a partir de computador ou dispositivo móvel (celular). Para computador, o acesso ocorre via navegador de internet. Por meio de celular, o acesso se dá a partir de um aplicativo disponível para plataforma Android. Em ambos os casos, o puzzle emite efeitos sonoros, que auxiliam na experiência de uso. Os efeitos sonoros ocorrem durante eventos de captura de moeda, movimentação do robô, colisões, e interação do usuário com a interface. Já em dispositivo móvel, além de efeitos sonoros, são emitidos efeitos de vibração do aparelho, que ocorrem durante encaixes de blocos e colisão do personagem.

4.2.1 Configuração dos itens

A configuração do puzzle de programação deve seguir os atributos da classe de sua mecânica. Os atributos são de quatro tipos: de cenário, de energia do personagem, de valores

esperados de resposta, e de tutoriais e mensagens (Quadro 5). As variáveis de cenário definem o mapa, obstáculos e posição do personagem. As variáveis de energia definem o nível de bateria do robô, os custos de movimentação e colisão. Os valores esperados, ao serem atribuídos aos itens, servem para comparação com atributos das respostas. Por fim, as variáveis de tutoriais e mensagens permitem configurar dados que não são propriamente do jogo, mas falas auxiliares e de aviso da interface.

```

class ItemProgramacao {

    // Cenário
    mapa!: Mapa[][];
    obstaculos!: Obstaculo[][];
    face!: Face;
    x: number = 0;
    y: number = 0;

    // Energia
    nivelBateria: number = 10
    nivelMaximoBateria: number = 10
    custoBateriaEmCadaMovimento: number = 1
    ganhoBateriaAoCapturarPilha: number = 2

    // Valores esperados
    comandosEsperados!: Comando[];
    tempoEsperado!: number
    tentativasEsperadas!: number

    // Tutoriais e mensagens
    acoesTutorial: AcaoTutorial[] = []
    falasAntesDeIniciar: string[] = []
    mensagemAoPularFase: string = 'Tem certeza que deseja pular essa fase?'
    mensagemAoSairDoJogo: string = 'Tem certeza de que deseja sair?';
}

```

Quadro 5. Classe de da mecânica de programação

O código do Quadro 6, exemplifica a instanciação de item utilizando algumas dessas variáveis. Nesse caso o item é um tutorial que ensina como funciona a seta para trás. Primeiro declara os comandos esperados, que é somente uma vez para trás (DOWN). Em seguida declara o mapa e os “obstáculos”, onde um é o próprio objetivo para finalizar a fase (moeda). Depois informa a direção da face e o posicionamento do personagem no mapa. Por fim, cria uma lista de ações tutoriais necessárias para finalizar o item. O usuário é obrigado a cumprir essas ações para poder avançar. O resultado dessa instanciação é o personagem de costas para a moeda (Figura 40).


```

function createItem() {
  let item = new ItemProgramacao();
  item.comandosEsperados = ['DOWN']

  item.mapa = [
    ['tile'],
    ['tile'],
  ]
  item.obstaculos = [
    ['coin'],
    ['null'],
  ]

  item.face = "down"
  item.x = 0
  item.y = 1

  item.acoesTutorial = [
    {elemento:"arrow-down",acao:'click'},
    {elemento:"btn-step",acao:'click'}
  ]
  return item;
}

```

Quadro 6. Instanciação do item de programação

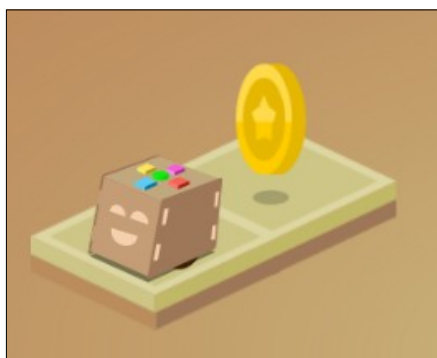


Figura 40: Resultado da instanciação do item

4.3 MECÂNICA DE REALIDADE AUMENTADA

Viana (2021) desenvolveu uma interface de realidade aumentada, chamada RoPE AR e conectou-a à plataforma. A aplicação é um ambiente onde o usuário usa cartões de papelão para programar e depurar os movimentos de um robô em mapas projetados (Figura 41). O RoPE AR visa verificar a qualidade das soluções criadas pelas crianças em cada programa, ao tentar mover o robô até a maçã. Um relato qualitativo do desenvolvedor sobre a plataforma pode ser conferido no apêndice I.

O sistema roda em um aplicativo de celular, que captura o ambiente utilizando a câmera. Ao

4.4 COMPONENTES DE SOFTWARE

Para ilustrar a organização dos elementos de software, é útil apresentar um diagrama de componentes. O sistema da plataforma e do puzzle apresentado tem cinco componentes: (1) servidor web, (2) banco de dados, (3) componente de comunicação com a linguagem R, (4) puzzle de programação e (5) um aplicativo Android. O servidor web, por sua vez, contém dois componentes: a parte de visualização da aplicação, nomeada no diagrama como *CT Puzzle Platform Frontend*, e a parte de comunicação com o banco de dados, nomeada no diagrama como *CT Puzzle Platform Backend* (Figura 43).

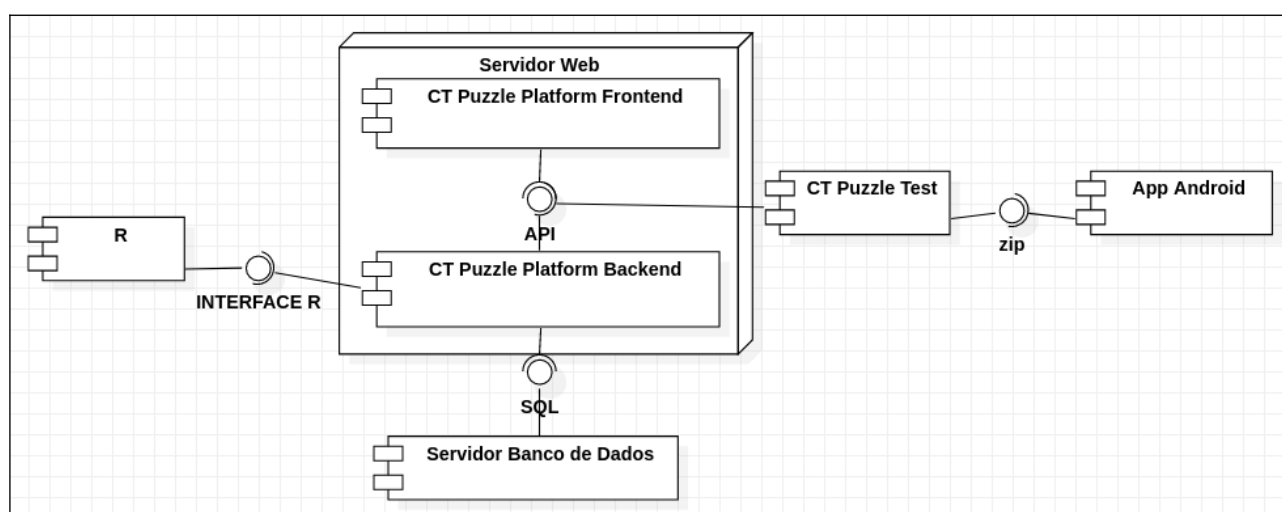


Figura 43: Diagrama de componentes

Fonte: Autor

Descrevendo pontualmente os componentes, temos:

- R: A linguagem R foi instalada no servidor para permitir auxiliar na execução de cálculos.
- Servidor de Banco de Dados: Os dados de mecânicas, itens, testes, aplicações e respostas são persistidos em um banco de dados relacional.
- *CT Puzzle Platform Backend*: É onde está a lógica da plataforma. Disponibiliza uma API, para consulta e inserção de dados na base. Também se comunica com o R, utilizando suas funções para geração de gráficos.

- *CT Puzzle Platform Frontend*: Apresenta as telas do sistema para os pesquisadores. Acessa a API do *backend* para inserir e buscar recursos.
- *CT Puzzle Test*: Implementação do puzzle propriamente dito. Foi implementada uma fase de programação como referência. Esse componente acessa a API do *backend* para solicitar o cenário de puzzle de cada item do teste. Está disponível tanto para acesso via navegador de internet como por aplicativo Android (próximo item).
- *App Android*: O *CT Puzzle Test* também funciona como um game *offline* em aparelhos com o sistema Android. A atualização da versão do puzzle no aparelho é feita sem a necessidade de atualizar o aplicativo. Cada nova versão publicada do puzzle recebe um número único. Ao abrir o aplicativo, o sistema verifica se o número local é diferente do remoto e, caso for, baixa um arquivo compactado com a nova versão. Depois, extrai o arquivo, substituindo a versão antiga.

4.5 CONSIDERAÇÕES

Este capítulo apresentou a plataforma de testes e os puzzles que a utilizam para configurar itens e guardar dados das interações. Iniciou com uma explicação geral, ilustrada em diagrama casos de uso, seguido do detalhamento de cada caso, com descrições e imagens das telas do sistema. Depois, mostrou dois puzzles implementados para se comunicar com a plataforma: um puzzle de programação virtual e um ambiente de realidade aumentada. Por fim, um diagrama de componentes ilustrou as partes do sistema e a relação entre elas.

No próximo capítulo, a plataforma é testada por meio de uma aplicação real com usuários. Um teste de programação foi elaborado e dividido entre grupo de teste e grupo de controle. Procurou-se avaliar se há efeito da depuração sobre o desempenho dos participantes. O conteúdo dos testes, a forma de divulgação e aplicação são explicados de forma a ilustrar um processo de uso da plataforma e resultados que se pode obter através dela.

5 RESULTADOS

Para testar a plataforma, foi realizado um experimento com voluntários de diferentes faixas etárias, que se dispuseram a tentar resolver com um conjunto de desafios de programação. O experimento não é um guia definitivo para outras implementações, mas ilustra o funcionamento da plataforma, em aplicação real. A base de dados gerada durante o experimento permitiu testar aspectos relacionados ao módulo estatístico.

O experimento consistiu em um teste, com itens do puzzle descrito no capítulo 4.2. Para testar o recurso de divisão de grupos, foi feita a divisão dos participantes em grupo de teste e controle. Como fator para a divisão de grupos, foi escolhido o uso de depuração, que é um recurso disponível no puzzle de programação (ambiente do teste).

As seções seguintes detalham a forma de divulgação, o contexto de aplicação, o conteúdo do teste, as características gerais dos participantes e os resultados obtidos. A avaliação sobre os resultados do teste é quantitativa e qualitativa. A avaliação sobre a plataforma é totalmente qualitativa.

5.1 TESTE DE DEPURAÇÃO

Programar envolve cometer erros. Ao indivíduo que constrói o programa, cabe a tarefa de corrigir esses erros, a fim obter o resultado desejado. A depuração facilita a correção, englobando o entendimento, teste e localização do problema (MCCAULEY et al., 2008). A execução passo a passo das instruções do programa faz parte da depuração e permite entender em que estado do programa a inconsistência acontece.

