PGCOMP - Programa de Pós-Graduação em Ciência da Computação
Universidade Federal da Bahia (UFBA)
Av. Milton Santos, s/n - Ondina
Salvador, BA, Brasil, 40170-110

https://pgcomp.ufba.br
pgcomp@ufba.br

Contexto: A demanda crescente da indústria de software por profissionais qualificados coloca em evidência a importância e necessidade de valorização da Educação em Engenharia de Software (EES) na formação de egressos dos cursos superiores da área de Computação. Educadores são desafiados continuamente a acompanhar as tendências emergentes na área, a identificar técnicas adequadas e incorporá-las em suas disciplinas de engenharia de software, de modo a promover o desenvolvimento de habilidades e competências que garantam aos egressos dos cursos da área de Computação uma formação que concilie o aprendizado de conceitos, processos, técnicas e ferramentas proporcionado no ambiente acadêmico com as necessidades e melhores práticas da indústria de software. Projetos de Software Livre/Código Aberto (FLOSS) trazem uma oportunidade para que estudantes e professores de cursos de graduação possam trabalhar o conteúdo da disciplina de engenharia de software em projetos realistas, que trazem uma experiência de mundo real próxima das exigências do mundo do trabalho. **Problema:** Um dos desafios associados ao uso de FLOSS na EES é como o educador pode avaliar o projeto escolhido como recurso de ensino e aprendizagem, bem como sua eficácia para o objetivo pedagógico proposto pela atividade ou disciplina. Torna-se relevante o uso de métodos ou modelos de avaliação que permitam ao docente mensurar e avaliar os resultados provenientes da sua escolha, de acordo com as exigências de aprendizagem necessárias após a conclusão da atividade ou disciplina. **Objetivo:** Este trabalho propõe a criação de um modelo para avaliar a eficácia do uso de projetos FLOSS no ensino da disciplina de Engenharia de Software. O modelo proposto é fundamentado no mapeamento de critérios de avaliação que foram estruturados de forma a facilitar a análise sobre o alcance dos objetivos pedagógicos planejados para o componente curricular. **Método de Pesquisa:** Por meio da Análise de Conteúdo, foram identificadas as competências e habilidades técnicas e sociotécnicas mais relevantes para a disciplina de Engenharia de Software, segundo as principais diretrizes curriculares do Brasil e do exterior, e investigados aspectos de influência tanto técnicas quanto pedagógicas, do uso de projetos FLOSS no ensino da disciplina de engenharia de software. Com base nestas informações, foram definidos os critérios de avaliação e, por meio de um diálogo teórico, identificas suas relações para propor um Modelo de avaliação da eficácia do uso de projetos FLOSS no ensino de Engenharia de Software. **Contribuições:** O uso de métodos para avaliar projetos FLOSS na EES mostra-se uma solução viável para o desenvolvimento real de competências e habilidades associadas a aspectos teóricos e práticos da engenharia de software, importantes na formação de profissionais das diversas vertentes da área de Computação. O uso de métodos para avaliar projetos FLOSS na EES pode estimular a inserção dos educadores no ecossistema de FLOSS, além de oferecer novas formas de avaliação aos que reconhecem e adotam tais projetos em sala de aula.
Palavras-chave: Análise de Conteúdo, Modelo de Avaliação, Software de Código Aberto, Currículo, Educação em Engenharia de Software.

MSC | 154 | 2023

Um Modelo para avaliar a eficácia do uso de Projetos FLOSS no Ensino de Engenharia de Software: Uma proposta baseada em Análise de Conteúdo

Erinaldo Santos Oliveira

UFBA

Um Modelo para avaliar a eficácia do uso de Projetos FLOSS no Ensino de Engenharia de Software: Uma proposta baseada em Análise de Conteúdo

Erinaldo Santos Oliveira

Dissertação de Mestrado

Universidade Federal da Bahia

Programa de Pós-Graduação em
Ciência da Computação

Março | 2023

Universidade Federal da Bahia
Instituto de Computação

Programa de Pós-Graduação em Ciência da Computação

UM MODELO PARA AVALIAR A EFICÁCIA DO USO DE PROJETOS FLOSS NO ENSINO DE ENGENHARIA DE SOFTWARE: UMA PROPOSTA BASEADA EM ANÁLISE DE CONTEÚDO

Erinaldo Santos Oliveira

DISSERTAÇÃO DE MESTRADO

Salvador
31 de Março de 2023

ERINALDO SANTOS OLIVEIRA

UM MODELO PARA AVALIAR A EFICÁCIA DO USO DE PROJETOS FLOSS NO ENSINO DE ENGENHARIA DE SOFTWARE: UMA PROPOSTA BASEADA EM ANÁLISE DE CONTEÚDO

Esta Dissertação de Mestrado foi apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Orientadora: Christina von Flach Garcia Chavez

Salvador
31 de Março de 2023

Sistema de Bibliotecas - UFBA

O48 Oliveira, Erinaldo Santos .
Um Modelo para avaliar a eficácia do uso de Projetos FLOSS no Ensino de Engenharia de Software: Uma proposta baseada em Análise de Conteúdo / Erinaldo Santos Oliveira – Salvador, 2023.
203p.: il.

Orientadora: Prof. Dr. Christina von Flach Garcia Chavez.
Dissertação (Mestrado) – Universidade Federal da Bahia, Instituto de Computação, 2023.

1. Engenharia de Software. 2. Software. 3. Currículo. 4. Educação - Engenharia de Software. I. Chavez, Christina von Flach Garcia . II. Universidade Federal da Bahia. Instituto de Computação. III Título.

CDU – 004.41

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DA BAHIA
INSTITUTO DE COMPUTAÇÃO
PGCOMP - Programa de Pós-Graduação em Ciência da Computação
http://pgcomp.ufba.br

"*Um Modelo para Avaliar a Eficácia do Uso de Projetos FLOSS no Ensino de Engenharia de Software: Uma Proposta baseada em Análise de Conteúdo*"

Erinaldo Santos Oliveira

Dissertação apresentada ao Colegiado do Programa de Pós-Graduação em Ciência da Computação na Universidade Federal da Bahia, como requisito parcial para obtenção do Título de Mestre em Ciência da Computação.

Banca Examinadora

Documento assinado digitalmente
gov.br CHRISTINA VON FLACH GARCIA CHAVEZ
Data: 03/09/2023 14:10:51-0300
Verifique em https://validar.iti.gov.br

Profª. Drª. Christina Von Flach Garcia Chavez (Orientadora PGCOMP)

Documento assinado digitalmente
gov.br LAIS DO NASCIMENTO SALVADOR
Data: 03/09/2023 20:03:44-0300
Verifique em https://validar.iti.gov.br

Profª. Drª. Laís do Nascimento Salvador (PGCOMP-UFBA)

Documento assinado digitalmente
gov.br ROBERTO ALMEIDA BITTENCOURT
Data: 03/09/2023 14:59:50-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Roberto Almeida Bittencourt (UEFS)

Deus seja louvado, pois tudo o que fiz e farei a ele dedico.

AGRADECIMENTOS

Agradeço a Deus todas as bençãos e a oportunidade de passar por esse desafio tão difícil. Agradeço minha mãe, minha dinda, minha vó e minha irmã por serem minha base. Agradeço a todos da minha família por todo apoio e suporte essencial para chegar até aqui. Agradeço a minha esposa linda pela motivação constante nesse trabalho e a minha Lua por existir. Agradeço a minha Professora e Orientadora Christina von Flach por seu apoio desde o início, por sua sensibilidade e competência, essenciais para me guiar até a conclusão desta etapa. Agradeço a todos os meus professores do PGCOMP por me fornecerem, cada um, um pouco do seu conhecimento. Agradeço ao colegiado do PGCOMP pela sensibilidade e carinho com nós estudantes. Agradeço a UFBA pela oportunidade concedida. Agradeço aos meus colegas e todos que por mim passaram e apoiaram para a conclusão deste trabalho. Obrigado a todos.

Tudo posso naquele que me fortalece.

—FILIPENSES (4:13)

RESUMO

Contexto: A demanda crescente da indústria de software por profissionais qualificados coloca em evidência a importância e necessidade de valorização da EES na formação de egressos dos cursos superiores da área de Computação. Educadores são desafiados continuamente a acompanhar as tendências emergentes na área, a identificar técnicas adequadas e incorporá-las em suas disciplinas de engenharia de software, de modo a promover o desenvolvimento de habilidades e competências que garantam aos egressos dos cursos da área de Computação uma formação que concilie o aprendizado de conceitos, processos, técnicas e ferramentas proporcionado no ambiente acadêmico com as necessidades e melhores práticas da indústria de software. Projetos de Software Livre/Código Aberto[1] trazem uma oportunidade para que estudantes e professores de cursos de graduação possam trabalhar o conteúdo da disciplina de engenharia de software em projetos realistas, que trazem uma experiência de mundo real próxima das exigências do mundo do trabalho. **Problema:** Um dos desafios associados ao uso de FLOSS na EES é como o educador pode avaliar o projeto escolhido como recurso de ensino e aprendizagem, bem como sua eficácia para o objetivo pedagógico proposto pela atividade ou disciplina. Torna-se relevante o uso de métodos ou modelos de avaliação que permitam ao docente mensurar e avaliar os resultados provenientes da sua escolha, de acordo com as exigências de aprendizagem necessárias após a conclusão da atividade ou disciplina. **Objetivo:** Este trabalho propõe a criação de um modelo para avaliar a eficácia do uso de projetos FLOSS no ensino da disciplina de Engenharia de Software. O modelo proposto é fundamentado no mapeamento de critérios de avaliação que foram estruturados de forma a facilitar a análise sobre o alcance dos objetivos pedagógicos planejados para o componente curricular. **Método de Pesquisa:** Por meio da Análise de Conteúdo, foram identificadas as competências e habilidades técnicas e sociotécnicas mais relevantes para a disciplina de Engenharia de Software, segundo as principais diretrizes curriculares do Brasil e do exterior, e investigados aspectos de influência tanto técnicas quanto pedagógicas, do uso de projetos FLOSS no ensino da disciplina de engenharia de software. Com base nestas informações, foram definidos os critérios de avaliação e, por meio de um diálogo teórico, identificas suas relações para propor um Modelo de avaliação da eficácia do uso de projetos FLOSS no ensino de Engenharia de Software. **Contribuições:** O uso de métodos para avaliar projetos FLOSS na EES mostra-se uma solução viável para o desenvolvimento real de competências e habilidades associadas a aspectos teóricos e práticos da engenharia de software, importantes na formação de profissionais das diversas vertentes da área de Computação. O uso de métodos para avaliar projetos FLOSS na EES pode estimular a inserção dos educadores no ecossistema de FLOSS, além de oferecer novas formas de avaliação aos que reconhecem e adotam tais projetos em sala de aula.

[1]Free/Libre Open Source Software (FLOSS)

Palavras-chave: Análise de Conteúdo, Modelo de Avaliação, Software de Código Aberto, Currículo, Educação em Engenharia de Software.

ABSTRACT

Context: The software industry's growing demand for qualified professionals highlights the importance and need for valuing EES in the training of graduates from higher education courses in the area of Computing. Educators are continually challenged to keep up with emerging trends in the area, to identify appropriate techniques and incorporate them into their software engineering disciplines, in order to promote the development of skills and competences that guarantee graduates of courses in the area of Computing an education that reconciles the learning of concepts, processes, techniques and tools provided in the academic environment with the needs and best practices of the software industry. Free/Libre Open Source Software (FLOSS) provide an opportunity for students and professors of undergraduate courses to work the content of the software engineering discipline into realistic projects, which bring a close real-world experience the demands of the world of work. **Problem:** One of the challenges associated with the use of FLOSS in the EES is how the educator can evaluate the chosen project as a teaching and learning resource, as well as its effectiveness for the pedagogical objective proposed by the activity or discipline. It is relevant to use evaluation methods or models that allow the teacher to measure and evaluate the results from his/her choice, according to the learning requirements necessary after the conclusion of the activity or discipline. **Objective:** This work proposes the creation of a model to evaluate the effectiveness of using FLOSS projects in the teaching of Software Engineering. The proposed model is based on the mapping of evaluation criteria that were structured in order to facilitate the analysis of the scope of the pedagogical objectives planned for the curricular component. **Research Method:** Through Content Analysis, the most relevant technical and sociotechnical skills and abilities were identified for the Software Engineering discipline, according to the main curricular guidelines in Brazil and abroad, and aspects of influence were investigated both technical and pedagogical, of the use of FLOSS projects in teaching the discipline of software engineering. Based on this information, the evaluation criteria were defined and, through a theoretical dialogue, their relations were identified in order to propose a Model for evaluating the effectiveness of the use of FLOSS projects in the teaching of Software Engineering. **Contributions:** The use of methods to evaluate FLOSS projects at the EES proves to be a viable solution for the real development of skills and abilities associated with theoretical and practical aspects of software engineering, important in the training of professionals from the various strands of Computing area. The use of methods to evaluate FLOSS projects in the EES can encourage the inclusion of educators in the FLOSS ecosystem, in addition to offering new forms of evaluation to those who recognize and adopt such projects in the classroom.

Keywords: Content Analysis, Assessment Model, Open Source Software, Curriculum, Software Engineering Education.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

Capítulo 1

INTRODUÇÃO

1.1 CONTEXTO

A importância da Educação em Engenharia de Software (EES) é reconhecida no ecossistema de engenharia de software e desperta crescente atenção da comunidade, dada a necessidade de técnicas, métodos e objetos que contribuam para a formação e desenvolvimento do profissional de software diante de novos e complexos desafios (DUARTE-FILHO et al., 2015). O domínio de técnicas, conceitos e métodos básicos de engenharia de software é requisito mínimo para a formação de qualquer profissional da área de computação (SHACKELFORD et al., 2005). Assim, o ensino eficaz da Engenharia de Software (ES) norteia pesquisas e estudos que contribuam para a aplicação profissional de suas práticas e técnicas (SANTOS et al., 2009b; PRIKLADNICKI et al., 2009).

Entretanto, o ensino efetivo de Engenharia de Software é um desafio na academia. Normalmente, a engenharia de software é lecionada em tópicos ou disciplinas em cursos de graduação da área de computação de forma superficial e de abordagem teórica, e portanto, não estimulam o aprendizado prático de tais conceitos e técnicas nas atividades profissionais (BEGOSSO et al., 2011; SANTOS et al., 2014). Assim, são necessários mecanismos educacionais que ponham em prática os conhecimentos da engenharia de software e suas especificidades. Diversos estudos buscam o uso de aplicações, técnicas e métodos para estimular o aprendizado prático da ES, como os programas de residência e uso de jogos no processo de ensino aprendizagem (BEGOSSO et al., 2011; SAVI; WANGENHEIM; BORGATTO, 2011; OLIVEIRA; OLIVEIRA; MEIRA, 2017).

Porém, há o desafio para compatibilizar tais elementos de ensino à realidade acadêmica e à prática real na indústria. Vários autores salientam que o uso de projetos e demais objetos de ensino de engenharia de software abordam a prática através de problemas de brinquedo ou de baixa complexidade, não alcançando a aproximação desejada entre a vivência da indústria e a academia (NASCIMENTO; BITTENCOURT; CHAVEZ, 2015; STAMELOS, 2011; CERONE; SOWE, 2010). Neste contexto, Projetos de Software Livre/Código Aberto (FLOSS)[1] surgem como opção realista para o ensino prá-

[1]Free/Libre Open Source Software (FLOSS)

tico da engenharia de software de forma a aproximar a didática acadêmica da prática na indústria (CHAVEZ et al., 2011; STAMELOS, 2011).

1.2 PROBLEMA

A seleção, uso e validação de projetos FLOSS adequados para os objetivos educacionais da ES é um desafio. Segundo (CHAVEZ et al., 2011, p.1) tradução nossa "Este ambiente abre muitas oportunidades para a Educação em Engenharia de Software". Algumas práticas didáticas dão liberdade aos estudantes na escolha do projeto FLOSS, entretanto, as disciplinas ou componentes curriculares possuem cronograma e tempo de aplicação limitados, cabendo ao educador o desafio de uso de um projeto FLOSS que promova a aprendizagem alinhada aos objetivos pedagógicos e o engajamento dos alunos na participação das atividades (STAMELOS, 2011; SOWE; STAMELOS, 2007).

O projeto FLOSS utilizado como material de ensino de forma compartilhada e reutilizada assume características de um Objeto de Aprendizagem (OA) ou Recurso de Ensino e Aprendizagem e, portanto, necessita de análise de seu uso técnico e pedagógico. A avaliação de um objeto ou material usado no processo de ensino-aprendizagem fornece ao educador o conhecimento e experiência para melhores decisões nos contextos de reutilização (MUSSOI; FLORES; BEHAR, 2010). Para que o professor de engenharia de software faça boas escolhas e tome boas decisões acerca do uso de FLOSS em suas práticas de ensino, a avaliação da eficácia do FLOSS escolhido para o alcance dos objetivos pedagógicos da disciplina ou da atividade torna-se relevante. Assim, torna-se pertinente compreender como avaliar a eficácia do uso de um Projeto FLOSS no ensino de engenharia de software.

A eficácia do ensino de engenharia de software manifesta-se no alcance dos objetivos de aprendizagem, necessidade comum a qualquer prática educativa. Tais objetivos de aprendizagem giram em torno do desenvolvimento de competências e habilidades que garantam aos futuros egressos o desenvolvimento profissional e domínio das práticas.

Segundo (ZORZO et al., 2017; BRANDÃO; BAHRY, 2005; NUNES; YARMAGUTI; NUNES, 2016), as competências tratam da capacidade de mobilizar conhecimentos, habilidades e atitudes a fim de proporcionar intervenção e ação na resolução de problemas previsíveis e imprevisíveis em um contexto cultural e profissional. Assim, várias diretrizes e referenciais estruturam os conhecimentos e habilidades necessários à aquisição de competências para efetiva formação profissional em engenharia de software, como os referenciais e corpos de conhecimento da Sociedade de Computação do Instituto de Engenharia Elétrica e Eletrônica (IEEE-CS) e da Associação para Maquinas de Computação (ACM). Diversos autores destacam a importância do estímulo à aquisição de competências que vão além dos conhecimentos e habilidades comumente atrelados ao domínio técnico em engenharia de software. Tais competências norteiam habilidades profissionais de aspectos humanísticos e sociais, chamados de sociotécnicos (CUKIERMAN; TEIXEIRA; PRIKLADNICKI, 2007; TEIXEIRA; CUKIERMAN, 2005).

1.3 OBJETIVOS

O objetivo geral desta pesquisa é *compreender competências associadas a Engenharia de Software e aspectos de influência citados na EES que consideram o uso didático de projetos FLOSS, para propor um modelo para avaliar a eficácia do uso desses projetos no Ensino de Engenharia de Software.*

Em geral, a avaliação de objetos e elementos utilizados no processo de ensino e aprendizagem é realizada por meio da análise dos efeitos e resultados obtidos segundo critérios relevantes identificados (BRAGA, 2014; MUSSOI; FLORES; BEHAR, 2010; GALHARDI; AZEVEDO, 2013). O modelo de avaliação proposto visa relacionar as influências técnicas ou pedagógicas estimulantes ou limitantes observadas em um projeto FLOSS (selecionado como material de ensino) com o alcance de objetivos de aprendizagem que desenvolvam competências técnicas e sociotécnicas de ES previamente planejadas.

São objetivos específicos desta pesquisa:

O1 Mapear competências técnicas e sociotécnicas relacionadas a Engenharia de Software;

O2 Mapear a relevância do conhecimento em Engenharia de Software e do conhecimento sociotécnico para a formação superior em Computação;

O3 Mapear aspectos técnicos ou pedagógicos que estimulem ou limitem o alcance das competências de Engenharia de Software por meio do uso de projetos FLOSS; e

O4 Relacionar competências e aspectos mapeados para propor um modelo conceitual de avaliação da eficácia do uso de Projetos FLOSS no ensino de Engenharia de Software.

Para alcançar os objetivos propostos será necessário responder algumas questões de pesquisa:

Q1 Quais as competências técnicas e sociotécnicas associadas com a Engenharia de Software segundo os principais Referenciais de Formação Superior em Computação no Brasil?

Q2 Qual a relevância do conhecimento em Engenharia de Software e do conhecimento sociotécnico segundo os principais guias curriculares internacionais?

Q3 Quais os principais aspectos de interferência no ensino e aprendizagem reportados em estudos sobre uso de Projetos FLOSS na Educação em Engenharia de Software?

Q4 Como organizar e modelar as competências e os aspectos mapeados como critérios para o educador avaliar a eficácia do uso de um Projeto FLOSS no ensino de Engenharia de Software?

1.4 MÉTODOS DE PESQUISA

A Análise Bibliográfica é o procedimento técnico que guiou o estudo. Para o desenvolvimento do modelo, várias fontes bibliográficas foram utilizadas para a coleta e posterior análise dos dados. Para mapear as competências relacionadas ao ensino de Engenharia de Software foram analisados os Referenciais de Formação para os Cursos de Graduação em Computação da Sociedade Brasileira de Computação. Para mapear a relevância dos conhecimentos técnicos de ES e do conhecimento sociotécnico para formação superior em Computação foram analisados os principais Guias Curriculares para Cursos de Graduação da ACM. Para mapear os aspectos de influência técnica ou pedagógica na aprendizagem em ES mediada pelo uso de Projetos OSS foram analisados 30 estudos selecionados no estudo secundário **FLOSS in Software Engineering Education: An Update of a Systematic Mapping Study** no contexto de uso de Projetos FLOSS no Ensino de Engenharia de Software.

Para a análise dos dados mapeados foi utilizada a técnica da Análise de Conteúdo baseada em Bardin (BARDIN, 2011). Tal técnica tem como objetivo investigar informações explícitas e implícitas provenientes de conteúdos manifestos. Após a análise dos dados, os mesmos foram dialogados teoricamente para trazer luz às suas relações e permitir contextualização técnica e pedagógica para propor um modelo conceitual de avaliação da eficácia do uso de Projetos OSS no Ensino de Engenharia de Software. Todo o processo de coleta e análise dos dados foi baseado numa abordagem mista intitulada por Creswell como Transformadora Concomitante. Sobre essa abordagem, (CRESWELL, 2007, p.223) explica que:

> [...] nas estratégias concomitantes, o pesquisador pode quantificar os dados qualitativos. Isso envolve criar códigos e temas qualitativamente; em seguida, contar o número de vezes que eles ocorrem nos dados de texto (ou possivelmente o quanto se fala sobre um código ou tema, contando linhas ou sentenças).

A técnica de enumeração na Análise de Conteúdo de Bardin permitiu esse processo. O mapeamento nos Referenciais Nacionais foi realizado por meio da análise de co-ocorrência. O mapeamento dos Guias internacionais se deu pela análise de presença e o mapeamento dos aspectos encontrados nos trabalhos sobre uso de OSS na EES deu-se pela análise da frequência.

1.5 CONTRIBUIÇÕES

As contribuições desta dissertação são:

- **A descrição das competências técnicas e sociotécnicas necessárias a prática de engenharia de software.** A definição de competências ou habilidades técnicas e sociotécnicas norteiam a avaliação dos objetivos de aprendizagem planejados pelas disciplinas, cursos e atividades de engenharia de software.
- **A descrição dos aspectos limitantes ou estimulantes mais relatados no uso de Projetos FLOSS no ensino de ES.** Através da prévia identificação de

itens que mais colaboram ou limitam com o cotidiano de ensino ou nos resultados deste, os educadores podem analisar suas ocorrências no projeto escolhido e avaliar o grau de interferência de tais itens no desenvolvimento das competências pretendidas pela disciplina ou atividade.

- **A descrição de um processo de avaliação de uso de projetos FLOSS no ensino de ES na forma de um modelo conceitual**. O uso de um modelo conceitual possibilita melhor visualização de seus processos envolvidos e estimula a compreensão de seus elementos. O modelo de avaliação do uso de projetos FLOSS no ensino de ES pode ajudar o professor a identificar tais critérios cotidianamente nas práticas educativas por meio de FLOSS, aprender sobre as limitações e vantagens de cada projeto FLOSS para cada contexto de ensino, e diminuir os riscos de escolhas ineficazes de projetos FLOSS que comprometam os resultados de aprendizagem pretendidos.

É importante lembrar que a avaliação da eficácia de um método de ensino deve ser contínua e deve permitir ajustes e melhorias para garantir que os alunos obtenham o máximo benefício do processo de aprendizagem.

1.6 MOTIVAÇÃO

A pesquisa sobre os efeitos do uso de projetos FLOSS na EES pode promover uma aproximação entre os projetos FLOSS e as práticas da indústria de software, justificando seu uso como potencializador do aprendizado de ES nos seus mais variados aspectos teóricos e práticos (NASCIMENTO; BITTENCOURT; CHAVEZ, 2015). O aprofundamento dos estudos entre a engenharia de software e o uso de projetos FLOSS como fator de ensino e prática pedagógica possibilita uma nova abordagem e maneira de ensinar e aprender, contribuindo para novos estímulos à prática docente e mais estudos que avaliem os benefícios e limitações do uso de FLOSS na educação de engenharia de software (CHAVEZ et al., 2011; SOWMYA; HIRIYANNAIAH; SRINIVASA, 2015; NASCIMENTO; BITTENCOURT; CHAVEZ, 2015; STAMELOS, 2011).

A análise de competências da engenharia de software, e dos aspectos atrelados ao ensino desta por meio de projetos FLOSS, possibilitam aos educadores uma imersão nos objetivos reais de aprendizagem de Engenharia de software e na identificação de problemas e benefícios que prejudicam ou estimulem o alcance desses objetivos, contribuindo para aquisição de experiência prática nas suas decisões pedagógicas(MUSSOI; FLORES; BEHAR, 2010; CAMARGO; FABRI, 2006; STAMELOS, 2011).

Poucos estudos abordam a relação entre o ensino de engenharia de software e o uso de projetos FLOSS sob a perspectiva do educador, sendo vasta a análise dos seus efeitos sob as visões e percepções do estudante. Desse modo, estudar as perspectivas e visões docentes sobre o resultado do processo de aprendizagem se torna pertinente por inserir novos conhecimentos e fundamentos teóricos sobre os efeitos do uso de projetos FLOSS como material de ensino(NASCIMENTO; BITTENCOURT; CHAVEZ, 2015).

Estruturar a análise de competências necessárias e interferências (facilitadores ou barreiras) técnicas e pedagógicas, possibilitam uma nova forma de estudar ou analisar

um projeto FLOSS como material de ensino, contribuindo para que docentes experientes com FLOSS otimizem suas análises e que os docentes inexperientes tenham estímulo e facilidade na inserção desta ferramenta em suas práticas educativas.

1.7 ORGANIZAÇÃO DA DISSERTAÇÃO

Além desse capítulo introdutório, essa dissertação está organizada da seguinte forma:

Capítulo 2 (Revisão Bibliográfica): Na revisão bibliográfica apresenta-se a concepção da pesquisa, ou seja, toda a perspectiva teórica que guiou o estudo por meio de uma revisão bibliográfica que permitiu elucidar e contextualizar a temática envolvida na pesquisa.

Capítulo 3 (Metodologia): Na metodologia apresenta-se a fase de planejamento da pesquisa, ou seja todo o delineamento metodológico do estudo, as decisões metodológicas são descritas e os procedimentos são detalhados de tal forma que todo o processo seja sequencialmente elucidado.

Capítulo 4 (Resultados): Neste capítulo todo o desenvolvimento da pesquisa é demonstrado, todo o processo de coleta e a análise de dados, a transformação para contabilização destes e a contextualização para o alcance dos objetivos propostos.

