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

ABSTRACT: This study presents a mapping on the assets present in the reference standards RF-SBC, CS-Curricula ACM / IEE 2013 and SWEBOK regarding the knowledge area about software process. The theme becomes relevant, as the job market increasingly requires professionals in continuous improvement in the delivery of high quality software products and who know how to better work their development process. The guiding research question of this work is: how to match the items related to the knowledge about the software engineering process, with respect to the ACM / IEEE curriculum, the references for the formation of undergraduate courses in Computing at SBC and the guide of knowledge in Software Engineering of SWEBOK? The methodology adopted was asset mapping, described in a specific section of the paper. The result was the correspondence of the assets present in the inputs and their reference to knowledge about the software process. The aim of this mapping is to enable subsidies in the formation of a course or discipline of software process based on industry (SWEBOK) and academia (RF-SBC and CS-Curricula ACM / IEEE 2013).

Keywords: Software Process, Mapping, Assets.

Um mapeamento sobre os ativos constantes nos padrões de referência da RF-SBC, do CS-Curricula ACM/IEE 2013 e do SWEBOK no que tange à área de conhecimento sobre Processo de Software

RESUMO: Este estudo apresenta um mapeamento sobre os ativos presentes nos padrões de referência RF-SBC, CS-Curricula ACM/IEE 2013 e SWEBOK no que tange a área de conhecimento sobre processo de software. A temática torna-se relevante, pois cada vez mais o mercado de trabalho solicita profissionais em aprimoramento contínuo na entrega de produtos de software de alta qualidade e que conheçam como melhor trabalhar o seu processo de desenvolvimento. A questão de pesquisa norteadora desse trabalho é: como corresponder os itens relacionados ao conhecimento sobre o processo de engenharia de software, no que tange ao curriculum da ACM/IEEE, as referências para a formação de cursos de graduação em Computação da SBC e o guia de conhecimento em Engenharia de Software do SWEBOK? A metodologia adotada foi o mapeamento de ativos, descrito em seção específica do trabalho. O resultado foi a correspondência dos ativos presentes nos insumos e a sua referência ao conhecimento sobre processo de software. Pretende-se com esse mapeamento possibilitar subsídio na formação de curso ou então disciplina de processo de software alicerçados sobre a indústria (SWEBOK) e a academia (RF-SBC e CS-Curricula ACM/IEEE 2013).

Palavras-chave: Processo de Software, Mapeamento, Ativos.
Agradecimentos: Este trabalho pertence ao projeto SPIDER/UFPA (*http://www.spider.ufpa.br*).

1. INTRODUÇÃO

A preocupação da produção de um software de qualidade não se restringe apenas a atingir os objetivos (ou requisitos) esperados pelos usuários, mas também em obter um ciclo de vida de produção de software que apresente as características desejáveis em qualquer processo de desenvolvimento de software, como boa manutenibilidade, alta reusabilidade e baixo acoplamento. Em relação à importância do software, nunca tantas pessoas dependeram tanto de sistemas de informação, pois quase tudo depende de software (MEIRA, 2015). Quanto à formação de profissionais, tipicamente, estes se formam em cursos de graduação na área de Tecnologia da Informação a fim de se preparar para atuar na indústria de software (NUNES, REIS e REIS, 2010).

O mercado de trabalho, no entanto, identifica falhas nas competências dos profissionais formados, pois os cursos de graduação não ensinam aos estudantes as competências necessárias para que eles possam começar a executar o seu trabalho com eficiência (WANGENHEIM e SILVA, 2009; MORENO *et al.*, 2012; MEIRA, 2015]. Desta forma, as empresas de software têm que complementar os conhecimentos dos recém-formados com treinamentos e prover habilidades relacionadas ao processo de desenvolvimento de software [BESSA, CUNHA e FURTADO, 2012].

Dentro dos conhecimentos pertinentes ao desenvolvimento de software perpassa o processo de software conceituado pelo *Guide of the Software Engineering Body of Knowledge* (SWEBOK) “os processos de engenharia de software estão preocupados com as atividades de trabalho realizadas pelos engenheiros de software para desenvolver, manter e operar software, como requisitos, design, construção, teste, gerenciamento de configuração e outros processos de engenharia de software” (BOURQUE e FAIRLEY, 2014).

Os processos de software configuram-se como importantes, pois permitem facilitar a compreensão, comunicação e coordenação humana; auxiliar o gerenciamento de projetos de software; medir e melhorar a qualidade dos produtos de software de maneira eficiente; apoiar a melhoria de processos; e fornecer uma base para o suporte automatizado da execução do processo (BOURQUE e FAIRLEY, 2014).