A função de execução passo a passo é disponível no puzzle de programação, sendo escolhido como tema para o experimento com a plataforma. O teste investigou se houve diferença de desempenho entre os participantes que receberam ou não instruções sobre o uso de depuração. A plataforma promoveu a divisão de grupos, sorteando os participantes aleatoriamente em grupo de teste e controle:

- Grupo de teste: recebeu explicações, através de tutoriais, sobre o funcionamento da instrução de passo a passo (depuração). As instruções de depuração consistiam em clicar no botão de passo a passo tantas vezes quantas fossem necessárias para o personagem completar o desafio.
- Grupo de controle: recebeu instruções iguais as instruções do grupo de teste *exceto* pelo fato de que ao invés de clicar diversas vezes no botão passo a passo, precisavam apenas clicar uma única vez no botão de *play*. Ao clicar, o personagem completava todos os passos, sem pausa, ou seja, sem necessidade da intervenção do usuário para efetuar o movimento seguinte.

Essa divisão permite formular diversas perguntas, como:

- Os participantes do grupo de teste utilizaram mais a depuração do que os participantes do grupo de controle?
- Os participantes que utilizaram a depuração cometeram menos erros?
- Qual a diferença entre o tempo resolução dos problemas entre os dois grupos?

Obviamente, há mais perguntas que podem ser feitas, mas essas servem para validar a utilidade da plataforma em auxiliar a na investigação pesquisa.

5.1.1 Divulgação e contexto de aplicação

A divulgação do teste se deu por dois meios: redes sociais e convite de professores a seus alunos. A divulgação em redes sociais ocorreu por compartilhamento em grupos de professores e relacionados a Pensamento Computacional. Já o convite direto de professores a seus alunos ocorreu durante aulas e por meio de tarefas de disciplina. Um termo de consentimento (APÊNDICE A – TERMO DE CONSENTIMENTO), esclarecendo o caráter voluntário da participação, acompanhou essas divulgações.

Dado o contexto de isolamento social, as aplicações aconteceram remotamente. As características do ambiente do puzzle (acessível por computador ou celular) contribuíram para esse

tipo de aplicação. Além disso, as participações não foram supervisionadas. Cada participante, após receber o endereço de participação, pode realizar o teste ao seu tempo, pedir ajuda ou desistir durante o teste caso optasse por fazê-lo. As participações ocorreram de forma livre durante o final de maio até metade de junho e mais um dia de participações em meados de julho. As participações do último dia ocorreram através de divulgação durante uma palestra online.

5.1.2 Participantes do teste

A faixa etária predominante do teste foi de adolescentes de 12 a 18 anos. Isso se deu porque a maioria dos participantes veio através de uma divulgação da avaliação em uma escola da cidade de Itajaí por um professor. Os participantes tem contato com programação, pois essa é uma disciplina ministrada. No entanto, há participações de outras faixas etárias, como revelam os dados da Tabela 1, obtidos a partir da plataforma:

Tabela 1. Faixas etárias dos participantes que finalizaram o teste e responderam o questionário

Grupo / Faixa etária	7 a 11	12 a 18	19 a 25	26 a 30	31 a 50	Total
Teste	1	70	5	1	3	80
Controle	0	67	2	2	4	75
Total	1	137	7	3	7	155

Fonte: Autor

Outro dado pertinente é a familiaridade dos participantes em relação a jogos e tecnologia. Isso pode servir para verificar se esse fator altera o desempenho dos participantes no teste. A pergunta que norteou essa divisão foi: O quão familiar ou fluente você se considera a respeito de jogos e tecnologia em geral?.

Tabela 2. Familiaridade com jogos e tecnologia. Inclui apenas participantes que finalizaram o teste.

Grupo / Familiaridade com tecnologia	Nada	Pouco	Normal	Muito	Totalmente	Total
Teste	2	11	37	20	8	78
Controle	0	7	32	21	12	72
Total	2	18	69	41	20	150

Fonte: Autor

Portanto, percebe-se que os participantes tem contato com tecnologia. A maioria (73%) considera ser normal ou muito familiar. Já nos extremos, apenas 2 responderam não ter nenhuma familiaridade e 20 responderam ser totalmente familiares.

5.1.3 Conteúdo do teste

O experimento consistiu em responder uma sequência de 12 itens e um questionário ao fim do teste. Dentre os itens, 5 itens eram tutoriais e 7 itens eram de resolução. O questionário conteve perguntas relacionadas à faixa etária, opiniões sobre o jogo e fluência em tecnologia. Entre os itens, a diferença era apenas sobre a apresentação da função de depuração. O mesmo questionário foi apresentado para ambos os grupos.

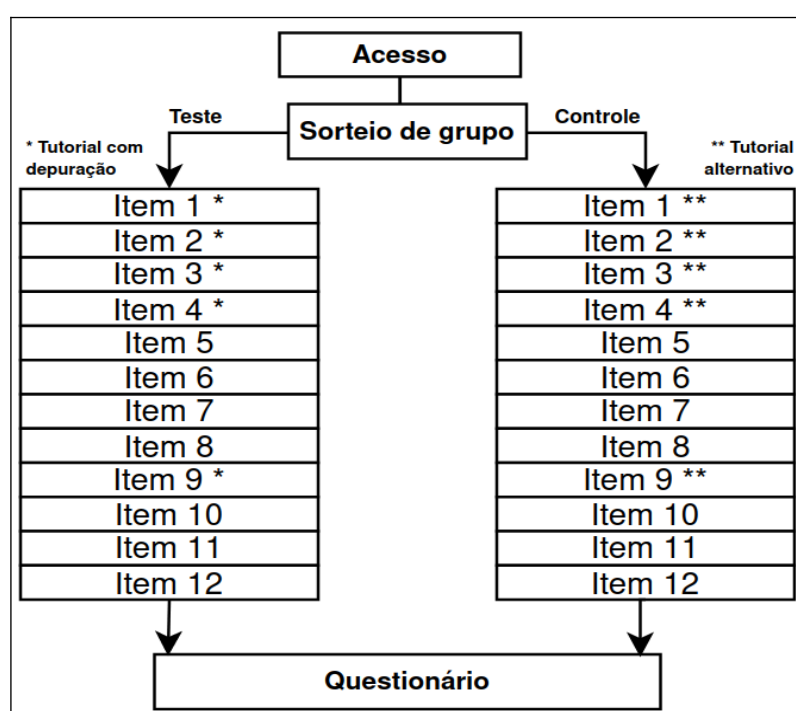


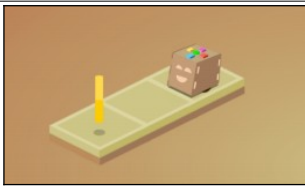
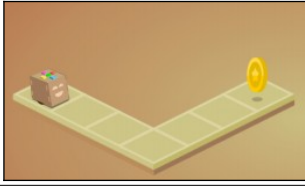
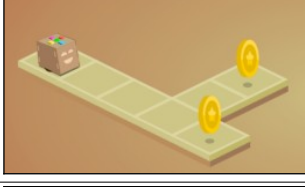
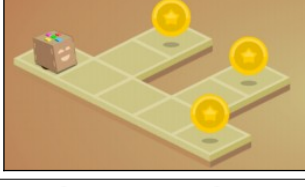
Figura 44: Visão geral do experimento

Fonte: Autor

A dificuldade dos itens (Quadro 7) é crescente, apresentando-se tutoriais básicos no início. Os quatro primeiros itens explicam os movimentos do personagem: andar para frente, para trás, giro para direita e para esquerda. Depois, seguem quatro itens que o usuário deve resolver sem auxílio.

Por fim, é apresentado um item de tutorial sobre o uso de funções, seguido por três itens mais difíceis.

Item	Descrição	Comandos necessários	Movimentos necessários
	Item 1 (tutorial) Nesta fase, o personagem deve apenas andar uma vez para frente. Antes de iniciar, é apresentado uma fala, explicando sobre o funcionamento do jogo.	1	1
	Item 2 (tutorial) Semelhante ao item anterior, explica a movimentação do personagem para trás.	1	1
	Item 3 (tutorial) Explica como funciona o comando de giro para direita. Nesse caso, é preciso utilizar dois passos: girar para direita e andar para frente.	2	2
	Item 4 (tutorial) Semelhante ao item anterior, porém o giro é para esquerda.	2	2
	Item 5 Nesse item, o personagem precisa dar dois passos para frente. Não é um item de tutorial.	2	2
	Item 6 Nesse item, o personagem precisa dar dois passos para frente, girar para esquerda e avançar para frente.	4	4
	Item 7 Nesse item, o personagem precisa dar um passo para frente, girar para esquerda e avançar duas vezes. É semelhante ao item 6, mas apresenta obstáculos para distração.	4	4
	Item 8 Esse item demanda o uso de dois giros: para direita e para esquerda. Assegura o entendimento dessas duas funções.	5	5

	Item 9 (tutorial) Esse item explica o funcionamento das funções. A tarefa é simples: andar duas vezes para frente. Mas o usuário é obrigado a usar funções para resolver.	4	2
	Item 10 Esse item pode ser resolvido com menos comandos do que o número de passos. Basta utilizar uma função para repetir os passos para frente.	6	7
	Item 11 Nesse item, o personagem precisa pegar duas moedas. O caminho com menos passos é capturar a moeda à frente, e depois voltar para seguir a bifurcação.	7	8
	Item 12 É o último item e o mais difícil. É imperativo o uso de funções para resolver o problema. O movimento de entrada e saída das “ruas” à esquerda do personagem pode ser isolado em funções.	11	16

Quadro 7. Itens do teste

Ao finalizar ou sair do jogo, o participante recebe um questionário (Quadro 8) a partir de uma página da plataforma. O questionário é formado por 12 perguntas, quatro grupos:

- Duas questões demográficas
- Quatro questões opinativas a respeito do jogo
- Três questões sobre a experiência do respondente com jogos e tecnologia em geral
- Três questões sobre o estado do respondente e do ambiente no momento da participação: pergunta sobre atenção, dificuldade e ajuda externa

Pergunta	Grupo	Tipo	Obrigatória	Opções
Quantos anos você tem?	Demográfica	Opções	Sim	menos de 7, 7 a 11, 12 a 18, 19 a 25, 26 a 30, 31 a 50, 51 a 65, mais de 65
Qual é seu gênero?	Demográfica	Opções	Não	MASCULINO, FEMININO
Qual sua opinião sobre a dificuldade do jogo?	Opinativa sobre o jogo	Opções	Sim	MUITO FÁCIL, FÁCIL, MÉDIO, DIFÍCIL, MUITO DIFÍCIL
Qual sua opinião sobre o número de fases do jogo?	Opinativa sobre o jogo	Opções	Sim	MUITO POUCAS, POUCAS, ADEQUADA, MUITAS, FASES DEMAIS

Qual foi seu nível de atenção ao resolver as fases?	Estado do respondente e ambiente	Opções	Sim	DESATENTO(A), POUCO ATENTO(A), NORMAL, ATENTO(A), MUITO ATENTO(A)
Teve alguma dificuldade em entender a lógica do jogo? Poderia explicar?	Estado do respondente e ambiente	Texto longo	Não	
Já jogou outros jogos parecidos? Quais?	Experiência em jogos e tecnologia	Texto longo	Não	
Você resolveu as fases sozinho(a) ou com alguém?	Estado do respondente e ambiente	Opções	Sim	FIZ SOZINHO(A), FIZ ALGUMAS FASES JUNTO COM OUTRA PESSOA, FIZ TODAS AS FASES JUNTO COM OUTRA PESSOA
O quanto familiar ou fluente você se considera a respeito de jogos e tecnologia em geral?	Experiência em jogos e tecnologia	Opções	Sim	Nada, Pouco, Normal, Muito, Totalmente
Com que frequência você costuma jogar (qualquer tipo de jogo)?	Experiência em jogos e tecnologia	Opções	Sim	NUNCA, RARAMENTE, AS VEZES, FREQUENTEMENTE, SEMPRE, PRATICAMENTE TODOS OS DIAS
O que mais gostou no jogo?	Opinativa sobre o jogo	Texto longo	Não	
O que menos gostou no jogo?	Opinativa sobre o jogo	Texto longo	Não	

Quadro 8. Itens do questionário

5.1.4 Plataforma como apoio à investigação de hipóteses

Durante a formulação dos testes foram levantadas perguntas. Investigou-se a influência da sugestão de uso de depuração (grupo de teste) em relação ao caso semelhante sem tutorial de depuração (grupo de controle). Verificou-se a possibilidade de responder essas perguntas a partir de análise de dados feita com a plataforma. Assim, testou-se a utilidade da ferramenta para produzir as informações úteis de pesquisa. A seguir descreve-se as perguntas e os recursos da utilizados para tentar respondê-las.

