Fecha de recepción: marzo 2023
Fecha de aceptación: abril 2023
Versión final: mayo 2023

Una mirada a la enseñanza del Diseño Computacional con el lenguaje RocketSocket

Hugo Cristo Sant'Anna(*)

Resumen: El Diseño Computacional (DC) es un abordaje de proyecto cuyos productos son algoritmos que computan soluciones para problemas de interés del diseñador. Desde 2021, la enseñanza del DC en la Universidade Federal do Espírito Santo (Ufes) es basada en el RocketSocket, un lenguaje creado para presentar principios, conceptos, prácticas y perspectivas de la Computación a estudiantes de la carrera de pregrado en Diseño. Este ensayo expone resultados preliminares de la experiencia didáctica en los tres primeros cuatrimestres del curso de DC en la Ufes (2021-2022), y presenta hipótesis sobre la investigación en curso.

Palabras clave: Diseño Computacional - Pensamiento Computacional - Principios de la Computación - Teoría del Control Perceptual - Método de Niveles

[Resúmenes en inglés y portugués en la página 262]

(*) Designer, Mestre e Doutor em Psicologia. Professor adjunto do Departamento de Desenho Industrial da Universidade Federal do Espírito Santo – Ufes (Vitória, ES – Brasil). Professor colaborador e orientador de mestrado do Programa de Pós-Graduação em Psicologia (PPGP). Coordenador do Grupo de Formalizações Matemáticas da Cognição e Design (Forma/Ufes) e membro do Laboratório de Estudos do Desenvolvimento Humano (LEDHUM/Ufes).

Introdução

Na formação de designers, os conhecimentos da área da Computação são aplicados às práticas e tomadas de perspectivas de projeto, geralmente denominadas como "design computacional", "design generativo" e "design algorítmico". Embora haja diferenças importantes entre essas vertentes, a adjetivação "computacional" contempla propriedades comuns a todas elas: Design Computacional (DC) corresponde a práticas cujos resultados

são algoritmos que computam soluções de projeto de interesse de seus autores (Sant'Anna, 2022). Frequentemente, designers computacionais são criadores de suas próprias ferramentas (Maeda, 2004), em vez de restringirem suas opções de projeto ao que é oferecido por pacotes tradicionais de *desktop publishing*.

Pode-se falar em um continuum de práticas de DC, em que o extremo menos computacional corresponde ao uso de linguagens de marcadores HTML e folhas de estilo CSS na estruturação de documentos hipertexto para a Web. O extremo oposto diz respeito a iniciativas de substituição integral do trabalho do designer por algoritmos, tais como aplicativos autônomos de diagramação, pintura digital e criação de identidades visuais, baseados em aprendizado de máquina (Sant'Anna, 2019).

Este ensaio apresenta reflexões sobre o ensino de DC utilizando a linguagem de programação RocketSocket (Sant'Anna, 2014, 2022), desenvolvida por mim e adotada no contexto do Curso de Design da Universidade Federal do Espírito Santo – Ufes (Vitória, ES – Brasil) desde 2021. Esta linguagem encontra-se em posição intermediária no continuum das práticas de DC, permitindo a estudantes de Design explorarem conceitos da Computação presentes em linguagens de programação profissionais, porém utilizando um ambiente orientado a iniciantes. Esta condição aproxima RocketSocket de outras linguagens e ambientes de programação de cunho didático, marcados por características como facilidade de acesso *online* ou instalação simplificada, oferta de exemplos, tutoriais e farta documentação para reduzir as barreiras de entrada.

No restante deste ensaio, discutirei as especificidades da linguagem RocketSocket e seus impactos no ensino de DC a estudantes de Design nos períodos letivos de 2021 e 2022. A Teoria do Controle Perceptivo, desenvolvida por William T. Powers (1926-2013) e colaboradores desde os anos 1960, subsidiará a apresentação das hipóteses sobre a pesquisa em andamento.

O que ensinar a designers sobre a Computação?