Capítulo 5 (Considerações Finais): Neste capítulo apresentamos as ameaças à validade, as discussões sobre os trabalhos futuros e a conclusão. O resumo metodológico da dissertação é descrito no ciclo da pesquisa conforme Figura 1.1

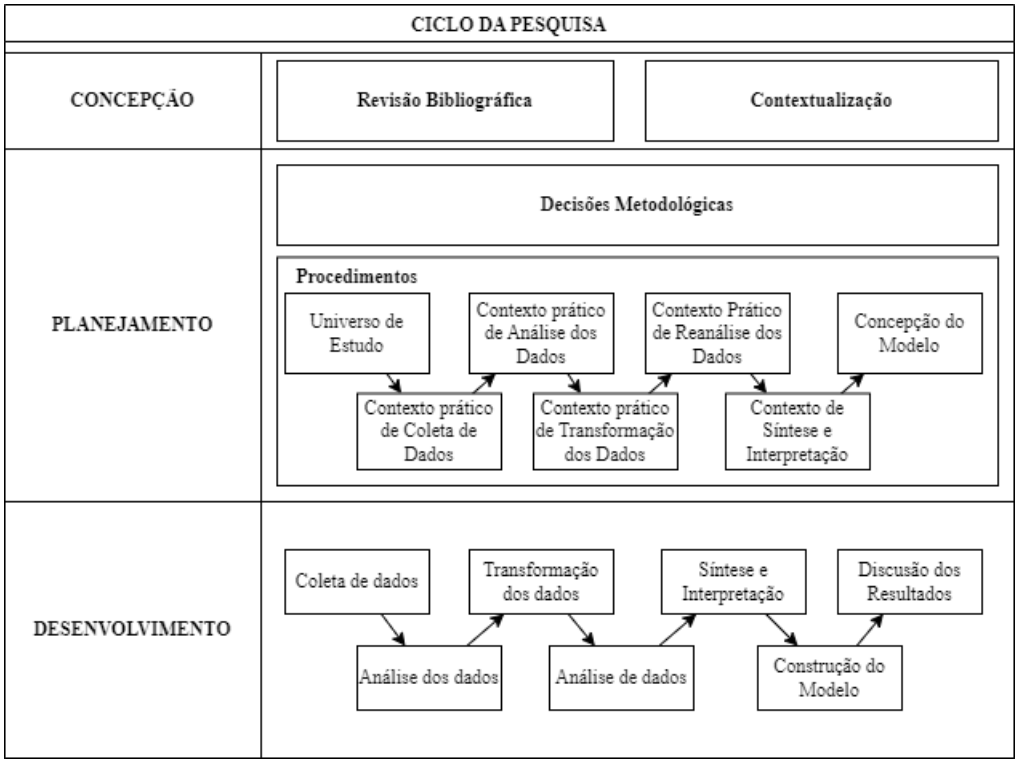

Figura 1.1 Ciclo da Pesquisa
Fonte: Elaboração Própria

Capítulo 2

REVISÃO BIBLIOGRÁFICA

Este Capítulo apresenta a fundamentação teórica para a estrutura conceitual do trabalho. A Seção 2.1 apresenta a necessidade de um ensino de ES voltado ao mundo do trabalho. A Seção 2.2 apresenta os principais guias e referenciais de formação para um currículo efetivo na formação do futuro profissional da Computação. A Seção 2.3 apresenta os principais aspectos que podem interferir no processo de ensino e aprendizagem de ES por meio do uso de FLOSS. A Seção 2.4 apresenta a importância do olhar sobre competências para a formação efetiva do profissional de ES e da Computação em geral. A Seção 2.5 aborda estudos que ajudam compreender a visão de um projeto FLOSS com um olhar voltado a Objetos de Aprendizagem. Finalmente, a Seção 2.6 apresenta referências para fundamentar a avaliação de projetos FLOSS como elemento de auxílio ao Ensino da Engenharia de Software.

2.1 NECESSIDADE DA PRÁTICA NO ENSINO DE ENGENHARIA DE SOFTWARE

A Educação em Engenharia de Software (EES) contribui para aquisição de conhecimentos que ultrapassam técnicas específicas ao desenvolvimento de software na formação dos futuros profissionais da área de Computação (BRITO et al., 2018; NASCIMENTO; BITTENCOURT; CHAVEZ, 2015; CAMARGO; FABRI, 2006).

A Engenharia de Software (ES) agrega competências e habilidades relevantes para a formação adequada do profissional das áreas da Computação. Segundo (ALVES; BENITTI, 2006), a ES tem uma abrangência de conhecimentos que vão além das características tecnicistas. Porém, há desafios associados ao ensino de ES que demandam por soluções que o tornem efetivo na formação do profissional de nível superior.

Um dos aspectos mais desafiadores para o ensino de ES é a inserção de conteúdos, conceitos e práticas de ensino que consigam aproximar o estudante da realidade da indústria de software. Segundo (WANGENHEIM; SILVA, 2009), há algum tempo setores da indústria queixam-se da falta de ensino de competências, em cursos de graduação, para o imediato e eficiente início de trabalho do egresso. Na tentativa de inserir os conteúdos e conhecimentos inerentes à ES da forma mais prática possível, muitas abordagens e

estudos consideram *projetos de desenvolvimento de software* nas práticas de ensino. Há mais de uma década, a Aprendizagem Baseada em Projetos (*Project-Based Learning*) é reconhecida e utilizada para a aplicação prática dos conteúdos e conceitos da ES.

Aspectos que permeiam a aplicação didática de projetos de software em cursos de graduação evidenciam alguns problemas a serem considerados. Tradicionalmente, estudantes trabalham com projetos "*toy*" em cursos de ES e desenvolvem pequenos programas a partir do zero (BUCHTA et al., 2006). Tal abordagem não contempla desafios da prática profissional, que envolve sistemas grandes e complexos, trabalho em equipes heterogêneas, em atividades de manutenção de software e, recentemente, DevOps (FERINO et al., 2023). Por outro lado, deve-se considerar a dificuldade de adequação do uso de projetos grandes, realistas e de longo prazo em um currículo de graduação (NASCIMENTO; BITTENCOURT; CHAVEZ, 2015).

Uma abordagem viável para tornar cursos de ES mais realistas e mais alinhados com as necessidades da indústria de software é envolver os alunos em Projetos Free/Libre Open Source Software (FLOSS) com a supervisão docente (JACCHERI; OSTERLIE, 2007; NASCIMENTO; BITTENCOURT; CHAVEZ, 2015). Segundo (STAMELOS, 2011), as comunidades FLOSS já demonstraram sua capacidade de produção de sistemas e aplicações de software de qualidade. A participação dos alunos em projetos de código aberto realistas os colocam em um ambiente propício a práticas dificilmente alcançadas em projetos menores ou em ambientes acadêmicos e pontuais. Assim, é pertinente conhecer e analisar os aspectos que norteiam o ensino de ES, as aptidões e competências necessárias na formação em suas diversas minúcias, áreas e disciplinas, bem como, os atributos e práticas mais exigidas pela indústria de software comuns a toda forma de atuação profissional. A Seção 2.6 apresenta aspectos relacionados ao uso e avaliação de projetos FLOSS na EES.

2.2 CURRÍCULOS E FORMAÇÃO EM ENGENHARIA DE SOFTWARE

A disciplina de ES pode ser considerada uma das mais relevantes dos cursos da área de Computação. Com a Sociedade moderna cada vez mais dependente de software, a demanda por profissionais que exercitem os métodos e técnicas da ES aumentou, sendo essencial seu ensino nos cursos de Computação (ANDRADE; SANTOS; LINHARES, 2015). Argumenta-se que seu ensino deveria ter destaque nos componentes curriculares de cursos superiores da área (PORTELA; VASCONCELOS; OLIVEIRA, 2015).

Posto isto, há uma busca constante pela aplicação curricular eficiente da ES como requisito formador e, portanto, conhecer suas dimensões e benefícios comuns à formação superior em cursos de Computação torna-se imprescindível.

2.2.1 A Engenharia de Software e as DCN para os cursos superiores de Computação segundo o Conselho Nacional de Educação (CNE)

No Brasil, os cursos superiores da área de Computação e suas áreas afins são definidos e regulados pelo Ministério da Educação (MEC), através da Secretaria de Educação e Regulação da Educação Superior (SERES) vinculada a este ministério, com o objetivo de

fiscalizar o cumprimento das legislações e normatizações na educação superior (BRASIL, 2017. Seção1. p.1).

Os cursos superiores da área de Computação são projetados para atender as demandas do mundo do trabalho no qual se inserem, além de atender aos planos e políticas de importância para os órgãos de regulação. Neste contexto, disciplinas de ES são concebidas e estruturadas de acordo com a proposta do curso de graduação, sendo a Computação uma atividade-meio ou atividade-fim, promovendo a aquisição, por parte dos egressos, de habilidades diversas de gerenciamento, acompanhamento e implementação de sistemas de software (BARBOSA; NELSON, 2015).

Os conteúdos de ES são materializados nos componentes curriculares que constam nos projetos de cursos, que por sua vez são orientados por regulamentações e diretrizes de iniciativa técnica e governamental. Por exemplo, projeto de um Curso Superior em Engenharia de Software proposto por (FIGUEIREDO et al., 2010) é fundamentado pelas diretrizes do CNE além das principais diretrizes curriculares estrangeiras e nacionais de associações em Computação.

Posto isto, a diretriz curricular mais recente proposta pelo CNE para os cursos de graduação na área de Computação do Brasil é a *Resolução Nº 5, de 16 de Novembro de 2016*, que tem como objetivo instituir as diretrizes curriculares abrangendo os cursos de Bacharelado em Ciência da Computação, Sistemas de Informação, Engenharia da Computação, Engenharia de Software e Licenciatura em Computação (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22). Tal documento estrutura os requisitos em diversos princípios relacionados com projeto pedagógico, perfil e competências do profissional formado, nas habilidades comuns a qualquer formação de nível superior, e na definição de formalizações e elementos burocráticos.

De acordo com a referida resolução (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22), os cursos de bacharelado e licenciatura na área de Computação devem garantir a formação de profissionais preocupados com as questões de cunho social e com os impactos que a Computação exerce na sociedade, além de postura crítica, empreendedora e voltada para o trabalho colaborativo, interdisciplinar e sensível às circunstâncias em um mundo de trabalho globalizado. Segundo (WANGENHEIM; SILVA, 2009), é importante a formação de profissionais com habilidades e competências diversas, por exemplo, comunicação interpessoal e trabalho em equipe, sendo o ensino da ES uma maneira de mitigar os problemas tradicionais associados com a indústria de software.

Para todos os cursos superiores da área de Computação, a referida diretriz cita como objetivo que seus egressos tenham habilidades comuns ao projeto e desenvolvimento de software, como a construção de aplicativos e ferramentas de software, a análise e o projeto de sistemas de computação, a criação controlada de software de alta qualidade, o desenvolvimento e gestão de soluções em Tecnologia da Informação e Comunicação (TIC), e a formação de usuários da infraestrutura de software dos computadores nas organizações.

Segundo (SANTOS et al., 2014; LESSA; JUNIOR, 2009), a ES é uma área da Ciência da Computação voltada para o projeto e desenvolvimento de software, justifica-se sua aplicação no processo ensino e formação dos egressos dos cursos da área de computação, no qual se estimula a obtenção de habilidades e competências para o mundo do trabalho e para a excelência profissional. Assim, a instrução adequada de conteúdos de ES em

cursos superiores de Computação pode contribuir para a efetividade dos objetivos propostos pelas Diretrizes Curriculares Nacionais para os cursos de graduação na área da Computação (DCN).

2.2.2 A Engenharia de Software e as Diretrizes Curriculares segundo a Associação para Maquinas de Computação (ACM)

A partir da década de 1970, ao perceber o crescimento e reconhecimento dos cursos de Ciência da Computação (SHACKELFORD et al., 2005; BAILEY; LUNT; ROMNEY, 2006), a Associação para "Máquinas" de Computação (ACM), voltada para o estímulo ao estudo e desenvolvimento da Computação como Ciência e como profissão, reconheceu a necessidade de recomendar elementos comuns aos diversos tipos de cursos de Computação. Em esforço conjunto com a Comunidade de Ciência da Computação do Instituto de Engenheiros Eletricistas e Eletrônicos (IEEE-CS), foram criados os primeiros relatórios para recomendações curriculares dos cursos da área de Computação. Tais recomendações deram a base para a organização dos primeiros currículos de cursos de graduação em Computação nas universidades brasileiras (JONATHAN, 2013; SANTOS et al., 2009a).

O desenvolvimento de recomendações e relatórios para a estrutura curricular por meio de órgãos especializados na área de Computação é importante e positivo, pois estimula a padronização educacional e busca por novos elementos que contribuam para o desenvolvimento da Educação Superior. Neste cenário, o relatório geral Computing Curricula 2005 (CC-2005) (SHACKELFORD et al., 2005, 2006) foi definido, por meio da colaboração entre as principais sociedades de Informática e das principais áreas da Computação.

O relatório geral Computing Curricula (2005) é uma evolução do Relatório Curricula Computing (2001) (CURRICULA, 2001). Foi desenvolvido pela ACM e pela Sociedade de Computação da IEEE (IEEE-CS), com o apoio da Associação de Sistemas de Informação (AIS). A estrutura do relatório está organizada em seções que abordam a evolução das áreas/disciplinas, a descrição de cada área abordada no relatório (Ciência da Computação, Engenharia da Computação, Sistemas de Informação, Tecnologia da Informação e Engenharia de Software), incluindo discussões, mudanças e considerações institucionais que norteiam as evoluções dos programas de graduação (BAILEY; LUNT; ROMNEY, 2006; SHACKELFORD et al., 2005).

O CC-2005 (SHACKELFORD et al., 2005) descreve o histórico das áreas de graduação e seus principais enfoques de abordagem no ensino de Computação. Em especial, destaca a ES como uma área voltada para o desenvolvimento e manutenção de software confiável e eficiente, sendo importante ao profissional da área desenvolver, manter e satisfazer os requisitos necessários aos clientes.

2.2.3 A Engenharia de Software no Curricula Computing 2005.

O CC-2005 apresenta a Engenharia de Software como curso de graduação e como disciplina dentro do currículo da Ciência da Computação. O Relatório mostra que os cursos de graduação em ES e Ciência da Computação têm muitas disciplinas em comum, relacionando a formação dos egressos em Engenharia de Software com a preocupação com confiabilidade e manutenção de software.

A maior parte das sub-áreas da Computação depende do uso de software para suas necessidades, evidenciando a importância da ES não apenas como curso ou disciplina dentro de um currículo, mas como área de conhecimento que trabalhará habilidades técnicas, humanísticas e sociais para tornar os sistemas de software confiáveis e eficientes.

No relatório CC-2005, a importância da ES para os cursos de graduação de Computação é explicitada através da comparação das ênfases dos tópicos de Computação nos cinco tipos de cursos analisados. O relatório mostra que os tópicos comuns de ES – requisitos, modelagem, design, validação, evolução, processo e qualidade de software, são tratados como relevantes para todos os tipos de cursos de graduação analisados, mas com diferentes ênfases em cada tópico de acordo com as necessidades de cada curso.

Há tópicos menos técnicos abordados no relatório com estreita relação com o ecossistema de ES – gerenciamento de risco, gerenciamento de projetos e comunicação interpessoal, que também são relevantes em todos os cursos de graduação analisados.

O Curricula Computing (2005) também compara as expectativas dos egressos dos cursos. Para engenheiros de Computação, indica a importância da ES na capacidade de projetar e implementar sistemas que integrem dispositivos de software e hardware. Para cientistas da Computação, o relatório recomenda que os egressos sejam capazes de preparar um amplo trabalho teórico voltado para o desenvolvimento de software. Para os graduados em sistemas de informação, orienta-se que os egressos sejam capazes de analisar requisitos de informação e seus processos, bem como a especificação e projetos de sistemas alinhados com objetivos organizacionais. Para os profissionais de tecnologia da informação, indica-se a necessidade de preparo para planejar, implementar, configurar e manter uma infraestrutura de computação em uma organização. Finalmente, para os cursos de ES, o CC-2005 recomenda a capacidade para executar e gerenciar as atividades em todas as fases do ciclo de vida de sistemas de software de grande escala.

O relatório (SHACKELFORD et al., 2005) conclui sua análise sobre a relação do currículo às necessidades dos graduados em Computação, afirmando que a ES pode se incluir nesse processo, por exemplo, ao relatar requisitos comuns a diferentes cursos de Computação, as competências e habilidades citadas por (STAMELOS, 2011). Ainda descreve dez requisitos profissionais comuns às cinco áreas analisadas, sendo seis requisitos relacionados com aspectos técnicos e de conhecimento construído no decorrer da graduação: domínio de aspectos essenciais da sua disciplina ou área, base nos conceitos e habilidades de programação de computadores, compreensão dos aspectos limitantes e possibilitadores das tecnologias de computador, compreensão dos conceitos de ciclo de vida, das fases do desenvolvimento de software, além da relação de qualidade e gestão do ciclo de vida de sistemas de computadores.

É importante destacar que, o relatório (SHACKELFORD et al., 2005) inclui a necessidade de que todas as áreas ou disciplinas formem profissionais com base em princípios e tecnologias da ES para garantir robustez e confiabilidade nas implementações de sistemas software apropriados ao público pretendido. Um dos requisitos relacionados é a compreensão do conceito essencial de processo relacionado à Computação e à atividade profissional, bem como o estudo de tópicos avançados de Computação que os ponham em contato com as fronteiras de sua disciplina ou área.

O relatório apresenta quatro requisitos sociotécnicos citados por (STAMELOS, 2011),

relacionados às habilidades de comunicação interpessoal, de gerenciamento e de atividades em equipe que, de acordo com (SHACKELFORD et al., 2005, p.36, tradução nossa), "para ter valor, as experiências de aprendizagem devem construir tais habilidades (não apenas transmitir que são importantes)". Adicionalmente, o relatório geral cita como requisitos sociotécnicos, a exposição a elementos que conectem o aluno à teoria – aprendizado na academia, e a ocorrências do mundo real, com atenção às questões éticas, legais e profissionais, e capacidade de integrar os vários elementos da experiência de graduação.

De forma direta e indireta, o Curricula Computing (2005) indica a importância e a necessidade real do aprendizado em ES para a formação eficaz dos profissionais em todos os programas e cursos analisados.

2.2.4 Engenharia de Software, SWEBOK e os principais guias da série Curricula Computing (ACM)

As recomendações curriculares segundo o CC-2005 geral para os cursos superiores da área de ciência da computação se tornaram referência para o desenvolvimento de guias e demais estudos e relatórios a cerca de cursos e programas de graduação da área.

O relatório Curricula Computing (2005) da ACM orientou o desenvolvimento de guias e diretrizes curriculares específicas para cada área abordada no relatório geral. Nota-se, em todos os guias específicos da série Curricula, uma abordagem às necessidades e habilidades profissionais que se inserem através do ensino de Engenharia de Software. Segundo (DRAFT, 2013, p.172, tradução nossa) "[...] os elementos da engenharia de software são aplicáveis ao desenvolvimento de software em todas as áreas da computação".

Para orientação de programas de graduação em Ciência da Computação foi criado o guia **Currículos de Ciência da Computação da ACM/IEEE (CS-2013)** (DRAFT, 2013). O referido relatório dá destaque à importância da ES para formação dos egressos em ciência da computação ao referenciar sua necessidade para desenvolvimento de sistemas de tempo real, cliente-servidor, distribuídos, paralelos, entre outros, além de orientar disciplinas ou componentes curriculares para o ensino de ES.

Para referenciar os currículos de graduação em Engenharia de Computação surge o guia **Currículos de Engenharia da Computação da ACM/IEEE (CE-2016)** (IMPAGLIAZZO et al., 2016), também baseado no relatório geral da CC-2005, o mesmo indica a ES como elemento de conhecimento e evolução das práticas relacionadas ao projeto e desenvolvimento de aplicativos e sistemas com maior ênfase em ferramentas de design e análises de complexidade.

Os programas para graduação em Sistemas de Informação tem como referência o Guia **Currículos de Sistemas de Informação da ACM/IEEE (SI-2010)**, advindo do Guia Curricular para programas de graduação em sistemas de informação (IS 2002), ambos os relatórios elaborados pela ACM e AIS, identificam a importância de conhecimentos da ES como requisitos aos seus egressos, como o domínio sobre análise e projetos de sistemas e softwares para soluções às organizações, testes e avaliação de eficácia e de qualidade, capacidade de comunicação, trabalho em equipe, colaboração e gerenciamento de riscos (TOPI et al., 2010; GORGONE et al., 2002).

Dando seguimento ao CC-2005 específico para os programas de graduação em tecno-

logia da informação surge o relatório **Currículos de Tecnologia da Informação da ACM/IEEE (TI-2017)**, uma atualização de sua primeira versão de 2008, o guia aborda diversos aspectos curriculares que devem ser analisados para a construção dos currículos dos cursos de Tecnologia da Informação.

O mesmo trata de princípios e aprendizagens necessárias para a boa formação dos profissionais, indicando a necessidade de atrelar soluções práticas de tecnologias para o tratamento de informações pertinentes às organizações, sendo elas software ou hardware. Tais tecnologias devem ser planejadas, gerenciadas e avaliadas de forma a garantir a qualidade do que é esperado por organizações e indivíduos.

Em seu corpo de conhecimento indicativo para as formações das grades curriculares, o relatório mostra os estudos de ES e seus tópicos (design, qualidade, gerenciamento e avaliação), como domínio essencial e complementar para os currículos de tecnologia da informação, dando ênfase também, à competências e habilidades de importância profissional e social, como o trabalho em equipe, ética, inovação, colaboração e relacionamento interpessoal (LUNT et al., 2008; SABIN et al., 2017).

O guia **Currículos de Engenharia de Software da ACM/IEEE (SE-2014)** (ARDIS et al., 2014b) é o mais específico da série *Curricula Computing* na análise do corpo de conhecimento necessário à ES, abordando também princípios sociotécnicos e sua relevância para que profissionais sejam capazes de planejar, desenvolver, acompanhar e avaliar software para garantir qualidade e eficácia (ARDIS et al., 2014b; LEBLANC et al., 2006). O relatório analisa a natureza da ES sob a perspectiva da área de Computação e destaca a sua importância para o desenvolvimento de modelos sistemáticos e técnicos para produção de software de alta qualidade. Sobre a ES como disciplina/área da Engenharia, o relatório orienta um estimulo às técnicas de engenharia no desenvolvimento de software para distinguir as próprias práticas da ES dos estudos de software na Ciência da Computação.

Como relatório norteador para o corpo de conhecimento da ES e suas especificações (ARDIS et al., 2014b), o relatório dá ênfase a importância do Guide to the Software Engineering Body of Knowledge (SWEBOK), o guia para o corpo de conhecimento de ES (BOURQUE; FAIRLEY et al., 2014; BOURQUE et al., 1999) produzido pela IEEE-CS. O SWEBOK organiza o domínio da ES em quinze áreas ou unidades de conhecimento. A parte mais técnica inclui as áreas de configuração, construção e design de software, infraestrutura, gerenciamento e processo de ES, evolução de software, manutenção de software, qualidade de software, requisitos e teste de software, modelos e métodos em ES e Economia de ES. Como versa sobre conhecimentos específicos ao domínio de ES, o SWEBOK também inclui as relações entre a prática profissional e aspectos sociotécnicos quando cita as Ciências Cognitivas e Fatores Humanos como componente relacionado ao ensino de ES.

A partir de todos os relatórios e guias para diretrizes curriculares analisados, bem como o guia de corpo de conhecimento SWEBOK, nota-se a importância da ES para a formação de todos os profissionais da área de Computação, identificando fatores gerais e específicos de inclusão da ES na formação profissional, dando suporte para que a comunidade educacional veja a ES como elemento de estímulo sociotécnico essencial para a eficácia de seus cursos de graduação, além de direcionar os conhecimentos e competências

a serem desenvolvidos na formação específica de cada profissional.

2.2.5 A Engenharia de Software e os Referenciais Curriculares segundo a Sociedade Brasileira de Computação (SBC)

A SBC discute o ensino de Computação no nível superior em diversos eventos, com destaque para o Workshop de Educação em Computação (WEI) (ZORZO et al., 2017). O WEI protagonizou diversas discussões sobre currículos de referência para diversos cursos de graduação no Brasil.

Os Referenciais de Formação para os Cursos de Graduação em Computação (RF), da Sociedade Brasileira de Computação (ZORZO et al., 2017), surgem com o objetivo de auxiliar os coordenadores de curso de graduação na elaboração de projetos pedagógicos. Tendo como base a DCN referente a Resolução n.5 (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22), o relatório aborda os RFs específicos para os cursos de graduação em Ciência da Computação, Engenharia da Computação, Engenharia de Software, Licenciatura em Computação e Sistemas de Informação.

A elaboração dos referenciais considerou a orientação de conteúdos a serem assimilados e direcionados a desenvolver competências esperadas aos egressos dos cursos (ZORZO et al., 2017). A metodologia aplicada no desenvolvimento dos RFs buscou o alinhamento com as Diretrizes Curriculares Nacionais e o embasamento modelado em competências.

2.2.6 Referenciais de Formação para Cursos de Bacharelado em Ciência da Computação (RF-CC).

Os RFs para os cursos de bacharelado em Ciência da Computação (RF-CC) tratam das competências e corpo de conhecimentos necessários à formação dos seus egressos. Tendo sua base pautada nos componentes curriculares da SBC (2005) (ROCHA et al., 2005) e do CS-2013 (DRAFT, 2013), a estrutura curricular aborda diversos tópicos de conhecimento e competências relacionadas aos estudos de ES. Segundo (ZORZO et al., 2017) o perfil dos profissionais formados em ciência da computação necessitam de sólida formação em ciência da computação e matemática, que tenham visão interdisciplinar de sistemas de computação, que sejam reflexivos, criativos e inovadores na análise e construção de sistemas de computação.

Especificando os requisitos e habilidades através de conteúdos, o RF-CC inclui a importância da Engenharia de Software nos seus eixos formadores, enfatizando o desenvolvimento, implantação e projeto de sistemas, indicando como habilidades a identificação, análise, especificação e validação de requisitos de software, incluindo também habilidades com teste, manutenção, avaliação, garantia de qualidade, projeto e planejamento de software, e definindo disciplinas de ES como componente curricular responsável ao desenvolvimento de tais habilidades (ZORZO et al., 2017).

Além das atividades profissionais específicas ao Bacharelado em Ciência da Computação, o RF-CC destaca as competências comportamentais, consideradas em qualquer corpo de conhecimento e a qualquer profissional da área de computação, como noções de ética e direitos na computação, comunicação profissional e comportamento humano. Competências essas de importância reconhecida nas atividades de Engenharia de Software

(ARDIS et al., 2014b; CUKIERMAN; TEIXEIRA; PRIKLADNICKI, 2007).

2.2.7 Referenciais de Formação para Cursos de Bacharelado em Engenharia da Computação (RF-EC).

Os Referenciais de Formação para cursos de bacharelado em Engenharia da Computação (RF-EC) têm como base principal as Diretrizes Curriculares Nacionais (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22), o CE-2016 (IMPAGLIAZZO et al., 2016) e o Currículo de Referência para cursos de Bacharelado em Engenharia da Computação da SBC de 2003 gerado das diretrizes (SBC, 1999).

O conhecimento de ES é indicado no RF-EC como requisito para domínio de habilidades técnicas e sociais voltados ao desenvolvimento de softwares em eixos de conhecimento (ZORZO et al., 2017), como especificação e validação de requisitos, projeto, implementação e verificação de software, implantação e documentação de software.

O RF-EC também cita princípios voltados ao projeto e experimento, como projetos e serviços de software, erros em projetos de software e hardware, análise de fracasso em software e hardware, e análise de riscos em projetos de software e hardware.

Habilidades sociotécnicas referidos como essenciais também são abordados no RF-EC em tópicos, como papéis e comportamento de equipes, papel dos gerentes de projetos, ética, legislação e cidadania (ZORZO et al., 2017). Nota-se que aspectos técnicos e sociais da ES tem destaque na estrutura curricular e objetivos pedagógicos indicados nos referenciais para graduação em engenharia da computação.