Pergunta 1: Os participantes do grupo de teste utilizaram mais a depuração do que os participantes do grupo de controle?

Esta resposta pode ser obtida através do diagrama de caixa do módulo estatístico. A comparação de dados entre aplicações de teste e controle, mostra que o grupo de teste utilizou mais a depuração (Figura 45). Observa-se que a variável contadora de depuração teve uma mediana de dois usos por item, enquanto o grupo de controle teve uma mediana de zero. Isso confirmou a hipótese inicial de que os participantes do grupo de teste utilizariam mais a depuração.

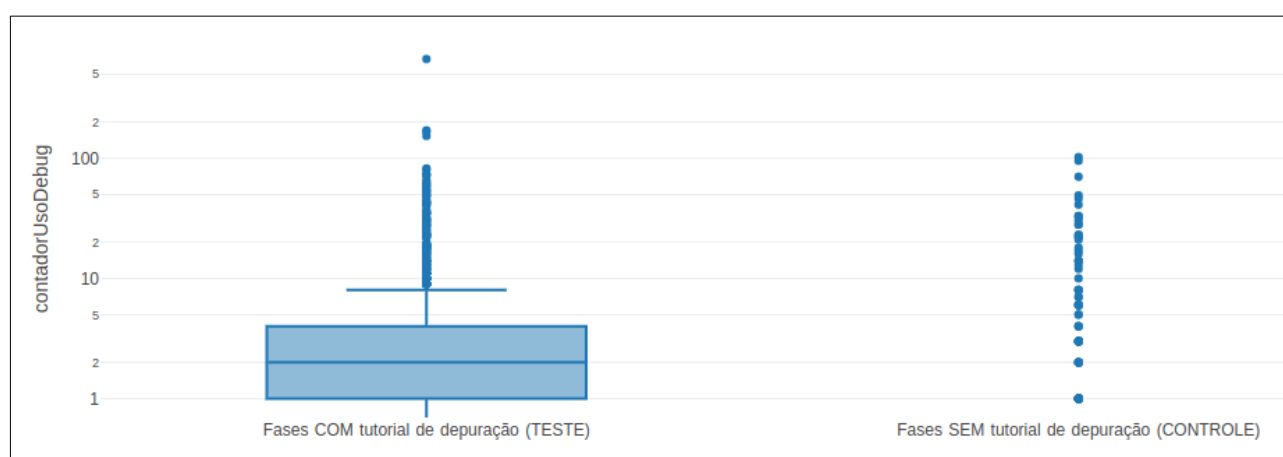


Figura 45: Diagrama de caixa comparando os usos do recurso de depuração entre aplicação de teste e aplicação de controle

Fonte: Autor

Pergunta 2: Os participantes que utilizaram a depuração cometeram menos erros?

Para responder essa pergunta utilizou-se o contador de uso da lixeira. Intuitivamente, quanto mais o usuário precisou usar a lixeira, significa que cometeu mais erros e precisou corrigi-los. No último item do teste a mediana de usos da lixeira foi de 5 para o grupo que recebeu tutoriais de depuração e de 8.5 para o grupo de controle. No entanto, essa diferença não foi significativa. Embora no grupo de controle tenha havido uma variação menor, os seus extremos dos quartis do diagrama de caixa (Figura 46) ficam contidos entre os quartis do grupo de teste.

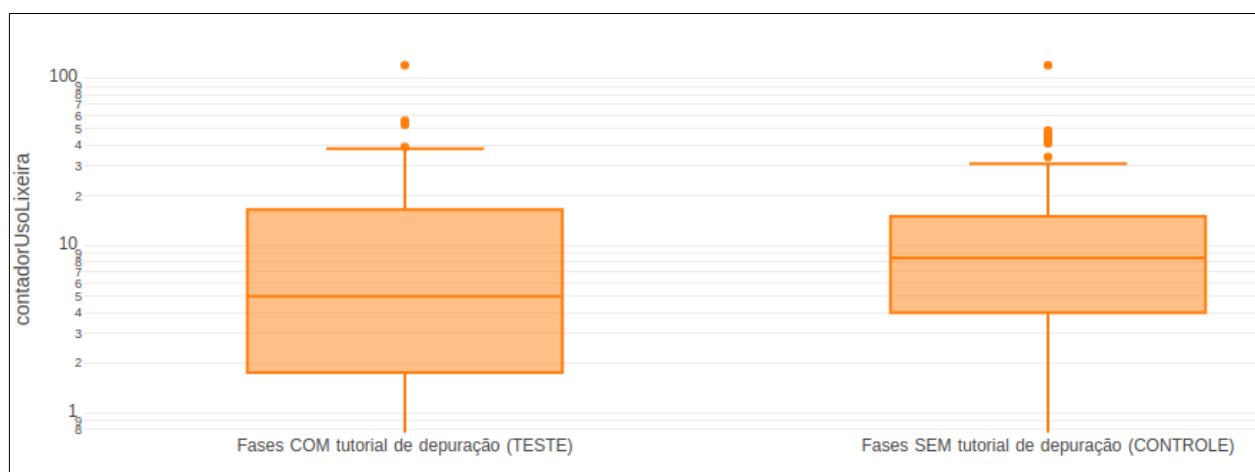


Figura 46: Diagrama de caixa comparando os usos do recurso de lixeira entre aplicação de teste e aplicação de controle para o último item de teste.

Fonte: Autor

Pergunta 3: Qual a diferença entre o tempo resolução dos problemas entre os dois grupos?

Para responder essa pergunta foi observado a informação sobre o contador de tempo em segundos médio. A mediana foi de 17 segundos para o grupo de teste e de 14 segundos para o grupo de controle. Ou seja, os usuários que precisaram usar a depuração demoraram mais tempo para resolver. Contudo, a interposição dos quartis ilustrada no gráfico (Figura 47), indica que essa diferença não foi significativa.

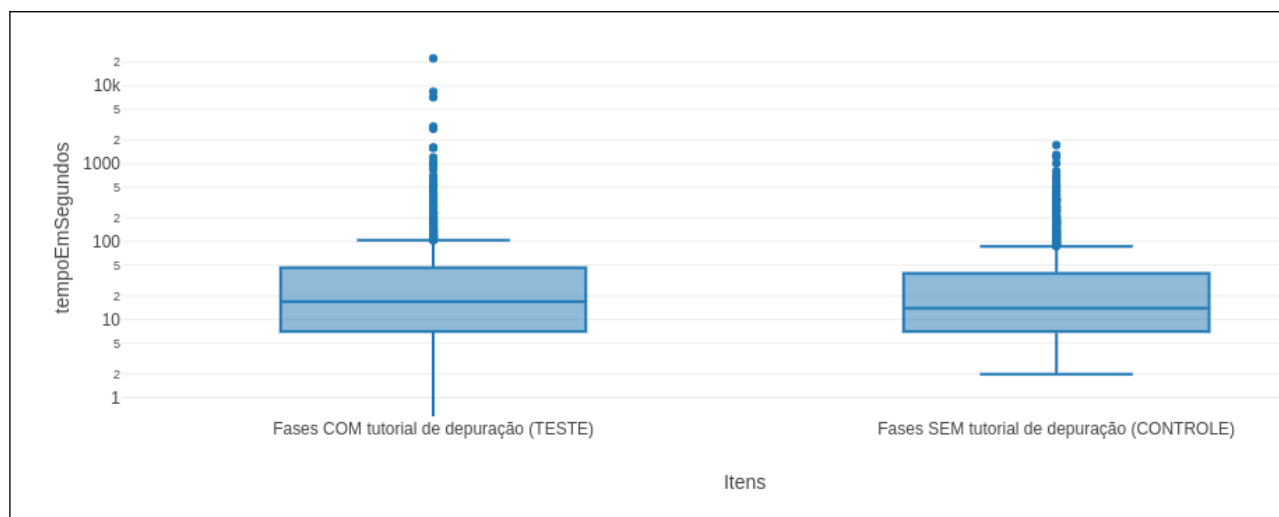


Figura 47: Diagrama de caixa comparando o tempo de resolução dos itens entre aplicação de teste e aplicação de controle

Fonte: Autor

5.1.5 Plataforma como apoio de análise qualitativa

A análise qualitativa das respostas do questionário ocorreu em três etapas: (1) exportação de dados a partir da plataforma, (2) análise de conteúdo através do software *Taguette* e (3) agrupamento dos dados. Esse processo ajudou a avaliar o puzzle de programação (capítulo 4.2) a partir da opinião dos participantes a perguntas abertas. Em geral, apenas os participantes que finalizaram o teste responderam às perguntas do questionário, que foi apresentado no fim do teste.

A exportação de dados foi feita a partir da tela de visualização de respostas da plataforma. Foram selecionadas as respostas das perguntas abertas, presentes no questionário. Essa exportação gerou quatro arquivos para cada grupo (um para cada resposta). Cada arquivo continha identificadores e respostas dos usuários. Os identificadores foram removidos e os arquivos resultantes foram importados no software *Taguette* para análise qualitativa.

O *Taguette* é um software livre e de código aberto para pesquisas qualitativas. Permite importar materiais, marcar, etiquetar o texto e exportar os resultados (RAMPIN; STEEVES; DEMOTT, 2021). Os textos foram importados na ferramenta e o conteúdo de cada resposta foi marcado e adicionado a uma ou mais categorias. Por fim, os dados foram exportados em formato de banco de dados sqlite3. Isso permitiu ordenar as categorias de resposta por frequência e remover as

categorias que apareceram somente uma vez. As perguntas do teste permitiram medir características relacionadas a dificuldade do teste, experiência prévia e a opinião dos participantes sobre o que gostaram ou não no puzzle de programação.