Denning & Martell (2015), sintetizando cerca de uma década de discussões sobre o ensino da Computação, propuseram "janelas" para os conhecimentos da área, denominadas "princípios" – *computation*, *communication*, *recollection*, *organization*, *evaluation* e *design*. As janelas, organizadas em uma estrutura hexagonal como uma casa de vidro, são uma metáfora para diferentes pontos de vista sobre problemas da Computação. Estes podem ser observados de modo particular, por janelas isoladas, ou de modo combinado, quando assumimos uma perspectiva em que a mesma questão é vista por várias janelas simultaneamente. Situações de ensino-aprendizagem dos princípios podem discutir, por exemplo, os limites do que pode ser computado; armazenamento, busca e recuperação confiáveis da informação; projeto e avaliação do desempenho de sistemas; e as estruturas por meio das quais tais sistemas são organizados e operam em conjunto.

Em uma segunda análise, relacionada ao funcionamento cognitivo de profissionais da Computação enquanto resolvem problemas, a ênfase recai sobre o desenvolvimento do

pensamento computacional ou *Computational Thinking* (CT). Este construto tem origem nos estudos pioneiros das aplicações de computadores à educação (cf. Papert, 1980), e voltou ao centro das atenções em meados dos anos 2000. Segundo argumentos mais populares pró-CT, a "forma de pensar" de cientistas da computação poderia ser aprendida por pessoas de qualquer área, e seria uma espécie de letramento essencial do século XXI (Wing, 2006, 2008). O desafio consistiria em criar as condições de desenvolvimento do CT entre pessoas de diversas idades, formações e interesses.

O quadro de referência elaborado por Brennan & Resnick (2012) se propôs a avaliar o desenvolvimento do CT por meio de: 1) aprendizagem de *conceitos* – sequências de instruções, laços, operadores, dados, condicionais, paralelismo e eventos; 2) adoção de *práticas* típicas de programadores com o CT, tais como desenvolver soluções de modo incremental, aprender a testar e depurar o código, reutilizar e recombinar soluções bem-sucedidas (sejam próprias ou de terceiros), e criar abstrações e modularizações no percurso rumo às soluções; e 3) tomada de diferentes *perspectivas* por aqueles que se expressam, se conectam a outras pessoas e questionam sua realidade por meio da Computação e do CT.

Embora orientado à linguagem Scratch (Maloney et al., 2010), o quadro de referência contempla a aproximação de iniciantes com a maioria das linguagens de programação disponíveis, seja porque os conceitos identificados são suficientemente gerais, existindo de alguma forma em todas as linguagens; seja porque as práticas e perspectivas do CT descritas manifestam-se em níveis mais elevados do funcionamento cognitivo, relativamente independentes de linguagens ou problemas particulares.

Sendo assim, o ensino de Design Computacional envolve tanto a apropriação dos princípios da Computação pelos estudantes em seu raciocínio de projeto, quanto o desenvolvimento de habilidades e competências típicas de programadores. Para projetar algoritmos que computam soluções para problemas de design, estudantes precisam interpretar o mundo em termos de processos de informação (Denning & Tedre, 2021), além de serem capazes de construir programas que implementem tais processos. A proposta da linguagem RocketSocket é satisfazer as duas demandas de modo integrado.

Da versão física para a versão Web de RocketSocket

Originalmente, RocketSocket foi desenvolvida como linguagem de programação "desplugada", de modo a ser utilizada em situações de ensino-aprendizagem sem a mediação de computadores (ver Sant'Anna, 2014, para detalhes construtivos). Os conceitos do CT eram abordados em missões em que estudantes controlavam o comportamento de um foguete que coletava estrelas, enquanto desviava de asteroides. O "espaço sideral" estava representado em um tabuleiro físico de 64 casas (8 linhas por 8 colunas), e cartas sorteadas para as missões indicavam a distribuição de estrelas, asteroides e a posição inicial do foguete sobre o tabuleiro. O trajeto do foguete durante a coleta era controlado por "programas" resultantes do encaixe de peças de MDF (*Medium-density Fiberboard*), referentes a movimentos básicos da espaçonave (p.ex., mover uma casa, girar múltiplos de 90°). Havia, ain-

da, estruturas para a construção de laços, decisões condicionais e realização de operações aritméticas.

Durante o processo de elaboração do programa, estudantes preenchiam o "plano de voo" (Sant'Anna, 2014, p. 195), em que imaginavam e representavam os sucessivos estados do tabuleiro, de acordo com a execução da sequência de instruções. Nesse sentido, planos de voo funcionavam como registros do raciocínio dos estudantes e, em caso de trajetos equivocados ou problemas na execução do programa, o estado imaginado e registrado no plano era comparado com estado real do tabuleiro. A discussão dos planos de voo ajudava estudantes a pensarem sobre o próprio pensamento e encontrar soluções para os problemas dos programas.