2.2.8 Referenciais de Formação para Cursos de Bacharelado em Engenharia de Software (RF-ES).

Os RFs dos cursos de bacharelado em Engenharia da Software (RF-ES) utilizaram como base os trabalhos da comunidade de ES do Brasil (NUNES; YARMAGUTI; NUNES, 2016), que definiram a estrutura e corpo de conhecimento baseado em competências, e as diretrizes descritas por (ARDIS et al., 2014b), (SHACKELFORD et al., 2005) e (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22).

O RF-ES utiliza as DCNs para descrever competências específicas do profissional de ES, como dominar processos e técnicas de construção, evolução e avaliação de software, análise, integração, qualidade, gerenciamento, modelo, problemas e negócios do ecossistema de software. O RF-ES dá atenção a capacidade de compreender e estruturar domínios de aplicação em contextos diversos, considerando a ética, aspectos sociais, aspectos profissionais e econômicos, de forma individual ou em grupo (ZORZO et al., 2017).

Todos os aspectos técnicos e sociotécnicos são descritos, referenciados e aplicados em eixos de conhecimento organizados em disciplinas ou componentes de ensino para cada tópico de conhecimento da ES com base nos relatórios do SWEBOK, SWECOM e o Relatório SE-2014 (BOURQUE; FAIRLEY et al., 2014; ARDIS et al., 2014a, 2014b).

Os autores dos RFs (ZORZO et al., 2017) detalham o processo de refinamento de competências através da taxonomia relatada por (ARDIS et al., 2014b), no qual aplicar métodos e técnicas aprendidas depende de entendê-los que, por sua vez, necessita conhecimento. Essa estrutura mapeia os componentes curriculares dentro dos eixos de

formação para determinar os objetivos de aprendizado.

Em todos os eixos de formação apontados pela RF-ES, organizam-se conteúdos ou disciplinas relacionados com a ES, nos quais destacam-se aquelas com aspectos vistos como sociotécnicos (CUKIERMAN; TEIXEIRA; PRIKLADNICKI, 2007), demonstrando a importância da aplicação prática de modelos, técnicas, conceitos atrelados aos processos que englobam aspectos profissionais, humanísticos e sociais.

Dentro dos eixos normativos, destacam-se componentes de abordagem sociotécnicas como Tomada de Decisão, Pensamento Sistêmico, Empreendedorismo, Direito e suas variações no ecossistema de software, técnicas de comunicação, treinamento e negociação, Formas de gestão, Competências cognitivas, cadeias de valor dentre outras (ZORZO et al., 2017).

A partir dessa referência curricular, percebe-se que a ES não permeia apenas os conceitos, técnicas, e modelos voltados às competências técnicas, mas sim, toma como requisito para a boa formação de seus profissionais, a vivência prática de aprendizado em aspectos profissionais, éticos, sociais e humanos.

2.2.9 Referenciais de Formação para Cursos de Licenciatura em Computação (RF-LC).

Para os cursos de Licenciatura em Computação, o RF-LC (ZORZO et al., 2017) define suas diretrizes com base nas DCNs de 2016 e nas Diretrizes Curriculares para a formação inicial de professores (MEC, 2015. Seção 1. p. 8) e estrutura todos os seus eixos, competências e conteúdos voltados para as práticas de ensino.

A Licenciatura em Computação é o programa mais próximo dos princípios de aplicação de teoria à prática com ênfase aos aspectos sociais para que seus profissionais sejam bem formados e capazes de prover formação técnica, social e humanística (ZORZO et al., 2017; SBC, 1999; MEC, 2015. Seção 1. p. 8).

O RF-LC engloba vários conteúdos, competências e habilidades desenvolvidos na aprendizagem de ES. Os eixos voltados para a aprendizagem das diversas áreas a serem lecionadas pelo educador a ser formado incluem diretamente estudos de ES, bem como disciplinas que estão relacionadas com suas práticas como a avaliação, gestão e interação de software educacional, além de verificação, análise, especificação e validação de sistemas (MEC, 2015. Seção 1. p. 8).

Vários componentes e competências de viés técnico-social são definidos neste referencial, como a ética, acessibilidade, comunicação oral e escrita, comunicação em espaços públicos de rede, gestão do trabalho, desenvolvimento e estímulo à autonomia e ao trabalho coletivo, comunicação com clareza, gestão e avaliação de recursos educacionais, atitude ética, crítica e reflexiva.

A ES inclusa no processo de ensino-aprendizagem do educador o possibilita estimular o Magistério e sua compreensão além da regência de classe (ARDIS et al., 2014b; MATOS, 2013).

2.2.10 Referenciais de Formação para Cursos de Bacharelado em Sistemas de Informação (RF-SI).

O Referencial de Formação para os cursos de Bacharelado em Sistemas de Informação (RF-SI) (ZORZO et al., 2017) foi estruturado para definir competências e habilidades a serem alcançadas pelos egressos dos cursos de Bacharelado em Sistemas de Informação, e eixos formadores com conteúdos voltados a esse objetivo. Embasado no currículo da SBC (SBC, 1999) e nas DCNs (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22), o RF-SI utilizou como metodologia para seu desenvolvimento a estrutura baseada em formação por competências e seu mapeamento em uma estrutura conceitual. Já nos aspectos de formação profissional em Sistemas de Informação o documento define como indissociáveis os problemas técnicos e comportamentais, dando ênfase a abordagem sociotécnica e a resolução de problemas do mundo real (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22).

Dentro desse contexto, objetiva-se que as competências sejam estimuladas em componentes curriculares voltados a aplicação da teoria à prática técnica e atitudinal. A ES tem presença como requisito educacional através de competências e conteúdos a serem desenvolvidos em seus eixos formativos.

Vários componentes curriculares do RF-SI adotam as práticas de ES como meio de alcance para as competências necessárias ao profissional de sistemas de informação. Tópicos como análise, especificação, validação, qualidade, projeto, requisitos e avaliação de softwares e sistemas de informação adotam práticas da ES.

Competências sociotécnicas ganham destaque em tópicos voltados aos estudos da ética nas organizações, comportamento organizacional, inovação e seus processos, cultura, ética e política no uso de SI, legislação, gerência de equipes entre outras (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22; MEC, 2015. Seção 1. p. 8).

A partir da análise de cada RF, nota-se que o Relatório RF (2017) adota a estrutura curricular voltada para aquisição de competências relacionando o desenvolvimento técnico e sociotécnico, mapeando conteúdos e fundamentos que terão como objetivo gerar as competências e habilidades profissionais necessárias.

Vários tópicos e conteúdos abordados nos RFs, definem princípios específicos da ES, bem como princípios sociais necessários a qualquer profissional e altamente recomendados para a real aprendizagem de ES.

Partindo por esse princípio, a ES e seu ensino efetivo podem contribuir para satisfazer os objetivos de aprendizagem, tanto de viés técnicos específicos como viés sociotécnicos e comportamentais, tornando o ensino de ES uma prática determinante para a boa formação dos profissionais de qualquer área da Computação.

2.3 PROJETOS FLOSS E FATORES DE INTERFERÊNCIA NO ENSINO DE ENGENHARIA DE SOFTWARE

O ensino de ES necessita de abordagens e técnicas que busquem sua otimização e facilitem o alcance de seus objetivos. Como área de conhecimento, a ES é vista como elemento importante na qualificação de todos os profissionais na área de Computação (SHACKELFORD et al., 2005; SANTOS et al., 2014, 2008). Assim, faz-se necessário pesquisar por

melhores práticas de ensino da ES para alcançar os objetivos de aprendizagem propostos na EES.

A busca por objetos ou materiais de aprendizagem que promovam a aquisição de conhecimento e sua aplicação prática no meio profissional deve ser atividade cotidiana do educador. Segundo (MUSSOI; FLORES; BEHAR, 2010, p.123), todo recurso digital utilizado para dar suporte à aprendizagem, estimulando a comunicação para a instrução de forma reutilizável, é um Objeto de Aprendizagem (OA).

Projetos FLOSS mostram-se como solução viável para contornar alguns problemas no ensino de ES (STAMELOS, 2011), em especial, a dificuldade de relacionar o aprendizado de conceitos e técnicas da ES com sua prática efetiva (NASCIMENTO, 2017; BUCHTA et al., 2006; BEGOSSO et al., 2011). O uso de Projetos FLOSS no ensino de ES faz com que eles assumam características típicas de um objeto de aprendizagem e, assim, devem ser tratados pelo docente como ferramenta técnica e pedagógica. Projetos FLOSS podem trazer um melhor alinhamento entre o ensino na universidade e as práticas da indústria, estimulando educadores a planejar sua adoção nas mais diversas práticas educativas.

Por outro lado, o uso de projetos FLOSS pode facilitar ou dificultar o processo de ensino-aprendizagem, e portanto, tais projetos devem ser selecionados de forma a mitigar possíveis fatores prejudiciais e aproveitar os fatores benéficos para a proposta educativa (REATEGUI; FINCO, 2010; STAMELOS, 2011).

Projetos FLOSS podem beneficiar a dinâmica de aprendizado conceitual, prático e realístico nas disciplinas de ES. As disposições da estrutura do código e do projeto de forma real em uma comunidade de desenvolvimento de software garantem ao educando ambientação com grandes e médios projetos. A interação e a colaboração no desenvolvimento com diversos agentes, muitas vezes profissionais experientes, inserem o educando no ecossistema heterogêneo de desenvolvimento comuns aos projetos de grande porte. A necessidade de conhecer o contexto, o objetivo, e o estado atual do projeto, estimula uma visão ampla do mesmo de forma prática.

Tais práticas estimulam a aprendizagem ativa e garantem o engajamento dos estudantes na rotina de aprendizado (NASCIMENTO; BITTENCOURT; CHAVEZ, 2015; FILHO; FF, 2017). Por exemplo, um estudo executado por (BARBOSA; NELSON, 2015) mostrou que a maioria dos estudantes envolvidos em atividades centradas na comunicação opinaram que a atividade cooperativa contribuiu para o aprendizado prático das técnicas de ES.

O desenvolvimento colaborativo real de Projetos FLOSS pode ser benéfico para o aprendizado. Um outro estudo realizado por (STAMELOS, 2011) identificou satisfação na prática de ensino de ES com projetos FLOSS mediado pelo educador, onde o contato e a colaboração entre educador e estudantes estimulou o aprendizado.

Mesmo podendo ser tratado como OA eficiente, o uso de projetos FLOSS no ensino da ES enfrenta desafios. Alguns problemas podem ser vivenciados pelos educadores e a análise prévia destes podem mitigar possíveis interferências no objetivo de aprendizagem (STAMELOS, 2011; NASCIMENTO; BITTENCOURT; CHAVEZ, 2015; JACCHERI; OSTERLIE, 2007).

- A dificuldade de acesso aos projetos e comunidades FLOSS, aspectos culturais e

personalidade dos estudantes podem interferir no andamento das atividades.

- A heterogeneidade de domínio de linguagem dos estudantes podem interferir nas suas efetivas participações nas atividades executadas no FLOSS.

- Problemas comportamentais de agentes participantes em comunidades FLOSS podem intimidar o engajamento dos estudantes.

- O tamanho e a complexidade do projeto podem não ser adequados para o tempo de execução da disciplina dentre outros fatores limitantes.

Por outro lado, projetos FLOSS podem ter benefícios que ultrapassam as demandas socio-técnicas do ensino de ES, sendo estimulante para abordagens técnicas específicas em outros tópicos e áreas. Por exemplo:

- Projetos FLOSS podem estimular a participação dos estudantes em pesquisas de interesse da universidade ou do curso de graduação. Alguns destes projetos são desenvolvidos em iniciativas da própria universidade ou de institutos de pesquisa e o engajamento dos estudantes é estimulado na contextualização de tais projetos em componentes curriculares.

- Muitos projetos FLOSS podem demandar soluções específicas a determinado tópico da ES, no qual a determinação de tópicos mais relevantes para o contexto da disciplina pode contribuir para o estudo destes de forma mais eficiente.

- Alguns projetos FLOSS podem estimular o aprendizado de metodologias de desenvolvimento compatíveis com o tempo e a proposta da disciplina dentre outros benefícios (ROCHA; SABINO; ACIPRESTE, 2015; BARBOSA; NELSON, 2015; PORTELA; VASCONCELOS; OLIVEIRA, 2015).

Enfim, do ponto de vista pedagógico, o uso de projetos FLOSS como OA tende a ser benéfico no ensino de ES, trazendo oportunidades de trabalhar o lado socio-técnico e multidisciplinar dos participantes. Entretanto, é necessário que o educador possa reconhecer e avaliar as características importantes nos Projetos FLOSS escolhidos para que estas possam atestar a qualidade e eficácia de uso para o objetivo pedagógico (REATEGUI; FINCO, 2010; STAMELOS, 2011; SAVI; WANGENHEIM; BORGATTO, 2011; FILHO; FF, 2017).

2.4 MAPEAMENTO DE COMPETÊNCIAS PARA O ENSINO DE ENGENHARIA DE SOFTWARE

O aprendizado efetivo de ES e suas especialidades passam por um processo de desenvolvimento de competências e habilidades que combinam o conhecimento técnico com as habilidades sociotécnicas (ARDIS et al., 2014b; CUKIERMAN; TEIXEIRA; PRIKLADNICKI, 2007; TEIXEIRA; CUKIERMAN, 2005).

A visão antiga da ES como área puramente técnica foi paulatinamente substituída por uma abordagem social da ES, ou seja, uma visão de ES preocupada com suas interferências na sociedade, desde sua prática profissional até os produtos oriundos desta. Segundo (PRESSMAN; MAXIM, 2016) os aspectos humanos são igualmente importantes ao conhecimento técnico na prática de ES. Por isso, não é interessante determinar como requisitos para boa formação em ES, o conhecimento teórico e técnico dos tópicos da ES sem sua aplicação prática próxima à realidade profissional que envolva, além de técnicas de engenharia e computação, processos com estreita relação aos aspectos humanos.

Dito isto, vários referenciais curriculares determinaram como competências a serem alcançadas pelos estudantes, o *conhecimento técnico das áreas específicas da ES atreladas a habilidades descritas como sociotécnicas*. Porém, a descrição e o mapeamento específico de atitudes que mostrem tais habilidades na prática profissional não são definidos de forma específica nos referenciais.

2.4.1 O Modelo de Competências em Engenharia de Software (SWECOM)

Para suprir a demanda por definições específicas de competências técnicas e sociotécnicas, o IEEE-CS desenvolveu um Modelo de Competências em Engenharia de Software (SWECOM) (ARDIS et al., 2014a). Esse modelo especifica competências como *um processo de conhecimento de conteúdos para posterior aplicação de forma efetiva, determinando o que o profissional deve ser capaz de fazer para ser considerado habilidoso*.

O Modelo SWECOM foi desenvolvido com base nos indicativos do guia SWEBOK, nas normas ISO/IEEE 12207 e ISO/IEEE 15288 relacionadas aos processos de engenharia de sistemas e de software (COMMISSION et al., 2008; STANDARDIZATION, 2008), além dos corpos de conhecimento de engenharia de sistemas e do currículo de ES ACM/IEEE-CS (LEBLANC et al., 2006). Para validar o modelo SWECOM e sua relevância, seus autores fizeram entrevistas e avaliações com revisores e engenheiros de software.

O Modelo SWECOM é organizado do ponto de vista comportamental, acadêmico e de ambiente profissional. O SWECOM também usa cinco elementos para definir *graus* de habilidades. As *habilidades cognitivas* (que se aplicam a todas as áreas e demais habilidades) referem-se à capacidade do profissional aplicar o conhecimento e raciocínio enquanto realiza as demais atividades do SWECOM. São habilidades cognitivas: os raciocínios indutivo, dedutivo e heurístico, habilidades de análise e manipulação de dados, além daquelas relacionadas à resolução de problemas e inovação.

Segundo (ARDIS et al., 2014a), os graus de *habilidades comportamentais* tratam da aplicação de produtividade, cognição e habilidades técnicas para otimizar os resultados destas. São atributos de habilidades comportamentais: aptidão, iniciativa, ética no trabalho, vontade, confiabilidade, sensibilidade cultural, postura comunicativa, participação em equipe e liderança técnica.

O foco principal do SWECOM é o mapeamento de *competências técnicas*, agrupadas em *habilidades em áreas do ciclo de vida do software* e *habilidades em áreas transversais do desenvolvimento de software*. O primeiro grupo do SWECOM, o de habilidades técnicas em áreas do ciclo de vida de software, relacionou as mesmas com áreas de conhecimento do SWEBOK (BOURQUE; FAIRLEY et al., 2014; ARDIS et al., 2014a). O

segundo grupo de habilidades técnicas da SWECOM trata das *habilidades transversais* que incluem áreas mais especializadas, como Segurança, Medição, Gerência e Interação Humano-Computador.

O Modelo SWECOM faz o mapeamento de habilidades de acordo com cinco *níveis de competência*: *técnico, praticante de nível de entrada, praticante, líder técnico e engenheiro de software sênior*. Para identificar o nível de competência de um profissional em cada área da ES, o modelo define atividades práticas que o profissional deve ser capaz de executar. Por exemplo, para a área de Requisitos de Software, o profissional no nível técnico consegue seguir os procedimentos definidos para dar suporte ao gerenciamento de requisitos, enquanto que o profissional no nível de competência de engenheiro de software sênior exige capacidade para modificar ou criar métodos, diretrizes, modelos e ferramentas para o gerenciamento de requisitos (ARDIS et al., 2014a).

Ao mapear habilidades dentro de áreas do corpo de conhecimento, o modelo mostra-se útil para a concepção curricular envolvendo o SWEBOK e os referenciais curriculares dos cursos de Computação. O SWECOM também pode ser a base para diversas atividades de pesquisas e discussões sobre como mapear competências e relacioná-las com perfis profissionais e atividades necessárias para tal.

2.4.2 Outras práticas para o Mapeamento de Competências na EES.

Um mapeamento de competências de ES foi realizado em um projeto-piloto de desenvolvimento de software na universidade, a Fábrica Acadêmica de Software - FAS (CAMARGO; FABRI, 2006), para avaliar se estudantes de um curso de Análise de Sistemas e Tecnologias da Informação atenderam às exigências do projeto. Os autores compararam competências adquiridas após o curso com as necessárias ao Projeto FAS, e com as propostas pelo SWEBOK, com objetivo de investigar a viabilidade do FAS ser usado na avaliação curricular de cursos da área.

O trabalho de (STAMELOS, 2011) discute oportunidades de pesquisa relacionadas ao uso de projetos FLOSS para ensinar ES, sendo uma das principais referências de aplicação pedagógica e técnica de FLOSS como objeto de aprendizagem em ES. Seu principal objetivo foi identificar o desenvolvimento de competências por meio do mapeamento de atividades da disciplina para projetos FLOSS do mundo real, alinhados com os requisitos de ensino de ES proposto em (BOURQUE; FAIRLEY et al., 2014). O autor definiu um esquema básico para o ensino de tópicos da ES com enfoque em aplicar conceitos em situações realistas extraídas de projetos FLOSS. (STAMELOS, 2011) também identificou possíveis atividades que o educador pode executar para desenvolver a competência prática nos tópicos da ES segundo o SWEBOK. Por fim, definiu como requisito para atividades de ensino com FLOSS, a adequação às exigências sociotécnicas estabelecidas pelo Curricula Computing (2005) (SHACKELFORD et al., 2005).

2.5 AVALIAÇÃO DE OBJETOS DE APRENDIZAGEM

O processo de ensino e aprendizagem sempre demanda por novos métodos, técnicas, objetos e tecnologias que otimizem os objetivos educacionais planejados. Dentro des-

sas tecnologias estão incluídos os *Objetos de Aprendizagem* (OAs). Segundo (BRAGA, 2014; IEEE, 2019), toda entidade, digital ou não, usada, reutilizada ou referenciada na aprendizagem, como imagens, vídeos, software e animações, são consideradas Objetos de Aprendizagem. O uso de projetos FLOSS como objeto de ensino de ES pode ser percebido como um OA, sendo relevante analisá-lo também por essa perspectiva.

O trabalho de (MUSSOI; FLORES; BEHAR, 2010) apresentou e discutiu aspectos básicos, formas e métodos de avaliação de *qualquer objeto de aprendizagem* com o objetivo de propor critérios para avaliação de um OA. Os autores defendem a necessidade de clareza na concepção epistemológica, características e objetivos.

Os critérios propostos partiram de algumas definições e características. A estrutura e organização dos critérios propostas no estudo se materializam em uma tabela de avaliação. As características comuns aos OA, segundo os autores, são a flexibilidade, facilidade de atualização, customização, interoperabilidade, aumento do valor de um conhecimento, indexação e procura. Os critérios e parâmetros de avaliação de um OA definidos no estudo foram a *usabilidade técnica* e a *usabilidade pedagógica*. Um OA foi usado em uma atividade a prática e avaliado por meio do mapeamento nos critérios organizados na tabela de avaliação. Na avaliação do OA, os autores consideraram suas características, os conteúdos abordados e seus objetivos. Na conclusão do trabalho afirmaram que, além de planejamento prévio e avaliação, os OAs devem ser validados junto ao seu público alvo, por meio de um *feedback* deste público sobre aspectos técnicos e pedagógicos.

O trabalho de (REATEGUI; FINCO, 2010) sobre avaliação de Objetos de Aprendizagem segue a linha do trabalho de (MUSSOI; FLORES; BEHAR, 2010) e recomenda uma abordagem epistemológica para alinhar o uso de um OA às práticas pedagógicas que se quer implantar. Os autores propõem a adoção de *critérios de avaliação* baseados em *aspectos técnicos e pedagógicos*.

As diretrizes de avaliação pedagógicas têm como base o acompanhamento efetivo do processo de aprendizagem, a esquematização de problemas, recursos educacionais, atividades de apoio a colaboração, proatividade nas atividades, suporte a comunicação, teste e acompanhamento das atividades entre outras. As diretrizes de avaliação técnica visam analisar se o OA é robusto, portável, emprega imagens, apresenta informações, permite orientação e navegação, é interativo, é estético e afetivo. As diretrizes devem auxiliar o processo de avaliação do OA escolhido pelo educador, sendo importante para subsidiar posteriores decisões nas escolhas e na adoção de determinados objetos na aprendizagem. Por fim, os autores descrevem a importância de investigar como o educador pode selecionar artefatos digitais alinhados com suas práticas pedagógicas.

2.6 AVALIAÇÃO DE PROJETOS FLOSS NO ENSINO DE ES

Avaliar o uso de um projeto FLOSS no ensino de uma disciplina de ES envolve reconhecer seus aspectos positivos e limitantes, suas características técnicas e seu estímulo ao desenvolvimento de competências e habilidades técnicas e socio-técnicas da ES, para que o aprendizado seja estimulado e os princípios da ES, bem como as habilidades sociotécnicas atreladas a ela, sejam assimilados de forma efetiva. A avaliação deve considerar critérios e aspectos pedagógicos e técnicos, além daqueles definidos nos corpos de conhecimento

e referenciais curriculares para a ES. Uma vez escolhido um projeto FLOSS, é necessário (i) avaliar se o mesmo atenderá os objetivos de ensino propostos pelo educador, e (ii) analisar a efetividade de tal projeto na aquisição das competências definidas.

O trabalho de (CERONE; SOWE, 2010) apresenta e discute os resultados de dois estudos piloto para avaliação da eficácia de projetos FLOSS no ensino de ES. O trabalho identifica duas características importantes para a aprendizagem em comunidades FLOSS: a *rede de colaboração* e a *comunidade de prática*. Os autores analisam a aprendizagem individual em comunidades FLOSS, identificando diversos fatores positivos na participação de estudantes. Além disso, definem *fases para o processo de aprendizagem em comunidades FLOSS*: socialização, externalização, combinação e internalização. Por fim, (CERONE; SOWE, 2010) argumentam em favor de *colocar a participação em projetos FLOSS como um dos pré-requisitos para a disciplina de ES*.

Primeiro Estudo-piloto. O primeiro estudo foi o trabalho de (SOWE; STAMELOS, 2007) que investigou o uso de projetos FLOSS em disciplinas de ES, considerando um currículo formal de curso de graduação. Durante o estudo, considerou três fases de participação dos estudantes: (i) apresentação de tópicos relacionados a FLOSS e seleção de um dos projetos, (ii) participação no projeto selecionado para encontrar, reportar e resolver *bugs*, e (iii) avaliação das atividades pelos professores. Observou-se que alguns estudantes sentiram-se motivados para continuar a participação nos projetos FLOSS selecionados. O trabalho também mostra uma estrutura base que serve para educadores adequarem o uso de projetos FLOSS no currículo de ES.

A avaliação dos estudantes teve como base uma apresentação de cada um sobre sua participação no FLOSS, com detalhes do projeto escolhido, suas interações e participações na comunidade. Por fim, o estudo identificou aspectos positivos e negativos na experiência de uso de projetos FLOSS no ensino e aprendizagem da disciplina.

Segundo Estudo-piloto. Esse estudo avaliou a prática de ensino de ES com uso de projetos FLOSS em disciplinas de pós-graduação (JACCHERI; OSTERLIE, 2007). As atividades que os estudantes deveriam realizar foram organizadas em tarefas de desenvolvimento de FLOSS e tarefas de identificação e definição de problemas de pesquisa.

No estudo de (CERONE; SOWE, 2010), dois aspectos considerados relevantes são o *grau de liberdade na escolha de projetos e atividades* e o *enfoque/motivação* do estudante. No primeiro estudo-piloto, os estudantes tinham total liberdade para escolher e realizar as atividades necessárias. No segundo estudo-piloto, os estudantes escolheram as questões de pesquisa, e as mesmas direcionaram ou restringiram a seleção de projetos. A escolha livre dos projetos pelos estudantes estimula sua motivação nas atividades e assegura os requisitos de aprendizagem em sala de aula. Entretanto, deve-se estar atento para fatores que possam prejudicar sua participação no projeto e até prejudicar a comunidade do projeto FLOSS. Os autores propõem soluções, tais como, regras de participação previamente definidas e limitação de atividades a serem executadas.

O trabalho de (LESSA; CHAVEZ, 2020) visou apresentar uma abordagem de seleção de Projetos baseados em critérios técnicos e sociais. Essa abordagem posteriormente foi avaliada segundo a visão de estudantes. A abordagem foi organizada em elucidação de critérios de seleção de Projetos FLOSS para a EES segundo revisão de literatura. Os critérios de seleção foram documentados e operacionalizados para permitir busca de

Projetos em repositórios baseados em tais critérios. Tal operacionalização permitiu o desenvolvimento de uma ferramenta de seleção dos projetos FLOSS embasados nos critérios analisados. A ferramenta FlossSeach.edu foi usada por alunos em um estudo de caso que visava avaliar sua facilidade de uso e utilidade. Os critérios são baseados em aspectos como tamanho do projeto, domínio, Linguagem de Programação, Tamanho da comunidade e se a mesma é ativa, aceitação de contribuições entre outros. A ferramenta foi indicada como útil e fácil de usar. Os critérios analisados na busca e seleção se mostram pertinentes para o professor avaliar os aspectos de escolha de um projeto FLOSS para ensinar Engenharia de Software e para avaliar a eficácia do projeto selecionado para os objetivos educacionais esperados.

Capítulo 3

METODOLOGIA

Este Capítulo apresenta todos os procedimentos metodológicos para o desenvolvimento da pesquisa. A Seção 3.1 apresenta os aspectos metodológicos gerais e a Seção 3.2 apresenta o planejamento e descrição das atividades desenvolvidas no Ciclo da Pesquisa do estudo.

3.1 CLASSIFICAÇÃO DA PESQUISA

Esta é uma pesquisa descritiva e exploratória, com Abordagem Mista (GIL et al., 2002). Descritiva porque objetiva identificar e explicar características de competências e habilidades associadas com a Educação em Engenharia de Software (EES), bem como aspectos que exercem influência no ensino de Engenharia de Software (ES) por meio de Projetos Free/Libre Open Source Software (FLOSS). Exploratória porque tem como objetivo investigar relações implícitas entre objetivos de aprendizagem, competências e aspectos que podem influenciar em tais objetivos.