5.1.5.1 Análise qualitativa sobre dificuldade do teste

A análise qualitativa sobre a dificuldade do teste se deu sobre a pergunta: “Teve alguma dificuldade em entender a lógica do jogo? Poderia explicar?” (Tabela 3). A maioria dos participantes (49) respondeu que não. Contudo, 31 participantes afirmaram ter dificuldades no final, principalmente na última fase. Essa dificuldade se deu principalmente durante o uso de funções e pelo fato de ser a fase mais difícil.

Alguns participantes citaram dificuldade para entender as instruções de giro, confundindo entre as setas de direita e esquerda. Um participante afirmou que a visão do jogo (isométrica) contribuiu para a dificuldade em entender a função de giro. Disse “tive um pouco de dificuldade para entender, qual o lado que o robzinho ia virar só depois de algumas fases eu peguei o jeito o que fica confuso é a função de virar para o lado, sugiro uma visão de cima, ou, uma vista mais direta, não diagonal [...]”.

Tabela 3. Respostas: “Teve alguma dificuldade em entender a lógica do jogo? Poderia explicar?”

Categoria de resposta	Descrição (O participante...)	Frequência	Grupo teste	Grupo controle
Não	Não considerou difícil	49	24	25
Sim	Considerou difícil usar ou entender algum elemento	31	18	13
No final	Teve dificuldades nas últimas fases do teste	12	6	6
Dificuldade com funções	Não conseguiu ou achou difícil usar funções F1 e F2	10	4	6
Expressão de contentamento	Expressou palavras como “interessante” ou “divertido”	7	4	3
Dificuldade com giro ou setas	Confundiu as setas de giro para esquerda e direita	7	3	4
No início	Teve dificuldade para entender o início do jogo	7	5	2
Experiência prévia	Citou ter experiência prévia com programação	6	2	4
Faltou explicar	Sentiu falta de explicações sobre algum ponto	5	1	4
Entendimento ao longo do teste	Explicou que entendeu a proposta do jogo ao longo das fases	4	3	1
Sugestão	Deu sugestões de melhorias sobre um ou mais aspectos	4	1	3

Entendimento além do explicado	Demonstrou entender conceitos além dos explicados nos tutoriais	3	3	1
Esquecimento de função de ré	Teve dificuldades porque esqueceu que o personagem anda para trás	2	2	0
Explicação sobre dificuldade	Explicou com exemplos ou analogias o motivo da dificuldade	2	2	0

Fonte: Autor

A análise entre grupos mostra que a frequência de resposta negativa para a pergunta sobre dificuldade foi a mesma: 24 participantes do grupo de teste e 25 participantes do grupo de controle afirmaram não ter tido dificuldade. Já para a resposta positiva (aqueles que afirmaram ter tido dificuldade), a frequência no grupo de teste foi maior: 18 participantes contra 13 do grupo de controle.

5.1.5.2 Análise qualitativa sobre experiência prévia

A análise qualitativa sobre a experiência prévia dos participantes foi norteada pela seguinte pergunta: “Já jogou outros jogos parecidos? Quais?” (Tabela 4). A maioria dos participantes (57) respondeu negativamente. Do outro lado, 44 participantes afirmaram ter jogado jogos semelhantes. Alguns participantes especificaram o nome ou a fonte dos jogos. 8 participantes citaram ter jogado mais de um jogo do tipo.

Tabela 4. Respostas: “Já jogou outros jogos parecidos? Quais?”

Categoria de resposta	Descrição (O participante...)	Frequência	Grupo Teste	Grupo controle
Não	Disse não ter jogado outros jogos parecidos	57	31	26
Sim	Afirmou ter jogado, pelo menos uma vez, algum jogo semelhante	44	23	21
Especificou quais	Especificou o nome ou citou a fonte do jogo com o qual teve contato	22	10	12
Não lembra nome	Disse não recordar o nome do jogo	18	12	6
Mais que um	Afirmou ter tido contato com mais de um jogo do tipo	8	3	5

Fonte: Autor

Esses dados revelaram que boa parte dos participantes (43%) teve contato com jogos do tipo, previamente. Isso revela um interesse pela área, provavelmente apoiado pelo fato de que a maioria dos participantes terem contato com programação no ambiente de ensino.

5.1.5.3 Análise qualitativa sobre pontos positivos do puzzle de programação

A opinião sobre os pontos positivos do puzzle foi pedido por meio da pergunta: “O que mais gostou no jogo?” (Tabela 5). Grande parte dos participantes (29) respondeu que apreciou o fato de o puzzle exigir raciocínio lógico. Outros pontos levantados foram o design, o fato de ser um jogo simples e com boa jogabilidade. Também houve respostas em que os participantes afirmaram ter gostado das funções e das fases finais, que exigiram mais raciocínio lógico.

Tabela 5. Respostas: “O que mais gostou no jogo?”

Categoria de resposta	Descrição (O participante...)	Frequência	Grupo teste	Grupo controle
Exige raciocínio lógico	Apreciou o desafio lógico de resolver o puzzle	29	12	17
Design	Gostou dos desenhos, layout, cores	14	7	7
Simple	Considerou o jogo simples, intuitivo	13	7	6
Jogabilidade	Considerou o jogo dinâmico, a mecânica fluida	11	2	9
Proposta	Gostou da ideia do jogo	9	2	7
Funções	Apreciou o uso de funções e a forma como são apresentadas	9	6	3
Tudo	Afirmou gostar de tudo no puzzle, não foi específico	8	5	3
Prende atenção	Disse que o puzzle é bom para capturar a atenção	5	2	3
Música/Sons	Gostou dos efeitos sonoros	5	2	3
Explicações	Apreciou o formato das explicações e tutoriais	5	2	3
Divertido	Considerou o puzzle divertido	5	5	0
Fases finais	Disse que as últimas fases foram mais instigantes, desafiadoras	5	4	1
Moedas	Gostou dos elementos de moedas	4	2	2
Personagem	Achou o personagem carismático	4	1	3
Ensina programação	Disse que o puzzle ensina programação	4	4	0
Dificuldade gradual	Apreciou o fato de a dificuldade aumentar de acordo com as fases	3	1	2
Passatempo	Afirmou que o puzzle serve para passar tempo	3	2	1
Diferentes públicos	Considerou o puzzle adequado para diferentes idades e níveis	2	1	1
Estimula senso de direção	Apoia o entendimento de direção, orientação espacial	2	2	0

Fonte: Autor

A análise entre os grupos mostra que os participantes do grupo de controle disseram com mais frequência ter gostado da “jogabilidade” do puzzle. Isso pode se porque não precisavam clicar todas as vezes no botão de depuração a cada passo do personagem como acontece com os participantes do grupo de teste. Já no grupo de teste, mais participantes afirmaram ter gostado das funções. A sugestão de resolução passo a passo (clcando todas as vezes no botão de próximo, facilitando o entendimento) favoreceu a resolução para esse tipo de questão para o grupo de controle.

5.1.5.4 Análise qualitativa sobre pontos negativos do puzzle de programação

Os “pontos negativos” do puzzle, na opinião dos participantes, foram coletados por meio da questão: “O que menos gostou no jogo?” (Tabela 6). A maioria dos respondentes (20) afirmou não haver pontos negativos no jogo. 12 respondentes consideraram o jogo muito curto e 6 o julgaram fácil demais. Isso pode ser atribuído ao fato de que grande parte dos participantes tem contato com programação, e que quase metade dos testes (5 questões das 12) são tutoriais.

Tabela 6. Respostas: “O que menos gostou no jogo?”

Categoria de resposta	Descrição (O participante...)	Frequência	Grupo teste	Grupo controle
Nada	Afirmou não haver nada que não gostou	20	10	10
Poucas fases	Considerou que o jogo poderia ter mais fases	12	7	5
Música/Sons	Achou a música muito alta, pouco adequada	10	2	8
Muito fácil	Considerou muito fácil o nível de algumas ou todas as fases	6	3	3
Lentidão do personagem ou elemento	Considerou a velocidade das animações muito lenta	5	4	1
Funções não explicadas	Disse que alguns elementos e funções não foram explicadas	5	2	3
Setas de giro	Afirmou que as setas de esquerda direita eram pouco intuitivas	5	4	1
Pouca performance ou resolução	Reclamou sobre problemas de renderização e performance	3	2	1
Início	Afirmou que o início do teste foi pouco claro	2	1	1
Funções (F1, F2)	Considerou o uso das funções confuso, pouco explicativo	2	1	1
Personagem	Considerou o personagem estranho ou pouco carismático	2	1	1

Fonte: Autor

Os efeitos sonoros também foram tema de reclamação. 10 participantes disseram que a música estava alta demais ou pouco adequada. A lentidão dos elementos (notadamente o movimento do personagem) também foi citado. Por fim, alguns participantes apontaram negativamente o fato de haver elementos não utilizados ou explicados.

5.1.6 Análise por TRI: Curvas características dos itens (CCI)

As curvas de probabilidade de acerto por nível de habilidade foram estimadas de maneira aproximada. Por tanto, os dados dos gráficos aqui apresentados servem para comparação entre os itens, mas não como valores verdadeiros de estimação de parâmetros.

Para o grupo de teste, as curvas foram consistentes com a análise qualitativa dos participantes. As fases finais (10, 11 e 12) situaram-se mais à direita do gráfico (Figura 48). Isso significa que é exigido um nível de habilidade maior para maximizar a probabilidade de acerto total.

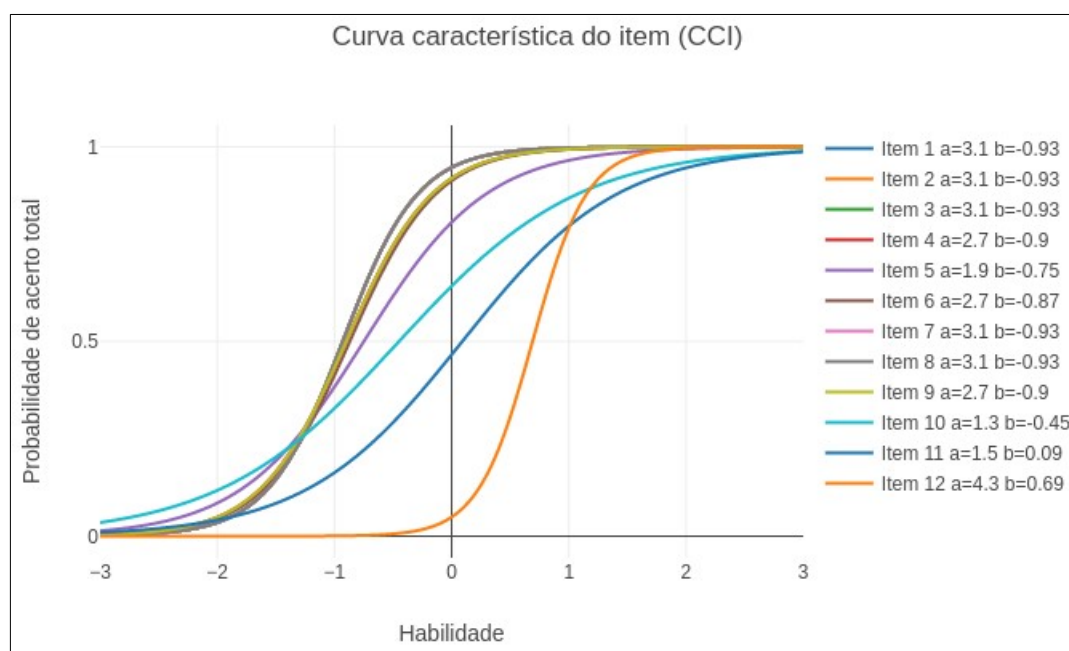


Figura 48: Curvas características dos itens do teste aplicado ao grupo de teste

O mesmo ocorreu para o caso do grupo de controle: os itens finais do teste situaram-se mais à direita do gráfico (Figura 49). O item 10 foi estimado próximo aos itens mais fáceis e tutoriais, o que é inconsistente já que é um item mais difícil. O item 11 e 12 foram estimados como os itens

mais difíceis, sendo que item 12 obteve uma curva em “S” mais abrupta, com maior capacidade de discriminação. Isso significa que pouca variação na habilidade é suficiente para observar uma grande variação na probabilidade de acerto total do item.

