DOI: 10.5748/19CONTECSI/PSE/EDU/6979

CONTENT AND COMPETENCES FOR TEACHING SOFTWARE DESIGN IN THE COMPUTER SCIENCE COURSE: A MAPPING OF CC-2020, RF-CC-2017 AND SWEBOK-V3.0 ASSETS

CONTEÚDOS E COMPETÊNCIAS PARA O ENSINO DE SOFTWARE DESIGN NO CURSO DE CIÊNCIA DA COMPUTAÇÃO : UM MAPEAMENTO DOS ATIVOS CC- 2020, RF-CC-2017 E O SWEBOK-V3.0

Vitor de Souza Castro ; https://orcid.org/0000-0003-3209-3806
UNIFESSPA

Sandro Ronaldo Bezerra Oliveira ; https://orcid.org/0000-0002-8929-5145
UFPA - Universidade Federal do Pará

CONTENT AND COMPETENCES FOR TEACHING SOFTWARE DESIGN IN THE COMPUTER SCIENCE COURSE: A MAPPING OF CC-2020, RF-CC-2017 AND SWEBOK-V3.0 ASSETS

ABSTRACT: The teaching of Software Engineering in higher computing courses needs to adapt to new technologies and tools that arise all the time. The Software Design area has the challenge of integrating the diversity of techniques in order to present a solution to a given problem, however the treatment of this challenge is largely not reflected in the curricula of Computing courses. In this sense, the objective of this work is to develop the mapping of assets between the curriculum of ACM/IEEE, SBC and SWEBOK guide specifically for the Software Design knowledge area in the Computer Science course. To this end, the methodology of analysis of reference documents was used to develop the mapping and the peer review process aimed at validating this mapping. As a result, we highlight the correlation between the assets of CC2020 and RF-CC-2017 in the vision of competencies and the mapping of contents between the three reference documents related to Software Design.

Keywords: Software Design, Asset Mapping, ACM/IEEE, SBC, SWEBOK.

CONTEÚDOS E COMPETÊNCIAS PARA O ENSINO DE SOFTWARE DESIGN NO CURSO DE CIÊNCIA DA COMPUTAÇÃO: UM MAPEAMENTO DOS ATIVOS CC-2020, RF-CC-2017 E O SWEBOK-V3.0

RESUMO: O ensino de Engenharia de Software nos cursos superiores de computação precisam adaptar-se as novas tecnologias e ferramentas que surgem a todo momento. A área de *Software Design* possui o desafio de integrar a diversidade de técnicas com o objetivo de apresentar uma solução para determinado problema, no entanto o trato deste desafio, em grande parte não se reflete nos currículos dos cursos de Computação. Neste sentido, o objetivo deste trabalho é desenvolver o mapeamento de ativos entre os *curriculum* da ACM/IEEE, da SBC e guia SWEBOK especificamente para área de conhecimento *Software Design* no curso de Ciência da Computação. Para tal, utilizou-se a metodologia de análise dos documentos de referência para desenvolvimento do mapeamento e o processo de revisão por pares objetivando a validação deste mapeamento. Como resultados, destacam-se a correlação entre os ativos da CC2020 e RF-CC-2017 na visão das competências e o mapeamento dos conteúdos entre os três documentos de referência relacionados ao *Software Design*.

Palavras-chave: Software Design, Mapeamento de Ativos, ACM/IEEE, SBC, SWEBOK.

Agradecimentos: Este trabalho pertence ao projeto SPIDER (*http://www.spider.ufpa.br*).

1 INTRODUÇÃO

O ensino de Engenharia de Software nos cursos superiores de computação precisam adaptar-se as novas tecnologias e ferramentas que surgem a todo momento. Um dos desafios para a área da Engenharia de Software é a alta heterogeneidade de soluções, levando a sistemas híbrido, com multi-paradigmas, por exemplo, misturando componentes monolíticos, baseados em serviço, *serverless*, de aprendizado de máquina, habilitados para *IoT* e com tecnologia quântica (Forti *et al.* 2022). Nota-se que essa diversidade de estratégias e técnicas para projetar um software precisaria obrigatoriamente ser abordada nos cursos de computação. No entanto, essa diversidade não é notada nos currículos.

Aniche *et al.* (2019) apresenta em seu trabalho desafios para a área de *Software Design*, tais como a forte influência das bibliotecas e *frameworks* na decisão do design do software, metáfora para os *stakcholders* e as diferentes representações do mesmo problema e a necessidade de medições no contexto da qualidade do *design*.

Em se tratando de projeto de software, (de Pádua Paula Filho, 2019) apresenta a fase de Desenho ou *Design*, como a responsável por definir uma estrutura implementável para um produto de software, que atenda aos requisitos especificados. Na mesma linha, (Pressman e Maxim 2016) afirma que a etapa de projeto de software abrange o conjunto de princípios, conceitos e práticas que levam ao desenvolvimento de um produto.

Para subsidiar os conteúdos necessários para o ensino de computação, existem diversas entidades que se propõem a desenvolver currículos e diretrizes para esses cursos, dentre os quais destaca-se a *Association for Computing Machinery* (ACM) e o *Institute of Electrical and Electronics Engineers* (IEEE) no contexto internacional e a Sociedade Brasileira de Computação (SBC) no contexto brasileiro.

Essas entidades promovem discussões com a comunidade científica e profissional para o desenvolvimento de *curriculum*, congregando diretrizes, conteúdos e competências importantes para o profissional da área de Computação. Esses *curriculum* servem de base para o desenvolvimento dos Projetos Pedagógico de Curso das Instituições de Ensino Superior.

Nesse contexto, o presente trabalho tem como objetivo o desenvolvimento do mapeamento de ativos da ACM/IEEE, da SBC e guia SWEBOK especificamente para área de conhecimento Software Design, para gerar um conjunto de conteúdos e competências para serem trabalhadas em sala de aula, alinhados aos referenciais nacionais e internacionais.

