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

Alcides Calsavara[1], Ana Paula Gonçalves Serra[2], Francisco de Assis Zampirolli[3], Leandro Silva Galvão de Carvalho[4], Miguel Jonathan[5], Ronaldo Celso Messias Correia[6]

[1]Escola Politécnica, Pontifícia Universidade Católica do Paraná (PUCPR) – Curitiba – PR; [2]Faculdade de Tecnologia e Ciências Exatas, Universidade São Judas Tadeu (USJT) – São Paulo – SP; [3]Centro de Matemática, Computação e Cognição, Universidade Federal do ABC (UFABC) – Santo André – SP; [4]Instituto de Computação, Universidade Federal do Amazonas (UFAM) – Manaus – AM; [5]Departamento de Sistemas e Computação, Universidade Federal do Rio de Janeiro (UFRJ) – Rio de Janeiro – RJ; [6]Departamento de Matemática e Computação, Universidade Estadual Paulista (UNESP) – Presidente Prudente – SP

alcides.calsavara@pucpr.br, prof.anapaula@usjt.br, fzampirolli@ufabc.edu.br, galvao@icomp.ufam.edu.br, jonathan@dcc.ufrj.br, ronaldo@fct.unesp.br

Resumo. *Os Referenciais de Formação para o Bacharelado em Ciência da Computação (RF-CC-17) da SBC organizam as competências e habilidades descritas nas Diretrizes Curriculares Nacionais para os Cursos de Graduação em Computação (DCN16) em eixos de formação e também indicam um conjunto de conteúdos associados. Este ensaio apresenta um método, baseado nos RF-CC-17, para elaborar um Mapeamento de Conformidade e Mobilização (MCM), como parte do descritivo de uma disciplina. Como exemplo, o método é aplicado na elaboração de dois descritivos distintos de introdução à programação, um baseado no paradigma imperativo e outro, orientado a objetos. Por fim, discute as vantagens de se usar o método para auxiliar na revisão de projetos pedagógicos de cursos vigentes tal que fiquem em conformidade com as DCN16.*

Abstract. *The SBC Computer Science Curricula 2017 (RF-CC-17) arranges the competences and skills described in the National Curricular Guidelines for Computing Undergraduate Courses (DCN16) into axial competences. Also, it associates content topics to each axial competence. This essay presents a method to elaborate a Mapping of Compliance and Mobilization (MCM) as a part of a discipline program, based on RF-CC-17. As an example, the method is applied in the elaboration of two distinct programs of introduction to programming, one based on the imperative paradigm and another, object-oriented. Finally, it discusses the advantages of using the method to review pedagogical projects of current courses in compliance to the DCN16.*

1. Introdução

Em novembro de 2016, o Ministério da Educação homologou as Diretrizes Curriculares Nacionais para os Cursos de Graduação em Computação (MEC 2016), simplesmente chamadas de **DCN16**, que estabelecem as normas legais para a organização e o

funcionamento de centenas de cursos brasileiros de graduação da área de computação. As DCN16, em seu Artigo 10, estipulam um prazo de dois anos, a partir da sua publicação, para que cada Instituição de Educação Superior (IES) implante as normas nelas estabelecidas aos alunos ingressantes. Ou seja, todo curso da área de computação deve, obrigatoriamente, revisar o seu Projeto Pedagógico de Curso (PPC) tal que fique em conformidade com as DCN16 até, o mais tardar, o ano letivo de 2019.

As DCN16 elencam o *perfil do egresso* e as *competências* e *habilidades* que um estudante deve adquirir durante a graduação; porém, não especificam quaisquer conteúdos básicos ou tecnológicos. Assim, as DCN16 seguem a tendência pedagógica de definir o que os egressos de um curso devem saber fazer e como devem se comportar na sua vida profissional, ao mesmo tempo que deixa cada curso livre para definir os componentes curriculares (incluindo as disciplinas do curso) e os correspondentes conteúdos, contanto (de acordo com o Artigo 6° das DCN16) que haja consistência com o perfil, as competências e as habilidades especificadas para o egresso.

Atingir a conformidade do PPC com as DCN16 representa tanto uma oportunidade como um desafio para o corpo docente de um curso, em especial para a sua coordenação e o seu Núcleo Docente Estruturante (NDE). Enquanto a oportunidade consiste em poder construir uma matriz curricular sem as limitações e as possíveis distorções decorrentes da rigidez do modelo tradicional baseado em conteúdo, o desafio reside justamente na mudança de paradigma, pois o modelo baseado em competências para construção de matrizes curriculares é, ainda, pouco explorado pelas IES brasileiras.