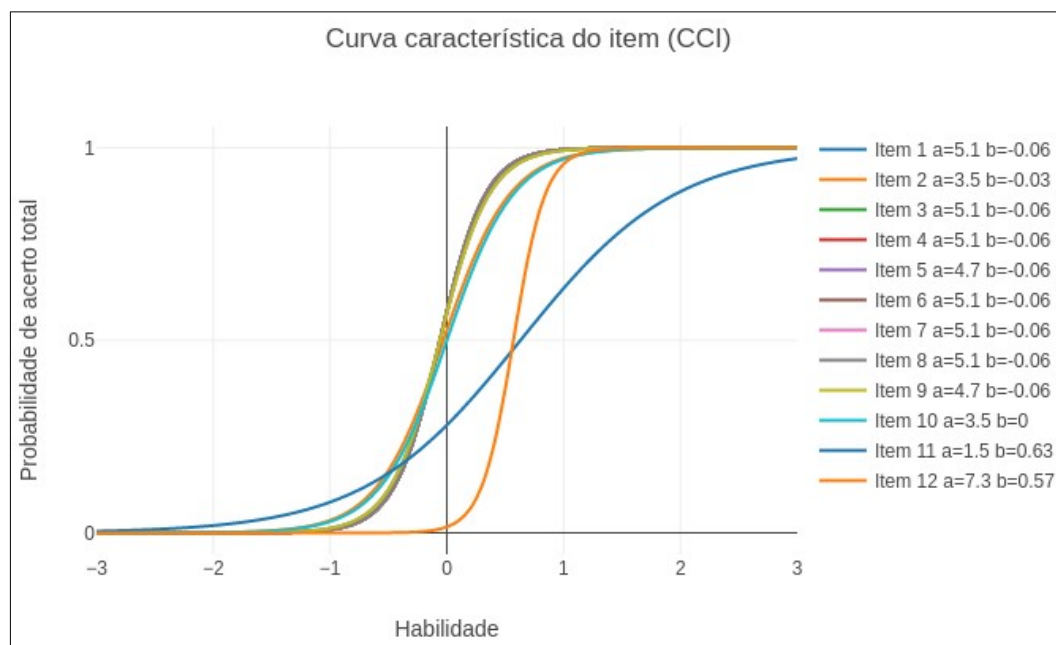


Figura 49: Curvas características dos itens do teste aplicado ao grupo de controle

Em relação aos dois últimos itens (11 e 12) os parâmetros de dificuldade (b) foram respectivamente 0.63 e 0.57 indicando que o item 11 é mais difícil. Isso não é consistente com as respostas do questionário, que citaram o item 12 como sendo mais difícil. Portanto, o método de estimação dos parâmetros deve ser revisado. Unir todas as respostas dos itens independente da aplicação seria uma forma direta de obter mais dados para estimação. Isso não foi feito durante esse trabalho, mas é uma oportunidade explicada no capítulo de trabalhos futuros.

5.2 CONSIDERAÇÕES

Esse capítulo mostrou os resultados do trabalho. Foram avaliados a plataforma e o puzzle de programação. A plataforma foi avaliada de forma prática, através de um teste aplicado aos participantes. O puzzle foi avaliado qualitativamente por meio das respostas dos participantes ao questionário de fim de teste.

O conteúdo dos testes, bem como sua estrutura e divisão foram esclarecidos no início do capítulo. A isso seguiu o processo e os meios de divulgação por redes sociais e em escolas por meio de convite. Algumas características dos participantes foram apresentadas, destacando-se que nenhum pessoal foi solicitado durante os testes para garantir o anonimato dos participantes.

O objetivo foi levantar hipóteses, testar a capacidade da plataforma em apoiar sua investigação e construir uma base de dados para permitir melhorar o puzzle de programação. Essas melhorias, se implementadas, devem beneficiar novos testes. Portanto, foi testado e apresentado de forma prática o uso do sistema construído durante o trabalho.

Os testes não atingiram coeficientes aceitáveis de alpha de Cronbach, como foi o caso do CTt e do CT Puzzle Test aprimorado por Couto (2019). Contudo, sendo esta a primeira versão do teste, deve-se alcançar coeficientes melhores a partir da reformulação de itens e aplicação com novos grupos de participantes.

6 CONCLUSÕES

O CT Puzzle Test é um ambiente de testes que tem um conjunto de puzzles relacionados com PC. Durante a pesquisa de Calbusch (2021) foram feitas as primeiras validações na direção de confirmar sua validade de critério e de construto. Contudo, a lógica de cálculo de escores do CT Puzzle Test é implementada diretamente na ferramenta, dificultando a flexibilização.

De uma perspectiva mais geral, observa-se que diversos aspectos do PC tornam difícil sua avaliação: a variedade de métodos, ambientes de ensino, se é interdisciplinar ou não, etc. Por ser um constructo que carece de uma definição consensual, estratégias de avaliação focam em diferentes dimensões (WEINTROP et al., 2021). Dessa forma, a coleta de dados psicométricos relacionados a habilidades do PC são importantes para o estudo desse tema.

A partir do entendimento sobre a pluralidade de definições e das limitações impostas por uma solução fixa, durante este trabalho foi construído um sistema flexível para criação de testes. Como efeito da linha de pesquisa existente, a plataforma foi aplicada com testes utilizando puzzles, conectando-se com um jogo externo, de programar. Testes foram criados e submetidos a voluntários. Os resultados foram analisados por meio dos recursos da própria ferramenta, apontando a direção para a evolução dos testes.

A ferramenta CT Puzzle Platform oferece flexibilidade para a criação dos testes, segurança para armazenamento de respostas, manutenção do cálculo de escores e auxílio durante a fase de análise. Os testes são disponíveis online, mas também podem ser adaptados para outros ambientes de aplicação. Assim, a plataforma não é o teste, mas sim a interface entre o ambiente de teste e o pesquisador.

Em comparação com testes como CTt de Román-González (2015), onde o conjunto de itens é fixo, a solução desse trabalho oferece mais liberdade ao pesquisador, que pode alterar o conjunto de itens. Foram criados dois testes que foram respondidos permitindo calcular o nível de dificuldade dos itens.

O pilar de algoritmos fundamentou o puzzle parametrizável criado durante o trabalho. O puzzle (um jogo virtual de programar) inspirou-se no projeto RoPE, do Laboratório LITE²⁰, da UNIVALI. A implementação desse jogo, suportando configurações por meio de texto no formato JSON, atendeu ao objetivo específico 1.

Após a criação do puzzle, foi iniciado o desenvolvimento da plataforma. A plataforma permite aos pesquisadores realizarem cadastro, criar mecânicas, testes e aplicá-los. Por fim, o módulo estatístico oferece informações a respeito da dificuldade do teste, com gráficos e informações qualitativas. A implementação da plataforma atendeu ao objetivo específico 2.

Dentro do módulo estatístico, foi implementado um gráfico de teoria da resposta ao item. Esse gráfico informa o grau de probabilidade de acerto dado o nível de habilidade do participante. Assim, embora o método de estimação de parâmetros possa ser aprimorado, a implementação deste recurso atendeu ao objetivo específico 3.

A partir do recurso de divisão de grupos (grupo de teste e grupo de controle) da plataforma foram criados dois testes: um baseado em depuração e outro sem depuração. Os testes foram divulgados via redes sociais e redes de contatos, contando com apoio de professores e alunos voluntários para participação. A aplicação desses testes atendeu ao objetivo específico 4.

6.1 CONTRIBUIÇÕES DA DISSERTAÇÃO

A principal contribuição deste trabalho é a ferramenta que permite criar novos testes a partir de itens existentes. A possibilidade de criar diferentes testes, permite que sejam feitos novos estudos a partir de conjuntos diferentes de itens, o que não era possível antes de forma fácil. Além disso, a estrutura de comunicação entre os instrumentos de coleta e a plataforma é neutro. Por isso, os ambientes de testes podem ser desenvolvidos utilizando qualquer tecnologia capaz de fazer requisições através do protocolo HTTP.

Durante esse trabalho foi desenvolvido um puzzle de programação. O puzzle é configurável, disponível para mobile e navegador. Como foi integrado na plataforma, serve de exemplo para

20 <http://lite.acad.univali.br/pt/>

implementações futuras. Sua aplicação em testes já ocorreu, e novas versões de testes podem ser criadas a partir dos resultados.

A ideia inicial da plataforma foi apresentada em um artigo sobre o CT Puzzle Test (RAABE; VIANA; CALBUSCH, 2020, p. 1689). O artigo, intitulado “CT Puzzle Test: Em direção a uma avaliação interativa do pensamento computacional” foi publicado nos anais do SBIE (Simpósio Brasileiro de Informática na Educação).

A aplicação com voluntários gerou uma base de dados de 256 participações válidas, totalizando 2546 respostas. A análise sobre essas respostas permitiu estudar os itens pela perspectiva de teoria da resposta ao item, gerando dados consistentes com as respostas qualitativas coletadas. Os itens mais difíceis dos testes realmente posicionaram-se mais à direita do gráfico de curva característica.

Por fim, além da ferramenta permitir análises quantitativas, gerou uma base de dados qualitativa, coletando por meio de questionários a opinião dos participantes sobre os testes criados. Essas informações, se filtradas, devem apoiar melhorias no puzzle de programação existente e auxiliar na definição das características de outros jogos que venham a ser integrados na plataforma.

6.2 TRABALHOS FUTUROS

Há diversos trabalhos que podem ser desenvolvidos a partir desta pesquisa. A criação de novos testes, o acoplamento de novos puzzles e melhorias na plataforma são algumas opções. Por tratar-se da primeira versão, a plataforma precisa de validações e correções, o que pode ser alcançado por meio de desenvolvimento iterativo. Esse desenvolvimento deve ser focado nas necessidades dos usuários, seja o pesquisador, os participantes dos testes, o desenvolvedor de puzzles, ou aquele interessado em conectar um ambiente de teste existente.