Além desta seção introdutória, este artigo está organizado como segue: a Seção 2 apresenta a fundamentação teórica desse trabalho; a Seção 3 é apresentado alguns trabalhos relacionados; a Seção 4 apresenta a metodologia para desenvolvimento do mapeamento; na Seção 5, é apresentada o mapeamento dos ativos; na Seção 6, é apresentada os resultados deste trabalho; a Seção 7 apresenta as ameaças a validade do trabalho; e, por fim, a Seção 8 apresenta as considerações finais, as limitações e os trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Esta seção tem como objetivo a apresentação dos elementos centrais do mapeamento proposto, sendo, o *curriculum* da ACM/IEEE, *curriculum* da SBC e o guia SWEBOK.

2.1 Computing Curricula 2020

A *Computing Curricula 2020*, CC2020, trata-se de uma iniciativa lançada em conjunto por várias sociedades profissionais de computação para resumir e sintetizar o estado atual das diretrizes curriculares para programas acadêmicos que conferem graus de bacharelado em computação (Force, 2020). O desenvolvimento de alinhamento das diretrizes curriculares internacionais em computação teve a sua primeira versão em 1968. Para o escopo deste

trabalho a versão do relatório a ser utilizada foi a CC2020, tendo como principais envolvidas na organização a (ACM) e a *IEEE Computer Society* (IEEE-CS).

O objetivo da CC2020 é ser uma extensão moderna do relatório CC2005, fornecendo orientação global em um ambiente de computação em evolução, pois afeta programas de bacharelado em computação em todo o mundo (Force, 2020).

O curso de Ciência da Computação (CC) foi o curso escolhido para ser objeto de estudo neste trabalho, em função de suas características direcionada para o processo de desenvolvimento de software, sendo pensar e resolver problemas, possuir um estilo que emerge das experiências obtidas através do estudo da área e da prática profissional em computação (Joint Task Force on Computing Curricula and Society 2013). Além disso, o curso de CC possui maior concentração de unidades de ensino com ênfase nos Fundamentos do Desenvolvimento de Software, o que faz possível a relação com as áreas de conhecimento do SWEBOK, seção 2.3.

Sobre a sua estrutura, o CC2020 está dividida nos ativos: *Courses*, *Knowledge Areas*, *Competencies* e *Elements of Foundation and Professional Knowledges*.

2.2 Referência de Formação em Computação da SBC

A Sociedade Brasileira de Computação (SBC) tem papel fundamental no direcionamento no ensino da computação. As comissões internas da SBC elaboram e aprovam os "Referenciais de Formação em Computação"' (RF) , que devem seguir as Diretrizes Curriculares Nacionais (DCN) e utilizar um modelo baseado em competência (Zorzo *et al*. 2017).

Neste trabalho, foi selecionado o curso de Ciência da Computação, pois trata-se de um curso que tem por objetivo formar cientistas da computação são responsáveis pelo desenvolvimento científico (teorias, métodos, linguagens, modelos, entre outras) e tecnológico da Computação (MEC 2016). Além disso, sua estrutura possui elementos centrais para o desenvolvimento de software, tais como Programação, Design de Software e Engenharia de Software. Os Referências de Formação especificamente em Ciência da Computação será citado neste trabalho como RF-CC-2017.

2.3 *Software Design*

O *Guide to the Software Engineering Body of Knowledge*, o SWEBOK, é um guia que estabelece uma linha base para o corpo de conhecimento na área de Engenharia de Software (Bourque e Fairley 2014).

O SWEBOK está dividido em *Knowledge Area* que estabelece uma caracterização do conteúdo de engenharia de software. Neste trabalho, será tratado a *Knowledge Area Software Design*, que faz a interface entre a área de Requisitos, com a área de Construção do software. Além disso, estabelece conteúdos necessários para desenvolvimento do Design de alto nível, especificando os componentes para facilitar a construção. Faz parte dessa *Knowledge Area* a decomposição do projeto de software, utilizando padrões de projetos para subsidiar o processo de construção do software.

Neste trabalho, foi selecionado a versão 3.0 do SWEBOK e para referenciar essa versão, será utilizado a sigla SWEBOK-v3.0.

A Figura 1 apresenta o desenho relacionando os elementos centrais utilizados para o mapeamento dos ativos. Destaca-se o curso de Ciência da Computação como elemento central, pois faz interface entre os dois referenciais (RF-CC-2017 e CC2020) e a área de conhecimento *Software Design* do SWEBOK.

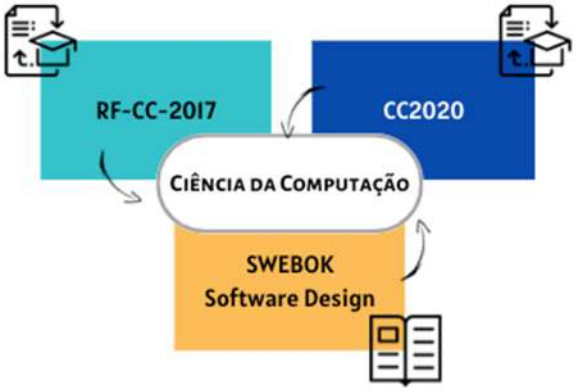

Figura 1 – Elementos Centrais do Mapeamento
Fonte: Elaboração própria (2022).

3 TRABALHOS RELACIONADOS

A presente seção tem como objetivo apresentar alguns trabalhos relacionados ao objeto da pesquisa. A metodologia utilizada para execução da pesquisa dos trabalhos relacionados seguiu os passos: (1) identificação de palavras-chave sobre o objeto da pesquisa; (2) definição das bases de conhecimento para a execução da pesquisa; e (3) seleção dos trabalhos relevantes.

Na execução da pesquisa foram encontrados alguns trabalhos relacionados que realizaram o mapeamento entre os ativos selecionados neste trabalho. No entanto, observa-se que para a área de conhecimento *Software Design*, nenhum trabalho relacionado foi encontrado.