Por outro lado, a Sociedade Brasileira de Computação (SBC) participou na homologação das DCN16, principalmente na elaboração da sua proposta anos antes (MEC 2012). Por isso, há anos vem promovendo estudos e discussões entre acadêmicos para o avanço do domínio do modelo baseado em competências para ensino e aprendizagem de computação. Em especial, tem organizado grupos de trabalho específicos para cada tipo de curso (Bacharelado em Ciência da Computação, Engenharia de Computação, Sistemas de Informação e Engenharia de Software, bem como Licenciatura em Computação) a fim de revisar os chamados *currículos de referência* segundo esse novo modelo. Como resultado desse longo trabalho, a SBC publicou, em outubro de 2017, os chamados *referenciais de formação* para cursos de graduação em computação (Zorzo et al. 2017). A partir de então, os referenciais de formação passaram a ser a recomendação oficial da SBC para a elaboração do PPC de cada IES, em substituição aos currículos de referência.

Coerentemente, os referenciais de formação baseiam-se fortemente nas DCN16 para fazer recomendações na elaboração de um PPC segundo o paradigma de competências. Incorpora, ainda, contribuições de outros documentos importantes, tais como os antigos currículos de referência da própria SBC (SBC 1999, SBC 2003 e SBC 2005) e o currículo de referência elaborado pela comunidade internacional da computação (ACM/IEEE 2013), além de contribuições advindas da experiência de muitas IES brasileiras. Assim, comparado com as DCN16, os referenciais de formação apresentam uma visão mais holística da área de computação, logo mais adequada para a elaboração do PPC, sem qualquer prejuízo à sua conformidade com as DCN16.

Muito embora os referenciais de formação representem um grande avanço promovido pela SBC e, de fato, facilitem a elaboração do PPC segundo o paradigma de competências, ainda pode-se considerar a criação de instrumentos adicionais para sua maior efetividade. Mais especificamente, os referenciais de formação do Bacharelado em Ciência da Computação (Calsavara et al., 2017), aqui referenciados como RF-CC-17,

associam conteúdos curriculares a cada competência (ou habilidade) presente nas DCN16 de acordo com o contexto em que é requerida, caracterizado como um *eixo de formação*. Estes são definidos por uma "macro-competência" do egresso. Cada competência das DCN16, chamada de *competência derivada*, pode ser requerida em mais de um eixo de formação. Além disso, os conteúdos associados estão, em sua maioria, presentes nos antigos currículos de referência sob o título de *matérias*, sendo, portanto, de fácil compreensão pela comunidade acadêmica. Entretanto, cabe a cada curso definir uma estratégia de como usar toda essa informação na elaboração do PPC, em especial, na descrição dos componentes curriculares do curso.

Neste trabalho, classificado como ensaio, propõe-se um método para auxiliar a elaboração da parte do PPC correspondente à descrição das disciplinas de um curso, a partir dos RF-CC-17. Adota-se aqui o termo *descritivo de disciplina* para designar o texto presente em um PPC que descreve cada disciplina do curso. Naturalmente, além de diferentes designações, o descritivo de disciplina pode assumir muitas formas, dependendo de cada IES, mas invariavelmente inclui uma *ementa*, que tipicamente constitui-se por uma lista de tópicos de estudo.

Devido à heterogeneidade de formas e exigências das IES para o descritivo de disciplina, o produto obtido com a aplicação do método proposto neste trabalho não constitui o descritivo da disciplina propriamente dito. Tampouco, constitui o plano de ensino (ou plano de estudos), que muitas IES desvinculam do descritivo de disciplina para dar mais perenidade ao PPC e, ao mesmo tempo, mais flexibilidade na operacionalização das disciplinas (pode haver um plano de ensino específico para cada turma da disciplina, por exemplo, com detalhamento de cronograma e instrumentos de avaliação). Constitui, sim, um instrumento para auxiliar a coordenação e o NDE de um curso a elaborarem estes dois documentos: descritivo de disciplina e plano de ensino. Tal instrumento, designado *Mapeamento de Conformidade e Mobilização* (MCM), visa garantir que a disciplina está em conformidade com os RF-CC-17 – logo, com as DCN16 – e, ainda, descreve o que o estudante deve realizar na disciplina para adquirir cada uma das competências da DCN vinculadas à disciplina, isto é, que recursos, incluindo conteúdos, o estudante deve mobilizar para adquirir cada competência.