O trabalho de Fábio Abib Hecktheuer iniciou-se paralelamente ao desenvolvimento da plataforma e propõe-se a avaliar a ferramenta, a partir da criação de testes e posterior aplicação com alunos de 11 a 14 anos. Esse processo será importante para identificar oportunidades de melhoria, desenvolver um manual e testar a ferramenta em um novo contexto. O testes farão uso da mecânica

de programação desenvolvida neste trabalho, servindo para testá-la novamente e propor modificações.

Nos trabalhos futuros, pode-se decidir o que fazer com os dados dinâmicos gerados pela plataforma. Novas implementações de funções de cálculo de escore podem tratar esses dados de forma diferente, considerando aspectos como o tempo de resolução, número de tentativas e desistências. O mesmo pode ser feito para outras mecânicas que venham a ser integradas com a plataforma, propondo diferentes abordagens de pontuação.

A ligação das mecânicas do CT Puzzle Test na plataforma possibilitaria a visualização de dados, acompanhamento de aplicações dos testes. Assim, a configuração dos itens do CT Puzzle Test poderia ser feita externamente, eliminando a necessidade de alterar o código fonte em caso de alteração de função de escore. O ajuste para comunicar o CT Puzzle Test com plataforma demandaria esforço de codificação. A adição de suporte para ordenação dos itens, envio de respostas e uso de função de escore, exigiria implementar uma lógica de ordenação dos itens no CT Puzzle Test adequando-a à configuração externa. Como análise teórica, poderiam ser comparados os resultados de testes anteriores do CT Puzzle Test a resultados computados a partir da plataforma.

Especificamente na plataforma, há melhorias que podem ser realizadas para dar mais visibilidade sobre a qualidade dos testes. Na versão desenvolvida durante este trabalho, o módulo estatístico apresenta dados orientados a sessões de aplicação de teste. Isso permite comparar dados entre grupos de diferentes aplicações, mas não oferece dados gerais. Agrupar os dados dos testes ajudaria a analisar populações maiores. Uma forma de fazer isso é adicionar um componente seletor de testes ao invés de um seletor de aplicações.

O método de estimação de parâmetros de TRI pode ser aprimorado. O método atual, embora tenha gerado resultados consistentes com a análise qualitativa, carece de um estudo mais aprofundado e uso de técnicas de estimação conhecidas. O método utilizado foi iterativo, com estimação de parâmetros por meio da aproximação da curva característica dos itens em relação a frequência de acerto observada em cada nível de habilidade. No entanto, há diversos métodos de estimação que podem ser empregados: máxima verossimilhança, máximo a posteriori (MAP), de a

posteriori esperado (PASQUALI, 2020). Aplicar esses métodos, ou mesmo testar outros modelos de TRI, é um trabalho futuro pertinente.

Do ponto de vista de qualidade de software, estudos podem ser realizados para verificar performance, segurança e a clareza das telas pelos usuários. Pode-se produzir um conjunto de atividades para participantes realizarem funções como pesquisadores ou participantes de testes. A aplicação de um questionário posterior levantaria necessidades de melhoria e dificuldades encontradas durante o uso do sistema.

As dúvidas dos usuários a respeito do sistema poderiam ser reunidas em formato de documentação, explicando sobre o uso de telas e o processo de integração com outros ambientes de teste. Além disso, o preenchimento automático de dados, como uma mecânica, itens, teste e uma aplicação de exemplo facilitariam ao usuário iniciante entender o fluxo. Essa funcionalidade poderia ser acompanhada de um tutorial interativo, que guiasse o usuário através das telas.

Alterações na forma de aplicação também são oportunidades de pesquisa. Nesse sentido, uma possibilidade é a inclusão de testes adaptativos. Na versão desenvolvida durante este trabalho, o pesquisador pode alterar a ordem dos itens da lista de teste. Mas no momento da aplicação, a ordem é fixa. Com testes adaptativos, os itens seriam definidos sob demanda, de acordo com o desempenho do participante no item anterior.

Como perspectiva mais direta, seria interessante integrar outros puzzles à plataforma. O uso de mecânicas diferentes no mesmo teste não foi considerado neste trabalho, que utilizou somente um puzzle de programação. O uso de ambientes de testes diferentes auxiliaria a avaliar diferentes aspectos do PC, já que cada ambiente pode requerer mais ou menos habilidade sobre cada pilar.

Técnicas como as implementadas no trabalho de Calbusch (2021), poderiam ser automatizadas na plataforma. O pesquisador criou um questionário sobre o teste para avaliação por especialistas. O questionário contém, entre outras questões, perguntas sobre a adequação de cada item de testes aos pilares do PC. Dado que a ferramenta desenvolvida durante este trabalho

possibilita principalmente análises quantitativas, seria útil adicionar opções de análise qualitativa por especialistas diretamente na plataforma.

Por fim, este trabalho é a sequência de pesquisas anteriores. E da mesma forma, pode contribuir para pesquisas futuras, servindo como uma ferramenta de apoio à criação de testes. Por isso, espera-se acompanhar sua evolução técnica, teórica e enquanto produto de software. Técnica, pautando-se em boas práticas de desenvolvimento e segurança da informação, Teórica, utilizando referências da psicometria e visando gerar resultados válidos e confiáveis. E enquanto produto de software, visando qualidade para apoio de pesquisas sobre a habilidade de Pensamento Computacional.

REFERÊNCIAS

- ARAUJO, A. L. S. O. et al. **Explorando Teoria de Resposta ao Item na Avaliação de Pensamento Computacional: um Estudo em Questões da Competição Bebras.** . In: XXIX SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO (BRAZILIAN SYMPOSIUM ON COMPUTERS IN EDUCATION). Fortaleza, Ceará, Brasil: 28 out. 2018. Disponível em: <<http://br-ie.org/pub/index.php/sbie/article/view/8025>>. Acesso em: 11 out. 2020
- BBC. **Computational thinking - KS3 Computing.** Disponível em: <<https://www.bbc.co.uk/bitesize/topics/z7tp34j>>. Acesso em: 5 jul. 2020.
- BEBRAS.ORG. **What is Bebras | www.bebbras.org.** Disponível em: <<https://www.bebbras.org/>>. Acesso em: 18 jul. 2020a.
- BEBRAS.ORG. **Bebras Lodge example 3 | www.bebbras.org.** Disponível em: <<https://www.bebbras.org/?q=lodge3>>. Acesso em: 21 jul. 2020b.
- BLIKSTEIN, P. **O Pensamento Computacional.** Disponível em: <http://www.blikstein.com/paulo/documents/online/ol_pensamento_computacional.html>. Acesso em: 11 jul. 2020.
- BRACKMANN, C. et al. **Pensamento Computacional Desplugado: Ensino e Avaliação na Educação Primária Espanhola.** . In: VI CONGRESSO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO. Recife, Pernambuco, Brasil: 27 out. 2017. Disponível em: <<http://www.br-ie.org/pub/index.php/wcbie/article/view/7487>>. Acesso em: 31 maio. 2020
- BRACKMANN, C. P. **DESENVOLVIMENTO DO PENSAMENTO COMPUTACIONAL ATRAVÉS DE ATIVIDADES DESPLUGADAS NA EDUCAÇÃO BÁSICA,** 2017.
- BRENNAN, K.; RESNICK, M. New frameworks for studying and assessing the development of computational thinking. v. 1, p. 25, 2012.
- CALBUSCH, L. **CTPUZZLE TEST: INSTRUMENTO PARA AVALIAÇÃO DO PENSAMENTO COMPUTACIONAL DE JOVENS DE 14 A 17 ANOS,** 2020.
- CAMBRIDGE, D. **Cambridge Dictionary | Puzzle.** Disponível em: <<https://dictionary.cambridge.org/dictionary/english-portuguese/puzzle>>. Acesso em: 9 dez. 2020.
- COUTO, N. E. R. **IMPACTO DA IMPLANTAÇÃO DE UMA DISCIPLINA DE PENSAMENTO COMPUTACIONAL EM ESTUDANTES.** p. 140, 2019.
- CSTA; ISTE. **Computational Thinking in K–12 Education teacher resources.** 2. ed. [s.l: s.n.].
- DANTAS, T. **Tangram.** Disponível em: <<https://mundoeducacao.uol.com.br/curiosidades/tangram.htm>>. Acesso em: 5 dez. 2020.

DE ANDRADE, D. F.; TAVARES, H. R. Teoria da Resposta ao Item: Conceitos e Aplicações. p. 164, 2000.

FEUERSTEIN, R. Feuerstein's Instrumental Enrichment. p. 17, 2000.

GONÇALVES, F. A. UM INSTRUMENTO PARA O DIAGNÓSTICO DO PENSAMENTO COMPUTACIONAL. p. 128, 2015.

GONZÁLEZ, M. R. COMPUTATIONAL THINKING TEST: DESIGN GUIDELINES AND CONTENT VALIDATION. 2015a.

GONZÁLEZ, M. R. **COMPUTATIONAL THINKING TEST: DESIGN GUIDELINES AND CONTENT VALIDATION**. Proceedings of EDULEARN15 Conference. **Anais...Barcelona - ES**: Unpublished, 2015b. Disponível em: <<http://rgdoi.net/10.13140/RG.2.1.4203.4329>>. Acesso em: 23 set. 2020

GOOGLE, T. **Google Trends**. Disponível em: <<https://trends.google.com.br/trends/explore?date=all&q=Computational%20thinking>>. Acesso em: 24 maio. 2020.

GROVER, S.; PEA, R. Computational Thinking in K–12: A Review of the State of the Field. **Educational Researcher**, v. 42, n. 1, p. 38–43, jan. 2013.

GUEDES, G. T. A. **UML 2: uma abordagem prática**. [s.l.] Novatec Editora, 2011.

HONG, J.-C. et al. Effects of cognitive style on digital jigsaw puzzle performance: A GridWare analysis. **Computers in Human Behavior**, v. 28, n. 3, p. 920–928, maio 2012.

ISTE; CSTA. **Operational Definition of Computational Thinking for K–12 Education**, 2011.

KONG, S.-C.; ABELSON, H. (EDS.). **Computational Thinking Education**. Singapore: Springer Singapore, 2019a.

KONG, S.-C.; ABELSON, H. **Computational Thinking Education**. [s.l.] Springer, 2019b.

KORKMAZ, Ö.; ÇAKIR, R.; ÖZDEN, M. Y. A validity and reliability study of the computational thinking scales (CTS). **Computers in Human Behavior**, v. 72, p. 558–569, jul. 2017.

KURSHAN, B. **Thawing from a Long Winter in Computer Science Education**. Disponível em: <<https://www.forbes.com/sites/barbarakurshan/2016/02/25/thawing-from-a-long-winter-in-computer-science-education/>>. Acesso em: 19 maio. 2020.

LAW, B. Puzzle Games: A Metaphor for Computational Thinking. In: **10th European Conference on Games Based Learning: ECGBL 2016**. [s.l.: s.n.]. p. 344.