O trabalho de (Calsavara et al. 2018) apresenta um estudo com o objetivo de desenvolver um instrumento para auxiliar a coordenação e o Núcleo Docente Estruturante (NDE) de um curso a elaborarem dois documentos: descritivo de disciplina e plano de ensino. Diferentemente de (Calsavara et al. 2018), este trabalho apresenta um mapeamento entre os ativos da ACM/IEEE CC2020, SBC RF-CC-2017 e o SWEBOK-v3.0.

Oliveira e Elgrably (2019) apresentam um estudo cujo objetivo foi desenvolver um mapeamento entre diversos ativos com o intuito de estabelecer uma base de conhecimento e ser utilizado como referência para o desenvolvimento de uma abordagem de Ensino e Treinamento de Testes com Métodos Ágeis. Para tal, realizou o mapeamento dos ativos da ACM/IEEE CC2013, SBC RF-CC-2005, SWEBOK-v3.0, TMMi e práticas ágeis.

Por fim, Quaresma e Oliveira (2020) realizou o mapeamento dos ativos da ACM/IEEE CC2013, SBC RF-CC-2017 e o SWEBOK-v3.0 para área de conhecimento *Software Process*. Diferentemente dos trabalhos de (Oliveira e Elgrably 2019) e (Quaresma e Oliveira 2020), o presente trabalho realizou o mapeamento dos ativos na área de conhecimento *Software Design* para o curso de Ciência da Computação.

Tabela 1 – Trabalhos Relacionados

Características/Trabalho	**Calsavara et al 2018**	**Oliveira and Elgrably 2019**	**Quaresma and Oliveira 2020**	**Este Trabalho**
Uso da CC2020	X			X
Uso da RF-CC-2017	X		X	X
Uso do SWEBOK-v3.0	X	X	X	X

Características/Trabalho	Calsavara et al 2018	Oliveira and Elgrably 2019	Quaresma and Oliveira 2020	Este Trabalho
Area de conhecimento *Software Design*				X

Fonte: Elaboração própria (2022).

4 METODOLOGIA DO MAPEAMENTO

A presente seção apresenta a metodologia utilizada para realização do mapeamento dos ativos, destacando os passos necessários e produtos gerados em cada passo.

Visando a organização do mapeamento, os pesquisadores definiram um conjunto de passos a serem seguidos, sendo:

1. Definição da temática a ser desenvolvida no mapeamento;
2. Identificação dos documentos de referências e *curriculum* em Computação;
3. Identificação e esquematização dos ativos dos documentos de referência identificados no passo 2;
4. Realização da correspondência entre os ativos, bem como justificativa para determinação da relação;
5. Realização da revisão por pares dos resultados obtidos com especialista;
6. Consolidação do mapeamento em sua versão final.

No passo 1, a definição da temática a ser desenvolvida no mapeamento considerou a revisão informal na literatura, realizada pelos pesquisadores de modo a identificar a área de conhecimento a ser estudada. A área escolhida foi o *Software Design*, que integra uma das áreas de conhecimento definida no SWEBOK-v3.0.

Em função da existência de Diretrizes Curriculares de referência no Brasil para área de Computação definida pela SBC e no contexto internacional a ACM/IEEE como entidade de referência para Computação, as diretrizes mantidas pelas duas entidades foram selecionadas para compor o conjunto de documentos para realização do mapeamento, atividade essa realizada para contemplar o passo 2.

No passo 3, foi realizado um estudo sobre os ativos constantes nos documentos de referências selecionados, sendo: SWEBOK-v3.0, SBC RF-CC-2017 e ACM/IEEE CC-2020. O objetivo deste passo é relacionar o objetivo de cada ativo para possibilitar o relacionamento entre si.

Com a identificação dos objetivos de cada ativo, no passo 4, foi realizado a correspondência entre os ativos dos documentos de referência selecionados, bem como a inclusão de justificativa para tal mapeamento.

Após a execução do passo 4, foi realizado o processo de revisão por pares com especialista, o passo 5. Neste passo, foi apresentado para o revisor o mapeamento realizado, bem como as justificativas para cada mapeamento. Esse processo foi realizado por meio de reunião remota e as melhorias necessárias foram relatadas durante o processo de apresentação.

Por fim, no passo 6 foi realizado a consolidação do mapeamento, adicionando as contribuições identificadas pelo revisor no passo 5.

5 MAPEAMENTO DOS ATIVOS

O mapeamento dos ativos visa a identificação, conceituação e relacionamento dos elementos constantes nos documentos de referência selecionados para esta pesquisa.

Iniciando pelo SWEBOK-v3.0, observa-se os seguintes ativos: *Knowledge Area, Topics e SubTopics*, conforme Tabela 2.

Tabela 3 – Ativos do SWEBOK-v3.0

Ativo	**Definição**
Knowledge Area	Uma caracterização do conteúdo de engenharia de software
Topic	Decomposição de uma Knowledge Area em um grupo de conteúdo
SubTopic	**Decomposição de um Topic contendo conteúdo específico**

Fonte: Elaboração própria (2022).

A estrutura do SWEBOK-v3.0 descreve o corpo de conhecimento em Engenharia de Software dividindo-a em 15 *Knowledge Area*, dentre as quais a *Software Design*, objeto desse mapeamento.

Sobre o documento de referência da SBC RF-CC-2017, os ativos identificados foram: Curso, Eixo de Formação, Competências derivadas e Conteúdos. Os cursos definidos no documento de referencia são Ciência da Computação, Engenharia da Computação, Engenharia de Software, Licenciatura em Computação, Sistemas de Informação e Cursos Superiores em Tecnologia. A Tabela 2 apresenta a conceituação dos ativos de acordo com a RF-CC-2017.