Portanto, destaca-se a importância do processo de software como agente definidor para facilitar e permitir um desenvolvimento de software com maior precisão, além do que é importante para proporcionar a partir de medição a melhoria dos produtos de software desenvolvidos, como é possível identificar através do SWEBOK.

Assim, um mapeamento para referenciar e identificar as correspondências sobre o assunto processo de software dentro dos padrões internacionais e nacionais, por ocasião a brasileira, faz-se necessário. Isto se deve pelo motivo de se gerar um padrão para o desenvolvimento de conteúdos para serem trabalhados dentro de sala de aula com os alunos de computação, conciliando informações da indústria com o que já é abordado em sala de aula. Logo, adotou-se neste trabalho como base os seguintes insumos: Referências para a formação dos cursos de Graduação em Computação da Sociedade Brasileira da Computação (RF-SBC), “Computer Science Curricula 2013” (CS-Curricula ACM/IEE) e, por fim, mas não menos importante, o SWEBOK, por serem documentos de referência no que tange à temática sobre processo de software.

Neste contexto, este trabalho têm a seguinte questão de pesquisa: Como corresponder os itens relacionados ao conhecimento sobre o processo de engenharia de software, no que tange ao curriculum da ACM/IEEE, as referências para a formação de cursos de graduação em Computação da SBC e o guia de conhecimento em Engenharia de Software do SWEBOK? Logo, temos como objetivo um mapeamento de correspondência

entre o currículo base adotado pela ACM/IEEE para as graduações e pós-graduação em Computação, as referências para a formação de cursos de graduação da Sociedade Brasileira da Computação e o Guia para conhecimento em Engenharia de Software (SWEBOK).

A metodologia adotada nesta pesquisa é realizar o mapeamento, ou seja, adotar como documentos base os insumos produzidos pelas organizações consideradas de referência para a área de engenharia de software e por consequência processo de software. No momento da definição do documento referenciar o insumo montado como proposta para o entendimento do conhecimento sobre processos de software, bem como justificar tal realização.

Além desta seção introdutória, o artigo está estruturado da seguinte forma: a seção 2 apresenta a fundamentação teórica; já a seção 3 aborda sobre a descrição da metodologia usada nesta pesquisa; na seção 4 comenta-se o resultado do mapeamento dos ativos, bem como sua descrição e justificativa; na seção 5 são apresentados os resultados esperados com esse mapeamento; e a seção 6 apresenta as conclusões, as limitações desse estudo e os trabalhos futuros.

2. FUNDAMENTAÇÃO TEÓRICA

A seção de fundamentação teórico apresenta alguns conceitos para o melhor entendimento desse trabalho, tais como: processo de software, currículo ACM/IEEE, Referência de Formação para cursos de Graduação em Computação da SBC e o Guia de conhecimento sobre Engenharia de Software (SWEBOK).

2.1. **Processo de Software**

Segundo o guia SWEBOK (BOURQUE e FAIRLEY, 2014), o processo de software é conceituado como um conjunto de atividades e tarefas inter-relacionadas que transformam produtos de trabalho de entrada em produtos de trabalho de saída. Para se definir processo de software o mesmo guia vai informar da necessidade de entradas necessárias, atividades de trabalho transformadoras e saídas geradas. Vale destacar que as atividades podem ser decompostas em tarefas menores.

Segundo BOURQUE e FAIRLEY (2014), um processo de software pode incluir subprocessos, ou seja, no desenvolvimento de um software ter um subprocesso especificado unicamente sobre coleta e validação de requisitos, ou então um subprocesso apenas para validação e testes do software. Outro ponto a ser tratar sobre processo é a sua notação, o qual, segundo os autores, podem incluir listas textuais de atividades e tarefas constituintes descritas em linguagem natural; diagramas de fluxo de dados; BPMN; e diagramas de atividades UML.

2.2. **CS-Curricula ACM/IEE**

O *Curriculum Guidilines for Undergraduation Degree Programs in Computer Science* resumindo em *Computer Science Curricula* 2013 (CS-Curricula ACM/IEE 2013) é uma parceria e esforço contínuo da ACM/IEEE para o desenvolvimento e alinhamento de diretrizes curriculares internacionais em computação, tendo como primeira versão a publicação realizada em 1968 (ACM/IEEE, 2013). A versão atual do currículo, que esta pesquisa está relacionada, representa uma revisão abrangente, na qual foi redefinido parte do corpo de conhecimento e foram inclusos exemplos de cursos e faculdades reais para fornecer orientação concreta sobre a estrutura e o desenvolvimento curricular em uma variedade de contextos (ACM/IEEE, 2013).