Considerando que os RF-CC-17 são recentes, não foram encontrados artigos relacionados ao seu uso. No entanto, existem esforços em escrever os PPCs de Ciência da Computação seguindo o modelo de competências, dentre eles Rezende, et al. (2004). Por outro lado, o currículo ACM/IEEE (2013), em seu Apêndice C, ilustra o uso do currículo por meio de 83 exemplos de disciplinas ministradas em diversas universidades, a maioria dos EUA. Os exemplos são descritos seguindo um *template* apresentado no próprio Apêndice C. Não é apresentado um método próprio para preenchimento do *template*, mas apenas uma explicação do significado de cada campo.

Este trabalho está organizado como se segue. A Seção 2 descreve o método proposto e o instrumento gerado com a sua aplicação: o MCM. A Seção 3 ilustra a aplicação do método para a disciplina de introdução à programação de duas IES distintas, uma que usa o paradigma de programação imperativo e a outra que usa a orientação a objetos, mostrando que o método proposto contempla as especificidades de cada contexto de aplicação. A Seção 4 faz algumas considerações sobre os benefícios do método proposto. Finalmente, a Seção 5 apresenta conclusões sobre o trabalho.

2. Método

O método parte do princípio que muitas coordenações de curso enfrentam o desafio de reformular um PPC já em vigor em sua instituição, e não construir um PPC inteiramente novo. Portanto, o método adota um procedimento incremental, partindo de uma abordagem baseada em conteúdos para chegar a uma proposta baseada em competências, tal como preconizada pelas DCN16. Em geral, o descritivo de cada disciplina contempla os seguintes campos: Ementa, Objetivos Gerais, Objetivos Específicos, Referências Bibliográficas Básicas e Complementares. Esta seção apresenta um método para se elaborar uma estrutura adicional, o *Mapeamento de Conformidade e Mobilização* (MCM), com o objetivo de auxiliar na revisão do descritivo segundo o paradigma de competências.

2.1. Descrição do Método

O método para elaborar o MCM de uma disciplina consiste nos seguintes passos:

Passo 1: *Seleção dos conteúdos dos RF-CC-17 pertinentes à disciplina*. Essa seleção pode se basear na descrição original da disciplina, normalmente formulada na abordagem conteudista. Por exemplo, para uma disciplina de introdução à programação, os seguintes conteúdos dos RF-CC-17 podem ser selecionados: algoritmos, estruturas de dados, técnicas de programação e programação imperativa. Além deles, podem ser incluídos conteúdos relacionados a competências transversais, tais como língua inglesa e trabalho em equipe.

Passo 2: *Seleção das competências derivadas*. Devem ser selecionadas as competências derivadas entre aquelas dos RF-CC-17 que possuem vínculo com os conteúdos selecionados no Passo 1. Essa seleção deve considerar a disciplina no contexto do curso e pode se basear nos objetivos da disciplina constantes na sua descrição original. Como podem existir competências derivadas com o mesmo nome em diferentes eixos de formação, mas com semânticas específicas para cada eixo, a seleção deve observar se a semântica no eixo é pertinente à disciplina.

Passo 3: *Contribuição da disciplina*. Deve-se explicar como a disciplina contribui para construir no estudante cada competência derivada selecionada no Passo 2. Essa explicação deve focar nas atividades que o estudante desenvolve na disciplina, incluindo os instrumentos e métodos utilizados, e quais são os objetivos específicos dessas atividades que sejam relacionados com a competência derivada. Também deve deixar claro qual o nível cognitivo – criar, aplicar, etc. (Ferraz e Belhot, 2010) – que se pretende desenvolver no estudante. Os RF-CC-17 já recomendam um nível cognitivo para cada competência derivada, mas cada curso pode redefinir esse nível de acordo com os objetivos da disciplina. A contribuição da disciplina para desenvolver uma competência derivada depende de muitos fatores de contexto, tais como perfil do corpo docente, perfil de estudantes, recursos disponíveis na IES, carga horária da disciplina e metodologia de ensino-aprendizagem. O mapeamento dessas relações está fora do escopo deste trabalho.

2.2. Estrutura do MCM

O MCM obtido com a aplicação do método proposto na seção anterior pode ser estruturado conforme mostra a Tabela 1.

Tabela 1. Estrutura do MCM.