MCCAULEY, R. et al. Debugging: a review of the literature from an educational perspective. **2008**, v. 18, n. 2, p. 28, 2008.

MORENO-LEÓN, J.; ROBLES, G.; ROMÁN-GONZÁLEZ, M. Dr. Scratch: Automatic Analysis of Scratch Projects to Assess and Foster Computational Thinking. p. 24, 2015a.

MORENO-LEÓN, J.; ROBLES, G.; ROMÁN-GONZÁLEZ, M. Dr. Scratch: Automatic Analysis of Scratch Projects to Assess and Foster Computational Thinking. p. 23, 2015b.

MORENO-LEÓN, J.; ROBLES, G.; ROMÁN-GONZÁLEZ, M. Dr. Scratch: Automatic Analysis of Scratch Projects to Assess and Foster Computational Thinking. **RED - Revista de Educación a Distancia**, v. 46, p. 24, 2015c.

NERING, M. L.; OSTINI, R. (EDS.). **Handbook of polytomous item response theory models**. New York, NY: Routledge, 2010.

OXFORD, L. D. **Oxford Advanced Learner's Dictionary | Puzzle**. Disponível em: <https://www.oxfordlearnersdictionaries.com/us/definition/english/puzzle_1?q=puzzle>. Acesso em: 9 dez. 2020.

PAPERT, S. **Mindstorms, Children, Computers and Powerful Ideas**, 1980.

PASQUALI, L. **Teoria dos Testes na Psicologia e na Educação**. 4. ed. [s.l.] Vozes, 2009a.

PASQUALI, L. PSICOMETRÍA. p. 8, 2009b.

PASQUALI, L. **TRI - Teoria de Resposta ao Item: Teoria, Procedimentos e Aplicações**. [s.l.] Editora Appris, 2020.

PEREZ, A. D. F.; VALLADARES, G. M. **Development and assessment of computational thinking: A methodological proposal and a support tool**. 2018 IEEE Global Engineering Education Conference (EDUCON). **Anais...** In: 2018 IEEE GLOBAL ENGINEERING EDUCATION CONFERENCE (EDUCON). Tenerife: IEEE, abr. 2018. Disponível em: <<https://ieeexplore.ieee.org/document/8363311/>>. Acesso em: 18 jul. 2020

RAABE, A.; VIANA, C.; CALBUSCH, L. **CT Puzzle Test: Em direção a uma avaliação interativa do pensamento computacional**. Anais do XXXI Simpósio Brasileiro de Informática na Educação (SBIE 2020). **Anais...** In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO. Brasil: Sociedade Brasileira de Computação, 24 nov. 2020. Disponível em: <<https://sol.sbc.org.br/index.php/sbie/article/view/12924>>. Acesso em: 11 set. 2021

RAMPIN, R.; STEEVES, V.; DEMOTT, S. **Taguette**. [s.l.] Zenodo, 2021.

ROMÁN-GONZÁLEZ, M.; MORENO-LEÓN, J.; ROBLES, G. Combining Assessment Tools for a Comprehensive Evaluation of Computational Thinking Interventions. In: KONG, S.-C.; ABELSON, H. (Eds.). **Computational Thinking Education**. Singapore: Springer Singapore, 2019. p. 79–98.

ROMÁN-GONZÁLEZ, M.; PÉREZ-GONZÁLEZ, J.-C.; JIMÉNEZ-FERNÁNDEZ, C. Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. **Computers in Human Behavior**, v. 72, p. 678–691, jul. 2017.

ROWE, E. et al. **Assessing implicit computational thinking in zoombinis gameplay**. Proceedings of the International Conference on the Foundations of Digital Games - FDG '17. **Anais...** In: THE INTERNATIONAL CONFERENCE. Hyannis, Massachusetts: ACM Press, 2017. Disponível em: <<http://dl.acm.org/citation.cfm?doid=3102071.3106352>>. Acesso em: 8 dez. 2020

SELBY, C. C. et al. Computational Thinking: The Developing Definition. p. 6, 2010.

SHUTE, V. J.; SUN, C.; ASBELL-CLARKE, J. Demystifying computational thinking. **Educational Research Review**, v. 22, p. 142–158, nov. 2017.

SOUZA, A. C. DE et al. Propriedades psicométricas na avaliação de instrumentos: avaliação da confiabilidade e da validade. **Epidemiologia e Serviços de Saúde**, v. 26, n. 3, p. 649–659, jul. 2017.

TANG, X. et al. Assessing computational thinking: A systematic review of empirical studies. **Computers & Education**, v. 148, p. 103798, abr. 2020.

TROCHIM, W.; DONNELLY, J. P. **The Research Methods Knowledge Base**. [s.l.: s.n.].

VAN DER LINDEN, W. J.; HAMBLETON, R. K. (EDS.). **Handbook of Modern Item Response Theory**. New York, NY: Springer New York, 1997.

WEINTROP, D. et al. Assessing computational thinking: an overview of the field. **Computer Science Education**, v. 31, n. 2, p. 113–116, 3 abr. 2021.

WERNER, L.; DENNER, J.; CAMPE, S. **The Fairy Performance Assessment: Measuring Computational Thinking in Middle School**, 2012.

WIEBE, E. et al. **Development of a Lean Computational Thinking Abilities Assessment for Middle Grades Students**. Proceedings of the 50th ACM Technical Symposium on Computer Science Education. **Anais...** In: SIGCSE '19: THE 50TH ACM TECHNICAL SYMPOSIUM ON COMPUTER SCIENCE EDUCATION. Minneapolis MN USA: ACM, 22 fev. 2019. Disponível em: <<https://dl.acm.org/doi/10.1145/3287324.3287390>>. Acesso em: 21 set. 2020

WING, J. **Computational Thinking** COMMUNICATIONS OF THE ACM, , mar. 2006.

YENI, S.; HERMANS, F. Design of CoTAS: Automated Computational Thinking Assessment System. p. 6, 2019.

APÊNDICE A – TERMO DE CONSENTIMENTO

Olá, tudo bem? Espero que sim.

Este formulário é um termo de consentimento de participação na minha pesquisa de mestrado em Computação Aplicada, pela UNIVERSIDADE DO VALE DO ITAJAÍ (UNIVALI).

O QUE É A PESQUISA?

A pesquisa envolve a participação em um teste.

O QUE É O TESTE?

O teste é formado por um conjunto de fases de um jogo. No jogo, você será desafiado a programar o robzinho RoPE guiando-o por caminhos de mapas. Cada fase tem um mapa diferente. Ao percorrer os mapas, o robzinho terá alguns objetivos e desafios:

- Coletar moedas
- Desviar e quebrar obstáculos
- Não deixar a bateria do robzinho acabar.

O QUE PRECISO FAZER?

Duas coisas:

- 1 - Tentar resolver o teste, com atenção.
- 2 - Responder as perguntas do formulário de fim de teste.

POSSO DESISTIR A QUALQUER MOMENTO?

Sim. A sua participação é totalmente voluntária. Caso queira desistir durante do teste, esse direito lhe é amplamente garantido.

BÔNUS E REMUNERAÇÕES

Não é oferecido qualquer tipo de ganho, recompensa de qualquer natureza para participação no teste. Aceitando esse termo, você concorda em participar voluntariamente.

DANOS

Caso se sinta prejudicado, sob quaisquer motivo ou circunstância ao realizar o teste, recomenda-se que pare imediatamente a participação no teste. O jogo contém elementos sonoros, gráficos, e desafios de lógica que podem ser desconfortáveis para algumas pessoas. Estaremos disponíveis para tentar reparar qualquer tipo de problema que possa a vir a surgir.

DADOS

Durante o teste, são coletados dados das seguinte natureza:

- Número de cliques
- Instruções informadas para o personagem
- Tempo das respostas
- Faixa etária
- Opiniões sobre o jogo

Não são coletados dados pessoais de qualquer natureza (e-mail, nome, localização, endereço, etc).

CONTATO

Para entrar em contato com o pesquisador, utilize os seguintes meios:

E-mails:

- cassiano.viana@univali.edu.br
- cassiano.pvianna@gmail.com

Telefone:

(47) 991341564

FINALIZAÇÃO

Caso concorde com os termos acima, clique no botão de submissão do formulário.

Se não concordar com algum ponto e prefira não participar da pesquisa, apenas feche essa página.

Grato, desde já:

Cassiano Pereira Viana

PARA ACESSAR O TESTE, VOCÊ DEVE CLICAR NO LINK PARA CELULAR OU PARA COMPUTADOR, QUE ESTARÁ DISPONÍVEL NA PRÓXIMA PÁGINA.

AO CLICAR PARA ENVIAR O FORMULÁRIO, VOCÊ AFIRMA TER LIDO E CONCORDADO COM TODOS OS TERMOS DE CONSENTIMENTO AQUI DESCRITOS.

APÊNDICE B – FERRAMENTAS ANALISADAS PARA DESENVOLVIMENTO DO PUZZLE

O Quadro 9 mostra as ferramentas analisadas para o desenvolvimento do puzzle de programação. Foram descartadas ferramentas com baixa popularidade, pagas ou dependentes de editor: melonJS, Ct.js, PlayCanvas e GameMaker Studio 2. Restaram então PixiJS e Phaser. Enquanto PixiJS aborda criação de conteúdo para web no geral (conteúdos ricos para site, marketing) o framework Phaser foca no desenvolvimento de games 2D, o que atende as necessidades do projeto. Por isso, optou-se por utilizar essa ferramenta.

Quadro 9. Ferramentas de desenvolvimento de jogos para web

Ferramenta	Uso	Dependente de editor	Popularidade	Gratuita	Repositório/site
PixiJS	Biblioteca de renderização HTML 5 e WebGL para criação de conteúdos ricos e games	Não	31.3k	Sim	https://github.com/pixijs/pixijs
Phaser	Framework HTML5 de jogos	Não	28.6k	Sim	https://github.com/photonstorm/phaser
melonJS	Motor de jogos HTML5 leve	Não	3.2k	Sim	https://github.com/melonjs/melonJS
Ct.js	Editor de games 2D	Sim	756	Sim	https://github.com/ct-js/ct-js
PlayCanvas	Motor e editor de games online	Sim	-	Não	https://playcanvas.com/
GameMaker Studio 2	Construtor de games no estilo arrasta e solta	Sim	-	Não	https://www.yoyogames.com/gamemaker

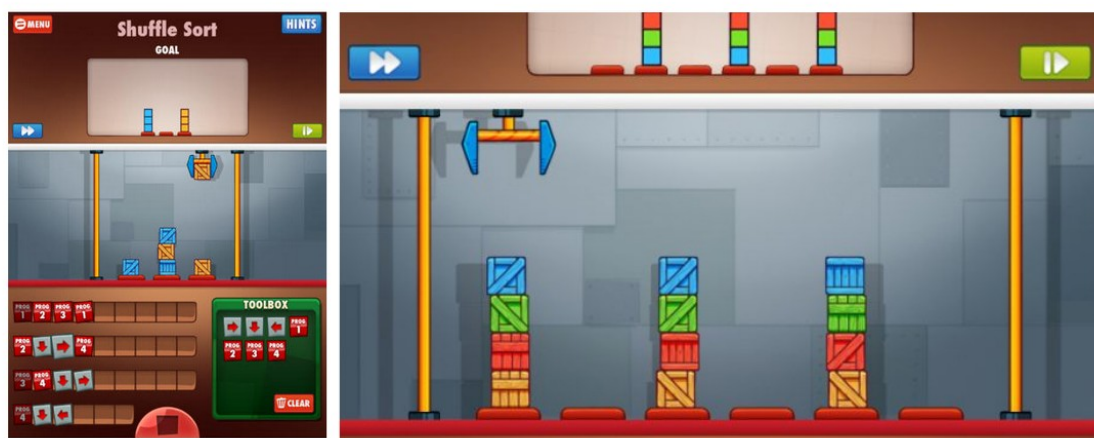
Fonte: Autor

APÊNDICE C – CARGO BOT

O Cargo Bot serviu de inspiração para parte do design do puzzle de programação.