Vale destacar que para o desenvolvimento do currículo os comitês gestores dos organismos internacionais, além de coletar informações via consulta pública a partir dos principais eventos de educação em computação, realizou um *survey* com professores e dirigentes de faculdades de computação para entender as necessidades da academia (ACM/IEEE, 2013). Portanto, o desenvolvimento do currículo solidifica-se através de um histórico de publicações, bem como busca informações da realidade atual, uma vez que, segundo os autores (ACM//IEEE, 2013), a computação tem uma alta volatilidade de assuntos referentes para a formação.

O documento foi definido através da seguinte estrutura: *Knowledge Areas, Knowledge Units, Topics (Core, Elective) e Learning Outcomes*, os quais serão descritos em uma seção futura sobre a descrição dos ativos usados no mapeamento.

2.3. **Referência para Formação de Cursos de Graduação em Computação da SBC**

As Referências para a Formação de Cursos de Graduação em Computação da SBC é um esforço da Comissão de Educação da SBC a partir de debates livres realizados nos principais eventos de educação em computação promovidos por esse organismo nacional (SBC, 2017). Dentro desse trabalho, vale destacar o papel de direcionador da educação em computação da SBC (SBC, 2017).

A Comissão de Educação coletou as informações dos debates e iniciou uma sistematização das informações para estar alinhada às Diretrizes Curriculares Nacionais, homologadas através da Resolução Nº05 de 16/11/2016 (MEC, 2016), e com um modelo baseado em competência (SBC, 2017). Após a sistematização com a modelagem das referências de formação com sua estrutura e definição, foi reaberta a consulta à comunidade para identificar possíveis melhorias (SBC, 2017).

Este documento de referência foi definido através da seguinte estrutura: Referência de formação, Curso, Objetivo geral do curso, Eixo de Formação, Conteúdos, Competência genérica e derivada, os quais também serão descritos em uma seção futura sobre a descrição dos ativos usados no mapeamento.

2.4. ***Guide to the Software Engineering Body of Knowledge* (SWEBOK)**

O *Guide to the Software Engineering Body of Knowledge* (SWEBOK) tem por finalidade: (i) caracterizar os conteúdos da disciplina de engenharia de software; (ii) promover uma visão consistente da engenharia de software em todo o mundo; (iii) esclarecer o lugar da engenharia de software em relação a outras disciplinas; (iv) fornecer uma base para materiais de treinamento e desenvolvimento curricular; e (v) fornecer uma base para certificação e licenciamento de engenheiros de software (BOURQUE e FAIRLEY, 2014). Portanto, descreve o conhecimento geralmente aceito sobre engenharia de software.

O guia encontra-se em sua terceira versão, a qual é uma compilação das versões anteriores com os comentários de aproximadamente de 150 revisores de 33 países, além disso encontra-se disponível gratuitamente em pdf pelo site da IEEE (BOURQUE e FAIRLEY, 2014). O documento contempla 15 *knowledge areas* e é subdividido em Subareas, Topics, Subtopics, os quais também serão descritos em uma seção futura sobre a descrição dos ativos usados no mapeamento.

3. METODOLOGIA DO MAPEAMENTO

O mapeamento proposto por esse trabalho segue como metodologia o cruzamento de informações para referenciar os ativos presentes nos insumos propostos por organizações de referência nacional e internacional no que tange tanto o processo de ensino e

aprendizagem, quanto o conhecimento adquirido acerca de determinado tema, processo de software. Logo, para a sua execução são adotados os seguintes passos:

- O primeiro passo para o desenvolvimento do mapeamento foi a definição da temática que seria abordada, onde através de uma revisão informal da literatura foi possível identificar o assunto de ensino de processo de software;
- O segundo passo foi a escolha das organizações e seus devidos insumos, sendo elas: (i) Sociedade Brasileira da Computação (SBC), por ser a organização que trata de computação no contexto brasileiro, com o insumo Referência para a Formação de Cursos de Graduação em Computação (RF-SBC); (ii) *Association for Computing Machinery / Institute of Electrical and Electronics Engineers* (ACM/ IEEE), por serem as organizações que tratam sobre assuntos relevantes à computação em nível internacional, onde as duas em parceria propuseram e desenvolveram o insumo currículo base para os cursos de computação denominado em sua última versão até a escrita dessa pesquisa "*Curriculum Guidilines for Undergraduation Degree Programs in Computer Science*", resumindo em "*Computer Science Curricula* 2013" (CS-Curricula ACM/IEE); e (iii) o IEEE, que promoveu o desenvolvimento de um outro insumo, na forma de um guia de conhecimento sobre Engenharia de Software, o qual fala sobre o processo de software denominado de "*Guide to the Software Engineering Body of Knowledge*" (SWEBOK);
- O terceiro passo foi a descrição dos ativos presentes nos insumos definidos acima. Neste passo foi analisado cada um dos documentos, bem como a bibliografia adotada por eles como base para o seu desenvolvimento. No passo em questão foi descrita a estrutura adotada pelo documento bem como o ativo;
- O quarto passo foi a correspondência dos ativos, onde a partir das informações sobre o que se tratavam cada um dos ativos foi possível fazer a referência entre eles nos documentos adotados, sendo para isso foi definida uma estrutura de equivalência;
- O quinto passo foi identificar o conhecimento de processo de software em relação a cada um dos ativos com a devida justificativa da relação.