Tabela 4 – Ativos do SBC RF-CC-2017

Ativo	**Definição**
Curso	Define o Perfil esperado para o egresso
Eixo de Formação	Capacitar o egresso em competências genéricas
Competência	Determina a necessidade de serem desenvolvidas em conteúdos específicos
Conteúdos	Assunto a ser estudado com o objetivo na obtenção das competências

Fonte: Elaboração própria (2022).

Por fim, o documento de referência da ACM/IEEE CC-2020 apresenta os seguintes ativos: *Knowledge Areas*, *Courses*, *Competencies* e *Elements of Foundation and Professional Knowledges*.

Tabela 5 – Ativos do ACM/IEEE CC2020

Ativo	**Definição**
Courses	Unidade de estudo reconhecida institucionalmente
Knowledge Areas	Conhecimento em foco para determinada área de estudo
Competencies	Coisas que um indivíduo deve demonstrar ser eficaz em um trabalho, papel, função, tarefa ou dever.
Elements of Foundation and Professional Knowledges	Habilidades fundamentais centrada no indivíduo, tais como apresentação, escrita, autogerenciamento do tempo, e outras.

Fonte: Elaboração própria (2022).

A Figura 2 sintetiza o mapeamento dos ativos selecionados neste trabalho. Observa-se que a figura geométrica de cada nó faz correspondência entre as estruturas, a exemplo do

losango que na SBC RF-CC-2017 trata-se do Conteúdos, na ACM/IEEE CC-2022 trata-se da *Knowledge Areas* e no SWEBOK a Topics.

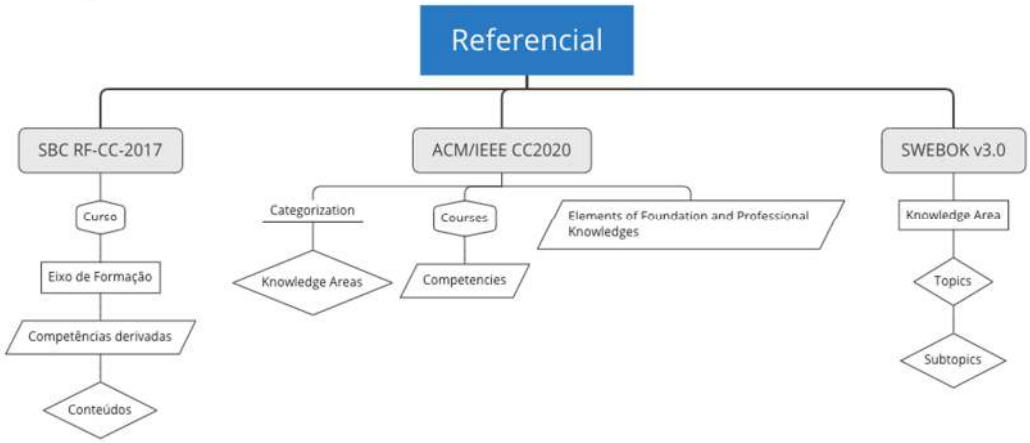

Figura 2 – Mapeamento dos Ativos
Fonte: Elaboração própria (2022).

O primeiro mapeamento realizado foi entre os SubTopics do SWEBOK-v3.0, os Conteúdos da SBC e as *Knowledge Areas* da ACM/IEEE, conforme Tabela 5. Nesse mapeamento, optou-se por realizar a relacionamento a nível de *SubTopics* pois trata-se de um nível específico do *Topic* do SWEBOK-v3.0 e por isso apresenta detalhamento sobre quais conteúdos são exigidos para atendimento ao *Topic*.

Para tal mapeamento, foi realizado a análise documental do SWEBOK-v3.0, na perspectiva de identificar elementos que caracterizem os *SubTopics*. Como exemplo, pode-se citar o *SubTopic General Design Concepts*, associado ao *Topic Software Design Fundamentals*, que apresenta como conteúdos os conceitos inicial de Design, representações, objetivos com o desenvolvimento do Design.

Nota-se que os conteúdos referente a Interação Humano-Computador (IHC) atende na totalidade os *SubTopics* relacionados ao *User Interface Design*. Neste sentido, observa-se nos Projeto Pedagógicos de Cursos a presença da disciplina de IHC de forma distinta das disciplinas de Análise e Projeto de Software.

Tabela 5 – Mapeamento *Topics/SubTopics*

SWEBOK (Topics)	**SBC (Conteúdos)**	**ACM/IEEE (Knowledge Areas)**
General Design Concepts	Engenharia de Software	Systems Analysis and Design
	Projeto de Algoritmo	Software Design
	Modelagem de Sistemas	Software Modeling and Analysis
Context of Software Design	Engenharia de Software	Software Process
Software Design Process	Modelagem de Sistemas	Software Modeling and Analysis
	Engenharia de Software	Systems Analysis and Design
Software Design Principles	Estrutura de Dados	Data Structures, Algorithms and Complexity

	Programação Funcional	Programming Fundamentals
	Programação Orientada a Objetos	
	Projeto de Algoritmo	
Concurrency	Engenharia de Software	Systems Analysis and Design
		Enterprise Architecture
		Parallel and Distributed Computing
Control and Handilg of Events	Programação Funcional	Programming Fundamentals
	Programação Orientada a Objetos	Security Issues and Principles
	Segurança de Sistemas Computacionais	
Data Persistence	Banco de Dados	Data Structures, Algorithms and Complexity
	Estrutura de Dados	Data and Information Management
Distribution of Components	Engenharia de Software	Data Structures, Algorithms and Complexity
	Modelagem de Sistemas	Platform Technologies
	Padrões de Projeto	Platform-Based Development
	Programação de Aplicativos para Dispositivos Móveis	Software Design
	Programação de Aplicações Web	Software Modeling and Analysis
	Internet das Coisas (IoT)	Systems Analysis and Design
	Realidade Virtual e Aumentada	
	Computação Gráfica	
Error and Exception Handling and Fault	Programação Funcional	Programming Fundamentals
	Programação Orientada a Objetos	Security Issues and Principles
	Segurança de Sistemas Computacionais	
	Programação de Aplicativos para Dispositivos Móveis	
	Programação de Aplicações Web	
Interaction and Presentation	Engenharia de Software	Systems Analysis and Design