Neste trabalho, optou-se pela abordagem Mista, pois compreende-se que esta tende a embasar as alegações de conhecimento em elementos pragmáticos, empregando estratégias de investigação com coletas de dados de forma simultânea ou sequencial, obtendo tanto informações de texto como informações numéricas (CRESWELL, 2007).

As competências e habilidades técnicas ou sociotécnicas contextualizadas com objetivos de formação superior em computação podem ser vistas ou descritas de formas variadas de acordo com normas ou referenciais formativos diversos (CUKIERMAN; TEIXEIRA; PRIKLADNICKI, 2007; ZORZO et al., 2017; SHACKELFORD et al., 2005).

Desta forma, a descrição dos aspectos que influenciam no desenvolvimento de competências ou habilidades como objetivos de aprendizagem em Engenharia de Software podem variar de acordo com as perspectivas dos docentes, pesquisadores e recomendações curriculares que os abordam.

Tendo em vista essas observações, compreende-se que o aprofundamento do estudo sobre os elementos citados não se alcança por métodos apenas interpretativos ou quantificáveis, sendo necessária uma análise mais minuciosa dos dados obtidos. Posto isto, o objetivo principal que norteia a pesquisa é:

Compreender Competências associadas a Engenharia de Software e Aspectos de Influência mais citados em estudos sobre ensino de Engenharia de Software auxiliado por projetos FLOSS para propor um Modelo para Avaliar a Eficácia do uso de Projetos FLOSS no Ensino de Engenharia de Software.

Por possibilitar uma estratégia de coleta e análise de dados tanto qualitativos como quantitativos, o Método Misto aproveita os pontos fortes de ambas abordagens, na qual o pesquisador tem a liberdade de adotar a estratégia que mais se alinha com os objetivos e questões de pesquisa (CRESWELL, 2007).

A estratégia de Método Misto que mais se alinha com os objetivos desta pesquisa é a Estratégia Transformadora Concomitante, pois esta "[...]possui uma perspectiva teórica norteadora do estudo cujo objetivo se sobrepõe ao uso dos métodos e ocorre com a coleta concomitante dos dados" (DAL-FARRA; LOPES, 2013, p.76).

Algumas características do Modelo Aninhado Concomitante estão presentes nesta estratégia, pois este Modelo adota um método predominante que guia o projeto, sendo ele qualitativo ou quantitativo, no qual o método com menor prioridade está embutido ou aninhado no método predominante (CRESWELL, 2007).

No caso desta pesquisa, o foco de análise dos dados se dará na transformação de dados qualitativos em dados quantitativos, pois o processo de análise se dará na identificação de termos ou temas mapeados nos textos para posterior classificação, quantificação e refinamento. Todos esses aspectos abordados mostram a relevância do uso do Método Misto como abordagem escolhida para esse estudo.

A Epistemologia Pragmática é a Alegação de Conhecimento que guiou esta pesquisa. Segundo (CRESWELL, 2007, p. 29) a Epistemologia Pragmática preconiza que "Os pesquisadores têm liberdade de escolha. Eles são "livres" para escolher métodos, técnicas e procedimentos de pesquisa que melhor se ajustem a suas necessidades e a seus objetivos".

Tal visão se torna relevante quando a busca pelas respostas aos problemas de pesquisa depende do rastreio e compreensão de fatores ou fenômenos que estão registrados ou estão implícitos em variados elementos textuais, como é caso da pesquisa em questão. O Pragmatismo é voltado para a consequência, pois observa as práticas de construção do conhecimento sob a visão de resultados práticos que produz (FERREIRA et al., 2020).

Neste estudo, a Análise Bibliográfica é o procedimento técnico predominante. É um tipo de pesquisa desenvolvida em material elaborado que objetiva investigar as principais contribuições teóricas sobre um tema (GIL et al., 2002; SOUZA; DIESEL, 2008). O processo de investigação sobre competências e habilidades inerentes a Educação em Engenharia de Software demanda uma análise em diversificados relatórios ou materiais que abordem o tema.

Além disso, investigar, identificar e conhecer os aspectos técnicos ou pedagógicos que mais influenciam objetivos de aprendizagem de ES apoiados por Projetos FLOSS demanda uma análise em diversos estudos que abordem ou que descrevem o uso de FLOSS como ferramenta de ensino de ES ou suas subáreas. Posto isso, o uso de Análise Bibliográfica mostra-se pertinente por reunir informações sob diferentes perspectivas, contextos, períodos e localidades.

Segundo (GIL et al., 2002, p. 45) "A principal vantagem da pesquisa bibliográfica

reside no fato de permitir ao investigador a cobertura de uma gama de fenômenos muito mais ampla do que aquela que poderia pesquisar diretamente". Essa afirmação é relevante, pois a variedade dos conceitos que abordam Competências e Aspectos de Interferência Técnico-Pedagógica em Engenharia de Software exige uma compreensão sob diferentes pontos de vista, nos quais provavelmente não poderiam ser observados em estudos diretos, justificando o uso da Análise Bibliográfica como Metodologia escolhida.

Nesta pesquisa, a técnica de análise e produção de dados utilizada será a Análise de Conteúdo. A Análise de Conteúdo é uma técnica que tem como objetivo investigar as características de conteúdos manifestos e suas relações visando contribuir com a percepção do teor de uma comunicação. Por meio de mensagens e seus fragmentos, busca-se a relação entre conteúdos, suas características implícitas e explícitas, a fim de investigar modelos ou padrões (CÂMARA, 2013). Para (CÂMARA, 2013, p.182):

> O esforço do analista é, então, duplo: entender o sentido da comunicação, como se fosse o receptor normal, e, principalmente, desviar o olhar, buscando outra significação,outra mensagem, passível de se enxergar por meio ou ao lado da primeira.

Tal compreensão do objetivo de que tange a Análise de Conteúdo mostra a importância desta técnica como método de investigação e extração dos dados para esta pesquisa. Nesta pesquisa, por meio dos vários documentos e recomendações que abordam as competências esperadas na formação de egressos de cursos superiores em Computação, é possível identificar, analisar, organizar e estruturar relações entre as mensagens e as comunicações que tais documentos mostram ou deixam implícito.

A Análise de Conteúdo também se mostra interessante como técnica para identificar mensagens e as relações entre estas contidas em trabalhos que abordam o uso de Projetos FLOSS na EES, identificando características, padrões e contextualizações diversas sobre termos, mensagens, impressões e visões que os pesquisadores ou participantes da pesquisa tiveram sobre os aspectos que exercem algum tipo de influência sobre a aprendizagem e o alcance das competências objetivadas pelo professor de ES. Sobre a importância desses objetivos de investigação (BARDIN, 2011, p.35) afirma:

> A superação da incerteza: o que eu julgo ver na mensagem estará lá efetivamente contido, podendo esta "visão" muito pessoal ser partilhada por outros? Por outras palavras, será a minha leitura válida e generalizável?
>
> E o enriquecimento da leitura: se um olhar imediato, espontâneo, é já fecundo, não poderá uma leitura atenta aumentar a produtividade e a pertinência? Pela descoberta de conteúdos e de estruturas que confirmam (ou infirmam) o que se procura demonstrar a propósito das mensagens, ou pelo esclarecimento de elementos de significações suscetíveis de conduzir a uma descrição de mecanismos de que a priori não possuíamos a compreensão.

Bardin corrobora com a compreensão da Análise de Conteúdo como uma técnica que tende ao objetivo de investigar o teor das comunicações de forma rigorosa e desbravadora, aspecto imprescindível para a os objetivos desta pesquisa.

Sobre as formas de comunicação contidas no estudo Bibliográfico aplicado a esta pesquisa, o acervo analisado conta com diversos tipos de informações, como quadros, tabelas, instruções, diagramas e textos que se complementam, auxiliam e promovem a compreensão dos seus conteúdos. Sobre as formas de comunicação que podem ser abordadas pela Análise de Conteúdo (BARDIN, 2011) afirma:

> Em última análise, qualquer comunicação, isto é, qualquer veículo de significados de um emissor para um receptor, controlado ou não por este, deveria poder ser escrito, decifrado pelas técnicas de análise de conteúdo.

Ora, tal afirmação corrobora exatamente com a conveniência da Análise de Conteúdo para os objetivos desse estudo e o porquê dessa técnica ser escolhida para tal. As Recomendações e Referenciais de Formação para os Cursos Superiores em Computação exercem uma comunicação relevante para os profissionais que buscam o desenvolvimento da educação superior na área. Da mesma forma, os diversos estudos que abordam o uso de Projetos FLOSS como elemento auxiliador no ensino de ES e suas subáreas exercem comunicação relevante com educadores e pesquisadores que se interessam pelo tema ou que busca identificar fatores de interferência nesse processo.

Antes do detalhamento dos procedimentos específicos para a análise dos conteúdos, é importante entender o ponto de partida metodológico sob a perspectiva teórica que originou todo o percurso de estudo de que se trata este texto. Essa perspectiva permite, passo a passo a compreensão de onde a análise do estudo parte, por onde ela percorre e onde se estabelece.

Segundo (MENDES; MISKULIN, 2017, p.1), "[...] a confecção de uma colcha de retalhos inicia-se com a escolha do retalho que comporá o "centro" do trabalho". Tal percepção contribui para identificar a perspectiva que influenciou nas tomadas de decisões metodológicas do trabalho. O esforço deste trabalho se inicia sobre a necessidade que a academia tem em buscar alternativas que aproximem o estudante de Computação com a realidade da Indústria de Software, encontrando nos Projetos FLOSS uma alternativa relevante para o Ensino de ES que estimulam a aprendizagem e vivência do aluno em uma dinâmica de desenvolvimento próxima ao real processo de desenvolvimento de software.

Porém, é desafiador ao docente encontrar Projetos que aproximem os objetivos de sua prática docente com as necessidades do mundo do trabalho e da formação acadêmica inerentes ao componente curricular ou curso que leciona.

O estímulo a aquisição de competências técnicas e sociotécnicas pelo estudante pode ser um fator relevante como parâmetro de avaliação de um projeto escolhido pelo docente ou instrutor de Engenharia de Software.

Nesse aspecto, obtêm-se a visão de competências ou habilidades sociotécnicas como as práticas dos profissionais tanto nos domínios tecnológicos e técnicos como nos domínios comportamentais.

No que tange a dinâmica docente ao usar Projetos FLOSS para ensinar Engenharia de Software, é importante a compreensão e a percepção de fatores que interfiram de forma positiva ou limitadora nos objetivos de aprendizagem esperados pelo docente ou instrutor, compreendendo como alvo abrangente de alcance de aprendizado exatamente a

aquisição de competências e habilidades técnicas e sociotécnicas ansiadas pela academia e pela indústria de software.

Percebendo essa linha de conceitos, é possível se debruçar em todos os aspectos que formam o trabalho. Os próximos tópicos desse capítulo descrevem o Ciclo da Pesquisa, focando nos contextos práticos dos procedimentos escolhidos para se atingir os objetivos do estudo.

3.2 CICLO DA PESQUISA

O Ciclo da Pesquisa deste trabalho se divide em atividades com Concepção, Planejamento e Desenvolvimento. A fase de Concepção é dividida em Contextualização e Revisão Bibliográfica, já discutidas nos capítulos anteriores. A fase de Planejamento é dividida em Decisões Metodológicas e Procedimentos. A fase de Desenvolvimento é determinada pela extração de dados, análise de dados, transformação de dados, interpretação e síntese de dados e geração do `Modelo Conceitual`.

A fase de Revisão Bibliográfica permitiu todo o aporte necessário para a perspectiva teórica adotada no estudo, sendo usada tanto para auxiliar a contextualização da pesquisa como para determinar a linha conceitual e teórica base para a pesquisa. Na fase de Contextualização foram definidos o contexto do tema, o problema de pesquisa, os objetivos, as questões e a justificativa.

As Decisões Metodológicas abordadas anteriormente justificam o Planejamento discutido adiante. A Figura 3.1. mostra o planejamento e descrição das atividades desenvolvidas no Ciclo da Pesquisa.

3.2.1 O Universo de Estudo

O Universo de Estudo dessa pesquisa foi delimitado a partir de dois cenários que estabelecem a fonte para as respostas aos problemas de pesquisa.

O primeiro cenário é *o mapeamento de competências e habilidades técnicas ou sociotécnicas esperadas aos estudantes de cursos superiores em Computação.* Tais competências são indicadas ou estipuladas por meio de instrumentos normativos, indicativos ou legislativos sobre objetivos educacionais ou instrucionais da formação superior em Computação.

O segundo cenário é *o mapeamento de aspectos relevantes de interferência nos processos de ensino e aprendizagem no ensino de Engenharia de Software ao usar FLOSS como ferramenta de ensino.* Tais aspectos são indicados em estudos ou pesquisas que abordaram o uso de Projetos FLOSS no ensino de Engenharia de Software.

3.2.2 Contexto da Coleta de Dados

A coleta de dados desse estudo se deu por um processo minucioso de análise bibliográfica, no qual o embasamento teórico que guiou o mesmo permitiu a adoção dos critérios de seleção e coleta necessários para a solução dos problemas de pesquisa.

Esta seção exibe a contextualização da coleta dos dados, analisando o aspecto teórico como guia para as decisões dos dados a serem coletados e os motivos metodológicos que embasam essas decisões. A linha conceitual auxilia a tomada de decisão e a prioridade

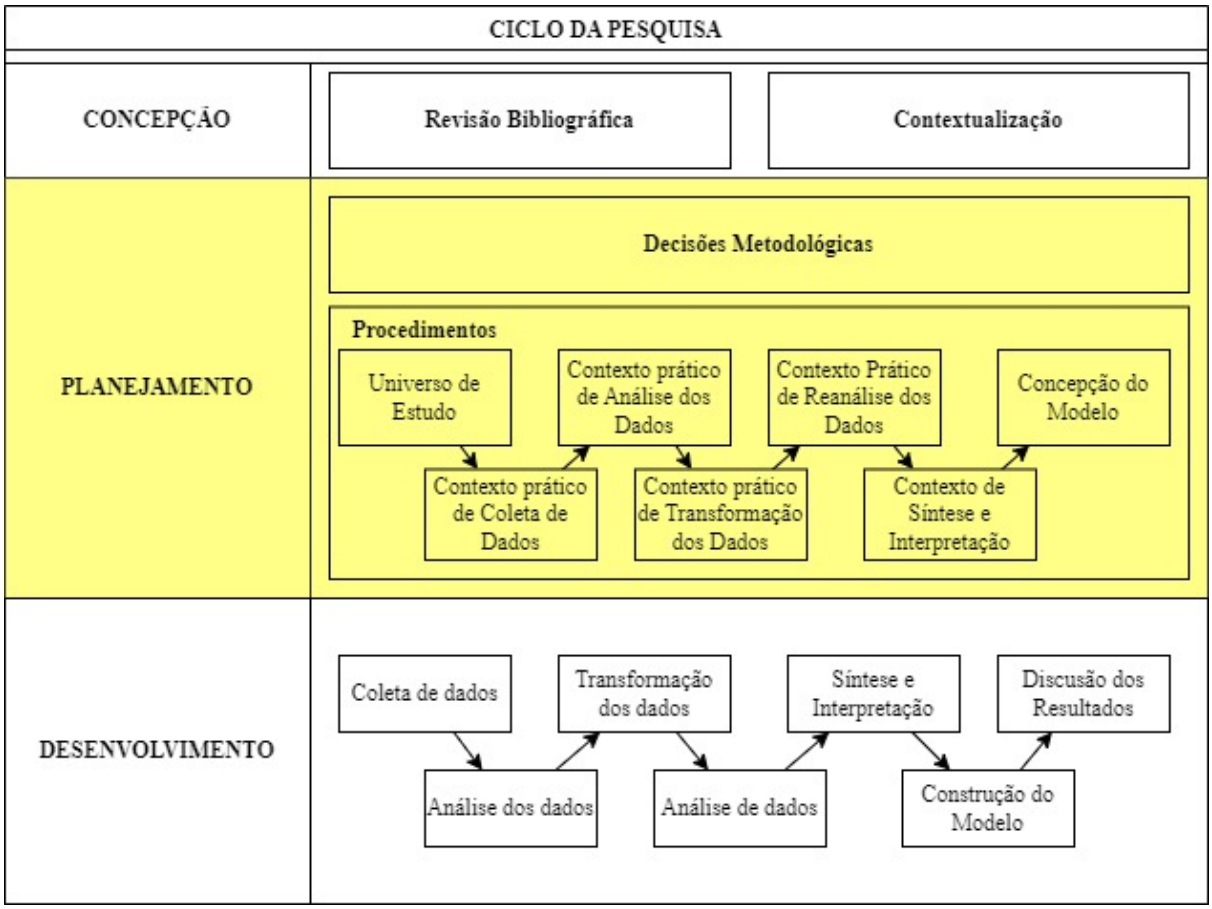

Figura 3.1 Ciclo da Pesquisa
Fonte: Elaboração Própria

de coleta dos dados, as análises destes, as interpretações e contabilizações necessárias e a organização dos resultados da coleta e análise para a concepção do modelo conceitual proposto como Objetivo Geral de Pesquisa. Vamos usar o termo ***Corpus***, de (BARDIN, 2011) para descrever as fontes selecionadas para coleta e posterior análise dos dados. Segundo (BARDIN, 2011, p.126), "O corpus é o conjunto dos documentos tidos em conta para serem submetidos aos procedimentos analíticos."

A Figura 3.2 mostra a organização linear da coleta de dados, estruturando as Fontes Bibliográficas escolhidas (*Corpus*). O ponto de partida na coleta de dados se deu na tentativa de compreender as competências técnicas e sociotécnicas estimuladas pelo ensino de Engenharia de Software e suas subáreas no Ensino Superior em computação. Para tal, decidiu-se pela análise dos principais guias e referenciais curriculares de formação para cursos superiores na área da computação no Brasil e no exterior.

3.2.2.1 Contexto de coleta nos Referenciais de Formação Nacionais. A escolha das fontes de estudos tem como meta responder a seguinte questão de pesquisa:

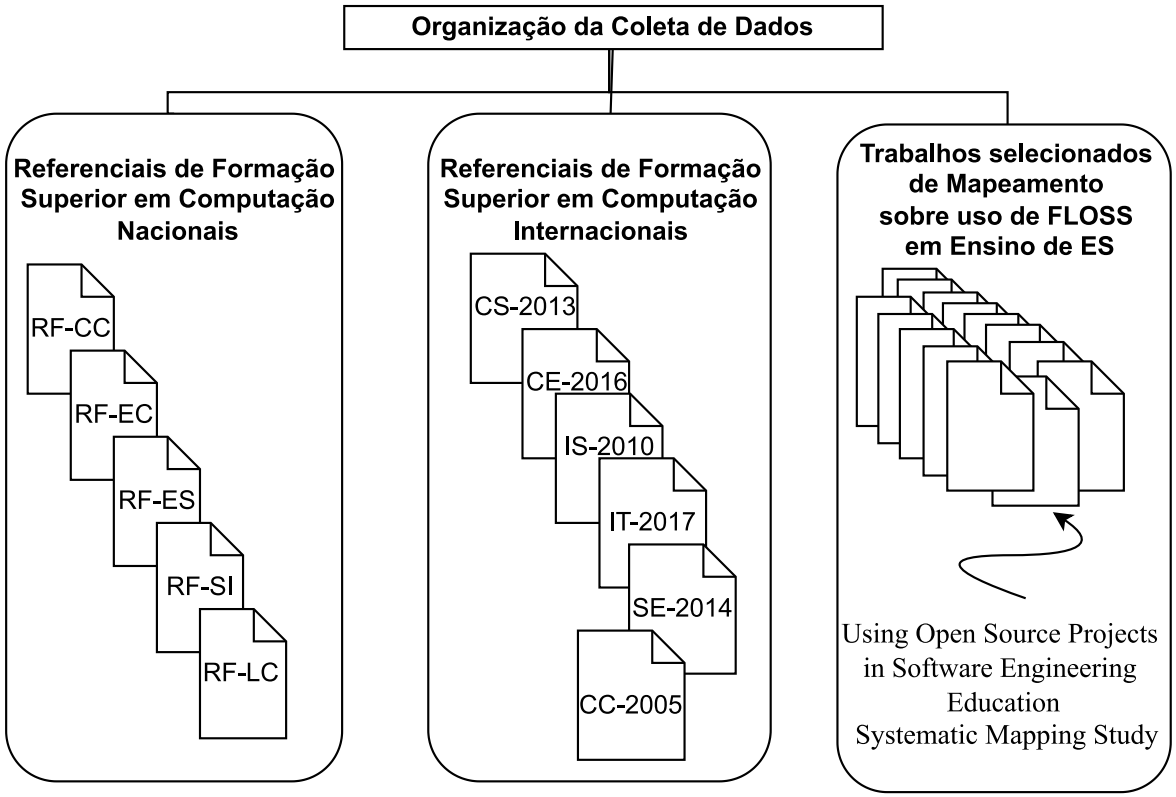

Figura 3.2 Contexto de Coleta de Dados
Fonte: Elaboração Própria

Q1 *Quais as competências técnicas e sociotécnicas associadas com a Engenharia de Software segundo os principais Referenciais de Formação Superior em Computação no Brasil?*

Os guias e referenciais de formação fazem parte de um trabalho conjunto em comissões ou grupos de trabalho com diversos profissionais da área de computação alinhados com o desenvolvimento educacional e de formação na área (SHACKELFORD et al., 2005; ZORZO et al., 2017).

Os critérios de escolha para fichamento e análise dos principais referenciais no Brasil e no exterior se baseiam em alguns aspectos como a relevância da associação ou sociedade científica que auxiliou no desenvolvimento do guia ou referencial, bem como, a instrumentação do guia/referencial através de aspectos da legislação específica da formação em Computação no Brasil. Dessa forma, foi consultada a referência legal sobre Diretrizes de Formação Superior no Brasil. A Resolução Nº 5 de 16 de Novembro de 2016 (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22), que instrui as Diretrizes Curriculares Nacionais para os cursos de graduação na área da Computação (DCN), é o instrumento legal base para a concepção de Referenciais de Formação superior em Computação no Brasil. Tais diretrizes formam a base para o desenvolvimento dos Referenciais de For-

mação para os Cursos de Graduação em Computação 2017 da Sociedade Brasileira de Computação (SBC).

Os Referenciais de Formação da SBC foram a base nacional utilizada para o mapeamento de competências e habilidades técnicas e socioténicas deste estudo. Neste documento, a SBC aborda Referenciais de Formação para cinco cursos de Graduação na área de Computação no Brasil, sendo eles os Referenciais de Formação para Cursos de Bacharelado em Ciência da Computação (RF-CC), os Referenciais de Formação para Cursos de Bacharelado em Engenharia da Computação (RF-EC), os Referenciais de Formação para Cursos de Bacharelado em Engenharia de Software (RF-ES), os Referenciais de Formação para Cursos de Bacharelado em Sistemas de Informação (RF-SI), e os Referenciais de Formação para Cursos de Licenciatura em Computação (RF-LC) (ZORZO et al., 2017).

Observa-se a pertinência em usar os cinco RFs como *corpus* no estudo devido a sua direção sobre dois aspectos básicos na linha teórica adotada, sendo eles o alinhamento com uma referência legal e o alinhamento com o conceito de competências. Isso fica claro quando (ZORZO et al., 2017, p.7) versa que "O trabalho das comissões foi conduzido pelos membros da Comissão de Educação[...], [...] buscando atender aos seguintes princípios básicos: 1. Estar alinhado com as DCNs; 2. Seguir um modelo baseado em competências.". Os Referenciais de Formação para os Cursos de Graduação em Computação (RF) foram desenvolvidos sobre bases metodológicas relevantes como versões de 1991 até 2005 dos Currículos de Referência da SBC, das próprias Diretrizes Curriculares Nacionais de 2016 e de Currículos de Computação da Série Curricula Computing da Associação para Maquinas de Computação (ACM).

Nota-se que os Referenciais de Formação para Cursos de Graduação na área da Computação são as mais relevantes fontes de consulta para a construção de currículos em Computação no Brasil. Os mesmos adotam liberdade para as realidades específicas dos cursos e instituições e possuem uma metodologia baseada em competências. A Figura 3.3 mostra a relação dos Referenciais de Formação RF com as bases metodológicas usadas na sua concepção.

A Tabela 3.1 apresenta as principais características dos dados nos cinco RFs usados no contexto de coleta.

Após essa explanação, observamos porque os Referenciais de Formação 2017 da SBC são escolhidos para o mapeamento de competências estimuladas pela Educação em Engenharia de Software no contexto brasileiro.

3.2.2.2 Contexto de Coleta nos Guias Curriculares Internacionais .

Os principais Guias Curriculares Internacionais são base para a metodologia de alguns Referenciais Curriculares e Diretrizes Curriculares no Brasil. Assim, eles devem ser compreendidos como instrumento de coleta dos dados na pesquisa. A escolha das fontes que serão abordadas adiante tem como meta responder a segunda questão de pesquisa:

Q2 *Qual a relevância do conhecimento em ES e do conhecimento sociotécnico segundo os principais guias curriculares internacionais?*

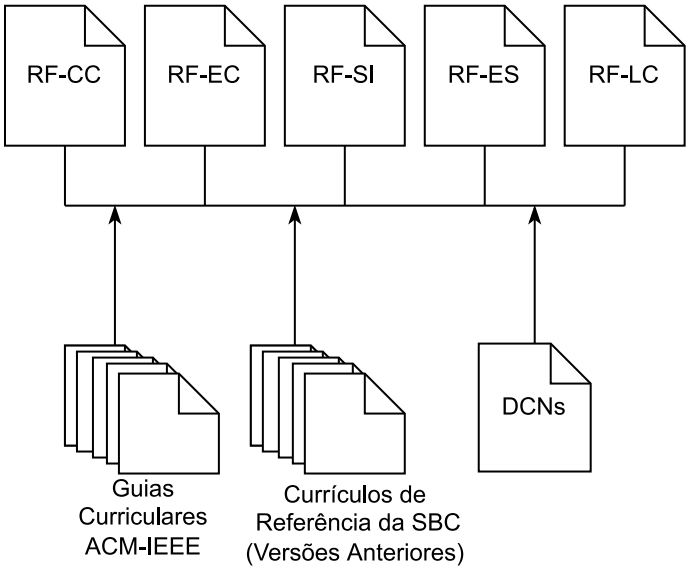

Figura 3.3 Referenciais Nacionais para coleta
Fonte: Elaboração Própria

Tabela 3.1 Características gerais dos Referenciais de Formação.

Características estruturais gerais dos Referenciais de Formação da SBC	
Tópico	Descrição
Apresentação	Demonstra todo o contexto histórico e técnico dos Referenciais
Estrutura conceitual	Descreve a metodologia usada para relacionar o perfil do egresso com eixos de formação, com as competências e os conteúdos que auxiliar o alcance de tais competências.
Apresentação do Curso	Demonstra todo o contexto técnico e a forma de desenvolvimento de cada RF
Breve Histórico do Curso	Exibe todo o contexto histórico que demonstra a importância do curso e do RF Equivalente
Benefícios do Curso para a Sociedade	Mostra os pontos que fazem o curso em questão ser importante para a sociedade e para o desenvolvimento profissional.
Aspectos relacionados com a formação profissional	Demonstra as características gerais e específicas dos profissionais de computação em cada curso.
Perfil do egresso.	Versa sobre as competências esperadas pelo egresso do curso.
Eixos de Formação, Competências e Conteúdos.	Relaciona as competências derivadas para cada eixo de formação e os conteúdos que permitem o alcance de tais competências.
Relação com as DCNs	Relaciona as competências desenvolvidas pelos conteúdos com as competências descritas pelas Diretrizes Curriculares Nacionais (DCN)
Estágios, TCC, Metodologias, Leis e agradecimentos.	Instruem sobre as diretrizes para aplicação de estágios e TCCs, Mostram os fundamentos legais e conclui o referencial.