Vale ressaltar que foi adotado um passo intercalar como medida para realizar em sua totalidade o passo quinto, o qual foi a correspondência de *Learning Outcomes* com *Topics* do CS-Curricula ACM/IEE.

4. MAPEAMENTO DOS ATIVOS PARA A FORMAÇÃO EM PROCESSO DE SOFTWARE

O mapeamento dos ativos é uma forma de referenciar sua correspondência em outros insumos, os quais tratam do mesmo assunto, no caso deste artigo processo de software. Para poder conseguir realizar esse cruzamento de informações, primeiro vamos caracterizar os ativos, na subseção 4.1, depois montar a estrutura de correspondência, na subseção 4.2, e, por fim, apontar qual os ativos específicos sobre processo de software bem como a justificativa de relacionamento, na subseção 4.3.

4.1. **Descrição dos ativos**

Nesse primeiro momento será necessário descrever os ativos que estão presentes dentro dos documentos, com o seu referido conceito, para indicar uma possível correspondência. Para isso, foi montada uma disposição de quadros, a qual tinha a seguinte estrutura: Ativo, Conceito e a referência adotada para a conceituação.

No Quadro 1 será feira a conceituação dos ativos referentes a ACM/IEEE, seguindo do mais abrangente hierarquicamente para o mais genérico.

Quadro 1 - Ativo e Descrição do curriculum da ACM/IEEE.

Ativo	Conceito	Referência
Knowledge Areas	Which representes a particular disciplinary subfield.	(ACM/IEEE, 2001)
Knowledge Units	The areas are broken down into smaller divisions calledunits, which represent individual thematic modules within an area.	(ACM/IEEE, 2001)
Topics (Core, Elective)	Each unit is further subdivided into a set of topics, which are the lowestlevel of the hierarchy. A Core Tier-1 topic should be a required part of every Computer Science curriculum. Core Tier-2 topics are generally essential in an undergraduate computer-science degree. Most programs will not cover all the elective material in the Body of Knowledge and certainly few, if any, students will cover all of it within an undergraduate program. All students of computer science should deepen their understanding in multiple areas via the elective topics.	(ACM/IEEE, 2013)
Learning Outcomes	Learning outcomes students are expected to achieve with respect to the topics specified	(ACM/IEEE, 2013)

Fonte: Elaboração própria (2020).

A segunda descrição trata dos ativos que compõem o documento do RF-SBC, os quais encontram-se descritos no Quadro 2.

Quadro 2 – Ativo e descrição do documento de referência da SBC 2017.

Ativo	Conceito	Referências
Objetivo geral do Curso	Perfil esperado para o egresso	(SBC, 2017)
Eixo de Formação	Capacitar o egresso em competências genéricas	(SBC, 2017)
Conteúdos	Conjunto de valores, conhecimentos, habilidades e atitudes	(VAN DER KLINK *et al.*, 2007)
Competência genérica e/ou derivada	Aquisição de conhecimentos, aptidões e atitudes para resolução de problemas de caráter profissional.	(VAN DER KLINK *et al.*, 2007)

Fonte: Elaboração própria (2020).

A terceira descrição refere-se aos ativos constantes no documento SWEBOK, que se encontra no Quadro 3.

Quadro 3 – Ativo e descrição do Guida de conhecimento sobre Engenharia de Software (SWEBOK).