Eixo	Competência derivada	Conteúdos	Contribuição da disciplina

Na coluna **Eixo**, são relacionados, em diferentes linhas, alguns dos sete Eixos de Formação contidos nos RF-CC-17, por meio dos quais a disciplina terá o papel de estimular algumas das 25 competências derivadas oriundas das DCN16. Por exemplo, uma primeira disciplina de programação em um curso de Ciência da Computação deve estimular o estudante a *Resolver Problemas* (Eixo 1 dos RF-CC-17). Como **Competência derivada**, podemos ter *Resolver problemas usando ambientes de programação* (terceira competência geral das DCN16). Para cada competência derivada, está associado um nível da Taxonomia de Bloom Revisada (Ferraz e Belhot, 2010). Neste caso, podemos ter *Criar*, indicando que o estudante deverá criar programas simples usando ambientes de programação, como o *Netbeans* ou *Eclipse*, e uma linguagem de programação, como C, Java ou Python. Na coluna **Conteúdos**, serão incluídos alguns conteúdos, como Algoritmos, Técnicas de Programação ou Estruturas de Dados. A maior parte deles foi descrita nos antigos Currículos de Referência (SBC, 1999, 2003 e 2005). Finalmente, a coluna **Contribuição da disciplina** descreve como a disciplina contribui para construir no estudante a competência derivada.

3. Aplicação do método proposto

Para ilustrar a aplicação do método proposto, realizou-se um estudo de caso a partir do descritivo tradicional da disciplina de introdução à programação oferecida a estudantes de primeiro período por duas IES brasileiras, neste trabalho denominadas de α e β , para manter o seu anonimato. As instituições adotam paradigmas de programação distintos nessa disciplina: imperativo e orientado a objetos. Não se pretende aqui definir um descritivo padrão para disciplinas de programação, mas somente ilustrar a aplicação do método especificamente para as duas IES consideradas.

3.1. Abordagem imperativa

Esta seção ilustra a aplicação do método para uma disciplina denominada *Introdução à Programação Imperativa* de uma IES α cujos cursos de computação e de engenharia compartilham essa disciplina em suas matrizes curriculares.

A IES α oferta dois tipos de turmas para a disciplina: *mista* e *exclusiva*. Em uma turma mista, pode haver estudantes de quaisquer cursos, enquanto que em uma turma exclusiva há estudantes de um único curso. Assim, por exemplo, pode haver uma turma composta exclusivamente de estudantes de Ciência da Computação, outra composta exclusivamente por estudantes de Engenharia Mecânica e outra mista, composta por estudantes de Ciência da Computação, de Engenharia Mecânica e de Engenharia Civil.

Independentemente do tipo de turma, o descritivo da disciplina é idêntico em todos os PPCs dos cursos, mas deve haver um plano de ensino específico para cada turma. Por isso, o descritivo da disciplina deve ser genérico o suficiente para se aplicar a todos os estudantes, independentemente do curso, mas também deve ser suficientemente preciso e detalhado a fim de permitir a derivação de um plano de ensino que se ajuste ao perfil dos estudantes de cada turma. Por exemplo, o plano de ensino para a turma exclusiva da Ciência da Computação pode estabelecer que a linguagem de programação C deve ser usada na disciplina, enquanto que o plano de ensino de uma turma mista pode estabelecer que a linguagem Python deve ser usada.

Além disso, por uma orientação geral da IES α, as disciplinas devem, sempre que aplicável, promover a multidisciplinaridade. Assim, embora a disciplina seja da área de computação, o seu descritivo estabelece uma forte interação com outras áreas, mais

especificamente por meio da proposição de atividades para os estudantes que envolvam a resolução de problemas de outras áreas, em especial dos diversos tipos de engenharia. Essa multidisciplinaridade é exigida não apenas para as turmas mistas, onde naturalmente já ocorre, mas também para as turmas exclusivas. Por isso, o plano de ensino de uma turma exclusiva deve prever atividades que exijam interação com estudantes (e, possivelmente, professores) de outros cursos.

A Tabela 2 mostra a parte do descritivo original da disciplina que contém informações úteis para a aplicação do método. O MCM obtido para a disciplina é composto pelas informações das Tabelas 3 e 4. A partir das informações da Tabela 1, no Passo 1, selecionam-se os seguintes conteúdos dentre os listados no RF-CC-17: Algoritmos, Estruturas de Dados, Técnicas de Programação e Programação Imperativa. Com essa lista de conteúdos, no Passo 2, são selecionadas seis competências derivadas (e correspondentes eixos) que aparecem vinculadas a esses conteúdos no RF-CC-17, conforme mostra a Tabela 3. Algumas competências derivadas não foram selecionadas, mesmo estando vinculadas aos conteúdos, pois não se adequam ao contexto da disciplina. Por fim, a Tabela 4 contém a contribuição da disciplina no desenvolvimento de cada competência derivada, o que, inclusive, justifica a própria seleção feita. Observa-se que o texto se concentra em explicar o que o estudante realiza na disciplina, isto é, que recursos, incluindo tópicos de estudo, mobiliza para adquirir a competência.