Fonte: (ZORZO et al., 2017)

Além do Brasil, a Educação Superior em Computação é tema importante entre educadores, pesquisadores e organizações científicas e governamentais no resto do mundo (NASCIMENTO; BITTENCOURT; CHAVEZ, 2015; SHACKELFORD et al., 2005; BAILEY;

LUNT; ROMNEY, 2006). A necessidade de um profissional de Computação capaz de assumir as atribuições e tarefas que lhes são destinadas em um mundo globalizado e em constante evolução justifica a busca por currículos e cursos superiores em computação que atendam tal necessidade.

Vários guias, modelos e recomendações se debruçam sobre as habilidades e competências que devem ser adquiridas pelo profissional ou estimuladas aos estudantes de Computação. Instituições como (ACM), Instituto de Engenheiros Eletricistas e Eletrônicos (IEEE), Modelo de Competências em Tecnologia da Informação do Estados Unidos (USTI) e Comité Europeu de Normalização (CEN), por exemplo, trabalham de forma constante para manter a área de Computação a mais atualizada e compatível possível com a realidade do mundo do trabalho.

Sob tal perspectiva, vários referenciais ou guias de formação para a formação na área de Computação são periodicamente desenvolvidos na busca de formar egressos atualizados e prontos para as demandas da indústria. A ACM juntamente com a IEEE são referências no desenvolvimento de guias e currículos para a formação superior em Computação. Quando se consulta os referenciais brasileiros em suas diversas versões, como os Referenciais da SBC de 1991 (SBC, 1991), de 1999 (SBC, 1999) e demais versões até os RF de 2017 (ZORZO et al., 2017), sempre nota-se trabalhos, documentos ou guias desenvolvidos pela ACM e pela IEEE.

É por meio desta abordagem que percebemos a importância dos principais currículos para os cursos de graduação na área de computação a nível mundial. São eles: o Currículo de Ciência da Computação ACM/IEEE de 2013; o Currículo de Engenharia da Computação ACM/IEEE de 2016; o Currículo de Sistemas de Informação ACM/AIS de 2010; o Currículo de Engenharia de Software ACM/IEEE de 2014 e o Currículo de Tecnologia da Informação ACM/IEEE de 2017.

Tais Currículos tem uma característica semelhante aos RFs brasileiros, os mesmos foram desenvolvidos com base em uma diretriz geral, no caso dos RFs a diretriz geral são as DCNs descritas na Resolução nº 5 de 2016 (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22), no caso dos Currículos da ACM a referência básica para o desenvolvimento dos Guias é o Relatório Geral para os cursos de Graduação na área da computação Computing Curricula 2005 (CC-2005).

O CC-2005 é o relatório resultante de uma força tarefa conjunta entre a ACM, IEEE e a Associação de Sistemas de Informação (AIS) que tem como objetivo fornecer uma visão geral dos programas de graduação na área de Computação. Foi por meio dessa visão geral que os guias de cada um dos cinco programas abordados foram desenvolvidos.

Como os Guias da série Computing Curricula 2005 estão em suas versões mais atuais, é pertinente focar nos mesmos para entrar no *corpus* da coleta dos dados necessários para compreensão das relações de conteúdos com competências no contexto internacional de ensino de computação.

Observamos a relevância dos Currículos de Ciência da Computação (CS-2013), de Engenharia da Computação (CE-2016), de Sistemas de Informação (SI-2010), de Engenharia de Software (SE-2014), e de Tecnologia da Informação (TI-2017) quando notamos sua presença nas fontes dos RFs Brasileiros e de outros documentos ou trabalhos pautados na abordagem de competência para os profissionais em computação no Brasil e no

Mundo (PORTELA; VASCONCELOS; OLIVEIRA, 2016; BEGOSSO et al., 2011; INFOCOMP(2012), 2012; BAILEY; LUNT; ROMNEY, 2006). A Figura 3.4 mostra a relação dos guias provenientes da série Curricula Computing CC-2005 com as bases metodológicas usadas na sua concepção.

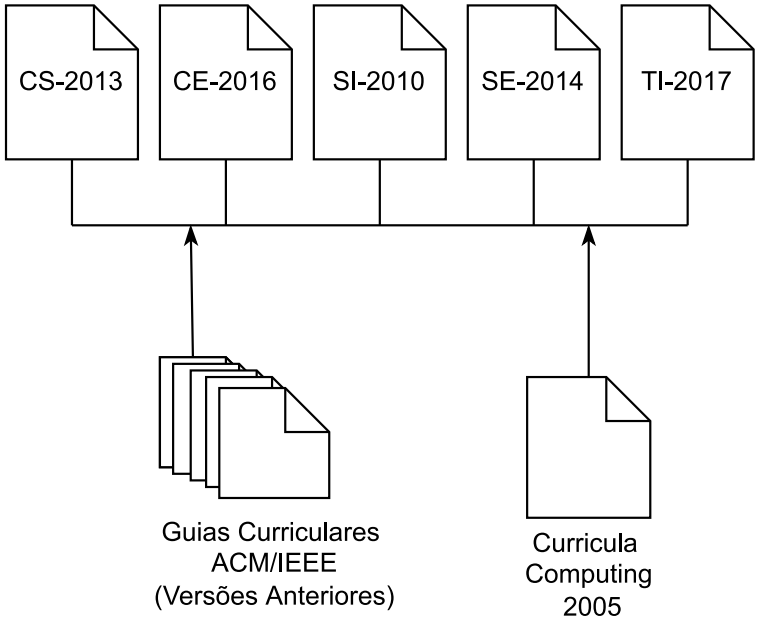

Figura 3.4 Referenciais Nacionais para coleta
Fonte: Elaboração Própria

Na Tabela 3.2 é possível observar as características gerais dos dados comuns aos cinco currículos da série Curricula (ACM) abordados. Conhecendo a importância e o teor das fontes citadas, fica explícita a relevância dos guias citados para a cumprimento dos mapeamentos e, portanto, a sua inclusão no *corpus* desta pesquisa.

3.2.2.3 Contexto de coleta em estudos sobre uso de FLOSS no ensino de Engenharia de Software. O passo seguinte é a seleção do *Corpus* voltado para a coleta de dados referentes a aspectos citados ou percebidos sobre interferências positivas ou limitadoras nos processos de ensino e aprendizagem na área de Engenharia de Software. A escolha das fontes descritas nesta etapa tem como meta responder a terceira questão de pesquisa desse estudo:

> **Q3** *Quais os principais aspectos de interferência no ensino e aprendizagem reportados em estudos sobre uso de Projetos FLOSS na Educação em Engenharia de Software?*

As fontes que mais tendem a contribuir com exposição de experiências de docentes ou estudantes sobre interferência de FLOSS no ensino/aprendizagem de ES são trabalhos e

Tabela 3.2 Características gerais dos Guias da Série Série Currícula Computing.

Características estruturais gerais dos Guias Curriculares da Série Curricula Computing	
Conteúdo	Descrição
Introdução	Demonstra a visão geral do documento o contexto histórico da área, fala sobre a evolução da área no aspecto acadêmico e profissional.
Princípios orientadores	Descrevem os fundamentos e princípios para a organização conceitual e estrutural do documento.
O área do curso como uma disciplina da computação.	Descreve os fundamentos da área do curso e sua importância para a área da computação. O perfil dos egressos e a posição profissional da área na computação
Corpo do conhecimento e competências	Descreve os conteúdos ou competências a serem abordadas no currículo, descrevendo o peso dos mesmos em blocos de prioridade de acordo com cada curso.
Benefícios do Curso para a Sociedade	Mostra os pontos que fazem o curso em questão ser importante para a sociedade e para o desenvolvimento profissional.
Expectativas de formação e aprendizagem	Fala sobre a expectativa ou resultados de aprendizagem esperados na aplicação do curso.
Estratégias ou fundamentos para o projeto e organização dos Currículos	Descreve os princípios ou informações relevantes a serem consideras na construção dos currículos.
Aspectos institucionais e avaliação do programa.	Fala sobre desafios institucionais, de aprendizagem, adaptação a realidades educacionais e institucionais e aspectos voltados a avaliação do curso.
Exemplos de Currículos ou instruções	Mostram escopos curriculares ou exemplos de currículos a serem usados como guia para desenvolvimento dos currículos,

Fonte: (DRAFT, 2013; IMPAGLIAZZO et al., 2016; TOPI et al., 2010; ARDIS et al., 2014b; SABIN et al., 2017)

estudos que versam sobre o uso de FLOSS na educação em Engenharia de Software e suas demais subáreas. Partindo por esse princípio, a busca por tais trabalhos devem obedecer critérios e métodos sistemáticos que consigam extrair, de uma grande base de dados, as pesquisas mais relevantes de acordo com os objetivos do estudo.

O Mapeamento Sistemático é um dos métodos de pesquisa mais relevantes para a investigação profunda de estado da arte a cerca de um tema (FALBO, 2018; DERMEVAL; COELHO; BITTENCOURT, 2020). Um Mapeamento Sistemático reúne estudos primários em um tópico de pesquisa sem fazer uma análise profunda de questões de pesquisa, objetivando apenas obter uma visão geral de uma determinada área (DERMEVAL; COELHO; BITTENCOURT, 2020; FALBO, 2018). Ora, como o objetivo principal dessa pesquisa é mapear aspectos dentro de um contexto teórico em determinado estado da arte, parece pertinente o uso de um Mapeamento Sistemático como guia para a obtenção de estudos a cerca do tema foco do contexto de coleta que aqui é descrito, ou seja, o uso de Projetos FLOSS na Educação em Engenharia de Software.

Para responder a esta questão de pesquisa, utilizamos os resultados de um mapeamento sistemático da literatura (SMS) realizado em 2018 (BRITO et al., 2018).
Tal estudo secundário tornou-se a base para a seleção dos trabalhos, dado o alinhamento entre questões de pesquisa e a compatibilidade temática necessária para nossa coleta e análise. A Tabela 3.3 apresenta o protocolo do mapeamento adotado e mostra sua compatibilidade com nossos requisitos de estudo para definição do *Corpus*.
O título, a descrição, a questão e a intervenção presentes na tabela 3.3 demonstram compatibilidade teórica com os tipos de trabalhos que foram selecionados e que esperamos usar para fazer nossa coleta e análise.

A população e aplicação são compatíveis com o contexto de nosso trabalho: ensino em cursos de graduação em Computação. As palavras-chave usadas para a busca são compatíveis com as palavras-chave que selecionamos a partir de uma busca *ad-hoc* por trabalhos que tratam do uso de Projetos FLOSS no ensino de Engenharia de Software.
Os estudos selecionados estão apenas na Língua Inglesa, pertinente por ser um dos idiomas mais usado para o registro dos trabalhos nas principais bases do mundo. Ao observar os métodos e motores de busca nota-se que os trabalhos foram retirados das bases científicas importantes.
Na linha dos critérios de inclusão e exclusão da Tabela 3.3, percebemos que o foco é buscar artigos que falam exclusivamente do uso de projetos FLOSS para aprender ou ensinar conteúdos de engenharia de software. Tais conteúdos são selecionados com base no corpo de conhecimento em Engenharia de Software da Guide to the Software Engineering Body of Knowledge (SWEBOK) ou afins.
Após os autores exercerem a extração e análise dos critérios em cada um dos trabalhos que foram trazidos das bases, restaram trinta e três artigos para classificação, análise e demais práticas.
No final do processo, *trinta artigos* considerados no trabalho de atualização do SMS (BRITO et al., 2018) convergem com a amostra bibliográfica necessária para fazer parte do *Corpus* de nossa pesquisa.

Tabela 3.3 Protocolo de Mapeamento Sistemático

Título	**Usando projetos de código aberto no ensino de engenharia de software: estudo de mapeamento sistemático**
Pesquisadores	Moara Sousa Brito; Fernanda Gomes Silva; Christina von Flach Garcia Chávez; Débora Maria Coelho Nascimento; Roberto Bittencourt;
Descrição	Uma possibilidade de trazer a prática para o Curso de Engenharia de Software é fazer com que os alunos participem de Projetos Open Source com supervisão de professores. Muitas pesquisas vêm se aproximando dessa ideia. O objetivo deste estudo de mapeamento sistemático é resumir todas as informações existentes sobre o uso de Projetos Open Source em Cursos de Engenharia de Software ou Ciência da Computação, e como a aprendizagem dos alunos tem sido hospedado, a fim de identificar possíveis lacunas que justifiquem novas atividades de pesquisa.
Questão	Como os projetos de código aberto têm sido usados no ensino de engenharia de software?
Intervenção	O uso de Projetos Open Source para melhorar o aprendizado dos alunos sobre Conceitos de Engenharia de Software.
População	Cursos de Graduação em Engenharia de Software
Resultados	Como os Projetos Open Source têm sido utilizados nos Cursos de Engenharia de Software, metodologias, propostas, oportunidades, estratégias, resultados obtidos, quais áreas específicas da Engenharia de Software têm sido utilizadas.
Aplicação	Universidades
Palavras-Chave	Aprendiz; Treinamento; Competências; Ciência da Computação; Informática; Curso; Currículo; Padrões de design; Educação; Educacional; FLOSS; FOSS; Software grátis; Aprendendo; Software Livre; Mentoria; OSP; OSS; Código aberto; Habilidades; Análise de Software; Arquitetura de software; Gerenciamento de Configuração de Software; Design de software; Desenvolvimento de software; Engenharia de software; Evolução de Software; Manutenção de software; Gestão de Software; Métricas de Software; Modelagem de Software; Processo de Software; Qualidade de Software; Requisito de Software; Teste de software; Validação de Software; Verificação de Software; Ensino; Treinamento; Tutoria;
Idioma dos estudos	Inglês
Métodos de busca	Pesquisa através de motores de busca na Web e pesquisa manual;
Motores de Busca	IEEE; Science Direct; Scopus; ACM; Engineering Village; Springer;
Critérios de Inclusão e Exclusão do Estudo	(I) O estudo explora o uso de Projetos Open Source para aprender/ensinar conteúdos de Engenharia de Software; (I) O estudo é apresentado como artigo completo; (I) O estudo é apresentado como um artigo curto; (I) O estudo é apresentado como trabalho em andamento; (E) Documento não escrito em inglês; (E) Documento cujo texto completo não esteja disponível; (E) O conteúdo do estudo não está associado à aprendizagem/ensino de Engenharia de Software; (E) O foco do estudo é apenas utilizar o Software Open Source como ferramenta na infraestrutura dos laboratórios; (E) O foco do estudo é apenas utilizar o Software Open Source como ferramenta de desenvolvimento de aplicativos; (E) O documento que trata de temas irrelevantes para a finalidade deste mapa; (E) O estudo está disponível apenas como resumo; (E) O estudo disponível apenas como apresentação em Powerpoint; (E) É uma descrição de um painel em uma conferência; (E) O documento é um processo; (E) O Software Open Source é uma ferramenta utilizada para auxiliar a aprendizagem; (E) O documento é um Índice; (E) O foco do estudo é ensinar sobre ou como usar um Software Open Source específico; (E) Projetos de código aberto não são usados; (E) Não há foco na utilização de um Projeto Open Source no processo de aprendizagem, o Software Open Source é apenas uma ferramenta; (E) Projetos de código aberto são usados como fonte de dados para pesquisa; (E) O documento é uma carta do editor; (E) É uma descrição de uma palestra ou workshop em uma conferência; (E) O documento descreve um tutorial;
Áreas de Extração de Informações	Áreas de Engenharia de Software abordadas={Engenharia de Software (geral), Requisitos de Software,Modelagem/Análise de Software,Design e Arquitetura de Software, Qualidade de Software: Verificação e Validação,Teste de Software,Evolução e Manutenção de Software, Processo de Software,Gerenciamento de Software,Gerenciamento de Configuração de Software , Desenvolvimento/Construção de Software}; objetivo da abordagem proposta no artigo={Aprender conceitos de SE usando projetos OSS, Aprender sobre Projetos OSS em um Curso de Engenharia de Software}; Abordagem de aprendizagem ativa aplicada={Não especificado explicitamente, Aprendizagem baseada em problemas, Aprendizagem baseada em projetos, Método de projeto,Aprendizagem baseada em casos, Aprendizagem em estúdio,Aprendizagem por pares,Aprendizagem baseada em jogos,Outro}; perspectiva de avaliação da aprendizagem={Não especificado explicitamente, perspectiva do aluno, perspectiva do professor, avaliação do produto: critérios de avaliação aplicados à produção do aluno}; tipo de avaliação aplicada para verificar a aprendizagem dos alunos={ nenhum,Exames,relatório, portfólio, artefatos de software,passando nos testes,seminário}; Como os projetos de código aberto são usados no currículo?={Sem especificação explícita, atividade extra, disciplina opcional, disciplina obrigatória}; nível de controle sobre o projeto={controle total,controle interno,sem controle}; Qual a função de usuário que um aluno assume?={usuário passivo, usuário ativo, co-desenvolvedores, desenvolvedores principais}; Tipo de Pesquisa={Relato de experiência, Estudo de caso, Experiência, Pesquisa/questionário, Etnografia, Filosófico, artigo de opinião, Proposta de solução};

Fonte (BRITO et al., 2018)

3.2.3 Contexto prático de Análise de Dados: Análise de Conteúdo.

A Análise de conteúdo, como já explicitado, permite extrair conhecimento de conteúdos de diversas formas e fontes. A riqueza desta técnica se apresenta nas suas variadas formas de compreender os conceitos, padrões e tendências extraídas dos dados (BARDIN, 2011; CÂMARA, 2013; FRANCO, 2020). Bardin faz uma descrição geral da Análise de Conteúdo da seguinte forma:

> Um conjunto de técnicas de análise das comunicações visando obter por procedimentos sistemáticos e objetivos de descrição do conteúdo das mensagens, indicadores (quantitativos ou não) que permitam a inferência de conhecimentos relativos às condições de produção/recepção (variáveis inferidas) dessas mensagens (BARDIN, 2011, p.48).

A análise de conteúdo é bem utilizada em pesquisas com métodos quantitativos de abordagem, mas também pode ser importante para pesquisas qualitativas. A frequência de aparição de elementos de uma mensagem está mais fundamentada a uma análise quantitativa, enquanto aspectos como presença, ausência e coocorrência podem ser associadas a uma abordagem mais qualitativa de análise (BARDIN, 2011).

Este trabalho absorve exatamente os dois aspectos de análise para uma demonstração mais rica dos resultados, a depender do objetivo específico de cada fase de análise, os resultados serão analisados sobre uma perspectiva qualitativa ou quantitativa.

Sobre a importância desta afirmação (BARDIN, 2011, p.145) versa:

> A abordagem quantitativa e a qualitativa não tem o mesmo campi de ação. A primeira obtém dados descritivos por meio de um método estatístico.[...] A segunda corresponde a um procedimento mais intuitivo, mas também mais maleável e mais adaptável a índices não previstos.

É importante descrever a estrutura metodológica da técnica para tornar compreensível os passos adotados desde os processos de leitura, como decisões na coleta e nas técnicas de análise dos mesmos. Segundo (BARDIN, 2011), a Análise de Conteúdo estrutura-se em três fases gerais: pré-análise, exploração do material, e tratamento dos resultados, inferência e interpretação, como mostra a Figura 3.5.

A pré-análise é a fase de organização da Análise de Conteúdo e tem como missão, escolher os documentos a serem submetidos à análise, a formulação das hipóteses e objetivos e os indicadores que fundamentem a interpretação final (BARDIN, 2011). Segundo Bardin, a fase de pré-análise é organizada pelas atividades indicadas na Figura 3.6.

- **Leitura flutuante:** Esta atividade tem como objetivo os primeiros contatos com as fontes a serem analisadas e com os textos a serem conhecidos para obter impressões e orientações. Paulatinamente a leitura se torna precisa de acordo com as hipóteses que emergem, com as teorias adaptadas sobre o material e com as possíveis técnicas utilizadas em materiais análogos. Foi por meio dessa leitura que os critérios para a escolha das fontes e os contextos práticos foram sendo delineados.

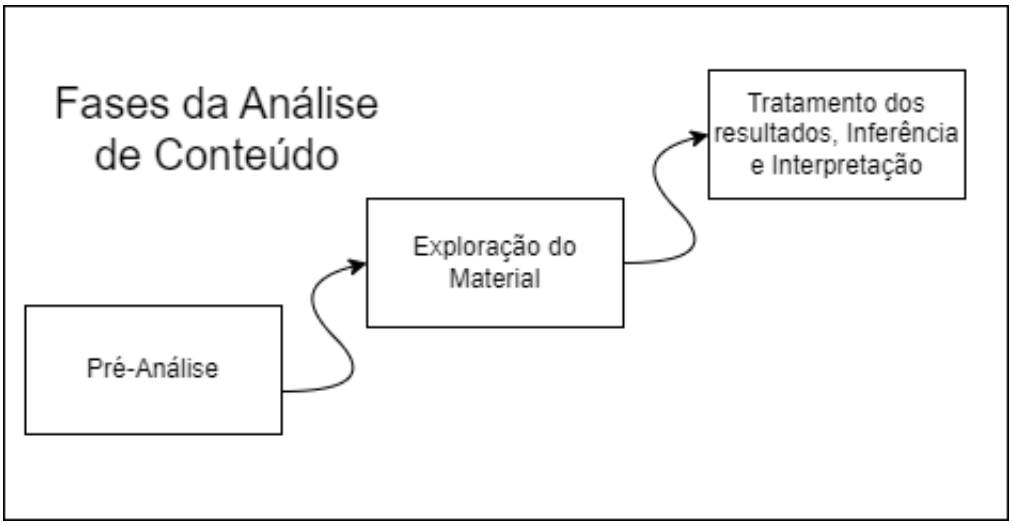

Figura 3.5 Fases da Análise de Conteúdo.
Fonte: (BARDIN, 2011)

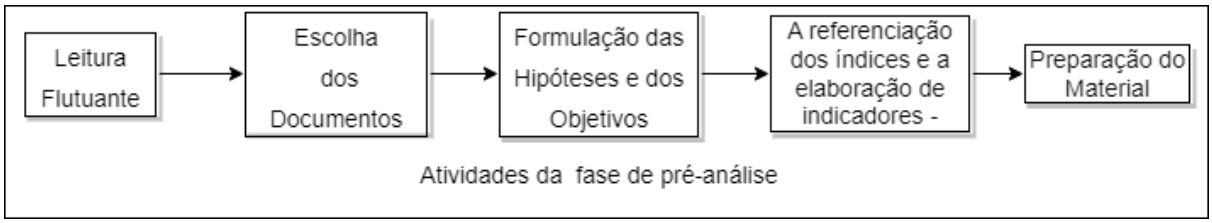

Figura 3.6 Pré-Análise da Análise de Conteúdo.
Fonte: (BARDIN, 2011)

- **A escolha dos documentos:** O universo de documentos a ser determinado. É na fase de escolha dos documentos que é constituído o *corpus* de análise. Quando já se tem o gênero de documentos demarcado, a constituição do *corpus* é necessária. Segundo Bardin, a constituição do *corpus* devem seguir algumas regras na delimitação:

  (I) *Regra da exaustividade* - Com o campo do *corpus* definido é necessário esgotar a análise em todos os elementos desse *corpus*, sem deixar elementos de fora da análise por razões que não possa ser justificável no plano do rigor.

  (II) *Regra da representatividade* - A análise pode ser feita numa amostra, esta se dirá rigorosa se for uma parte significativa do universo inicial, podendo ter resultados obtidos para amostra generalizados.

  (III) *Regra da homogeneidade* - as fontes escolhidas devem ser homogêneas, obedecendo critérios precisos de escolha.

  (IV) *Regra da pertinência* - As fontes devem ser pertinentes ao objetivo de análise, ou seja, devem ser adequados como fonte de informação.

- **Formulação das hipóteses e dos objetivos:** As hipóteses são ideias provisórias

que se propõe a verificar por meio dos procedimentos de análise, ou seja uma suposição que é levantada de forma intuitiva que permanece suspensa enquanto não for submetida a prova. Os objetivos tratam da finalidade geral que se propõe, bem como o quadro teórico ou pragmático que se embasará dos resultados utilizados.

- **Referenciação dos Índices e a elaboração de indicadores:** Índices são manifestações dos textos que podem ser explícitas e escolhidas, como menções que se apresentam em importância de acordo com sua frequência ou percepção. Os indicadores são construídos após os índices, como a frequência de apresentação, a ausência ou presença de correlações dos índices.

- **Preparação do Material:** É a preparação formal das fontes a serem analisadas, ou seja, a organização dos recortes, textos e demais fontes para serem registrados ou editados para a análise.

  Analisando as atividades de pré-análise, já é possível observa-las já na descrição dos contextos de coleta, como aspectos voltados ao *corpus* e as decisões tomadas para tal. Nos capítulos posteriores, cada etapa será detalhada em cada um dos procedimentos adotados nos elementos do *corpus* definido.

Depois da Pré-Análise, (BARDIN, 2011) descreve a Exploração do Material como segunda fase da Análise de conteúdo. Esta se trata da fase de Codificação, que é a aplicação sistemática das decisões tomadas. A codificação é organizada pela definição das *Unidades de Contexto, Unidades de Registro, Eixos Temáticos* e *Categorias de Análise* (RODRIGUES, 2016; BARDIN, 2011). A Figura 3.7 mostra as fases da Exploração do Material.

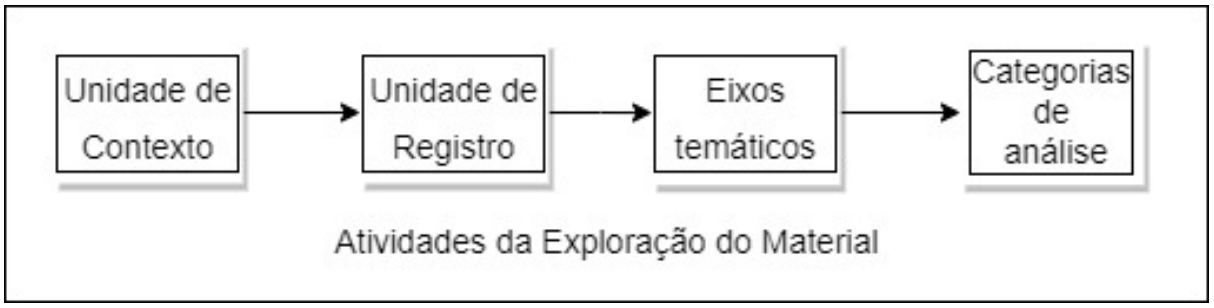

Figura 3.7 Exploração do material.
Fonte: (RODRIGUES, 2016)

Unidades de Contexto são os segmentos da mensagem que possuem dimensões superiores às da Unidade de Registro para permitir a compreensão do significado das mesmas. As unidades de contexto podem ser extraídos de um parágrafo, de uma frase, de um quadro, de uma imagem e os excertos são as unidades de classificação ou registros extraídas das unidades de contexto. As dimensões das Unidades de Contexto devem obedecer a dois critérios: custo e pertinência. A Unidade de Contexto não deve ser demasiado grande ou insuficientemente pequeno, estando pertinentes ao quadro teórico (BARDIN, 2011).

Unidades de Registro é a unidade de significação codificada, podendo ser temas, palavras, objetos e personagens, que são posteriormente usadas em operações de contagem ou enumeração, como presença ou ausência, intensidade, direção e ordem de aparição, permitindo categorização posterior para nortear discussões e interpretações (BARDIN, 2011; RODRIGUES, 2016).

Eixos Temáticos são feitos por meio de articulações e refinamento semântico das Unidades de Registros por meio de interpretação de similaridades em acordo com o universo e os objetivos da pesquisa. Sobre a importância dos eixos temáticos (BARDIN, 2011, p.41) afirma que "A análise de conteúdo pode ser uma análise dos "significados" (exemplo: a análise temática)[...]".