<https://altermanchess.wixsite.com/cargobot>

Cargo Bot



A idéia do jogo é ordenar as caixas de acordo com o objetivo com a garra usando lógica de programação, o jogo permite você movimentar, agarrar e soltar objetos com o auxílio de comandos e funções.

Paulo Eduardo Martins

APÊNDICE D – DADOS

- Respostas obtidas durante as aplicações com grupo de teste e grupo de controle

Grupo teste	https://hd1.nyc3.cdn.digitaloceanspaces.com/mestrado/respostas_aplicacao_teste_2021_7_8_0_46.csv
Grupo controle	https://hd1.nyc3.cdn.digitaloceanspaces.com/mestrado/respostas_aplicacao_controle_2021_7_8_0_45.csv

- Base de dados PostgreSQL da plataforma

<https://hd1.nyc3.cdn.digitaloceanspaces.com/mestrado/dump2021-08-07-03-00.sql.gz>

- Bases de dados de análise qualitativa sobre as respostas do questionário

Teve alguma dificuldade em entender a lógica do jogo? Poderia explicar?	https://hd1.nyc3.cdn.digitaloceanspaces.com/mestrado/2021-08-07_p1_dificuldade.sqlite3
Já jogou outros jogos parecidos? Quais?	https://hd1.nyc3.cdn.digitaloceanspaces.com/mestrado/2021-09-18_p2_jogos_parecidos.sqlite3
O que mais gostou no teste?	https://hd1.nyc3.cdn.digitaloceanspaces.com/mestrado/2021-09-18_p3_o_q_gostou.sqlite3
O que não gostou no teste?	https://hd1.nyc3.cdn.digitaloceanspaces.com/mestrado/2021-09-18_p4_o_q_no_gostou_teste.sqlite3

Para separar os dados em grupos (análise qualitativa), pode-se utilizar o seguinte trecho de SQL:

```
select documents.name,
       tags.path,
       sum(1) total
from   tags
       join highlight_tags on highlight_tags.tag_id = tags.id
       join highlights on highlights.id = highlight_tags.highlight_id
       join documents on highlights.document_id = documents.id

group by
  documents.id,
  tags.id

-- Filtra resultados com mais de uma resposta no grupo
having sum(1) > 1
order by
  documents.name asc,
  total desc;
```

APÊNDICE E – CÓDIGO

Seguem os dados e os recursos utilizados para desenvolvimento. E-mail do desenvolvedor original do puzzle de programação e da plataforma de testes: cassiano.pvianna@gmail.com.

- Código fonte do puzzle de programação: <https://github.com/CassianoViana/CTPuzzleGame>
- Código fonte da plataforma: <https://github.com/CassianoViana/CTPuzzlePlatform>

APÊNDICE F – FERRAMENTAS

- Para edição de imagens do jogo de programação: <https://inkscape.org/pt-br/>
- IDE utilizada para desenvolvimento: <https://code.visualstudio.com/>
- CI/CD: GitLab (<https://gitlab.com/>)

APÊNDICE G – SUGESTÃO PARA DESENVOLVEDORES

Para aqueles que desejarem alterar o jogo de programação, recomenda-se:

- Realizar tutoriais disponíveis na página do projeto: <http://phaser.io/learn>.

Porém, é importante salientar que o jogos a ser integrados com a plataforma podem ser desenvolvido em qualquer tecnologia que o desenvolvedor prefira utilizar. Basta respeitar o protocolo de comunicação descrito no trabalho.

Para desenvolvedores da plataforma, recomenda-se:

- Estudar sobre TRI (Teoria da Resposta ao Item). Pesquisar principalmente por, Baker, Pasquali e os listados nas referências.

APÊNDICE H – FUNÇÃO DE CÁLCULO DE ESCORE

```

function calculaScore(item: ItemProgramacao, resposta: RespostaItem) {

  let fatorNotaComandos = 0// 0 a 1
  let fatorNotaCaminho = 0// 0 a 1
  let nrFatoresValidos = 0

  if (resposta.comandosUtilizados) {

    // calcula número de passos
    if (item.numeroMinimoPassos) {
      nrFatoresValidos++

      let nrPassosMinimo = item.numeroMinimoPassos

      let fns: Map<Comando, Comando[]> = new Map<Comando, Comando[]>()
      fns.set('PROG_0', [])
      fns.set('PROG_1', [])
      fns.set('PROG_2', [])

      resposta.comandosUtilizados.map(cmd => cmd.toString()).forEach(cmdName => {
        let alocou = false
        fns.forEach((comandos: Comando[], key: Comando) => {
          if (cmdName.indexOf(`[${key}]`) > -1) {
            comandos.push(cmdName.replace(`[${key}]`, '')) as Comando
            alocou = true
          }
        })
        if (!alocou) {
          fns.get('PROG_0')?.push(cmdName as Comando)
        }
      })

      console.log(fns)

      let contadorPassos = 0

      let contaPassos = function (comandos: Comando[]) {
        if (contadorPassos > 1000) {
          nrFatoresValidos = 0
          fatorNotaCaminho = 0
          // provável recursividade
          return
        }
        comandos?.forEach((comando: Comando) => {
          if (comando !== 'PROG_1' && comando !== 'PROG_2') {
            contadorPassos++
          }
          if (comando === 'PROG_1') {
            let f1 = fns.get('PROG_1')
            if (f1) {
              contaPassos(f1)
            }
          }
          if (comando === 'PROG_2') {
            let f2 = fns.get('PROG_2')
            if (f2) {
              contaPassos(f2)
            }
          }
        })
      })

      let funcaoPrincipal = fns.get('PROG_0')
      if (funcaoPrincipal) {

```

```

        contaPassos(funcaoPrincipal)
    }

    console.log('Total passos dados: ', contadorPassos)
    console.log('Passos mínimo: ', nrPassosMinimo)

    if (contadorPassos > 0) {
        fatorNotaCaminho = nrPassosMinimo / contadorPassos
        if (contadorPassos < nrPassosMinimo) {
            fatorNotaCaminho = 1
        }
    }
}

// calcula nota por nr comandos
if (item.numeroMinimoComandos) {
    nrFatoresValidos++

    let nrCmdsMinimo = item.numeroMinimoComandos
    let nrCmdsUsados = resposta.comandosUtilizados.length;

    console.log('Total comandos usados: ', nrCmdsUsados)
    console.log('Número mínimo: ', nrCmdsMinimo)

    if (nrCmdsUsados > 0) {
        fatorNotaComandos = nrCmdsMinimo / nrCmdsUsados
        if (nrCmdsUsados < nrCmdsMinimo) {
            fatorNotaComandos = 1
        }
    }
}

let nota = 0.0;

let somaFatores = 0.0
somaFatores += fatorNotaComandos
somaFatores += fatorNotaCaminho

console.log("Nota por uso de menos passos: ", fatorNotaCaminho)
console.log("Nota por uso de menos comandos: ", fatorNotaComandos)
console.log("Total = ", somaFatores)

nota += somaFatores * 10 / nrFatoresValidos

console.log("Escore (total/nr fatores válidos) = ", nota)

if (resposta.finalizou != undefined && resposta.finalizou == false) {
    console.log("Não finalizou. Desconta 25% da nota")
    nota = nota * 0.75
}

return { score: nota, max: 10 }
}

```

APÊNDICE I – RELATO QUALITATIVO SOBRE O USO DA PLATAFORMA NA VISÃO DE UM DESENVOLVEDOR

Esse é o relato de Cesar Viana, desenvolvedor do RoPE AR: ambiente de realidade aumentada para interação com o brinquedo RoPE:

“Já faz um tempo que eu configurei o aplicativo que utiliza a plataforma. Mas se eu lembro de algumas coisas então vou falar. Eu tinha a necessidade de registrar as interações de crianças enquanto elas interagiam com brinquedo RoPE. Então eu precisava guardar isso de alguma forma e poder analisar. Outra necessidade que eu tinha era configurar fases de um jogo onde o RoPE estaria inserido. Fica meio abstrato falando assim, mas o jogo é projetado no chão e o RoPE está sobre esse jogo. Então existe a necessidade de ter várias fases e também guardar as interações: clique nos botões e os programas que são criados.

Eu tinha criado uma API em node. Só que gastei pelo menos umas duas semanas criando ela. E estava tudo fixo. Por exemplo, eu configurei as fases especificamente pra esse jogo. E no caso, se eu quisesse editar alguma coisa eu teria que mexer no código fonte e publicar. Usando a plataforma, ela resolveu o meu problema. No caso, a mesma lógica que eu tinha configurado na API eu consegui reproduzir na plataforma com as diferentes fases. Foi fácil de entrar e fazer login, o login foi enviado um e-mail para mim, e eu confirmei. E daí já fui aí na parte de mecânica, configurei a mecânica do jogo, usando uma matriz.

Também existe a possibilidade de configurar como vai ser a resposta. Isso que eu achei bem interessante. Além da mecânica do jogo, dizer como vai ser a resposta e o cálculo do score, que diz se a resposta foi boa ou não. Outra vantagem que também que achei legal, é poder testar as respostas, simular as respostas estão vindo. Eu não tinha essa possibilidade no ambiente que eu tinha programado antes. Eu poderia talvez receber algumas respostas que não conseguiria utilizar depois. A plataforma me ajudou nisso.

Em resumo, a lógica de como usar ficou bem claro: tem as mecânicas e itens que correspondem a essas mecânicas; a resposta, como ela vai ser recebida; o cálculo do score. Isso tudo foi bem fácil de entender e aplicar.

Sobre a parte de URL, a plataforma permite configurar uma URL onde a pessoa vai clicar e vai abrir uma página WEB. No meu caso, consegui abrir o aplicativo do celular quando a pessoa clicava nessa URL. Foi bem simples a parte de configuração.

O que eu não testei muito foi a parte de análise de dados. Mas pelo que eu olhei tem os gráficos que são bem fáceis. Em termos de tempo de resposta, está bem rápido. A usabilidade está fácil de entender, os filtros também. A única coisa que ainda vejo, são dados de outras mecânicas, que outras pessoas criaram. Poderia mostrar só a minha.”