Tabela 2. Parte do descritivo original da disciplina Introdução à Programação Imperativa da IES α (auxilia em todos os passos do método).

NOME DISCIPLINA: Introdução à Programação Imperativa
Ementa: Conceitos de algoritmos e programação estruturada. Tipos de dados, constantes, variáveis e atribuição. Pseudolinguagem e fluxogramas. Estruturas de seleção. Estruturas de repetição. Vetores. Matrizes. Funções. Entrada e saída de dados. Leitura e escrita em arquivos textos. Depuração de programas. Melhores práticas de programação.
Objetivo Geral: Criar algoritmos básicos para solucionar problemas de natureza técnico-científica e os implemente em uma linguagem de programação.
Metodologia: Apresentação fundamentos sobre manipulação e tratamento de dados e informações, utilizando explicações e experimentações dos conceitos de lógica de programação.

Tabela 3. Eixos e competências derivadas selecionadas para a disciplina Introdução à Programação Imperativa da IES α e conteúdos associados (Passo 2 do método).

Eixo	**Competência derivada**	**Conteúdos**
Resolução de Problemas	C.1.1. *Identificar problemas que tenham solução algorítmica* [Avaliar]	Algoritmos
	C.1.3. *Resolver problemas usando ambientes de programação* [Criar]	Algoritmos; Técnicas de Programação; Estruturas de Dados
	C.1.5. *Reconhecer a importância do pensamento computacional no cotidiano e sua aplicação em circunstâncias apropriadas e em domínios diversos* [Aplicar]	Algoritmos; Estruturas de Dados
Desenvolvimento de Sistemas	C.2.1. *Resolver problemas usando ambientes de programação* [Criar]	Algoritmos; Programação Imperativa
Desenvolvimento de Projetos	C.3.7. *Reconhecer a importância do pensamento computacional no cotidiano e sua aplicação em circunstâncias apropriadas e em domínios diversos* [Aplicar]	Algoritmos; Estruturas de Dados
Aprendizado Contínuo e Autônomo	C.6.6. *Compreender os fatos essenciais, os conceitos, os princípios e as teorias relacionadas à Ciência da Computação para o desenvolvimento de software e hardware e suas aplicações* [Avaliar]	Algoritmos

Tabela 4. Contribuição da disciplina Introdução à Programação Imperativa da IES α para cada Competência Derivada (CD) selecionada (Passo 3 do método).

CD	Contribuição da disciplina
C.1.1	O estudante experimenta a aplicação de alguns algoritmos simples em problemas de domínios diversos, incluindo algoritmos de ordenação e busca em conjuntos de dados armazenados em memória e em arquivo. Com essa experiência, o estudante passa a compreender o potencial da computação na resolução de problemas que envolvam o tratamento de grandes volumes de dados e pode avaliar a possibilidade de aplicação desses algoritmos em outros contextos. Por exemplo, dado um algoritmo simples de ordenação, o estudante deve ser capaz de simular (em papel) a sua execução. Além disso, o estudante deve ser capaz de escrever (em pseudocódigo e fluxograma) a solução de problemas simples, como ordenar uma lista de estudantes pelo conceito final de uma disciplina, ou exibir os nomes dos estudantes em ordem alfabética.
C.1.3	O estudante resolve problemas simples que envolvam o tratamento de dados numéricos e textuais (*strings*) por meio da implementação de programas em linguagem imperativa, com base em algoritmos que empregam comandos de atribuição, desvio, seleção e repetição, variáveis e constantes de tipos primitivos (numéricos, textuais e booleanos) e estruturas de dados de baixa complexidade (vetores e matrizes). Os algoritmos podem ser selecionados da literatura ou desenhados especificamente para os problemas propostos. Com isso, o estudante passa a ter o domínio básico de uma linguagem e de um ambiente de programação, que constituem a ferramenta prática mais fundamental da computação para a resolução de problemas do mundo real.
C1.5	Idem a C.1.1, com ênfase na aplicação da computação em diversas áreas do conhecimento, ou seja, o estudante é apresentado a problemas típicos de outras áreas, tal que compreenda o essencial do processo de transposição de conhecimentos da computação para outras áreas. Por exemplo, o estudante deve ser capaz de transformar uma imagem colorida na correspondente imagem em preto e branco, ambas representadas por matrizes. Este é um primeiro passo para análise de imagens mais complexas, como imagens de satélite, microscópicas do tecido humano, do espaço, etc. Esta análise também pode ser útil em robótica, para veículos autônomos, por exemplo. O estudante deve ser capaz de desenvolver algoritmos simples para automação de casas, fábricas, escritórios, etc, quando os problemas possam ser resolvidos usando apenas a lógica de programação.
C.2.1	O estudante constrói programas em linguagem imperativa com base em algoritmos especificados por meio de pseudocódigo, fluxograma ou formulação matemática. A complexidade dos programas propostos aumenta gradativamente ao longo do período letivo, tal que, a partir de certo ponto, os programas sejam, necessariamente, estruturados em funções e procedimentos. Finalmente, os programas são validados pelo estudante, inicialmente seguindo um plano de testes proposto e, posteriormente, a partir de um plano de testes elaborado pelo próprio estudante. Dessa forma, o estudante experimenta três etapas fundamentais do ciclo de desenvolvimento de sistemas de software: a especificação, a implementação e a validação.
C.3.7	Idem a C.1.5, incluindo o desenvolvimento de um projeto em outra área do conhecimento, a fim de permitir que estudante vivencie a prática de transposição de conhecimentos da computação para outra área.
C.6.6	O estudante realiza estudos sobre algoritmos simples de ordenação e busca por meio de consultas a livros e autores clássicos dessa área de estudo da computação, além de analisar artigos científicos sobre o assunto, em especial com avaliações e propostas de melhorias de algoritmos. Assim, o estudante toma conhecimento dos meios de publicação de conhecimentos científicos que o auxiliarão no restante do curso e, futuramente, na sua carreira profissional.