Categoria de Análise são resultados de operações de classificação dos elementos que constituem um conjunto por diferenciação para depois serem reagrupados com critérios definidos (BARDIN, 2011; RODRIGUES, 2016; FRANCO, 2020). Bardin também fala que essas Categorias podem ser vistas como classes ou rubricas que agrupam elementos (as Unidades de Registro) com um título genérico de acordo com as características comuns destes elementos.

O critério de categorização pode ser *semântico*, como em categorias temáticas, pode ser *sintático*, como em verbos e adjetivos, pode ser *léxico*, como em proximidade e sinônimos entre palavras, e expressivo como perturbações da linguagem.

É importante distinguir as unidades de registro com suas regras de enumeração. A unidade de registro é aquilo que se define e se conta, a enumeração é o modo como se conta e como se marca (BARDIN, 2011; RODRIGUES, 2016; FRANCO, 2020).

A regra de *presença (ou ausência)* analisa quando elementos estão presentes no texto de forma a servir como indicador. A regra da *frequência* é a mais usada e determina a importância de um elemento de acordo com a frequência de aparição. A *frequência ponderada* analisa a importância da frequência de aparição de um elemento de acordo com um grau de importância previamente determinado. A regra da *Intensidade* analisa a frequência de um elemento em determinados níveis semânticos de viés quantitativo de aparição no "texto". A regra de *direção* é analisada por meio da ponderação da frequência de forma qualitativa de acordo com polos direcionais (escala) demarcados por índices qualitativos. A regra da *Ordem* analisa a importância do elemento de acordo com a ordem de aparição das Unidades de Registros. A regra da *Coocorrência* analisa a presença simultânea de duas ou mais unidades de registro numa unidade de contexto (BARDIN, 2011).

Após realizadas as atividades da Exploração do Material, parte-se para a terceira fase da Análise de Conteúdo, a fase de **tratamento dos resultados e interpretação**. Tal fase tem como atividades um aprofundamento da categorização, aqui chamado de *Movimento Dialógico das Categorias de Análise* e a *Interpretação e Inferências*. A Figura 3.8 mostra as atividades na etapa de tratamento dos resultados.

O Movimento Dialógico é importante para aprofundar o conhecimento extraído das categorias de análise. Portanto, as categorias de análise devem ser estipuladas considerando princípios que as tornem fieis para a passagem de dados brutos para dados organizados (RODRIGUES, 2016). Também (BARDIN, 2011, p.149) versa que "Geralmente as categorias terminais provêm do reagrupamento progressivo de categorias com

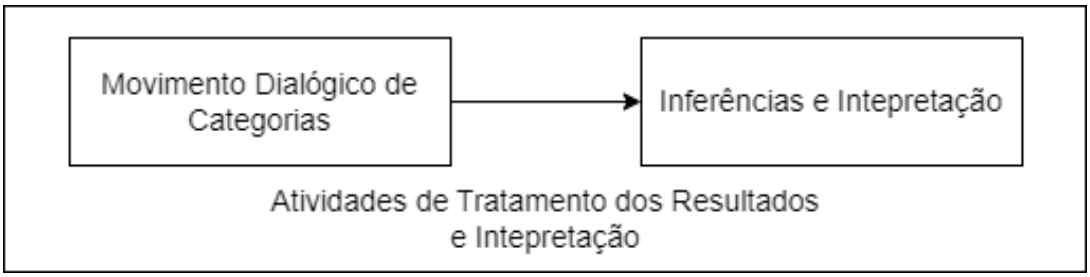

Figura 3.8 Tratamento dos resultados .
Fonte: (RODRIGUES, 2016)

uma generalidade mais fraca.". Ou seja, o cuidado no reagrupamento é iniciado a partir das escolhas de boas categorias. Bardin fala que um conjunto de boas categorias deve ter estas qualidades:

(i)*Exclusão mútua:* Este princípio deriva da importância de um elemento fazer parte de apenas uma divisão, não podendo se enquadrar em mais de uma categoria.

(ii)*Homogeneidade:* O princípio de exclusão mútua depende da homogeneidade das categorias. Um único princípio de classificação deve governar a organização (BARDIN, 2011, p.150).

(iii)*Pertinência:* As categorias devem ser pertinentes e, para tal, precisam estar adaptada ao material escolhido e pertencente ao quadro teórico definido, passando uma ideia de adequação ótima.

(iv)*Objetividade e fidelidade:* As diferentes partes de um mesmo material devem ser codificadas da mesma maneira quando nelas se aplicar a mesma grade categorial, sendo fieis, objetivas e aplicadas em toda a análise (BARDIN, 2011; MENDES; MISKULIN, 2017).

(v)*Produtividade:* Um conjunto de categorias é produtivo se fornece resultados férteis: em índices de inferências, em hipóteses e em dados exatos(BARDIN, 2011, p.150).

As inferências surgem na tentativa de tornar resultados brutos em dados significativos. O analista de posse dos resultados significativos podem usar as inferências para posteriores interpretações no que tange os objetivos previstos. Da mesma forma, os resultados obtidos e o confronto com o material e inferências alcançadas pode basear outras análises por meio de outras dimensões teóricas.

3.2.3.1 Estratégia de desenvolvimento da Análise de Conteúdo - Análise, transformação e reanálise. Aqui serão apresentados os procedimentos e técnicas que escolhemos para a análise de conteúdo da pesquisa, evidenciando os três movimentos de análises gerais de acordo com os objetivos estipulados.

O primeiro procedimento da análise de conteúdo foi a **Leitura Flutuante**. Por meio dela percorremos todos os componentes do *Corpus*, os mesmos provenientes dos contextos de coleta explicitados no início deste capítulo, identificando os aspectos teóricos e os recortes temáticos que mais chamaram atenção de acordo com os objetivos estipulados e com as metodologias conceituais das fontes de dados (FRANCO, 2020). Nesta fase, já seguimos um aspecto teórico bem definido para a especificação do universo da pesquisa.

Tal delimitação do universo de acordo com os pressupostos teóricos tornou a próxima etapa mais eficiente e objetiva.

Na fase da **Escolha dos Documentos**, fizemos a análise das decisões de escolhas do *Corpus* de acordo com as principais regras (exaustividade, representatividade, homogeneidade e pertinência). Salienta-se que todas essas regras foram observadas com cuidado. Nenhum RF nacional ou guia internacional mais atualizado ficou de fora do *corpus*, a amostra dos estudos sobre uso de FLOSS no ensino de ES foi selecionada de um mapeamento sistemático com regras metodológicas totalmente alinhadas com nossos objetivos de coleta, fazendo valer as regras principais da constituição do *corpus* abordadas por Bardin.

Os RFs Nacionais e os Currículos internacionais são os mais pertinentes possíveis da nossa investigação sobre as relações de competência e relevância da Engenharia de Software com a formação dos graduados em Computação. O mesmo se observa nos estudos selecionados do Mapeamento Sistemático de (BRITO et al., 2018), visto que o foco é exatamente na abordagem educacional alvo do nosso objetivo de pesquisa. Tal aspecto mostra que a regra da representatividade também foi observada, já que o universo da amostra das fontes selecionadas abarca um espaço de tempo relevante (cinco anos), em um dos idiomas mais utilizados para registros científicos (Língua Inglesa) e extraída de bases científicas relevantes.

Os RFs nacionais e guias internacionais seguem os mesmos princípios e métodos comuns ao seu próprio universo, o que também leva a crer a obediência ao princípio da homogeneidade. Todas as fontes descritas nos contextos de coleta se mostram pertinentes ao estudo, pois, por meio destas, é possível extrair os conteúdos e conhecimentos que mais se aproximam do nosso quadro teórico.

Após a fase de Escolha dos Documentos, partimos para a fase de **Preparação do Material**. Todo o contexto de coleta foi armazenado e identificado, a classificação tanto das fontes como os fragmentos de conteúdos mais alinhados com os objetivos foram armazenados em tabelas descritivas.

As fontes teóricas ou documentos base para a metodologia da construção dos Referenciais e Guias também tiveram seus dados contextualizados para basear os procedimentos. As informações dos estudos a cerca de Uso de FLOSS no ensino de Engenharia de Software foram referenciados e organizados em tabelas descritivas, com aspectos importantes identificados para posteriores análises e discussões.

Com os processos de pré-análise concluídos partimos para a **Exploração do Material**, com os processos qualitativos de identificação e seleção das unidades de contextos, das demarcações das unidades de registros e a análise destes para a obtenção dos eixos temáticos em linha com os objetivos da pesquisa e com o quadro teórico já delineado.

Após feitos os recortes e o alinhamento qualitativo das unidades de registros, das unidades de contexto e dos eixos temáticos com os aspectos teóricos e *insights* provenientes destes, partimos para os processos de análise e enumeração.

A abordagem descritiva das unidades de registro utilizada foi a temática, pois esta permite as diversas concepções e alinhamentos conceituais de termos e significados para os registros levantados.

A enumeração foi determinada de acordo com o objetivo esperado de cada contexto de

coleta, de acordo com as metodologias de desenvolvimento de cada um dos elementos do *corpus* ou de acordo com os aspectos teóricos alinhados ao objetivo de estudo proveniente.

Nos RFs nacionais, utilizamos a regra de **coocorrência** como técnica de enumeração. Esta foi possível após o levantamento qualitativo das unidades de registros associadas e passível de inferência de acordo com nossos objetivos. Nos Guias da Série *Curricula Computing* da ACM, a técnica de Enumeração das unidades de Registros é a **presença**, pois esta é a que mais se aproxima do objetivo de análise específico a este *corpus*. Nos estudos advindos do Mapeamento Sistemático, guia para nossa pesquisa (BRITO et al., 2018), a regra de enumeração escolhida foi a análise da **frequência**.

Após o levantamento qualitativo das unidades de Registros, partiu-se para as construções qualitativas dos eixos temáticos e depois a contabilização da frequência dos temas de acordo com algumas bases teóricas já conhecidas que serviram como guia para o processo qualitativo. Esse processo de levantamento qualitativo e posterior contabilização assume *a característica da abordagem mista* que esse estudo adota, chamada de Estratégia Transformadora Concomitante (CRESWELL, 2007).

As etapas de seleção ou fichamento dos dados, as classificações das unidades de contexto e unidades de registro, e os processos de enumeração e análise estão descritos na apresentação dos resultados nos próximos capítulos.

Após a fase de exploração do material, partimos para o **tratamento dos resultados**. Nesta fase é feita toda a reanálise dos dados já classificados e o movimento dialógico entre as categorias e temas levantados, as características de apresentação destes nos textos de acordo com a enumeração e refinamento dos registros e categorias, usando como base prerrogativas teóricas que auxiliem nos delineamentos adotados.

Por meio das unidades de registros, unidades de contexto e eixos temáticos levantados em cada contexto de análise, partimos para as inferências e interpretações diversas. Estas resumem no debruçar dos indicadores para assim significar os conteúdos caracterizados. A Tabela 3.4 resume as etapas da análise de conteúdo realizada e as decisões em cada processo.

A partir do estudo realizado, o diálogo com as perspectivas teóricas e com os fundamentos norteadores do percurso metodológico, é possível contextualizar as relações entres os elementos identificados e assim aplicar soluções aos problemas e questões de pesquisa levantados.

Tabela 3.4 Procedimentos da Análise de Conteúdo

Resumo dos procedimentos da Análise de Conteúdo				
Leitura Flutuante	- Escolha dos Documentos - Constituição do Corpus -Preparação do Material.	Exploração do Material		Inferência e Interpretação (De acordo com hipóteses e quadro teórico)
		Tipo de Análise	Tipo de enumeração	
	Referenciais de Formação (RF (SBC))	Tema	Coocorrência	Relação de Conteúdos e Competências
	Guias Curriculares (Série Currícula (ACM))	Tema	Presença	Relevância de Conteúdos e Competências
	Estudos selecionados Mapeamento Sistemático (BRITO et al., 2018)	Tema	Frequência	Relação de Aspectos de Interferência técnica ou pedagógica na aprendizagem.

Fonte:Elaboração Própria

3.2.3.2 Estratégia para organização dos conteúdos mapeados como critérios de avaliação e construção do *Framework* Conceitual. Após toda a análise de conteúdo nos diferentes elementos do *corpus* da pesquisa, parte-se para o diálogo conceitual e teórico entre os temas mapeados. Esse diálogo usa como base os objetivos da pesquisa, os problemas e questões levantados e alguns aspectos teóricos já debatidos em estudos norteadores. Essa etapa do estudo tem como objetivo responder a principal questão de pesquisa:

> **Q4** *Como organizar e modelar as competências e os aspectos mapeados como critérios para o educador avaliar a eficácia do uso de um Projeto FLOSS no ensino de Engenharia de Software?*

Esta etapa se deu com a tentativa de mapear conceitos em critérios, critérios esses que serão resumidos e significados para uma classificação conceitual que permita o docente observar, perceber e relacionar aspectos da sua prática educativa com os elementos mapeados.

Dos estudos sobre os Referenciais nacionais, extraímos conhecimento sobre a relação de competência técnica e sociotécnica da área de Engenharia de Software com as competências(necessidades) de formação. Dos Guias Internacionais extraímos conhecimento sobre a relevância da Engenharia de Software e dos conteúdos complementares(sociotécnicos) para a formação superior na área de computação. Dos estudos extraídos do Mapeamento Sistemático realizado por (BRITO et al., 2018) buscamos identificar relatos ou fatores de interferência na aprendizagem quando do uso de Projetos FLOSS para ensinar ES.

Após a organização dos conhecimentos extraídos, buscamos dialogar teoricamente e qualitativamente os aspectos de influência entre os conceitos ou elementos mapeados, como se percebe tal correlação e como estruturar de forma visual e pragmática a relação entre os conceitos mapeados. Para tal procedimento, usamos alguns guias teóricos de comparação, voltados ao pensamento por competência na engenharia de software como o Modelo de Competências em Engenharia de Software (SWECOM) (ARDIS et al., 2014a) e voltados a avaliação de aprendizagem como a Taxonomia de Bloom (FERRAZ; BELHOT et al., 2010; GALHARDI; AZEVEDO, 2013).

Essa estrutura conceitual e sistemática de procedimentos resume todos os passos adotados para a resposta da principal questão de pesquisa. Todo o delineamento metodológico aqui apresentado pode ser conceitualmente observado na figura 3.9.

Após a construção do modelo, partimos para a pré-análise do mesmo usando o diálogo teórico entre os elementos mapeados e elementos equivalentes discutidos em estudos guia. Nos capítulos a seguir apresentam-se todas as minúcias da pesquisa e a discussão dos resultados provenientes.

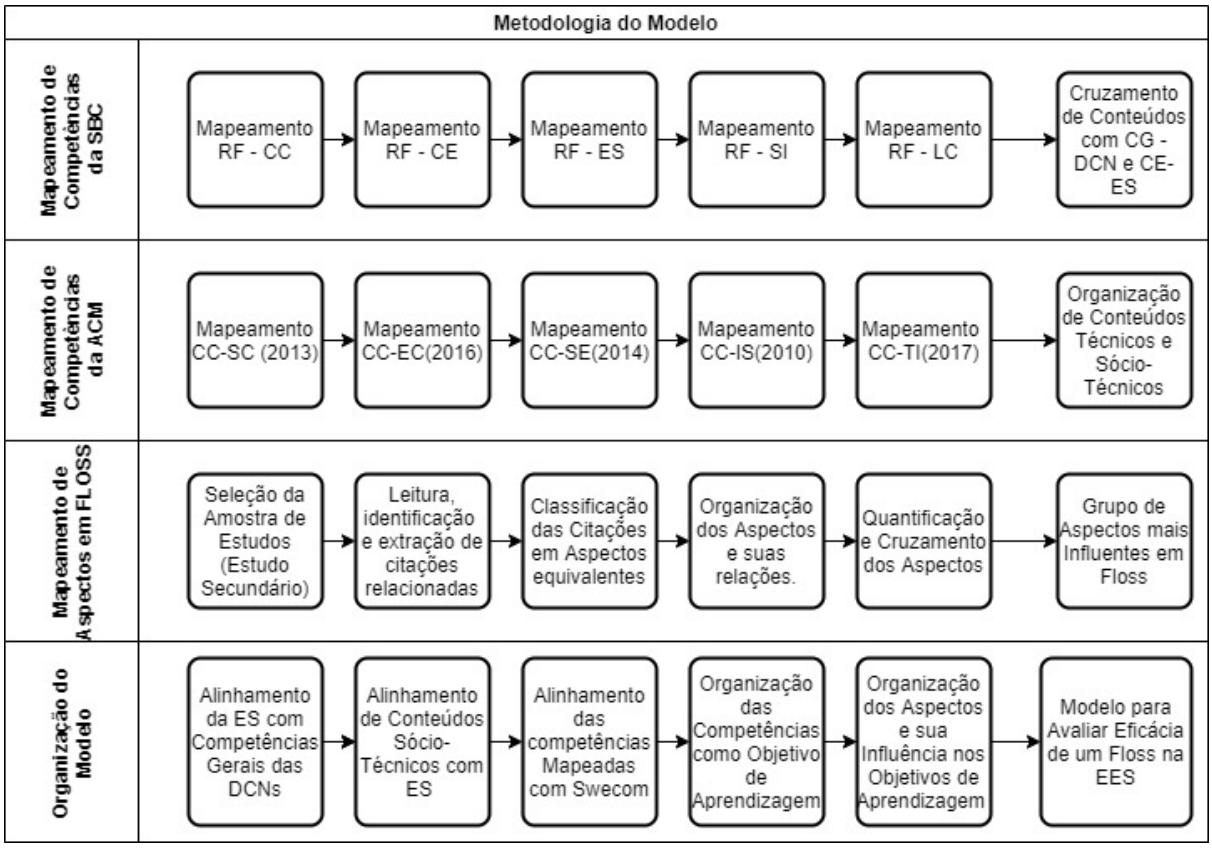

Figura 3.9 Resumo Metodológico de Construção do Modelo .
Fonte: Elaboração Própria

Capítulo

4

RESULTADOS

Este Capítulo apresenta todas as etapas do processo de alcance dos resultados. A Seção 4.1 apresenta o mapeamento das competências de Engenharia de Software (ES) segundo os Referenciais Nacionais. A Seção 4.2 apresenta o mapeamento das competências de ES e destaca sua relevância segundo os Guias internacionais. A Seção 4.3 apresenta o mapeamento os aspectos de interferência na aprendizagem em ES mediado por uso de Free/Libre Open Source Software (FLOSS). A Seção 4.4 apresenta o movimento de contextualização e diálogo dos elementos mapeados para o desenvolvimento de um `Modelo Conceitual`.

4.1 MAPEAMENTO DAS COMPETÊNCIAS DA ES SEGUNDO OS REFERENCIAIS DA SBC

Definidos e discutidos o planejamento dos aspectos que explicam a escolha das fontes bibliográficas em cada um dos contextos de coleta, parte-se para a descrição das estratégias planejadas e adotadas para análise dos dados em cada um dos contextos da pesquisa de acordo com o universo de análise.

Para compreender como os dados foram coletados e analisados, além de sua relação com nossos objetivos de estudo, é importante *compreender a estrutura conceitual de cada Referencial de Formação*. Os dois princípios básicos para todos os RFs são o *alinhamento com as Diretrizes Curriculares Nacionais* e *seguir um modelo baseado em competências*. É importante, também, discutir as visões que os Referenciais de Formação para os Cursos de Graduação em Computação (RF)s têm sobre as competências voltadas ao desenvolvimento dos futuros profissionais.

Os RFs seguem uma estrutura conceitual baseada em uma metodologia que *muda o paradigma de formação de currículos orientada por conteúdos* a serem assimilados por alunos *para uma orientação de competências esperadas aos egressos dos cursos* (ZORZO et al., 2017). Os autores versam sobre as diversas interpretações no que tange a definição de competências e por isso usam o modelo de referência baseado na `Taxonomia de Bloom Revisada` (FERRAZ; BELHOT et al., 2010).

Sobre as vantagens do uso de abordagem por competência, (ZORZO et al., 2017) fala da reconhecida capacidade de significar conteúdos de conhecimento que compõem o currículo, ampliando o mesmo para inclusão de habilidades e atitudes além do conhecimento, bem como de maior aderência ao perfil do egresso esperado pelos cursos. Embasados nos princípios citados, foi definida a estrutura conceitual usada para os RFs como mostra a Figura 4.1.

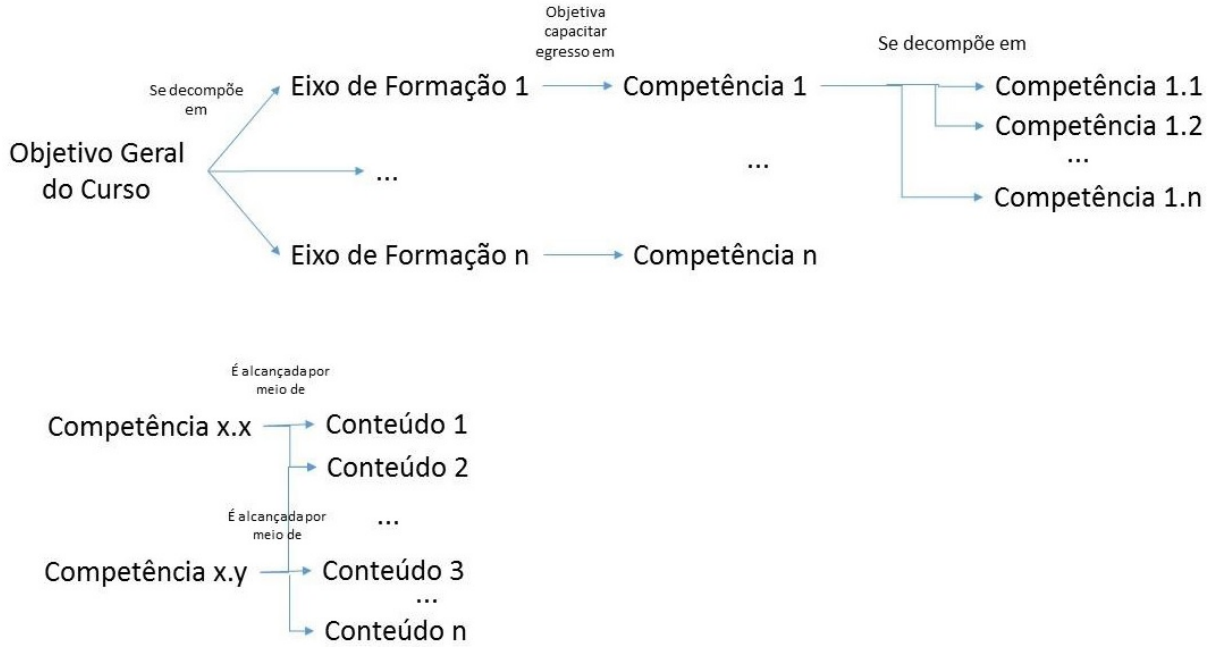

Figura 4.1 Estrutura Conceitual usada nos Referenciais de Formação
Fonte:(ZORZO et al., 2017, p.8)

Para explicar a estrutura conceitual utilizada (ZORZO et al., 2017, p.7) descreve o seguinte:

> Em linhas gerais, o **perfil** esperado para o egresso determina o objetivo geral do curso, decomposto em diferentes **eixos de formação**. Os eixos de formação objetivam capacitar o egresso em competências genéricas. Para alcançar cada **competência**, são relacionadas diversas **competências** derivadas, que determinam a necessidade de serem desenvolvidas em **conteúdos** específicos.

Ratificando, os Referenciais de Formação informam cada eixo de formação, as competências de eixo associadas, as competências derivadas cada um dos eixos, e todos os conteúdos associados a tais competências derivadas. Além da estrutura curricular, cada RF apresenta um quadro que relaciona as *Competências Gerais comuns a todos os cursos de Graduação em Computação segundo as DCNs (CG)* e as *Competências Específicas para cada Curso de Graduação em Computação segundo as DCNs (CE)* com as competências derivadas (C.n.n). Finalmente, cada eixo se decompõe em competências (ZORZO et al., 2017), apresentando uma estrutura como mostra a Figura 4.2.

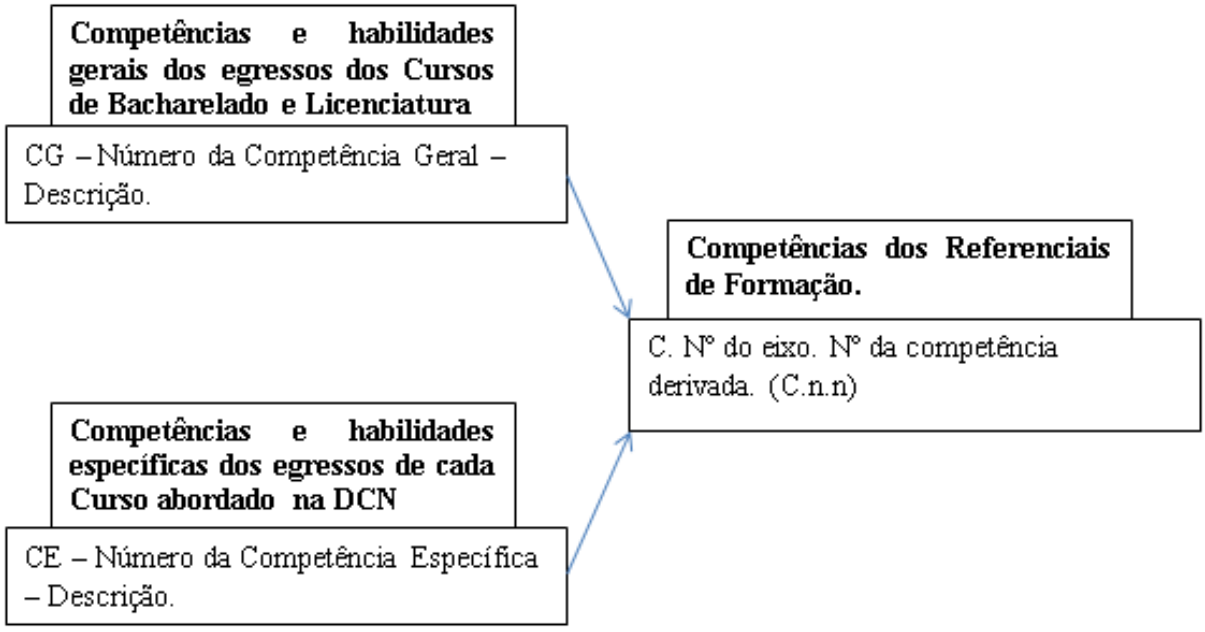

Figura 4.2 Estrutura conceitual de relação entre Competências de RFs com Competências Gerais e Específicas segundo as DCNs
Fonte: Elaboração Própria

Analisando a estrutura conceitual, adotamos a estratégia de fichamento e análise focada em identificar o grau de alcance que os conteúdos de Engenharia de Software têm para cada CG e CE em cada RF, além de compreender o nível de interação entre conteúdos sóciotécnicos e conteúdos de Engenharia de Software no alcance de competências esperadas segundo os Referenciais de Formação para Cursos de Bacharelado em Engenharia de Software (RF-ES). Em síntese:

- Analisamos cada Referencial de Formação (RF), buscamos os *conteúdos da área da Engenharia de Software* e identificamos competências derivadas nas quais estes conteúdos estão relacionados;

- Mapeamos as Competências Gerais (CG) segundo as DCNs que são estimuladas por tais conteúdos;

- Mapeamos as Competências Específicas (CE) nas DCNs que também são estimuladas por tais conteúdos no Referencial do Bacharelado em Ciência da Computação, de Engenharia da Computação e de Sistemas de Informação, além do curso de Licenciatura em Computação; e

- Mapeamos *exclusivamente as Competências Específicas ao Graduado em Engenharia de Software (CE-ES)* por meio de competências derivadas que são estimuladas por meio de conteúdos sociotécnicos em concomitância com conteúdos de Engenharia de Software no RF-ES.