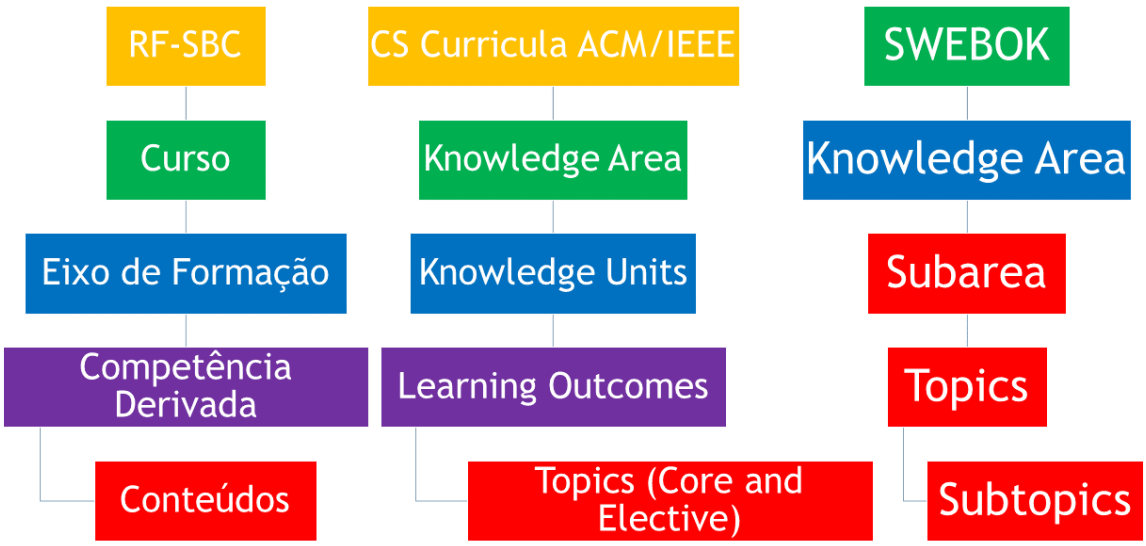
Ativo	Conceito	Referência
Knowledge Areas	Consensuson a list of Knowledge Areas.	(BOURQUE e FAIRLEY, 2014; WILLE *et al.*, 2004)
SubArea	Consensuson a list of subarea.	(WILLE *et al.*, 2004)
Topics/ Subtopics	Consensuson a list of topics and relevant reference materials for each Knowledge Area	(BOURQUE e FAIRLEY, 2014; WILLE *et al.*, 2004)

Fonte: Elaboração própria (2020).

4.2. **Correspondência dos ativos**

Dentro da pesquisa, buscou-se trabalhar com um mapeamento dos ativos dos modelos, currículos e referências de formação para cursos superiores com o intuito de identificar os pontos de similaridade. No levantamento foi gerada uma estrutura para identificar o relacionamento existente entre os ativos definidos na seção 4.1 a partir dos seus respectivos conceitos, como pode ser visto na Figura 1.

Figura 1. Mapeamento dos ativos entre CS Curricula ACM/IEEE x RF-SBC x SWEBOK.



Fonte: Elaboração própria (2020).

A Figura 1 apresenta as correspondências entres os insumos para todas os conhecimentos aos quais se deseja desenvolver para computação e em específico para Engenharia de Software. Na seção seguinte será trabalhada a correspondência desses ativos

em relação ao conhecimento necessário sobre processo de software, bem como será identificada uma justificativa para essa escolha.

4.3. **Justificativa para o mapeamento dos ativos entre CS Curricula ACM/IEEE x RF-SBC x SWEBOK em relação a Processo de Software**

O primeiro mapeamento refere-se aos documentos de referência da SBC, com a ACM/IEEE, que representam os guias que contêm os ativos para a formação em Computação das duas entidades.

No que tange à área de conhecimento a ser tratada neste trabalho, será usado o assunto referente a processo de software. A segunda equivalência é: Curso (RF-SBC), *Knowledge Area* (ACM/IEEE) e o guia do SWEBOK. Dentro das Referências de Formação da SBC as bases para a formação do documento são os cursos, no caso da ACM/IEE a *Knowledge Area* são as bases para a formação do currículo, que correspondem ao SWEBOK, que é o guia de conhecimento sobre Engenharia de Software. As correspondências sobre processos de software nos documentos são: na SBC tem-se o curso de Engenharia de Software; na ACM/IEEE tem-se a *Knowledge Area* de *Software Engineering*; e no SWEBOK tem-se o próprio guia orientador do corpo de conhecimento.

A terceira equivalência presente é referente às competências genéricas que o aluno deve ser capacitado. No documento da SBC este item refere-se ao Eixo de Formação. Para a ACM/IEEE tem-se o eixo temático de formação, o qual capacita também o aluno, por ocasião tem-se o ativo *Knowledge Units*. No que tange ao SWEBOK, tem-se a *Knowledge Area*, a qual é a subclasse de conhecimento em relação ao tema de Engenharia de software. No que tange a esse trabalho, os mapeamentos entre os referidos ativos são: o eixo de formação é o Gerenciamento e Processo de Software; a *Knowledge Units* é a *Software Process*; e a *Knowledge Area* é a *Software Engineering Process*; onde todos esses ativos referem-se às competências para a capacitação na área de processo de software.

Para alcançar as competências requeridas pelos documentos base para a formação dos alunos e profissionais de engenharia de software, são necessários os conteúdos de formação. No documento de referência dos cursos da SBC esse ativo é denominado de Conteúdo. Já no documento de referência da ACM/IEE esse ativo é tratado como *Topics*, onde ele é subdividido em: *core-tier1*, *core-tier2* e *elective*; como definido na seção 4.1. Em relação ao SWEBOK, este ativo é descrito na forma de uma hierarquia com três níveis, sendo eles: *Subarea, topics e subtopics*. O Quadro 6 trata do mapeamento entre esses ativos.