3.2. Abordagem orientada a objetos

Esta seção ilustra a aplicação do método para uma disciplina denominada *Programação Orientada a Objetos* de uma IES β na qual essa disciplina é oferecida aos estudantes dos cursos de Ciência da Computação e Sistemas de Informação, sendo adotado um único descritivo e um único plano de ensino para ambos os cursos. Uma mesma turma pode ser composta por somente estudantes de Ciência da Computação, ou por estudantes de Ciência da Computação e Sistemas de Informação, em uma configuração mista. Para ambos os cursos, essa é uma disciplina de primeiro semestre e a primeira disciplina de programação. A IES β usa a orientação objetos como primeiro paradigma de programação, não havendo nenhuma outra disciplina anterior de abordagem estruturada.

Parte do descritivo original da disciplina é mostrada na Tabela 5. Aplicando-se o Passo 1, são selecionados os seguintes conteúdos: Algoritmos; Técnicas de Programação; e Programação Orientada a Objetos. A Tabela 6 apresenta o resultado do Passo 2,

selecionando as competências derivadas. A Tabela 7 completa o MCM, descrevendo como cada competência derivada contribui para a formação do estudante (aplicação do Passo 3).

Tabela 5. Parte do descritivo original da disciplina Programação Orientada a Objetos da IES β (auxilia em todos os passos do método).

NOME DISCIPLINA: Programação Orientada a Objetos
Ementa: Conceitos básicos de orientação a objetos (classe, objeto, atributos, métodos, encapsulamento). Estruturas básicas de programação orientada a objetos. Tipos de dados, constantes, variáveis e atribuição. Estrutura de seleção. Estruturas de repetição. Vetores. Matrizes. Prática de desenvolvimento de algoritmos e programação orientada a objetos. Interação entre classes por relacionamento de associação. Classe de interface gráfica. Classe de negócio. Classes persistentes.
Objetivo Geral: Criar algoritmos orientados a objetos para exercitar a abstração e a capacidade de resolução de problemas computacionais com soluções de programas orientados a objetos.
Metodologia: Apresentação dos conceitos de orientação a objetos, por meio de explicação, experimentação dos conceitos, utilizando interface gráfica e interação com várias classes, por meio de técnicas colaborativas de aprendizado.

Tabela 6. Eixos e competências derivadas selecionadas para a disciplina Programação Orientada a Objetos da IES β e conteúdos associados (Passo 2 do método).