A escolha pela estratégia de mapear conteúdos de ES com as Competências Gerais e Específicas esperadas por egressos de cada curso é fundamentada pelo próprio RF quando versa que "No Brasil, a área de Engenharia de Software está presente, como disciplina, em praticamente todos os currículos dos cursos da área de Computação" (ZORZO et al., 2017, p.57). Além disso, é reconhecido que todo profissional de Computação deve desenvolver competências e habilidades inerentes ao planejamento, desenvolvimento, avaliação e gerenciamento de software (CUKIERMAN; TEIXEIRA; PRIKLADNICKI, 2007; CAMARGO; FABRI, 2006; STAMELOS, 2011). Os conteúdos de Engenharia de Software foram mapeados de acordo com a terminologia de subáreas ou tópicos de Engenharia de Software segundo o Software Engineering Body of Knowledge - SWEBOK (BOURQUE; FAIRLEY et al., 2014). A Figura 4.3 apresenta a estrutura do método de mapeamento definido de conteúdos associados às CG (DCN) e CE-ES (DCN).

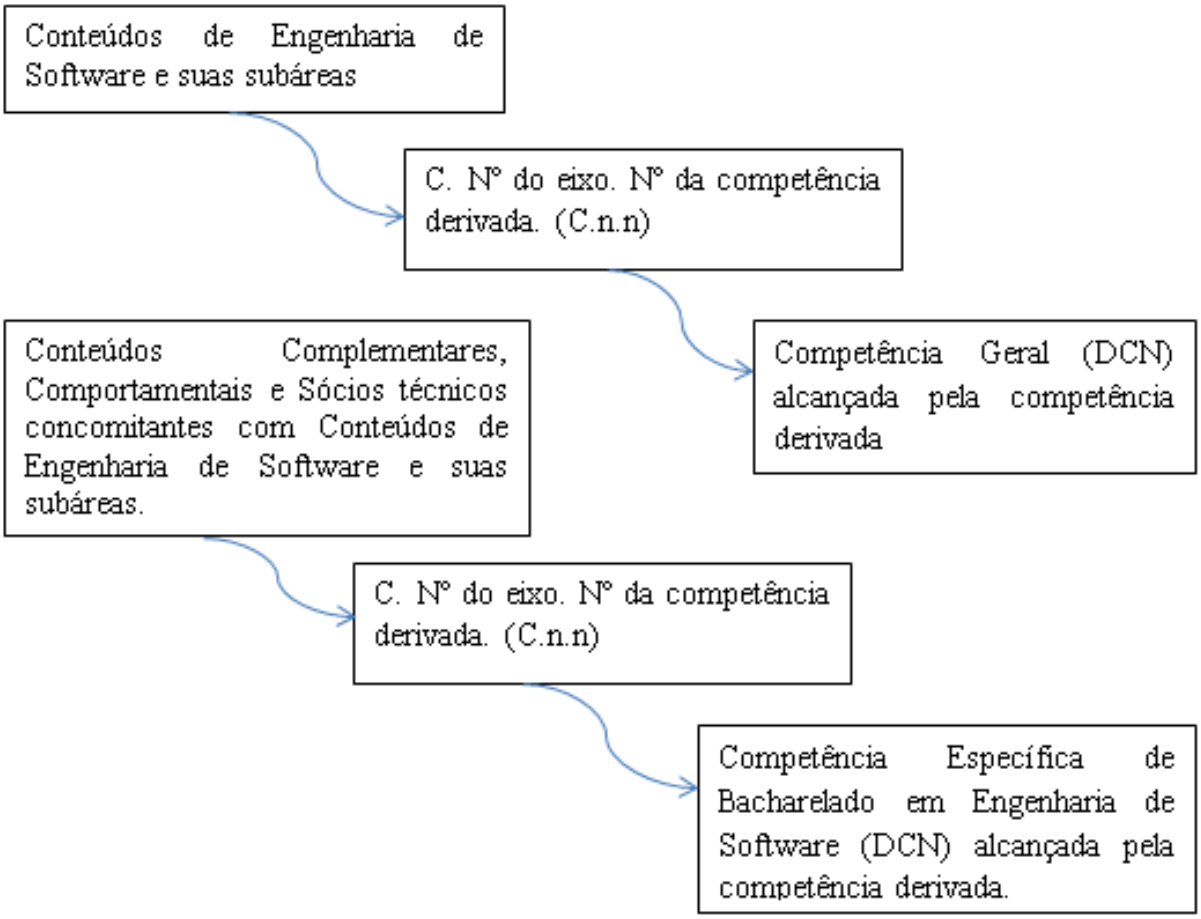

Figura 4.3 Método de mapeamento de conteúdos associados às CG (DCN) e CE-ES (DCN) (Fonte: Elaboração Própria)

É importante justificar o fichamento de apenas conteúdos de desenvolvimento social, profissional e comportamental (sociotécnico) relacionados com Competências Específicas esperadas pelos Egressos dos Cursos de Bacharelado em Engenharia de Software presentes nas DCNs (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22). As competências que

desenvolvem a boa formação de um profissional em Computação perpassam por habilidades técnicas e sociotécnicas. Um profissional da área de Computação e Engenharia de Software, especificamente, também deve desenvolver habilidades de aspectos profissionais, comportamentais e sociais esperados pelo mundo do trabalho (CUKIERMAN; TEIXEIRA; PRIKLADNICKI, 2007; YAMAMOTO et al., 2005).

O Bacharelado em Engenharia de Software é focado em atividades acadêmicas e científicas tanto de desenvolvimento profissional comum a qualquer área da Computação quanto o desenvolvimento de profissionais preparados para a indústria de software. Sobre isso, o próprio RF-ES versa que "[...] o curso de Bacharelado em Engenharia de Software visa a formação de profissionais qualificados para a construção de software de qualidade para a Sociedade." (ZORZO et al., 2017, p.58).

Ora, se a área de Engenharia de Software tem enfoque nos processos e conhecimentos para as boas práticas de desenvolvimento de software, um curso de bacharelado na área tende a agrupar as competências e habilidades direcionadas a tal princípio (ARDIS et al., 2014b; FIGUEIREDO et al., 2010; LESSA; JUNIOR, 2009). Portanto, percebe-se que todos os eixos de formação e todas as Competências específicas para egressos de Bacharelados em Engenharia de Software tendem a possuir vários conteúdos da área de ES.

Assim, buscamos descobrir quais competências são alcançadas por conteúdos complementares, comportamentais ou sóciotécnicos em concomitância com conteúdos da área de ES como mostra o esquema na Figura 4.3. Com essa estratégia, espera-se ter uma visão do alcance sociotécnico que a área da ES pode absorver e promover para a formação dos profissionais da Engenharia de Software e da Computação em geral.

É por meio dessa concepção que serão estruturados os fichamentos do texto deste universo de análise, a análise de conteúdo e estratégia de codificação. Antes de demonstrar o processo de codificação, é importante contextualizar a principal referência conceitual para o desenvolvimento dos RFs, a *Resolução Num.5 de 16 de Novembro de 2016*. Esta resolução, que trata das Diretrizes Curriculares Nacionais para os cursos de graduação na área da Computação (DCN), está organizada em artigos, parágrafos e incisos que abrangem diversos aspectos das diretrizes, como a organização dos projetos pedagógicos dos cursos, o perfil dos egressos, competências e habilidades esperadas, conteúdos curriculares, organização curricular, estágios e trabalhos de conclusão entre outros aspectos (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22).

É com base no Art.5 das DCNs que os referenciais organizam as competências esperadas para os egressos, tanto em nível de formação geral a qualquer graduado em Computação (CG) como em nível específico de formação dos egressos para cada curso (CE). Sobre os conteúdos, as Diretrizes Curriculares versam o seguinte:

> Art.6: Os currículos dos cursos de bacharelado e licenciatura da área da Computação deverão incluir conteúdos básicos e tecnológicos referentes à área da Computação, comuns a todos os cursos, bem como conteúdos básicos e tecnológicos específicos para cada curso, todos selecionados em grau de abrangência e de profundidade de forma consistente com o perfil, as competências e as habilidades especificadas para os egressos. (EDUCAÇÃO-MEC, 2016.

Seção 1. p. 22, p.2).

Observando essa afirmação, podemos dizer que os conteúdos selecionados nos currículos têm relação direta com o alcance dos objetivos de formação de cada curso de acordo com a abrangência e a profundidade pertinente com a vocação esperada para a formação dos futuros profissionais. Ou seja, os conteúdos da área da Engenharia de Software podem ter relação fundamental com esses objetivos de formação e com o perfil dos egressos a serem formados. Além disso, *a formação sociotécnica pode ter papel decisivo da efetiva formação dos egressos em acordo com as instruções das DCNs e das necessidades profissionais da Industria de Software.* Ainda sobre a compreensão dos conteúdos a serem incluídos, (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22, p.2) versa o seguinte:

> § 1º Estes conteúdos não consistem em disciplinas obrigatórias, mas no conjunto substantivo de conhecimentos que poderão ser selecionados pelas Instituições de Educação Superior para compor a formação dos egressos em cada curso em questão.

Tal afirmação instrui que os conteúdos podem ser articulados e combinados para prover uma formação condizente com o perfil de egresso no qual o curso objetiva, organizando um conjunto de conhecimentos que qualifique o egresso para as necessidades da indústria e do mundo do trabalho como um todo. Essa percepção fica mais clara quando se observa os objetivos da abordagem por Competências segundo os próprios referenciais de formação (ZORZO et al., 2017, p.7) que tratam da "[...]sua reconhecida capacidade em dar significado ou razão aos conteúdos de conhecimento que compõem o currículo".

Assim, se as competências e habilidades dão significado aos conhecimentos que advêm dos conteúdos, significa que as relações entre os conteúdos descritos nos (RF)s e as competências derivadas que os mesmos estimulam têm influência no desenvolvimento das competências gerais e específicas descritas nas (DCN)s.

Então, a partir de toda a essa análise, mostra-se conveniente conhecer as competências e os conhecimentos que, por meio de conteúdos de Engenharia de Software, são desenvolvidos. É também oportuno conhecer as competências especificas à formação em Engenharia de Software que são alcançadas por conhecimentos de teor sociotécnico. A Figura 4.4 apresenta a abordagem de análise usada neste trabalho. A hipótese que guiou tal abordagem foi:

> *Se o conhecimento em Engenharia de Software é norteador para a formação de competência geral e específica em cursos de graduação em Computação, é importante compreender o grau de importância que a formação sociotécnica tem para o desenvolvimento da própria formação em Engenharia de Software.*

Neste contexto, o objetivo da análise de conteúdo no contexto de análise de RFs foi definido:

> *Identificar as relações entre conteúdos da Engenharia de Software com competências esperadas aos graduados em Computação, além de identificar as relações entre conteúdos sociotécnicos com competências esperadas aos egressos dos cursos de graduação em Engenharia de Software.*

Depois de realizadas a leitura flutuante, parte-se para o fichamento das *unidades de contextos* e das *unidades de registros* extraídos dos eixos de cada um dos Referenciais de Formação. As unidades de contexto foram selecionadas seguindo a lógica metodológica dos RFs (Figura 4.3).

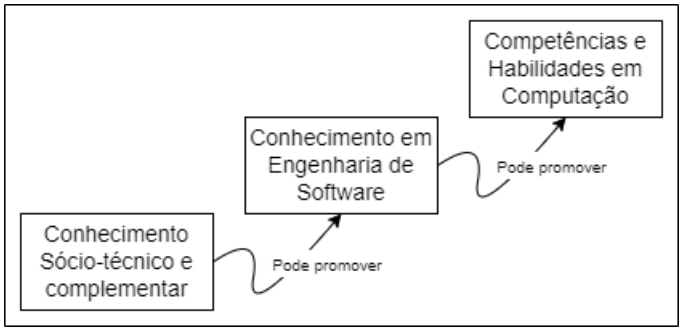

Figura 4.4 Abordagem hipotética de investigação
(Fonte: Elaboração Própria)

4.1.1 Mapeamento dos Referenciais de Formação para os Cursos de Graduação em Ciência da Computação (RF-CC).

Partimos para a Análise de Conteúdo dos Referenciais de Formação para os cursos de Graduação em Ciência da Computação. Presentes no mesmo documento dos demais referenciais aqui abordados (ZORZO et al., 2017), o RF-CC segue a mesma estrutura conceitual abordada para todos os referenciais.

Isso quer dizer que as competências esperadas aos egressos do curso segundo as Diretrizes Curriculares Nacionais foram organizadas em eixos de formação que por sua vez estrutura a competência do eixo. No eixo de formação se organizam as várias competências derivadas e estas organizam os conteúdos que auxiliam seu alcance. Porém, *este Referencial decidiu relacionar diretamente as competências derivadas com as competências gerais e específicas presentes nas DCNs* (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22). Sobre isso o RF-CC versa:

> A descrição de uma competência derivada é, em princípio, livre, desde que adequada ao correspondente eixo de formação. Os RF-CC-17, entretanto, descrevem uma competência derivada como sendo exatamente igual a alguma competência prevista nas DCN16. Essa abordagem visa explicitar a coerência e a complementaridade entre os dois documentos (ZORZO et al., 2017, p.11).

Explicitando esses detalhes característicos do RF-CC, partimos para as descrições das Unidades de Contexto e Unidades de Registro do RF-CC. Para a seleção das unidades de registros nos apoiamos nos seguintes índices:

- *Competências gerais ou específicas.* São descritas nas DCNs como competências esperadas aos egressos de todos os cursos de graduação (CG) e competências específicas (CE) esperadas aos egressos de cada um dos cursos abordados na Resolução;

- *Conteúdos da Engenharia de Software ou equivalentes.* São todos os conteúdos atribuídos a área de Engenharia de Software ou com termo e características semelhantes, seguindo terminologias da área segundo o corpo de conhecimento do SWEBOK, o Modelo de Competências em Engenharia de Software (SWECOM) e Referenciais anteriores, por exemplo, os Referencias da SBC de 1999 (BOURQUE; FAIRLEY et al., 2014; ARDIS et al., 2014a; SBC, 1999).

Para deixar claro o percurso do fichamento das unidades de contexto é importante recorrer ao esquema de identificação de competências e conteúdos no RF-CC. Esse esquema obedece a metodologia conceitual dos Referenciais. Sobre o esquema de identificação da estrutura conceitual, o (ZORZO et al., 2017, p.16) versa:

> Cada eixo de formação para os RF-CC-17 corresponde a uma macro competência e relaciona um grupo de competências derivadas (competências e habilidades oriundas das DCN16), as quais, se desenvolvidas em conjunto, levarão o estudante a atingir a competência do eixo. Em conjunto, possibilitam o egresso de um Bacharelado em Ciência da Computação a lidar profissionalmente com as várias facetas das atividades de Computação.

Complementando, os eixos tratados no RF-CC são estruturados por código numérico do eixo, o título, a descrição do eixo, a competência de eixo (Competência Associada ao Eixo de formação), as Competências derivadas. Sobre a identificação das competências derivadas observa-se a descrição segundo (ZORZO et al., 2017, p.17):

> **Competências derivadas**: lista de competências, oriundas das vinte e cinco competências e habilidades, gerais e específicas, definidas pelas DCN16, necessárias para construir a competência de eixo. As competências gerais das DCN16 são indicadas pelo identificador CG e as específicas do curso de Bacharelado em Ciência da Computação, pelo identificador CE. Por sua vez, cada competência derivada é constituída dos seguintes subcampos:
>
> - **código**: é formado pela junção da letra C (inicial da palavra "competência"), do código do eixo (1 a 7) e de um número indo-arábico que ordena sequencialmente a competência derivada no contexto do eixo de formação.
> - **classificação**: um dos seis níveis do processo cognitivo da Taxonomia de Bloom Revisada (Ferraz e Belhot, 2010).
> - **conteúdo**: lista de conhecimentos que devem ser trabalhados para desenvolver a competência derivada. Cada conteúdo é definido simplesmente por um título, oriundo da listagem constante nas seções 3.1 e 3.2 do Parecer CNE/CSE 136/2012 (MEC, 2012).

Conhecendo o esquema metodológico do RF-CC, é possível compreender como as unidades de registros foram extraídas dos Eixos de Formação. A tabela descritiva A.1 (Apêndice A) mostra a relação das Unidades de Contexto e Unidades de Registros extraídos de cada Eixo do RF-CC. A Relação de competências dos referenciais de formação com as competências descritas nas DCN são apresentadas em (ZORZO et al., 2017, p.33).

Por meio da Tabela de exploração do RF-CC, percebemos que várias competências Gerais e Específicas estão relacionadas com conteúdos de Engenharia de Software. Observa-se que não se encontram conteúdos de ES relacionados a competências no Eixo 1, possivelmente devido a competência do eixo ser direcionada a competências e habilidades voltadas a fundamentos matemáticos e problemas de soluções algorítmicas, aspectos mais profundos de aprendizagem teórica na Ciência da Computação (SBC, 1999; ZORZO et al., 2017; DRAFT, 2013).

O Referencial aborda um aspecto importante no que tange as competências alcançadas para o desenvolvimento dos eixos de formação: a metodologia de ensino aprendizagem. Este aspecto está diretamente relacionado com a temática de nosso trabalho, a saber, a metodologia de ensino de Engenharia de software por meio de projetos FLOSS. Sobre esse aspecto, o RF-CC versa:

> A competência 9 das DCN16 (*Adequar-se rapidamente às mudanças tecnológicas e aos novos ambientes de trabalho*) não foi relacionada como uma competência derivada em nenhum dos eixos de formação, pois se trata de uma habilidade que deve ser trabalhada por meio da metodologia de ensino-aprendizagem, e não por conteúdos específicos. Apesar disso, ela foi incluída na descrição da competência geral de três eixos de formação. (ZORZO et al., 2017, p.33)

Os eixos de formação que tem como Competência Alvo a Competência Geral IX da DCN são os eixos 3, 6 e 7. Como mostrado na Tabela A.1, estas competências de eixos também são alcançadas por competências derivadas que podem ser desenvolvidas por conteúdos de Engenharia de software e afins (qualquer conteúdo colaborativo no eixo 6) para o alcance da competência do eixo e, por isso, a Competência Geral IX foi extraída da unidade de contexto.

Essa relação pode indicar, além da necessidade de um projeto curricular preocupado com conteúdos efetivos no alcance de competências, a necessidade de combinar a aquisição de tais conhecimentos com metodologias efetivas para o desenvolvimento da aprendizagem dos estudantes (ANDRADE; SANTOS; LINHARES, 2015; NUNES; YARMAGUTI; NUNES, 2016; NASCIMENTO; BITTENCOURT; CHAVEZ, 2015).

Este adendo é importante para o movimento dialógico dos contextos da pesquisa que serão abordados em seções adiante. Após extraídas das unidades de contexto e agrupadas as unidades de registros semelhantes, listam-se na tabela descritiva 4.1 abaixo, as unidades de Registros identificadas.

Foram identificados onze Unidades de Registros após o agrupamento: duas unidades de registros representando Conteúdos de Engenharia de Software ou equivalente, e nove unidades de Registros representando Competências Descritas nas DCNs. É importante salientar que as Diretrizes Curriculares Nacionais listam vinte e cinco competências esperadas aos Egressos dos Bacharelados em Ciência da Computação, sendo doze competências Gerais e comuns a todos os Cursos e treze competências Específicas ao egresso do Bacharelado em Ciência da Computação. *Portanto, segundo nosso mapeamento, nove das vinte e cinco competências importantes para a formação superior em Ciência da Com-*

Tabela 4.1 Lista de Unidades de Registros do RF-CC

Unidades de Registros extraídas do Referencial de Formação para Graduação em Ciência da Computação	
1	Competência Geral IV
2	Competência Geral VII
3	Competência Geral VIII
4	Competência Geral IX
5	Competência Geral XII
6	Competência Específica IV
7	Competência Específica V
8	Competência Específica VII
9	Competência Específica VIII
10	Engenharia de Software
11	Conteúdo Aplicado por Práticas Colaborativas

Fonte:Elaboração Própria

putação podem ser estimuladas ou alcançadas direta ou indiretamente pelo conhecimento em Engenharia de Software.

Essa relação de coocorrência entre conteúdos de Engenharia de software com competências esperadas aos egressos mostra a vocação que a área pode assumir para uma formação superior em Ciência da Computação, condizente com as necessidades inerentes ao próprio perfil profissional dos egressos do curso e com as necessidades da indústria de software e do mundo do trabalho. Segundo (WANGENHEIM; SILVA, 2009, p.1), "[...] o ensino adequado de ES é importante para melhorar o estado atual de desenvolvimento de software e ajudar a mitigar muitos dos problemas tradicionais associados à indústria de software." A Tabela 4.2 apresenta as relações de coocorrência entre as unidades de registros de acordo com o mapeamento realizado e extraído das unidades de contexto.

Tabela 4.2 Tabela de coocorrência das Unidades de Registros no RF-CC

Competências	Eixos Formadores (Macro Competências) e Conteúdo Associado						
CG e CE das DCNs	Eixo 1	Eixo 2	Eixo 3	Eixo 4	Eixo 5	Eixo 6	Eixo 7
CG-IV		Eng de Software (C.2.2)					
CG-VII				Eng de Software (C.4.3)			
CG- VIII		Eng de Software (C.2.3)					
CG-IX			Eng de Software (C.3.8)(C.3.9)				Eng de Software (C.7.6)
CG-XII						Qualquer conteúdo que use práticas colaborativas (C.6.5)	
CE-IV		Eng de Software (C.2.7.)	Eng de Software (C.3.8)		Eng de Software (C.5.8)		Eng de Software (C.7.6
CE-V		Eng de Software (C.2.2)(C.2.3) (C.2.7)(C.2.8) (C.2.9)					
CE-VII		Eng de Software (C.2.8)		Eng de Software (C.4.8)			
CE-VIII		Eng de Software (C.2.9)	Eng de Software (C.3.9)				

Fonte:Elaboração Própria

Ao observar atentamente a tabela de coocorrências do RF-CC, percebemos que as competências que podem ser estimuladas por conhecimento em Engenharia de software

distribuem-se em seis dos sete eixos de formação determinados pelo referencial. Isso dá a entender que o conhecimento em Engenharia de Software pode auxiliar ao egresso no alcance das competências de eixos; estas, por sua vez, percorrem por diversas facetas de formação necessárias aos egressos do curso.

A importância da descoberta desse grau de alcance dos conhecimentos em Engenharia de software para tantas competências é descrita no próprio RF-CC, pois, segundo o mesmo referencial tais competências desenvolvidas "Em conjunto, possibilitam o egresso de um Bacharelado em Ciência da Computação a lidar profissionalmente com as várias facetas das atividades de Computação." (ZORZO et al., 2017, p.16).

4.1.2 Mapeamento dos Referenciais de Formação para os Cursos de Graduação em Engenharia da Computação (RF-EC).

O Referencial de Formação para Cursos de Bacharelado em Engenharia da Computação (RF-EC) compartilha vários aspectos com o RF-CC. Por exemplo, adota a metodologia conceitual comum a todos os referencias, com eixos, competências derivadas e conteúdos. O Referencial de Engenharia da Computação também tem como foco alinhar as competências de acordo com o quinto artigo das Diretrizes Curriculares Nacionais, informando as Competências Gerais comuns a todos os egressos (CG) e as Competências Específicas (CE) esperadas aos egressos de tais cursos. O RF-EC também é organizado por Eixos de Formação. Indica uma competência de eixo, alcançada por competências derivadas que, por sua vez, são alcançadas por meio dos conteúdos a serem selecionados. A identificação das competências derivadas se dá também pela letra C de competência, seguido pelo número do Eixo e pelo numero da competência em ordem. Na competência derivada encontram-se a classificação do processo cognitivo segundo a taxonomia de Bloom e a lista de conteúdos que podem servir ao alcance da competência derivada. Por meio da combinação das competências derivadas se alcançam as vinte e três competências gerais e específicas segundo as DCNs. A relação entre competências dos Referenciais de Formação com as competências descritas nas DCNs é descrita em (ZORZO et al., 2017, p.50). As unidades de contexto e unidades de registros associados estão descritos na tabela A.2 (Apêndice A).

Ao listarmos as unidades de registros, percebemos as constantes coocorrências entre conteúdos de Engenharia de software, competências gerais e específicas das DCNs e as competências derivadas. Percebe-se que, das vinte e três competências gerais e específicas das DCNs listadas no RF-EC, cerca de quatorze competências destas podem ser estimuladas por conhecimentos da área de Engenharia de Software. Algumas competências derivadas tiveram papel fundamental na relação entre a Engenharia de software e as competências das DCNs. As competências derivadas *C.3.3 (Identificar normas e documentações técnicas necessárias em projetos, serviços e experimentos de Engenharia de Computação)* e *C.3.4(Aplicar metodologias de gestão de projetos, serviços e experimentos de Engenharia na área de Computação)*, foram as competências que podem ser alcançadas por conteúdos de ES mais frequentes. A Tabela 4.3 apresenta a lista de Unidades de Registros do RF-EC.

Tabela 4.3 Lista de Unidades de Registros do RF-EC

Unidades de Registros extraídas do Referencial de Formação para Graduação em Engenharia da Computação	
1	Competência Específica I
2	Competência Derivada C.1.4
3	Requisitos de Sistema
4	Competência Específica V
5	Competência Específica VIII
6	Competência Específica II
7	Competência Derivada C.2.2
8	Engenharia de Software
9	Competência Específica VII
10	Competência Geral VII
11	Competência Derivada C.3.4
12	Ciclo de vida de produtos de software e Hardware
13	Técnicas para especificação de Requisitos
14	Competência Derivada C.3.3
15	Documentação técnica em projetos de Hardware e Software
16	Competência Geral VIII
16	Competência Geral IX
18	Competência Geral X
19	Competência Geral XI
20	Competência Geral XII
21	Competência Específica I
22	Competência Específica III
23	Competência Derivada C.3.1
24	Erros, motivos de fracasso e riscos envolvidos no projeto de sistemas de software e Hardware.
25	Competência Específica V
26	Competência Específica VII

Fonte: Elaboração Própria

Após o reagrupamento, obtivemos vinte e seis Unidades de Registros diferentes. Das vinte e três Competências Gerais e Específicas relacionadas ao curso de Bacharelado em Engenharia da Computação relatadas nas Diretrizes Curriculares nacionais, Quatorze tiveram competências derivadas associadas como meio de estímulo, competências derivadas estas que podem ser alcançadas por conteúdos da área de Engenharia de Software. Seis Unidades de Registros extraídas retratam conteúdos equivalentes a área de Engenharia de Software e cinco unidades de registros retratam as competências derivadas dos eixos de formação que estão relacionadas com as competências das DCNs. A partir do reagru-

pamento das unidades de registro no RF-ECs, partimos para a exibição das coocorrências na Tabela 4.4.