A versão Web de RocketSocket (Sant'Anna, 2022) surgiu em resposta às demandas do ensino remoto emergencial, durante o período de isolamento social causado pela pandemia de Covid-19 (2020-2021). O tabuleiro, foguete, estrelas e asteroides foram substituídos por encarnações digitais animadas, acessíveis por navegadores Web. As instruções em peças de encaixe de MDF foram adaptadas para comandos em português, digitados no editor de código. As "missões" foram transformadas em "fases", criadas por mim e disponibilizadas para as turmas diretamente no ambiente, em tabuleiros de até 32x32 casas. Os programas para controlar o foguete, durante a coleta, passaram a ser editados e armazenados na nuvem pelos estudantes, e posteriormente avaliados por mim e monitores da disciplina.

Aprimorando a versão física, RocketSocket na Web facilitou o fornecimento de exemplos e documentação, integrados ao ambiente de escrita de código. Ademais, permitiu aos estudantes verificarem o sucesso de seus programas imediatamente, sem a mediação do professor, bastando ordenar a execução do trajeto pelo "computador virtual" do foguete. Este, além de interpretar as instruções e informar sobre erros e acertos do código, proporcionou oportunidades de inspeção do funcionamento de computadores como recurso didático adicional. É possível visualizar quais instruções estão sendo executadas pelo computador virtual a cada momento, acompanhar passo a passo o histórico das instruções e seus efeitos sobre o tabuleiro, e avaliar o uso de memória pelo programa. Isto é, a versão Web incorporou recursos para que as "janelas" de Denning e Martell (2015) pudessem ser abordadas de modo integrado aos conceitos, práticas e perspectivas do pensamento computacional (Brennan & Resnick, 2012).

Finalmente, a versão Web introduziu o modo "Papert", em que o mesmo conjunto de instruções das missões de coleta pode ser utilizado para desenhar. Neste modo, o foguete é substituído por uma caneta, o tabuleiro torna-se uma folha em branco de até 512 por 512 casas (*pixels*) e os trajetos dão origem a formas coloridas. Instruções adicionais tornam-se disponíveis no modo desenho, tais como levantar e abaixar a caneta, para movê-la sem riscar a tela; mudar a espessura, cor da tinta da caneta e do fundo da tela; e escrever textos com fontes disponíveis no sistema do usuário.

Propósitos e Design Computacional

A aplicação dos conhecimentos sobre coletar estrelas a problemas de desenho consiste no desafio atual da pesquisa sobre RocketSocket. Com base nas atividades realizadas pelas turmas de Design Computacional dos semestres letivos 2021/1, 2021/2 e 2022/1 (cerca de 180 estudantes), é possível observar a ocorrência de um "salto" qualitativo nas competências por meio das quais estudantes se expressam utilizando conhecimentos e práticas da Computação.

Discentes iniciam a disciplina construindo trajetos simples, formados por linhas horizontais e verticais sobre uma malha de baixa resolução (até 1024 ou 2^{10} casas). Até o final do semestre, superam a ortogonalidade dos movimentos e produzem desenhos complexos, com temáticas diversas, com até 2^{18} casas (*pixels*) de resolução: ilustrações, letreiros personalizados, reproduções de logotipos, pictogramas de jogos olímpicos, padrões para ladrilhos hidráulicos e estampas, brasões e escudos de times, personagens e elementos de jogos, séries e filmes (ver Sant'Anna, 2022, para exemplos).

Parece haver indícios de que a autonomia para escolher as aplicações de RocketSocket contribui para acelerar a apropriação da linguagem pelas turmas. Nos primeiros momentos da disciplina, ainda que as missões se assemelhem a jogos e contenham relativo grau de desafio lúdico, os objetivos e propósitos da aprendizagem são definidos pelo professor. Na transição para as atividades avançadas de desenho, os objetivos são estabelecidos por cada estudante, e os propósitos de aprendizagem da linguagem parecem adquirir outros significados. Frente a este cenário, a investigação e compreensão dos propósitos e motivações dos estudantes tornaram-se centrais para avaliar a pertinência de RocketSocket como ferramenta didática.