	Modelagem de Sistemas	
Security	Segurança de Sistemas Computacionais	Security Issues and Principles
	Criptografia	Security Technology and Implementation
Architectural Strutures and Viewpoints	Engenharia de Software	Enterprise Architecture
	Modelagem de Sistemas	Systems Analysis and Design
		Software Design
		Software Modeling and Analysis
Architectural Styles	Modelagem de Sistemas	Enterprise Architecture
	Padrões de Projeto	Systems Analysis and Design
		Software Design
		Software Modeling and Analysis
Design Patterns	Padrões de Projeto	Software Design
		Software Modeling and Analysis
Families of Programs and Frameworks	Programação Orientada a Objetos	Platform-Based Development
	Projeto de Algoritmo	Programming Fundamentals
	Programação de Aplicativos para Dispositivos Móveis	Platform Technologies
	Programação de Aplicações Web	Integrated Systems Technology
	Internet das Coisas (IoT)	
	Realidade Virtual e Aumentada	
	Computação Gráfica	
General User Interface Design Principles	Interação Humano-Computador	User Experience Design
	Engenharia de Software	
User Interface Design Issues	Interação Humano-Computador	User Experience Design
The Design of User Interaction Modalities	Interação Humano-Computador	User Experience Design
The Design of Information Presentation	Interação Humano-Computador	User Experience Design
User interface Design Process	Interação Humano-Computador	User Experience Design
	Engenharia de Software	
Localization and Internationalization	Interação Humano-Computador	User Experience Design
Metaphors and Conceptual Models	Interação HumanoComputador	User Experience Design

Quality Attributes	Engenharia de Software	Software Quality, Verification and Validation
	Avaliação de Desempenho	
Quality Analysis and Evaluation Techniques	Engenharia de Software	Software Quality, Verification and Validation
	Avaliação de Desempenho	
Measures	Avaliação de Desempenho	Software Quality, Verification and Validation
Structural Descriptions	Engenharia de Software	Data Structures, Algorithms and Complexity
	Estrutura de Dados	Systems Analysis and Design
	Modelagem de Sistemas	Software Design
	Padrões de Projeto	Software Modeling and Analysis
	Projeto de Algoritmo	
Behavioral Descriptions	Engenharia de Software	Data Structures, Algorithms and Complexity
	Estrutura de Dados	Systems Analysis and Design
	Modelagem de Sistemas	Software Design
	Padrões de Projeto	Software Modeling and Analysis
	Projeto de Algoritmo	
General Strategies	Engenharia de Software	Systems Analysis and Design
	Projeto de Algoritmo	Data Structures, Algorithms and Complexity
Function-Oriented Design	Padrões de Projeto	Data Structures, Algorithms and Complexity
	Programação Funcional	Programming Fundamentals
	Projeto de Algoritmo	Software Design
	Estrutura de Dados	Software Modeling and Analysis
	Modelagem de Sistemas	Systems Analysis and Design
Object-Oriented Design	Programação Orientada a Objetos	Programming Fundamentals

	Engenharia de Software	Software Design
	Projeto de Algoritmo	Software Modeling and Analysis
	Estrutura de Dados	Systems Analysis and Design
	Padrões de Projeto	Data Structures, Algorithms and Complexity
	Modelagem de Sistemas	
Data Structure-Centered Design	Estrutura de Dados	Data Structures, Algorithms and Complexity
	Banco de Dados	Software Design
	Padrões de Projeto	Software Modeling and Analysis
	Modelagem de Sistemas	Systems Analysis and Design
Component-Based Design	Engenharia de Software	Software Design
	Modelagem de Sistemas	Software Modeling and Analysis
	Padrões de Projeto	Systems Analysis and Design
		Platform Technologies
Others Methods	Engenharia de Software	Enterprise Architecture
	Modelagem de Sistemas	Integrated Systems Technology
	Padrões de Projeto	Platform Technologies
		Platform-Based Development
		Software Design
		Software Modeling and Analysis
		Systems Analysis and Design
Software Design Tools	Engenharia de Software	Software Design
	Modelagem de Sistemas	Software Modeling and Analysis
		Systems Analysis and Design

Sobre a Tabela 5, observa-se que para o relacionamento com os conteúdos da SBC RF-CC-2017, os pesquisadores realizaram o mapeamento fazendo uso do conhecimento observado nos Currículos das Instituições de Ensino Superior sobre o determinado conteúdo, haja vista que na RF-CC-2017, não há detalhamento ou hierarquia de conteúdos. Para o contexto da CC-2020, as *Knowledge Areas* possuíam algum nível de detalhamento que possibilitou o mapeamento entre os ativos.

Em função da necessidade da vinculação entre os *SubTopics* e o Topics, destacada no processo de revisão por pares, seção 5.1, a Tabela 6, apresenta esta relação.

Tabela 6 – Relacionamento entre *Topics* e *SubTopics* SWEBOK-v3.0

Topics	***SubTopics***
Software Design Fundamentals	General Design Concepts Context of Software Design Software Design Process Software Design Principles
Key Issues in Software Design	Concurrency Control and Handilg of Events Data Persistence Distribution of Components Error and Exception Handling and Fault Tolerance Interaction and Presentation Security
Software Struture and Architecture	Architectural Strutures and Viewpoints Architectural Styles Design Patterns Families of Programs and Frameworks
User Interface Design	General User Interface Design Principles User Interface Design Issues The Design of User Interaction Modalities The Design of Information Presentation User interface Design Process Localization and Internationalization Metaphors and Conceptual Models
Software Design Quality Analysis and Evaluation	Quality Attributes Quality Analysis and Evaluation Techniques Measures
Software Design Notation	Structural Descriptions Behavioral Descriptions
Software Design Strategies and Methods	General Strategies Function-Oriented Design Object-Oriented Design Data Structure-Centered Design Component-Based Design Others Methods
Software Design Tools	Sem SubTopics

Fonte: Elaboração própria (2022).