Para realizar o mapeamento total dos ativos constantes na ACM/IEEE referentes aos *topics*, foi necessário considerar ativos de conteúdo da SBC constantes no eixo de formação Qualidade de Software, uma vez que este eixo de formação possui cruzamento de conteúdos referentes à área de processo de software.

Quadro 4 – Quinta Equivalência

ACM/IEEE (*Topics*)	**SWEBOK (*Subarea/ Topics/ Subtopics*)**	**SBC (Conteúdos)**
Systems level considerations, i.e., the interaction of software with its intended environment (crossreference	-	Gerenciamento do ciclo de vida de produção
		Gerenciamento do fluxo de trabalho
		Engenharia de produto
		Modelos de ciclo de vida: história e perspectivas

ACM/IEEE (*Topics*)	SWEBOK (*Subarea/ Topics/ Subtopics*)	SBC (Conteúdos)
IAS/Secure Software Engineering)		Segurança do processo de Engenharia de Software
Introduction to software process models (e.g., waterfall, incremental, agile) • **Activities within software lifecycles**	Software Process Definition	Teoria Geral de Processos (modelagem, especificação, análise e controle, adaptação)
		Conceito de processo de software
	Software Engineering Process Tools	Representação de processo de software
	Software Life Cycle	Modelos de processo de software
Programming in the large vs. individual Programming	-	Refatoração
		Engenharia reversa
		Reengenharia
		Análise de impacto
		Manutenção
		Depuração
		Práticas de laboratório no desenvolvimento e evolução de software
Evaluation of software process models	Software Measurement	Artefatos de software, papéis, métricas de processo de software
Software quality Concepts	Software Measurement	Modelos e normas de qualidade de processo (nacionais e internacionais)
		Métricas de processo de Software
Process Improvement	Software Process Assessment and Improvement	Modelos e normas de qualidade de processo (nacionais e internacionais)
Software process capability maturity models	Software Process Assessment and Improvement	Modelos e normas de qualidade de processo (nacionais e internacionais)
		Normatização e certificações de qualidade

Fonte: Elaboração própria (2020).

Uma das bases do currículo da ACM/IEEE é a taxonomia de Bloom, necessária para o desenvolvimento de competências no aluno. Para cada uma das *Knowledge Units* foi proposto tanto *Learnning Outcomes* quanto *Topics*, porém estes ativos não possuem uma relação direta. Entretanto, na SBC os ativos de Conteúdo estão relacionados diretamente com as Competências Derivadas. Desta forma, uma vez que o Conteúdo na SBC equivale a *Topics* na ACM/IEEE, e as Competências Derivadas na SBC equivalem a *Learning Outcomes* na ACM/IEEE, para que se consiga relacionar as *Learning Outcomes* da ACM/IEEE com as Competências Derivadas da SBC faz-se necessário definir um relacionamento de *Learning Outcomes* com *Topics* da ACM/IEEE (para caracterizar as competências/aprendizados esperados em cada um dos tópicos inclusos no documento de

referência), possibilitando o mapeamento de competências propostas entre os dois currículos. Este relacionamento encontra-se mapeado no Quadro 7.

Quadro 5 – Relacionamento de *Topics* com *Learning Outcomes* da ACM/IEEE.

Tipo	***Topics***	***Learning Outcomes***	**Justificativa**
Core-Tier1	Systems level considerations, i.e., the interaction of software with its intended environment (crossreference IAS/Secure Software Engineering)	1. Describe how software can interact with and participate in various systems including Information management, embedded, process control, and communications systems. [Familiarity]	As considerações a nível de sistema permitem ao aluno compreender as interações entre os participantes e os sistemas para o descrever.
	Programming in the large vs. individual Programming	5. Describe how programming in the large differs from individual efforts with respect to understanding a large code base, code reading, understanding builds, and understanding context of changes. [Familiarity]	Com o conteúdo de Programação e Considerações a nível de sistema é possível identificar como programar e a necessidade de desenvolvimento para a produção do software.
	Introduction to software process models (e.g., waterfall, incremental, agile) (Activities within software lifecycles)	2. Describe the relative advantages and disadvantages among several major process models (e.g., waterfall, iterative, and agile). [Familiarity]	A introdução ao modelo de software permite descrever as vantagens e desvantagens sobre i uso de modelo de processo.
		3. Describe the different practices that are key components of various process models. [Familiarity]	A introdução permite conhecimento aos conceitos básicos sobre os componentes do processo de software.
		4. Differentiate among the phases of software development. [Familiarity]	Dentro da fase introdutória é possível identificar e diferenciar as fases do desenvolvimento de software.
Core-tier2	Evaluation of software process	6. Explain the concept of a software lifecycle	Com o conhecimento de avaliação de