A *Perceptual Control Theory* (PCT) ou Teoria do Controle Perceptivo (Powers, 1973) argumenta que controlamos nossas experiências de modo a manter o estado de coisas em *condições de referência* que consideramos *ideais*, segundo nossos propósitos. Percebemos o nosso entorno, as relações que mantemos com outras pessoas e objetos, e realizamos comparações contínuas com aquilo que entendemos como ideal para cada situação. Caso haja diferenças entre o estado desejado e o percebido (o "erro"), nos engajamos em ações para reduzi-las e reaproximar a situação de condições satisfatórias.

Aplicações da PCT a contextos educacionais (cf. Carey, 2012) descrevem o papel do professor como criador de oportunidades para que estudantes experimentem novas condições de referência, e possam ser ajudados a desenvolver as habilidades necessárias para reduzir as diferenças entre o que percebem e o que desejam atingir. Nesses termos, RocketSocket expõe estudantes a novos estados de coisas sobre a Computação, eventualmente distintas daqueles previamente estabelecidos. Estudantes de Design dificilmente estão alheios às práticas criativas com os dispositivos computacionais contemporâneos, embora, conforme mencionado no início deste ensaio, assumam mais frequentemente os papéis de *usuários* que de *autores* daquelas tecnologias. A mudança de papéis discentes requer revisão de suas competências para a ação, para que possam vislumbrar novos propósitos e objetivos.

Ações, na PCT, são parte do laço de controle do organismo para que as percepções sejam coerentes com nossos propósitos (Powers, 1973). Há uma organização hierárquica de unidades Perceber-Comparar-Atuar (PCAs), desde a percepção de intensidades pelas terminações nervosas nos níveis inferiores da hierarquia, até a percepção de valores, crenças, e visões de mundo nos níveis superiores (Carey, 2006). A interação entre os níveis da hierarquia opera pela propagação ascendente de sinais perceptivos, e pelo fornecimento das condições de referência no sentido descendente. Em outras palavras, as "ações" dos níveis mais altos da hierarquia podem não ser públicas (movimentos, gestos, vocalizações), mas fornecem padrões para a comparação de sinais perceptivos dos níveis mais baixos. No sentido ascendente da hierarquia dos propósitos, pergunta-se o *porquê* de as pessoas quererem ou desejarem que as coisas sejam de tal forma, enquanto no sentido descendente, questiona-se *como* as pessoas querem que as coisas sejam (Mansell et al., 2012).

Desse modo, a hipótese sob investigação articula os dois sentidos da hierarquia das unidades PCA. Os *porquês* relatados pelos estudantes, em suas justificativas de projeto com RocketSocket, envolvem tanto desejos de expressão individual e de conexão com seus pares, mediados pela tecnologia (as tomadas de perspectiva de Brennan & Resnick, 2012), quanto interesses em compreender as "janelas" da Computação (Denning & Martell, 2015), de modo a pensarem suas realidades a partir daqueles princípios. São propósitos de alto nível, que fornecem condições de referência para as aplicações de conceitos e adoção de práticas do pensamento computacional nos níveis inferiores, perseguindo, assim, os objetivos desejados.

No entanto, perseguir objetivos e propósitos não significa ausência de dificuldades. A PCT explica que a persistência de erros que não conseguem ser eliminados pode gerar mal-estar e sensação de perda de controle (Carey, 2006). Por outro lado, a existência de propósitos incompatíveis e perseguidos simultaneamente pode causar conflitos e sofrimento (Mansell et al., 2012). Essas questões têm implicações importantes para as dificuldades de aprendizagem e frustrações enfrentadas por estudantes, enquanto exploram conhecimentos da Computação ou de qualquer área.

A possibilidade de desenvolver projetos motivados por interesses pessoais, com expectativas definidas pelos próprios estudantes, parece mitigar parte das dificuldades que identifiquei nos três primeiros semestres da disciplina Design Computacional. Os problemas experimentados nas missões de coleta de estrelas são mais simples e limitados, e ainda assim parecem causar mais desconforto que os projetos de desenho, que têm temática livre e execução mais complexa.

Assim sendo, outro aspecto da pesquisa em andamento consiste na adaptação do *Method of Levels* (MOL), ou Método dos Níveis, a contextos educacionais. MOL consiste em forma transdiagnóstica de Terapia Cognitivo-Comportamental, aplicação direta da PCT à psicoterapia (Carey, 2006, 2008; Mansell et al., 2012). Esta adaptação pretende investigar as motivações dos estudantes enquanto desenvolvem seus programas, considerando a hierarquia de propósitos ("porquês" e "comos") da PCT.