Eixo	**Competência derivada**	**Conteúdos**
Resolução de Problemas	C.1.1. Identificar problemas que tenham solução algorítmica [Avaliar]	Algoritmos
	C.1.3. Resolver problemas usando ambientes de programação [Criar]	Algoritmos; Técnicas de Programação
Desenvolvimento de Sistemas	C.2.1. Resolver problemas usando ambientes de programação [Criar]	Algoritmos; Programação Orientada a Objetos
Aprendizado Contínuo e Autônomo	C.6.6. Compreender os fatos essenciais, os conceitos, os princípios e as teorias relacionadas à Ciência da Computação para o desenvolvimento de software e hardware e suas aplicações [Avaliar]	Algoritmos

4. Discussão sobre o método proposto

De um lado, as DCN16 relacionam 25 competências e habilidades mínimas que os cursos de Ciência da Computação devem desenvolver em seus egressos. De outro, o parecer CNE/CES Nº 136/2012 (MEC, 2012), que acompanha as diretrizes, elenca um vasto conjunto de conteúdos curriculares a serem ministrados pelos cursos. Essas duas listagens são apresentadas de forma desconexa, em documentos diferentes. Assim, para as coordenações de curso e NDEs que desejam atualizar seus PPCs em conformidade com as DCN16, a principal vantagem de utilizar os RF-CC-17 e o método aqui proposto é explicitar a ligação entre competências e conteúdos curriculares.

Essa ligação pode ser realizada de forma diferente em cada IES. Por exemplo, a IES α decidiu desenvolver a competência C.1.5, diferentemente da IES β. Isso não implica que a IES β esteja em falta com as DCN, desde que ela desenvolva a competência derivada em outra disciplina da matriz curricular.

Além disso, a listagem de competências e habilidades nas DCN16, divididas entre gerais e específicas, apresenta sobreposições. Por outro lado, os RF-CC-17 agrupam essas competências e habilidades em sete eixos de formação. Dessa forma, as coordenações de curso e NDEs, ao usarem o método de geração do MCM, contam com um roteiro e um estudo de caso para facilitar a compreensão das diretrizes, a fim de elaborar um descritivo de disciplina orientado a competências. De qualquer maneira, o método proposto não dispensa a leitura e o conhecimento das DCN16 e dos RF-CC-17.

Tabela 7. Contribuição da disciplina Programação Orientada a Objetos da IES β para cada Competência Derivada (CD) selecionada (Passo 3 do método).

CD	Contribuição da disciplina
C.1.1	O estudante modela classes que simulam objetos do mundo real, e identifica métodos que podem ser implementados por meio de algoritmos simples. O estudante desenvolve a capacidade de abstração que o habilita a compreender como mapear problemas do mundo real no computador e a construir soluções orientadas a objetos computacionais eficazes.
C.1.3	O estudante implementa programas usando uma linguagem orientada a objetos em um ambiente de programação simples, sem muitos recursos de *plugin* e/ou interface gráfica. Na implementação de cada classe, o estudante emprega conceito de encapsulamento, técnicas de representação de dados (variáveis, constantes, tipos primitivos e estruturas simples, como vetores e matrizes) e comandos de atribuição, desvio, seleção e repetição. Com isso, o estudante passa a ter domínio básico de técnicas de programação e de um ambiente de programação, que é a ferramenta prática mais fundamental da computação para a resolução de problemas do mundo real.
C.2.1	O estudante constrói programas em linguagem orientada a objetos com base em um problema computacional, implementando classes de interface gráfica, classes de negócio e acesso a banco de dados, além de realizar testes unitários e testes de validação e verificação. Dessa forma, o estudante experimenta quatro etapas fundamentais do ciclo de desenvolvimento de sistemas de software, a saber: a especificação (problema a ser resolvido), projeto (identificação das classes e suas responsabilidades utilizando o princípio de arquitetura em camadas), implementação, e testes unitários e de verificação e validação.
C.6.6	O estudante realiza estudos sobre programação orientada a objetos por meio de consultas a livros e autores clássicos dessa área de estudo da computação, além de analisar artigos científicos sobre o assunto, em especial sobre boas práticas de programação orientação a objetos e plataformas que podem ser aplicadas. Assim, o estudante toma conhecimento dos meios de publicação de conhecimentos científicos que o auxiliarão no restante do curso e, futuramente, na sua carreira profissional. A metodologia de ensino apoia o aprendizado contínuo, por meio de aulas práticas com técnicas colaborativas de aprendizado professor-estudante e estudantes-estudantes. O objetivo é ensinar os estudantes a escreverem código de programação orientado a objetos de uma forma mais prática, dinâmica e colaborativa, com maior facilidade e qualidade, incentivando o entendimento de conceitos e técnicas de orientação a objetos, o gosto pela programação e exercitando o trabalho em equipe. O estudante participa ativamente da escrita do código, todos os alunos interagem e compartilham suas dúvidas, para isso, é utilizada uma técnica conhecida como *Coding Dojo*.