Outro mapeamento realizado foi entre as Competências derivadas da SBC RF-CC-2017 com as *Competencies* da ACM/IEEE CC-2020. Esse mapeamento considerou o curso de Ciência da Computação, em função da relação existente, Figura 2, entre o Curso e suas competências.

Tabela 7 – Mapeamento de Competências

SBC (Competências Derivadas)	ACM/IEEE (Competencies)
Resolver problemas usando ambientes de programação (C1.3 e C.2.1)	Create a simple application, together with help and documentation, that supports a graphical user interface for an enterprise and conduct a quantitative evaluation and report the results.
	Demonstrate program pieces (such as functions, classes, methods) that use generic or compound types, including for collections to write programs.
	Design and develop a user interface using a standard API and that incorporates visual and audio techniques used for a local organization
	Design and develop an interactive application for a local charity, applying a user-centered design cycle with related vocabulary, tools, and techniques that optimize usability and user experience.
	Design, implement, test, and debug an industry program that uses fundamental programming constructs including basic computation, simple and file I/O, standard conditional and iterative structures, the definition of functions, and parameter passing.
	Present the design and implementation of a class considering object-oriented encapsulation mechanisms (e.g., class hierarchies, interfaces, and private members).
	Write the correct input validation code for a cybersecurity company after classifying common input validation errors.
Reconhecer a importância do pensamento computacional no cotidiano e sua aplicação em circunstâncias apropriadas e em domínios diversos (C.1.5)	Analyze an industry problem to determine underlying recurrence relations and present the solution to professionals by using a variety of basic recurrence relations.
Conceber soluções computacionais a partir de decisões, visando o equilíbrio de todos os fatores envolvidos (C.1.6. e C.4.7. - CE-VI)	Analyze an industry problem to determine underlying recurrence relations and present the solution to professionals by using a variety of basic recurrence relations.
	Create a simple, formal mathematical model of a real-world situation and use that

	model in a simulation for a local technology company.
	Model a real-world problem using appropriate graphing strategies (e.g., trees, traversal methods for graphs and trees, spanning trees of a graph) and determine whether two graph approaches are isomorphic.
	Present to a peer group some practical examples of an appropriate set, function, or relation model, and interpret the associated operations and terminology in context.
Aplicar temas e princípios recorrentes, como abstração, complexidade, princípio de localidade de referência (caching), compartilhamento de recursos, segurança, concorrência, evolução de sistemas, entre outros, e reconhecer que esses temas e princípios são fundamentais à área de Ciência da Computação (C.1.7.)	Analyze an industry problem to determine underlying recurrence relations and present the solution to professionals by using a variety of basic recurrence relations.
	Apply consistent documentation and program style standards for a software engineering company that contribute to the readability and maintainability of software, conducting a personal and small-team code review on program component using a provided checklist.
	Contrast appropriate data models, including internal structures, for different types of data, and present an application to a group of professionals for the use of modeling concepts and notation of the relational data model.
	Demonstrate to a group of security professionals some ways to prevent a race condition from occurring and ways to handle exceptions.
	Design a scalable parallel algorithm for a computer firm by applying task-based decomposition or data-parallel decomposition.
	Present the analysis of a mobile industrial system and illustrate correct security vulnerabilities.
	Present the design and implementation of a class considering object-oriented encapsulation mechanisms (e.g., class hierarchies, interfaces, and private members).
	Use type-error messages, memory leaks, and dangling-pointer to debug a program for an engineering firm.

Tomar decisões e inovar, com base no conhecimento do funcionamento e das características técnicas de hardware e da infraestrutura de software dos sistemas de computação, consciente dos aspectos éticos, legais e dos impactos ambientais decorrentes (C.2.2.)	Analyze an industry problem to determine underlying recurrence relations and present the solution to professionals by using a variety of basic recurrence relations.
Avaliar criticamente projetos de sistemas de computação (C.2.3. e C.4.4. - CG-VIII)	Analyze and evaluate a user interface that considers the context of use, stakeholder needs, state-of-the-art response interaction times, design modalities taking into consideration universal access, inclusiveness, assistive technologies, and culture-sensitive design.
	Contrast appropriate data models, including internal structures, for different types of data, and present an application to a group of professionals for the use of modeling concepts and notation of the relational data model.
	Create and conduct a simple usability test to analyze and evaluate a user interface that considers the context of use taking into consideration universal access and culturally sensitive design.
Empregar metodologias que visem garantir critérios de qualidade ao longo de todas as etapas de desenvolvimento de uma solução computacional (C.2.8. e C.4.8. - CE-VII)	Apply consistent documentation and program style standards for a software engineering company that contribute to the readability and maintainability of software, conducting a personal and small-team code review on program component using a provided checklist.
Analisar quanto um sistema baseado em computadores atende os critérios definidos para seu uso corrente e futuro (adequabilidade) (C.2.9. e C.3.9. - CE-VIII)	Analyze an industry problem to determine underlying recurrence relations and present the solution to professionals by using a variety of basic recurrence relations.
	Analyze and evaluate a user interface that considers the context of use, stakeholder needs, state-of-the-art response interaction times, design modalities taking into consideration universal access, inclusiveness, assistive technologies, and culture-sensitive design.
Aplicar os princípios de interação humano-computador para avaliar e construir uma grande variedade de produtos incluindo interface do usuário, páginas WEB, sistemas multimídia e sistemas móveis (C.2.11.)	Create a simple application, together with help and documentation, that supports a graphical user interface for an enterprise and conduct a quantitative evaluation and report the results.