Tipo	*Topics*	*Learning Outcomes*	Justificativa
	models	and provide an example, illustrating its phases including the deliverables that are produced. [Familiarity]	modelo de processo de software é possível identificar os ciclos de vida e possíveis exemplos de uso.
		7. Compare several common process models with respect to their value for development of particular classes of software systems taking into account issues such as requirement stability, size, and non-functional characteristics. [Usage]	Com o assunto de avalição dos modelos de processo de software é possível realizar uma comparação entre os modelos.
Elective	Software quality Concepts	8. Define software quality and describe the role of quality assurance activities in the software process. [Familiarity]	Os conceitos sobre qualidade de software permitem descrever a mesma.
	Process Improvement	9. Describe the intent and fundamental similarities among process Improvement approaches. [Familiarity]	O conteúdo de melhoria de processo permite conhecer os seus fundamentos.
		12. Explain the role of process maturity models in process improvement. [Familiarity]	Através do conteúdo melhoria de processo é possível explicá-lo.
	Software process capability maturity models	10. Compare several process improvement models such as CMM, CMMI, CQI, Plan-Do-Check-Act, or ISO9000. [Assessment]	Dentro do conceito de maturidade no processo é possível fazer um comparativo dos que já estão disponíveis.
	Software process measurements	11. Assess a development effort and recommend potential changes by participating in process Improvement (using a model such as PSP) or	Dentro da medição de software é possível identificar o esforço além de práticas para melhorar o desenvolvimento.

Tipo	*Topics*	*Learning Outcomes*	Justificativa
		engaging in a project retrospective. [Usage]	
		13. Describe several process metrics for assessing and controlling a project. [Familiarity]	Na medição de software é possível descrever as métricas do produto e do processo de software.
		14. Use project metrics to describe the current state of a project. [Usage]	Ao entender as métricas do projeto é possível identificar o estágio do seu desenvolvimento.

Fonte: Elaboração própria (2020).

Portanto, a partir do relacionamento definido entre *Topics* e *Learning Outcomes* do *Curricula* da ACM/IEEE, podemos por dedução identificar a equivalência das *Learning Outcomes* com as Competências Derivadas da RF-CS-SBC. Esta equivalência encontra-se definida no Quadro 8.

Quadro 6 – Equivalência entre *Learning Outcomes* e Competências Dericadas

Learning Outcomes	Competências derivadas
1. Describe how software can 12roject12 with and participate in various systems including Information management, embedded, process control, and communications systems. [Familiarity]	**C.4.5.** Aplicar técnicas, ferramentas e práticas para gerenciamento do processo da produção, aquisição e evolução de um software
	C.4.9. Revisar o processo geral de Engenharia de Software de forma a garantir segurança
2. Describe the relative advantages and disadvantages among several major process models (e.g., waterfall, iterative, and agile). [Familiarity]	**C.4.1.** Conhecer os fundamentos da teoria de processos / **C.4.2.** Aplicar processos de construção de software
3. Describe the 12roject12e practices that are key 12roject12ent of various process models. [Familiarity]	
4. Differentiate among the phases of software 12roject12ent. [Familiarity]	
5. Describe how 12roject12ent in the large differs from individual efforts with respect to understanding a large code base, code 12roject, understanding builds, and understanding context of changes. [Familiarity]	**C.4.3.** Aplicar técnicas e procedimentos de manutenção e evolução de software
	C.4.2. Aplicar processos de construção de software
6. Explain the concept of a software lifecycle and provide isso example, illustrating its phases including the deliverables that are produced. [Familiarity]	**C.4.5.** Aplicar técnicas, ferramentas e práticas para gerenciamento do processo da produção, aquisição e evolução de um software

Learning Outcomes	Competências derivadas
7. Compare several common process models with respect to their value for 13roject13ent of particular classes of software systems taking into account issues such as requirement stability, size, and non-functional characteristics. [Usage]	
8. Define software quality and describe the role of quality assurance activities in the software process. [Familiarity]	**C.7.4.** Entender as normas e modelos de qualidade de produto e processo de software/ **C.7.5.** Aplicar conceitos de qualidade de processo para a definição de um processo de software
9. Describe the 13roject and fundamental similarities among process Improvement approaches. [Familiarity]	
10. Compare several process 13roject13ent models such as CMM, CMMI, CQI, Plan-Do-Check-Act, or ISSO9000. [Assessment]	
11. Assess a 13roject13ent effort and recommend potential changes by participating in process Improvement (using a model such as PSP) or engaging in a 13roject retrospective. [Usage]	
12. Explain the role of process 13roject13 models in process 13roject13ent. [Familiarity]	
13. Describe several process metrics for assessing and controlling a 13roject. [Familiarity]	
14. Use 13roject metrics to describe the current state of a 13roject. [Usage]	

Fonte: Elaboração própria (2020).