5. Conclusão

O Mapeamento de Conformidade e Mobilização (MCM), obtido com a aplicação do método proposto, constitui uma base confiável para elaborar os descritivos de disciplinas de um PPC em conformidade com as DCN16. Isso foi verificado por meio da aplicação do método para a disciplina de introdução à programação em duas IES brasileiras. O MCM obtido para cada IES seleciona algumas das competências especificadas nos RF-CC-17 pertinentes à disciplina e descreve como os recursos vinculados, incluindo os conteúdos ministrados, são mobilizados pelo estudante para adquirir cada competência.

Apesar de o estudo de caso contemplar somente a disciplina de introdução à programação, o método proposto é genérico e pode ser utilizado para reestruturar as demais disciplinas de PPCs vigentes. Além disso, o MCM auxilia o docente responsável pela disciplina a planejar, organizar e definir as atividades que serão desenvolvidas pelo discente, isto é, a elaborar o plano de ensino.

O método proposto agiliza a reestruturação dos PPCs, principalmente para coordenações e NDEs pouco experientes com a abordagem baseada em competências. Dessa forma, contribui para que as IES cumpram o prazo legal estabelecido pelas DCN16, ou seja, para os estudantes ingressantes no ano letivo de 2019.

Os MCMs produzidos neste ensaio estão disponíveis em goo.gl/YgHi7h. Como trabalho futuro, convidamos a comunidade a utilizar o método proposto para expandir esse repositório, produzindo o MCM para outras disciplinas de Computação ministradas aos

cursos de graduação. Esse trabalho colaborativo pode proporcionar ainda mais agilidade no processo de reestruturação dos PPCs, bem como pode contribuir para a sua melhor qualidade e maior conformidade com as DCN16.

Referências

ACM/IEEE (2013). Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science. Final Report. ACM, New York, NY, USA. 2013. Disponível em: http://dx.doi.org/10.1145/2534860. Último acesso em: 16/03/2018.

Calsavara, A., Serra, A. P. G., Zampirolli, F. A., Carvalho, L. S. G., Jonathan, M., Correia, R. C. M. (2017). Referenciais de Formação: Bacharelado em Ciência da Computação. In: Zorzo, A. F., Nunes, D., Matos, E., Steinmacher, I., Leite, J., Araujo, R. M., Correia, R., Martins, S. (Org.). Referenciais de Formação para os Cursos de Graduação em Computação, 2017, p. 9-39.

Ferraz, A. P. C. M., Belhot, R. V. (2010). Taxonomia de Bloom: revisão teórica e apresentação das adequações do instrumento para definição de objetivos instrucionais. Gest. Prod., São Carlos, 17(2), 421-431.

MEC (2012). Proposta de Diretrizes Curriculares Nacionais para os cursos de graduação em Computação. Disponível em: https://goo.gl/esgE8f. Parecer CNE/CES nº 136/2012, aprovado em 8 de março de 2012. Último acesso em: 16/03/2018.

MEC (2016). Diretrizes Curriculares Nacionais para os Cursos de Graduação em Computação (DCN16). Disponível em: https://goo.gl/35CmzT. Resolução CNE/CES nº 5, de 16 de novembro de 2016. Último acesso em: 16/03/2018.

Rezende, L., Segre, L. M., Campos, G. H. (2004). O modelo das competências e as implicações para o currículo do curso de ciência da computação. In Anais do XXIV Congresso da Sociedade Brasileira de Computação (WEI). Salvador (Vol. 2).

SBC (1999). Currículo de Referência da SBC para cursos de Graduação em Computação (CR99). http://lad.dsc.ufcg.edu.br/ec/cr99.pdf. Último acesso em: 16/03/2018.

SBC (2003). Currículo de Referência da SBC para cursos de Graduação em Computação e Informática (CR03). https://goo.gl/FXncde. Último acesso em: 16/03/2018.

SBC (2005). Currículo de Referência da SBC para cursos de Graduação em Bacharelado em Ciência da Computação e Engenharia da Computação (CR05). https://goo.gl/VL7dcD. Último acesso em: 16/03/2018.

Zorzo, A. F., Nunes, D., Matos, E., Steinmacher, I., Leite, J., Araujo, R. M., Correia, R., Martins, S. (2017). Referenciais de Formação para os Cursos de Graduação em Computação. Sociedade Brasileira de Computação (SBC). 153p, 2017. ISBN 978-85-7669-424-3.