	Create and conduct a simple usability test to analyze and evaluate a user interface that considers the context of use taking into consideration universal access and culturally sensitive design.
	Design an interactive application, applying a user-centered design cycle with related tools and techniques (modes, navigation, visual design), to optimize usability and user experience within a corporate environment.
	Design and develop an interactive application for a local charity, applying a user-centered design cycle with related vocabulary, tools, and techniques that optimize usability and user experience.
	Design for a client a responsive web application utilizing a web framework and presentation technologies in support of a diverse online community.
	Write event handlers for a web developer for use in reactive systems such as GUIs.
Identificar e analisar requisitos e especificações para problemas específicos e planejar estratégias para suas soluções (C.3.8. - CE-IV)	Analyze an industry problem to determine underlying recurrence relations and present the solution to professionals by using a variety of basic recurrence relations.
	Analyze and evaluate a user interface that considers the context of use, stakeholder needs, state-of-the-art response interaction times, design modalities taking into consideration universal access, inclusiveness, assistive technologies, and culture-sensitive design.
	Design a simple parallel program for a corporation that manages shared resources through synchronization primitives and use tools to evaluate program performance.
	Design a simple sequential problem and a parallel version of the same problem using fundamental building blocks of logic design and use appropriate tools to evaluate the design for a commercial organization and evaluate both problem versions.
	Design and conduct a performance-oriented, pattern recognition experiment incorporating state machine descriptors and simple schedule algorithms for exploiting redundant information and data correction

	that is usable for a local engineering company and use appropriate tools to measure program performance.
	Present to a client the design of a simple software system using a modeling notation (such as UML), including an explanation of how the design incorporated system design principles.
	Produce a document that is helpful to others that addresses the effect of societal change due to technology.

No mapeamento de competências, Tabela 7, foi realizado o agrupamento de Competências Derivadas da SBC identificadas como equivalentes, como por exemplo, Resolver problemas usando ambientes de programação, que se apresenta associada ao Eixo Resolução de Problemas (C1.3) e Desenvolvimento de Sistemas (C2.1).

No mapeamento das competências, apesar de citar como um dos ativos da CC-2020 os *Elements of Foundation and Professional Knowledges*, não houve relacionamento com as competências derivadas da SBC, pois os *Elements of Foundation and Professional Knowledges* tratam-se de competências centradas no indivíduos, sendo independente do curso, como por exemplo: *Collaboration and Teamwork, Time Management e Written Communication*

.

5.1 Revisão por pares

Após a conclusão do mapeamento dos ativos e justificativas para as relações, o processo de revisão por pares foi iniciado. O processo de revisão por pares é realizado por especialistas na área em questão e que não fez parte da construção do mapeamento dos ativos, com o objetivo de realizar uma avaliação crítica da pesquisa e fornecer feedback de melhorias.

Neste trabalho, o processo de revisão por pares foi realizado por um pesquisador com experiência de 20 anos na área de Engenharia de Software e atuação em pesquisas sobre Educação em Computação.

Após o convite e o aceite para realizar a revisão, foi realizado dois encontros de forma remota para a apresentação dos objetivos da pesquisa e o resultado do mapeamento realizado.

Ao final deste processo de revisão por pares, o Revisor apresentou um conjunto de melhorias para o mapeamento, conforme:

- Remover as competências derivadas da SBC RF-CC-2017: C.3.2. Preparar e apresentar seus trabalhos e problemas técnicos e suas soluções para audiências diversas, em formatos apropriados (oral e escrito); C.3.5. Empreender e exercer liderança, coordenação e supervisão na sua área de atuação profissional (CG-XI); e C.3.6. Ser capaz de realizar trabalho cooperativo e entender os benefícios que este pode produzir (CG-XII). A justificativa para a remoção foi que essas competências são transversais ao curso e portanto não seria específicas para contemplar o *Software Design*.
- Remover os Knowledge Areas da ACM/IEEE CC2020: *Security Policy and Management* e *IS Management and Leadership*. A justificativa para remoção foi que essas Knowledge Areas, apesar de relacionadas com *Software Design*, não estão

claramente descritas no SWEBOK-v3.0 como elementos essenciais para o *Software Design*.

- Indicação de justificativas para o mapeamento entre os *SubTopics*, Conteúdos e *Knowledge Areas*. A justificativa foi a necessidade de compreender quais unidades de ensino e conteúdos subsidiaram a relação.
- Relacionar a que *Topic* pertence o *SubTopic* do SWEBOK-v3.0. A justificativa foi para facilitar a visualização da informação para o leitor.

No passo 6 da pesquisa, as contribuições oriundas do processo de revisão por pares foi incorporada ao mapeamento, gerando um versão final deste trabalho.

6 RESULTADOS

A presente seção tem como objetivo apresentar os resultados alcançados com a realização do mapeamento dos ativos.

O primeiro resultado alcançado no presente trabalho foi a correlação entre os ativos da ACM/IEEE CC2020 e da SBC RF-CC-2017 na visão das competências, Tabela 7, alinhadas ao *Software Design*. Outro resultado importante foi o mapeamento dos conteúdos, *SubTopics* e *Knowledge Areas*, conforme apresentado na Tabela 5, especificamente para o *Software Design*.

O desenvolvimento do mapeamento envolvendo os referenciais da SBC, da ACM/IEEE e o SWEBOK, especificamente para a area de conhecimento *Software Design* possibilitam a identificação de quais conteúdos específicos e competência necessárias para inclusão em um Currículo que trate deste área de conhecimento.

Por fim, o presente trabalho possibilitou a compreensão e a amplitude dos conteúdos necessários para o *Software Design* e como esses conteúdos estão representados nos principais *curriculum* de Ciência da Computação.

7 AMEAÇAS À VALIDADE

Esta seção apresenta uma análise dos riscos e estratégias de mitigação em relação ao mapeamento proposto neste trabalho. Para tal análise, as subseções foram divididas em validade interna, externa, de construção e de conclusão.