Para concluir, é importante ressaltar a necessidade de identificação das unidades PCA que integrariam a hierarquia de controle descrita. Se as suposições da PCT e MOL estiverem corretas, o acompanhamento dos progressos dos estudantes permitirá investigar em que

medida as "janelas" para os conhecimentos da Computação e tomadas de perspectiva do pensamento computacional estabelecem as condições ideais para as práticas concretas de Design Computacional, utilizando a linguagem RocketSocket.

Referências

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada*, *1*, 25.

Carey, T. A. (2006). *The method of levels: How to do psychotherapy without getting in the way*. Living Control Systems Publishing.

Carey, T. A. (2008). *Hold that Thought: Two Steps to Effective Counseling and Psychotherapy with the Method of Levels*. newview.

Carey, T. A. (2012). *Control in the classroom: An adventure in learning and achievement*. Living Control Systems Publishing.

Denning, P. J., & Martell, C. H. (2015). *Great Principles of Computing*. MIT Press.

Denning, P. J., & Tedre, M. (2021). Computational Thinking: A disciplinary perspective. *Informatics in Education*, *20*(1), 361–390. https://doi.org/10.15388/infedu.2021.21

Maeda, J. (2004). *Creative Code*. Thames and Hudson.

Maloney, J., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. (2010). The scratch programming language and environment. *ACM Transactions on Computing Education (TOCE)*, *10*(4), 16.

Mansell, W., Carey, T. A., & Tai, S. J. (2012). *A transdiagnostic approach to CBT using method of levels therapy: Distinctive features*. Routledge.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic Books.

Powers, W. T. (1973). *Behavior: The control of perception*. Aldine Publishing Company.

Sant'Anna, H. C. (2014). *Ação, Computação, Representação: Uma investigação psicogenética sobre o desenvolvimento do Pensamento Computacional* [Doutorado em Psicologia, Universidade Federal do Espírito Santo]. http://repositorio.ufes.br/handle/10/9083

Sant'Anna, H. C. (2019). Revisão crítica das aplicações de Aprendizado de Máquina no Design Visual: Bases teóricas, desempenho dos modelos e novos paradigmas de projeto. *Anais do SIIMI/2019 VI Simpósio Internacional de Inovação em Mídias Interativas*. SIIMI/2019 VI Simpósio Internacional de Inovação em Mídias Interativas, Buenos Aires.

Sant'Anna, H. C. (2022). Entre foguetes, estrelas e canetas: Relato de experiência de ensino de princípios da Computação para estudantes de graduação em Design. *Projetica*, *13*(3), Art. 3. https://doi.org/10.5433/2236-2207.2022v13n3p166-183

Wing, J. M. (2006). Computational Thinking. *Communications of the ACM*, *49*(3), 33–35. https://doi.org/10.1145/1118178.1118215

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, *366*(1881), 3717–3725. https://doi.org/10.1098/rsta.2008.0118

Abstract: Computational Design (CD) is a design approach whose products are algorithms that compute solutions to problems of interest to the designer. Since 2021, the teaching of CD at the Federal University of Espírito Santo (Ufes) is based on RocketSocket, a language created to present principles, concepts, practices, and perspectives of Computing to undergraduate Design students. This essay describes preliminary results of the didactic experience in the first three terms of the CD class at Ufes (2021-2022) and presents hypotheses about the ongoing research.

Keywords: Computational Design - Computational Thinking - Computing Principles - Perceptual Control Theory - Method of Levels.

Resumo: O Design Computacional (DC) é uma abordagem de projeto cujos produtos são algoritmos que computam soluções para problemas de interesse do projetista. Desde 2021, o ensino de DC na Universidade Federal do Espírito Santo (Ufes) é baseada em RocketSocket, uma linguagem criada para apresentar princípios, conceitos, práticas e perspectivas da Computação a estudantes de gradação em Design. Este ensaio relata resultados preliminares da experiência didática nos três primeiros semestres da disciplina de DC na Ufes (2021-2022), e apresenta hipóteses sobre a investigação em andamento.

Palavras chave: Design Computacional - Pensamento Computacional - Princípios da Computação - Teoria do Controle Perceptivo - Método dos Níveis

[Las traducciones de los abstracts fueron supervisadas por su autor]