5. RESULTADOS ESPERADOS COM O MAPEAMENTO

Nesta seção, apresentam-se os resultados esperados com o mapeamento. Em um primeiro momento o mapeamento permite uma referência entre os principais documentos de referência sobre o ensino e o conhecimento sobre engenharia de software. Após esse resultado, é possível identificar a partir do mapeamento uma referência sobre o conhecimento sobre processo de software em nível nacional e internacional, o que tange as práticas acadêmicas e de mercado.

O mapeamento possibilita também o entendimento sobre as principais competências definidas em nível nacional e internacional, quando abordamos a área temática de processo de software. Além disso, é possível visualizar os conhecimentos necessários para o desenvolvimento delas.

Com as informações a respeito de processo de software no que tange as principais referências ao seu conhecimento a nível nacional e internacional nas esferas de guia de conhecimento industrial e acadêmico, pode-se pensar em abordagens para o desenvolvimento de conteúdos, os quais possibilitam o conhecimento sobre processo de software.

6. CONCLUSÕES E TRABALHOS FUTUROS

O mapeamento dos ativos dos insumos pulicados e mantidos pelas organizações nacional e internacional sobre computação permite criar uma sintonia sobre o que é necessário para realizar um alinhamento acerca de outros conhecimentos sobre engenharia de software, ou até mesmo da própria engenharia de software. Além do que, permite uma correspondência entre documentos, o qual ainda não foi visto na literatura, quando se aborda sobre a área temática de processo de software.

Portanto, o trabalho ao desenvolver o mapeamento permite um referencial sobre os principais conteúdos e competências necessários ao se desenvolver o processo de ensino e aprendizagem sobre processo de software. Esta informação configura-se como um importante instrumento para a colaboração futura na elaboração de novas abordagens de ensino em processo de software.

Como trabalhos futuros espera-se: desenvolver uma abordagem centrada no aluno para o ensino e aprendizagem sobre processo de software; construir um curso/disciplina alicerçados no mapeamento sobre processo de software; desenvolver o mapeamento para outros conhecimentos sobre processo de software.

REFERÊNCIAS BIBLIOGRÁFICAS

ACM/IEEE. (2001). Computer Science Curricula 2001. *ACM and IEEE Computer Society, Incorporated: New York, NY, USA*.

ACM/IEEE. (2013). Computer Science Curricula 2013. *ACM and IEEE Computer Society, Incorporated: New York, NY, USA*.

BESSA, B., CUNHA, M., FURTADO, F. (2012). “Engsoft: Ferramenta para simulação de ambientes reais para auxiliar o aprendizado baseado em problemas (pbl) no ensino de engenharia de software”. In Anais do XX Workshop sobre Educação em Informática. Curitiba-PR.

BOURQUE, P. e FAIRLEY, R. (2014). Software engineering body of knowledge (SWEBOK) v3. *IEEE Computer Society, EUA*.

BOURQUE, P. e FAIRLEY, R. E. (2014). Guide to the software engineering body of knowledge (SWEBOK (R)): Version 3.0. IEEE Computer Society Press.

MEIRA, S. (2015). “Sistemas de Informação e Engenharia de Software – Cadê as Escolas?”. Revista da SBC Engenharia de Software - Qual é o impacto da ES no mercado de Computação e na sociedade como um todo? Porto Alegre, Brasil.

MORENO, A. M., SANCHEZ-SEGURA, M. I., MEDINA-DOMINGUEZ, F., CARVAJAL, L. (2012). “Balancing software engineering education and industrial needs”. Journal of systems and software, 85(7), 1607-1620.

NUNES, D. J., REIS, C., REIS, R. (2010). “Educação em Engenharia de Software: A carreira de pesquisador em engenharia de software: princípios, conceitos e direções”. Salvador.

Sociedade Brasileira de Computação (2017). Referenciais de Formação para os Cursos de Graduação em Computação.(2017). Retrieved May, 11, 2018.

VAN der KLINK, M., BOON, J., SCHLUSMANS, K. (2007). Competências e ensino superior profissional: presente e futuro. Revista Europeia de Formação Profissional, 40(1), 72-89.

WANGENHEIM, C. G., SILVA, D. A. (2009). “Qual conhecimento de engenharia de software é importante para um profissional de software?”. Proceedings of the Fórum de Educação em Engenharia de Software, 2, 1-8.

WILLE, C., DUMKE, R. R., ABRAN, A., DESHARNAIS, J. M. (2004). “E-learning infrastructure for software engineering education: Steps in ontology modeling for SWEBOK”. In IASTED Conf. on Software Engineering (pp. 520-525).