7.1 Validade interna

A validade interna refere-se especificamente a se um tratamento/condição experimental faz diferença ou não, e se há evidências suficientes para apoiar a afirmação (Travassos e do Amaral 2002).

O desenvolvimento do mapeamento dos ativos seguiu um conjunto de passos, descritos na seção 4. Além disso, um dos passos para conclusão do mapeamento foi o processo de avaliação por pares, antes da geração da versão final.

7.2 Validade externa

A validade externa refere-se à generalização dos resultados do tratamento/condição (Wohlin et al. 2012).

Para mitigação dos riscos à validade externa, foi desenvolvido uma metodologia para realização do mapeamento, além disso, foram identificados trabalhos relacionados que apresentam passos semelhantes no processo de construção do mapeamento.

7.3 Validade de construção

A validade de construção está preocupado com a relação entre teoria e observação (Wohlin et al. 2012).

Por tratar-se de um estudo teórico e que o mapeamento levou em consideração a interpretação do pesquisador autor, foi incluído neste processo a avaliação por pares, atividade esta que mitiga a ameaça à validade de construção. Além disso, desenvolver a metodologia para realização do mapeamento é um indicador para minimizar os riscos que ameaçam a validade de construção.

7.4 Validade de conclusão

A validade de conclusão é relacionada à habilidade de chegar a uma conclusão correta na relação entre tratamento e resultado (Travassos e do Amaral 2002).

Dentre os trabalhos relacionados, seção 3, há estudos alinhados que colaboram para minimizar a ameaça à validade de conclusão, pois conseguiram realizar o mapeamento entre os ativos da ACM/IEEE e SBC de outras versões. Pode-se adicionar como medida para mitigar os riscos dessa validade, a inclusão do processo de revisão por pares, como mecanismo de validação do mapeamento realizado.

8 CONCLUSÃO

O presente trabalho teve como objetivo realizar o mapeamento dos principais ativos dos *curriculum* da ACM/IEEE, da SBC e do SWEBOK, especificamente para área de conhecimento *Software Design*.

Na seção 5 foi apresentado a conceituação dos ativos e os relacionamento entre Topics, Conteúdo e Knowledge Areas, e o relacionamento enter e as Competências Derivadas e as *Competencies*. Além disso, para o resultado gerado houve o processo de revisão por pares que sinalizou melhorias para a versão final, conforme seção 5.1 .

Como resultados, destacam-se a correlação entre os ativos da CC2020 e RF-CC-2017 na visão das competências relacionadas ao *Software Design*. Além disso, os Topics/*SubTopics* do SWEBOK para o *Software Design* foram mapeados com os ativos da CC2020 e RF-CC-2017.

Como limitações, este trabalho abordou somente a área de conhecimento *Software Design*, de maneira isolada, sem a integração com outras áreas, tais como Requisitos e Construção do Software.

Por fim, como trabalhos futuros, pretende-se propor um plano de ensino de contemple os conteúdos e competências identificadas no mapeamento para o ensino do *Software Design*. Além disso, avaliar como os Projetos Pedagógico de Cursos de graduação em Ciência da Computação em Universidades brasileiras estão em sinergia com os conteúdos e competências para área de conhecimento apresentada neste trabalho.

REFERÊNCIAS

Aniche, M., Yoder, J., e Kon, F. (2019). Current challenges in practical object-oriented software design. In *2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*, páginas 113–116.

Bourque, P. and Fairley, R. E., editors (2014). *SWEBOK: Guide to the Software Engineering Body of Knowledge*. IEEE Computer Society, Los Alamitos, CA, version 3.0 edition.

Calsavara, A., Serra, A. P. G., de Assis Zampirolli, F., de Carvalho, L. S. G., Jonathan, M., e Correia, R. C. M. (2018). Método baseado nos referenciais de formação da SBC para reestruturação de descritivos de disciplinas de ciência da computação em conformidade com as DCN de 2016. In *Anais do XXVI Workshop sobre Educação em Computação*. SBC.

de Pádua Paula Filho, W. (2019). *Engenharia de Software: produtos*. LTC, Rio de Janeiro.

Force, C. T. (2020). *Computing Curricula 2020: Paradigms for Global Computing Education*. Association for Computing Machinery, New York, NY, USA.

Forti, S., Breitenbucher, U., e Soldani, J. (2022). Trending topics in software engineering. *SIGSOFT Softw. Eng. Notes*, 47(3):20–21.

Joint Task Force on Computing Curricula, A. f. C. M. A. and Society, I. C. (2013). *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. Association for Computing Machinery, New York, NY, USA.
MEC (2016). Resolução nº 05, de 16 de novembro de 2016, diretrizes curriculares nacionais para os cursos de graduação em computação.

Oliveira, S. R. B. e Elgrably, I. S. (2019). Uma proposta de ensino e aplicação de testes com foco em métodos ágeis feita através de um mapeamento de ativos. volume 1. TECSI.

Pressman, R. and Maxim, B. (2016). *Engenharia de Software: Uma abordagem profissional*. Mc Graw Hill Education, Porto Alegre.

Quaresma, J. A. e Oliveira, S. R. B. (2020). A mapping of the assets included in the reference standards of rf-sbc, cs-curricula acm / iee 2013 and swebok regarding the knowledge area about software process.

Travassos, G. H. e do Amaral, D. G. E. A. G. (2002). Relatório técnico - introdução à engenharia de software experimental.

Wohlin, C., Runeson, P., Host, M., Ohlsson, M. C., Regnell, B., e Wesslén, A. (2012). *Experimentation in software engineering*. Springer Science & Business Media.

Zorzo, A. F., Nunes, D., Matos, E. S., Steinmacher, I., Leite, J. C., Araujo, R., Correia, R. C. M., e Martins, S. (2017). *Referenciais de Formação para os Cursos de Graduação em Computação*.