Tabela 4.4 Tabela de Coocorrência do RF-EC

Competências	**Eixos Formadores (Macro Competências) e Conteúdo Associado**				
CG e CE das DCNs	**Eixo 1**	**Eixo 2**	**Eixo 3**	**Eixo 4**	**Eixo 5**
CG VII			Ciclo de vida de produtos de software e hardware; Técnicas para Especificação de Requisitos (C 3.4)		
GG VIII			Ciclo de vida de produtos de software e hardware; Técnicas para Especificação de Requisitos (C 3.4). Documentação técnica em projetos de Hardware e Software (C 3.3).		
CG IX			Ciclo de vida de produtos de software e hardware; Técnicas para Especificação de Requisitos (C 3.4). Documentação técnica em projetos de Hardware e Software (C 3.3).		
CG X			Ciclo de vida de produtos de software e hardware; Técnicas para Especificação de Requisitos (C 3.4). Documentação técnica em projetos de Hardware e Software (C 3.3).		
CG XI			Ciclo de vida de produtos de software e hardware; Técnicas para Especificação de Requisitos (C 3.4).		
CG XII			Ciclo de vida de produtos de software e hardware; Técnicas para Especificação de Requisitos (C 3.4). Documentação técnica em projetos de Hardware e Software (C 3.3).		
CE I	Requisitos de sistemas (C 1.4)		Ciclo de vida de produtos de software e hardware; Técnicas para Especificação de Requisitos (C 3.4).		
CE II		Engenharia de Software (C.2.2)			
CE III			Erros , Motivos de fracasso e Riscos envolvidos no projeto de sistemas de software e hardware (C 3.1). Ciclo de vida de produtos de software e hardware; Técnicas para Especificação de Requisitos (C 3.4)		
CE V	Requisitos de sistemas (C 1.4)		Ciclo de vida de produtos de software e hardware; Técnicas para Especificação de Requisitos (C 3.4). Documentação técnica em projetos de Hardware e Software (C 3.3).		
CE VII		Engenharia de Software (C.2.2)	Ciclo de vida de produtos de software e hardware; Técnicas para Especificação de Requisitos (C 3.4). Documentação técnica em projetos de Hardware e Software (C 3.3).		
CE VIII	Requisitos de sistemas (C 1.4)				
CE IX	Requisitos de sistemas (C 1.4)				
CE X			Ciclo de vida de produtos de software e hardware; Técnicas para Especificação de Requisitos (C 3.4). Documentação técnica em projetos de Hardware e Software (C 3.3).		

Fonte:Elaboração Própria

É interessante observar na tabela de coocorrências que, diferente da distribuição de conteúdos encontrada no mapeamento do Referencial de Ciência da Computação, os conteúdos análogos a área de Engenharia de software se concentram com mais frequência

no Eixo 3.

Das vinte e cinco coocorrências entre conteúdos da Engenharia de software com as competências das DCNs, dezenove delas são indicadas no Eixo três, que tem como indicação de Macro competência "Gerenciar projetos, serviços e experimentos de Engenharia na área de Computação, de forma colaborativa em equipes multidisciplinares e em grupos sociais" (ZORZO et al., 2017, p.46).

Percebe-se que a ES pode assumir um papel norteador no desenvolvimento de competências para o efetivo gerenciamento de serviços e experimentos em Computação de forma colaborativa, ou seja, estimulando o aprendizado técnico alinhado com o desenvolvimento sociotécnico. Isso pode ser relevante para a compreensão do poder da Engenharia de software como área da Computação flexível e expansível para os propósitos de formação nas mais diversas áreas da Computação. Para discutir mais sobre esse aspecto vamos para o mapeamentos dos Referenciais de Formação para Bacharelados em Sistemas de Informação RF-SI.

4.1.3 Mapeamento dos Referenciais de Formação para os Cursos de Graduação em Sistemas de Informação (RF-SI).

A metodologia de construção deste referencial é o mesmo dos demais RFs, com Eixos de Formação e sua macro-competência associada, a distribuição de diversas competências derivadas que geram a competência do eixo e que estão diretamente associadas com as DCNs, seguida da classificação da abordagem de aprendizado segundo a Taxonomia de Bloom, e todos os conteúdos associados a cada competência derivada. Assim como nos demais RFs, cada competência derivada é identificada com a letra C de competência, número do eixo de formação e o número da competência. A relação entre cada competência derivada com as respectivas competências das DCNs são apresentadas em (ZORZO et al., 2017, p.129).

O referido referencial corrobora com as perspectivas já observadas no contexto teórico que guiou esse estudo, que é a capacidade de conhecimentos serem estruturados de forma estratégica para o desenvolvimento de competências compatíveis com a realidade de cada área de formação. Segundo (ZORZO et al., 2017, p.107) a abordagem de formação baseada em competências é "também utilizada pelas DCNs e amplamente empregada atualmente no que se refere à definição de objetivos de formação e aprendizagem". (ZORZO et al., 2017, p.107) continua ao afirmar que o RF-SI "Organiza as competências e conteúdos necessários para alcance do perfil do egresso em eixos de formação, com flexibilidade para detalhamento específico para as estratégias de cada curso". Ou seja, o conhecimento em Engenharia de software pode ser usado como objetivo de aprendizagem norteador para a formação dos egressos dos cursos de graduação em Sistemas de Informação.

Para investigar essa possibilidade, partimos para a exploração do RF-SI, extraindo as Unidades de Contexto e as Unidades de Registros do Referencial. Lembrando que as unidades de Contextos são os excertos de cada competência das DCNs seguida por competências derivadas que são alcançadas por conteúdos de Engenharia de Software. A Tabela A.3 (Apêndice A) mostra em detalhes a exploração do RF-SI.

É interessante observar uma grande quantidade de relações entre competências e conteú-

dos de Engenharia de software no eixo 3. Este tem como macro competência:

> Gerenciar os sistemas de informação em contextos sociais e organizacionais, avaliando as necessidades de informatização nestes sistemas, especificando soluções de software para sistemas de informação, produzindo o software para o atendimento destas necessidades, aplicando processos, técnicas e ferramentas de desenvolvimento de software, implantando o software em contextos sociais e organizacionais de sistemas de informação, mantendo sua operação e avaliando o impacto de seu uso (ZORZO et al., 2017, p.120).

Podemos observar diversos aspectos técnicos e sociais indicados como fatores de desenvolvimento da formação do estudante no eixo, estes fatores são alcançados pelas diversas competências derivadas distribuídas no eixo, onde algumas tem vários tópicos de Engenharia de software associados. Todos os conteúdos de Engenharia de software, competências derivadas e competências das DCNs que se associam foram agrupados para identificar as unidades de registros na Tabela 4.5.

Tabela 4.5 Lista de Unidades de Registros do RF-SI

URs extraídas do Referencial de Formação para Graduação em Sistemas de Informação			
1	Competência Geral I	20	Competência Derivada C.3.3
2	Competência Geral II	21	Competência Derivada C.3.4
3	Competência Geral III	22	Competência Derivada C.3.5
4	Competência Geral IV	23	Competência Derivada C.3.6
5	Competência Específica I	24	Competência Derivada C.3.7
6	Competência Específica II	25	Competência Derivada C.3.4
7	Competência Específica III	26	Competência Derivada C.4.3
8	Competência Específica IV	27	Competência Derivada C.4.3
9	Competência Específica V	28	Engenharia de Requisitos
10	Competência Específica VI	29	Arquitetura de Software
11	Competência Específica VIII	30	Projeto de Software
12	Competência Específica X	31	Verificação e Validação de Software
13	Competência Específica XI	32	Qualidade de Software
14	Competência Específica XIII	33	Gerência de Configuração de Software
15	Competência Específica XIV	34	Teste de Software
16	Competência Específica XV	35	Engenharia de Requisitos de Sistemas
17	Competência Específica XVI	36	Manutenção de Software
18	Competência Derivada C.3.1	37	Engenharia de Requisitos
19	Competência Derivada C.3.2	38	Construção de Software

Fonte: Elaboração Própria.

Após o reagrupamento das Unidades de Registros, temos trinta e oito unidades; destas, dezesseis são relacionadas com competências das DCNs, dez unidades são relacionadas a competências derivadas de eixos de formação do RF, e onze unidades são referentes a conteúdos da área de Engenharia de Software.

Percebemos maior de granularidade de conteúdos de Engenharia de Software no RF de Sistemas de Informação quando comparados ao RF de Ciência da Computação (dois conteúdos) e Engenharia da Computação (seis conteúdos). Isso pode indicar uma variação na estratégia curricular para o perfil de formação em Sistemas de Informação dentro da própria área, onde o educador pode selecionar os conhecimentos mais relevantes de acordo com a vocação institucional ou da estratégia de formação a ser adotada (WANGENHEIM; SILVA, 2009; BRANDÃO; BAHRY, 2005).

Além disso, temos várias competências das DCNs associadas a competências derivadas através do conhecimento da Engenharia de Software e suas sub-áreas. Das vinte e oito competências alocadas a formação dos bacharéis em Sistemas de Informação, dezesseis competências podem ser alcanças pelo conhecimento das mais diversas áreas da ES. A Tabela 4.6 mostra as relações de coocorrências das Unidades de Registros do RF de Sistemas de Informação.

Um aspecto observável nas coocorrências descritas na Tabela 4.6 é a grande influência da Engenharia de software que pode ser planejada para o alcance das competências específicas aos egressos de Bacharelados em Sistemas de Informação. De dezesseis Competências Específicas ao graduado em Sistemas de Informação (SI), treze podem ser alcançadas por competências desenvolvidas pelo conhecimento em Engenharia de software.
Isso pode confirmar a tendência que a Engenharia de Software tem para prover a formação não apenas de competências comuns a qualquer graduado na área da Computação, mas também prover formação de competências específicas nas suas subáreas.

Vamos continuar analisando essas e demais impressões explorando o Referencial de Formação para os Cursos de Graduação em Licenciatura em Computação.

Tabela 4.6 Tabela de coocorrência do RF-SI

Competências	Eixos Formadores, Competências e Conteúdos Associados						
CG e CE das DCNs	Eixo 1	Eixo 2	Eixo 3	Eixo 4	Eixo 5	Eixo 6	Eixo 7
CG I			Engenharia de Requisitos (C3.3); Arquitetura de Software (C.3.3); Projeto de Software(C.3.3)(C.3.4) ; Verificação e Validação de Software (C.3.3)(C.3.4); Qualidade de Software (C.3.3)(C.3.4); Gerência de Configuração de Software(C.3.3)(C.3.4); Teste de Software (C.3.4)				
CGII			Projeto de Software(C.3.4) ; Verificação e Validação de Software (C.3.4); Qualidade de Software (C.3.4); Gerência de Configuração de Software(C.3.4); Teste de Software (C.3.4)				
CG III			Projeto de Software(C.3.4) ; Verificação e Validação de Software (C.3.4); Qualidade de Software (C.3.4); Gerência de Configuração de Software(C.3.4); Teste de Software (C.3.4)	Engenharia de Requisitos (C.4.3)			
CG IV			Engenharia de Requisitos de Sistemas(C.3.1)				
CE I			Engenharia de Requisitos de Sistemas (C.3.1) (C.3.2); Qualidade de Software (C.3.2)(C.3.5)(C.3.6); Gerência de Configuração de Software(C.3.2)(C.3.6); Verificação e Validação de Software(C.3.5)(C.3.6); Manutenção de software(C.3.6); de Requisitos (C.3.6);Projeto de Software(C.3.6)				
CE II			Engenharia de Requisitos de Sistemas (C.3.1) (C.3.2); Qualidade de Software (C.3.2)(C.3.3)(C.3.4)(C.3.5)(C.3.6); Gerência de Configuração de Software(C.3.2)(C.3.3) (C.3.4)(C.3.6); Engenharia de Requisitos (C3.3)(C.3.6) ; Arquitetura de Software (C.3.3); Projeto de Software(C.3.3)(C.3.4)(C.3.6); Verificação e Validação de Software (C.3.3)(C.3.4)(C.3.5)(C.3.6); Teste de Software (C.3.4) Manutenção de software (C.3.6)	Engenharia de Requisitos (C.4.3)			
CE III			Engenharia de Requisitos de Sistemas(C.3.1) Engenharia de Requisitos (C3.3); Arquitetura de Software (C.3.3);Projeto de Software(C.3.3)(C.3.4); Verificação e Validação de Software (C.3.3)(C.3.4); Qualidade de Software (C.3.3)(C.3.4); Gerência de Configuração de Software(C3.3)(C.3.4); Teste de Software(C.3.4)	Engenharia de Requisitos (C.4.3)			
CE IV			Engenharia de Requisitos de Sistemas(C.3.1)	Engenharia de Requisitos (C.4.3)			
CE V			Qualidade de Software(C.3.5); Verificação e Validação de Software(C.3.5);				
CE VI			Engenharia de Requisitos de Sistemas (C.3.2); Gerência de Configuração de Software(C.3.2)(C.3.3); Qualidade de Software (C.3.2)(C.3.3); Engenharia de Requisitos(C.3.3); Arquitetura de Software (C.3.3); Projeto de Software (C3.3) (C.3.4); Verificação e Validação de Software (C.3.3)(C.3.4)	Engenharia de Requisitos (C.4.3)			
CE VIII			Engenharia de Requisitos de Sistemas(C.3.1)(C.3.2); Qualidade de Software (C.3.2)(C3.3)(C.3.7); Gerência de Configuração de Software(C.3.2)(C3.3)(C.3.4)(C.3.7); Engenharia de Requisitos(C.3.3)(C.3.7); Arquitetura de Software(C.3.3); Projeto de Software(C.3.3)(C.3.4)(C.3.7) ; Verificação e Validação de Software (C3.3)(C.3.4) Teste de Software(C.3.4)(C.3.7); Construção de Software(C.3.7); Manutenção de Software(C.3.7)				
CE X			Engenharia de Requisitos de Sistemas (C.3.2); Qualidade de Software (C.3.2)(C.3.5); Gerência de Configuração de Software(C.3.2); Verificação e Validação de Software (C.3.5)	Engenharia de Requisitos (C.4.3)			
CE XI			Engenharia de Requisitos de Sistemas(C.3.1)(C.3.2); Qualidade de Software (C.3.2)(C3.3)(C.3.4)(C.3.5)(C.3.7); Gerência de Configuração de Software(C.3.2)(C3.3) (C.3.4)(C.3.7); Engenharia de Requisitos(C3.3)(C.3.7); Arquitetura de Software(C.3.3); Projeto de Software (C.3.3)(C.3.4)(C.3.7); Verificação e Validação de Software(C.3.3)(C.3.5); Tesde de Softtware(C.3.4)(C.3.7);Construção de Software(C.3.7); Manutenção de Software(C.3.7);				
CE XIII			Engenharia de Requisitos de Sistemas (C.3.2); Qualidade de Software (C.3.2); Gerência de Configuração de Software(C.3.2)				
CE XIV			Engenharia de Requisitos de Sistemas(C.3.1)(C.3.2); Qualidade de Software (C.3.2)(C3.3) (C.3.4); Gerência de Configuração de Software(C.3.2)(C.3.3) (C.3.4); Engenharia de Requisitos(C.3.3); Arquitetura de Software (C.3.3); Verificação e Validação de Software (C.3.3) (C.3.4); Projeto de Software (C.3.3)(C.3.4)				
CE XV			Engenharia de Requisitos de Sistemas(C.3.1); Engenharia de Requisitos(C.3.7); Projeto de Software(C.3.7); Construção de Software(C.3.7);Teste de Software(C.3.7); Manutenção de Software(C.3.7);Qualidade de Software(C.3.7); Gerência de Configuração de Software(C.3.7)				
CE XVI			Manutenção de software(C.3.6);Engenharia de Requisitos(C.3.6); Projeto de Software(C.3.6); Qualidade de Software(C.3.6); Gerência de Configuração de Software(C.3.6); Verificação e Validação de Software(C.3.6);				

Fonte: Elaboração Própria

4.1.4 Mapeamento dos Referenciais de Formação para os Cursos de Licenciatura em Computação (RF-LC).

O Referenciais de Formação para Cursos de Licenciatura em Computação (RF-LC) é direcionado a um curso que tem como fundamento de formação a inserção de educadores da área da Computação, com o objetivo de prepará-los para a sociedade, podendo atuar em escolas e organizações como professores de Computação ou como agentes de integração ou estímulo do uso da tecnologia na educação (ZORZO et al., 2017). Segundo (MATOS, 2013, p.27):

> As habilidades para o mundo do trabalho no século XXI passam pelas tecnologias digitais de informação e comunicação (TDIC). Estas tem ampliado as possibilidades de comunicação e interação entre as disciplinas e áreas de conhecimento, na busca por soluções integradoras e desenvolvimento de competências adequadas para a realidade que se apresenta na atualidade.

Ou seja, além de competências comuns a qualquer formação em Computação, os Licenciados em Computação tem como objetivos desenvolver competências que apoiem a educação através de tecnologias contemporâneas, por meio de conhecimentos técnicos e científicos da área da Computação que possibilitem a articulação de tais conhecimentos com aqueles oriundos de sua formação pedagógica (ZORZO et al., 2017; MATOS, 2013). Nesse aspecto, podemos afirmar que o conhecimento em Engenharia de Software se torna relevante para o alcance desses objetivos.

Para compreender o grau de alcance da Engenharia de Software para as competências gerais e específicas dos egressos de Licenciatura em Computação, apresentamos o mapeamento destas por meio da exploração do RF-LC. A estrutura conceitual do RF é a mesma adotada pelos RFs já apresentados, seus eixos formadores, suas competências derivadas e sua identificação, e a lista de conteúdos associados. A relação entre as competências do RF-LC com as Competências Gerais e Específicas para Licenciatura em Computação das DCNs é apresentada em (ZORZO et al., 2017, p.97). Um aspecto nessa relação difere dos demais RFs. Enquanto os RFs até aqui explorados relacionam diretamente competências derivadas nos eixos com competências das DCNs, o RF de Licenciatura em Computação relaciona a competência geral do eixo com as competências das DCNs; estas descrevem as suas competências derivadas que, por sua vez, listam os conteúdos que as possibilitam. A Tabela A.4 (Apêndice A) mostra em detalhes as relações entre as competências e conteúdos da ES exploradas.

Podemos observar que no RF-LC existe uma concentração maior de unidades de registros no Eixo 5. Este eixo tem como competência Geral "Conceber, desenvolver, avaliar e gerir recursos tecnológicos para fins educacionais." (ZORZO et al., 2017, p.93). Tal relação pode indicar a Engenharia de Software como conhecimento auxiliador para gestão e inovação tecnológica de recursos educacionais da Computação. Sobre a importância dessa perspectiva, o RF versa o perfil do curso em formar egressos para o exercício da docência em Computação e "[...] para atuar no projeto, no desenvolvimento, na avaliação e na gestão de sistemas educacionais e de tecnologias contemporâneas, relacionadas à Computação articulada à Educação." (ZORZO et al., 2017, p.86). A partir da exploração do RF, extraímos e agrupamos as unidades de registros como mostra a Tabela 4.7.

Tabela 4.7 Lista de Unidades de Registros do RF-LC

Unidades de Registros extraídas do Referencial de Formação para Graduação em Licenciatura em Computação			
1	Competência Geral I	11	Competência Específica VI
2	Competência Geral III	12	Competência Específica VII
3	Competência Geral V	13	Competência Específica IX
4	Competência Geral VI	14	Competência Derivada C.2.1
5	Competência Geral XI	15	Competência Derivada C.5.2
6	Competência Geral XII	16	Competência Derivada C.6.6
7	Competência Específica I	17	Competência Derivada C.6.7
8	Competência Específica II	18	Competência Derivada C.6.9
9	Competência Específica III	19	Engenharia de Software
10	Competência Específica V	20	Todos os Conteúdos

Fonte:Elaboração Própria

Podemos observar que há vinte Unidades de Registros distintas após o agrupamento das mesmas. Oito unidades são referentes a competências derivadas de eixos de formação que são desenvolvidas por conteúdos de Engenharia de Software ou equivalentes. Das vinte e uma competências das DCNs para Licenciatura em Computação, dez delas são alcançadas por competências desenvolvidas pelo conhecimento em Engenharia de Software ou equivalente, como mostram as dez unidades de registros referentes a competências das DCNs extraídas. No RF-LC observamos apenas um conteúdo específico de ES extraído e um conteúdo que pode ser alocado a ES (*Todos os Conteúdos*). Após a extração, identificação e agrupamento das unidades de registros, partimos para a análise das coocorrências entre estes (Tabela 4.8).

Podemos observar a relação da Engenharia de Software com várias competências especificas aos egressos da Licenciatura em Computação segundo os DCNs. De nove competências específicas das DCNs para LC, sete delas podem ser estimuladas por competências desenvolvidas por conhecimentos em ES. Isso corrobora com a tendência da ES como uma área promovedora da gestão e da inovação de sistemas e recursos tecnológicos da Computação alinhados com o desenvolvimento das aptidões técnicas de docência em Computação. Podemos observar que as competências derivadas estimuladas por *Todos os Conteúdos* retratam aspectos de desenvolvimento sociotécnico. Tais competências derivadas promovem três das doze competências gerais segundo as DCNs. Isso é esclarecedor sobre a importância do desenvolvimento técnico e sociotécnico como mola mestra integrada de aprendizagem para a formação superior em Computação. Para continuar com novas observações partimos para a exploração do Referencial de Formação para os Cursos de Bacharelado em Engenharia de Software (RF-ES).

Tabela 4.8 Tabela de coocorrência do RF-LC

Competências CG e CE das DCNs	Eixos Formadores, Competências e Conteúdo Associado					
	Eixo 1	**Eixo 2**	**Eixo 3**	**Eixo 4**	**Eixo 5**	**Eixo 6**
CG I		Engenharia de Software (C.2.1)				
GG III		Engenharia de Software (C.2.1)				
CG V		Engenharia de Software (C.2.1)				
CG VI						Todos os Conteúdos (C.6.6)(C.6.7)(C.6.9)
CG XI						Todos os Conteúdos (C.6.6)(C.6.7)(C.6.9)
CG XII						Todos os Conteúdos (C.6.6)(C.6.7)(C.6.9)
CE I					Engenharia de Software(C.5.2)	
CE II					Engenharia de Software(C.5.2)	
CE III					Engenharia de Software(C.5.2)	
CE V					Engenharia de Software(C.5.2)	
CE VI					Engenharia de Software(C.5.2)	
CE VII					Engenharia de Software(C.5.2)	
CE IX					Engenharia de Software(C.5.2)	

Fonte: Elaboração Própria

4.1.5 Mapeamento dos Referenciais de Formação para os Cursos de Bacharelado em Engenharia de Software(RF-ES).

A metodologia de construção do RF-ES é a mesma usada nos demais referenciais. Além dos fundamentos matemáticos e de Computação, o RF destaca a importância dos conhecimentos em processos de produção baseados em Engenharia de sistemas e Engenharia de produção combinadas com as práticas já conhecidas da Engenharia de software (ZORZO et al., 2017). Isso mostra que a formação de um engenheiro de software conta com aprendizagem que vai além de seus próprios conhecimentos, sendo importante contribuir tanto com a Computação como também absorver demais conhecimentos advindos de áreas da Engenharia. Assim como nos demais RFs, a estrutura do RF conta com eixos e sua macrocompetência, as competências derivadas que formam a competência do eixo, o aspecto de aprendizagem baseado na Taxonomia de Bloom e os conteúdos associados. As competências derivadas também possuem a mesma classificação dos demais RFs (C.Número do Eixo.numero da competência).

Porém, o mapeamento do RF de Engenharia de Software passa por uma análise além dos RFs já mapeados. Para iniciar a contextualização, observamos o que versa (ZORZO et al., 2017, p.57): "A Engenharia de Software foca sua preocupação em manter o controle sobre todas as fases do processo de desenvolvimento do software por meio de métricas

voltadas ao controle produtivo dessas aplicações". Ou seja, a Engenharia de software, como uma das subáreas da Ciência da Computação, pode ser vista com relevância para a formação de qualquer profissional da área da Computação.

O objetivo do Bacharelado em Engenharia de Software (ZORZO et al., 2017, p.57) inclui a "[...] formação de profissionais qualificados para a construção de software de qualidade para a Sociedade". Para concluir, o RF versa que "A formação em Computação requer[...]conhecimentos das tecnologias utilizadas para a implementação e implantação de software" (ZORZO et al., 2017, p.58). Ou seja, o Bacharelado em Engenharia de Software não reflete apenas os objetivos de formação específico ao desenvolvimento de software, mas sim, se apropria de um dos objetivos essenciais da Computação.

Essa perspectiva nos possibilita usar a formação em Engenharia de Software como parâmetro de formação na área da ES como subárea de formação comum a qualquer área da Computação. Sendo assim, além da relação de conteúdos da ES com Competências Gerais constantes nas DCNs, investigamos as Competências Específicas aos egressos de Bacharelado em Engenharia de Software que podem ser alcançadas por conteúdos sociotécnicos em concomitância com conteúdos de ES. A relação entre as competências derivadas e as competências das DCNs são exibidas em (ZORZO et al., 2017, p.74).

Conhecer conteúdos sociotécnicos que se relacionem com competências que também podem ser alcançadas com a área da ES, pode trazer luz a uma vocação sociotécnica na formação das diversas áreas da Engenharia de Software. A formação sociotécnica é fundamental para o efetivo desenvolvimento de profissionais preparados para o mundo do trabalho e para a rápida inclusão na indústria de software (CUKIERMAN; TEIXEIRA; PRIKLADNICKI, 2007; WANGENHEIM; SILVA, 2009).

Quando se conhece aspectos sociotécnicos alinhados com a ES no alcance das competências específicas aos egressos de graduações em Engenharia de Software, permite-se compreender o grau de alcance e interação do conhecimento em ES alinhado ao aprendizado sociotécnico na formação superior em Computação. Quando se observa a relação direta entre o conhecimento técnico em ES e sua capacidade de interação com conhecimento sociotécnico, a relação de formação sobre as duas visões tende a não mais permitir dissociação sobre suas práticas.

Aprender Engenharia de Software sem conhecer e assumir seu lugar como profissional e como agente do desenvolvimento técnico e tecnológico no mundo globalizado tende a não ser mais suficiente para os objetivos de formação de profissionais da Computação preparados para os desafios de sua profissão. Dessa forma, os critérios para a seleção dos conteúdos e sua relação com a ES são:

- **Sociotécnicos.** Conteúdos de contexto comportamental ou profissional, conteúdos de contexto social e ético, conteúdos complementares de questões filosóficas, históricas, econômicas ou comerciais, conteúdos de contexto empreendedor e conteúdos de contexto científico ou de promoção do conhecimento.

  Que desenvolvam as competências derivadas do eixo 3 que representa como Competência Geral: *Conhecer os direitos e deveres de sua área de atuação, os melhores métodos de ensino, pesquisa e consultoria, saber trabalhar cooperativamente, além*

de negociar e se comunicar de forma eficaz, inclusive na língua inglesa. E que desenvolvam as mesmas competências derivadas outros conteúdos.

- **Técnicos.** Conteúdos específicos da Engenharia de Software e afins, Conteúdos de contexto da Engenharia e de produção, Conteúdos de contexto de visão sistêmica.

Os contextos sociotécnicos definidos como conteúdos a serem extraídos foram baseados em trabalhos que versam sobre os aspectos sociotécnicos, comportamentais e complementares na formação de competências da ES, como o SWECOM (ARDIS et al., 2014a) e estudos sobre visão sociotécnica e profissionais (CUKIERMAN; TEIXEIRA; PRIKLADNICKI, 2007; WANGENHEIM; SILVA, 2009; BRANDÃO; BAHRY, 2005). A escolha pelos conteúdos técnicos tem base na herança que a Engenharia de Software também adquire de princípios de construção e gestão de software com aspectos da Engenharia e de gestão de processos. Sobre isso, (ZORZO et al., 2017, p.57) afirma que:

> [...]devem ser estabelecidos e usados sólidos princípios de Engenharia para que se possa obter economicamente um software que seja confiável e que funcione eficientemente em máquina reais. [...]Os conhecimentos de processos de produção são baseados nas experiências de outras áreas, particularmente em Engenharia de Sistemas e Engenharia de Produção, que foram combinadas com as metodologias já bem-sucedidas da Engenharia de Software.

A Competência de eixo 3 comprova a importância que o desenvolvimento sociotécnico assume na efetiva formação na área da Engenharia de Software, o que fomenta a formação de futuros profissionais da Computação atentos a questões sociais, éticas e atitudinais no mundo do trabalho (CUKIERMAN; TEIXEIRA; PRIKLADNICKI, 2007; WANGENHEIM; SILVA, 2009). Na Tabela A.5 (Apêndice A), apresentamos as Unidades de Contextos e Unidades de Registros extraídos do RF de Engenharia de Software.

Há várias relações entre competências e conteúdos. Os conteúdos que mais se relacionam com Competências Gerais (DCN) estão concentrados no Eixo 3 e no Eixo 6. Segundo (ZORZO et al., 2017, p.65), o Eixo 3 tem como competência macro "Conhecer os direitos e deveres de sua área de atuação, os melhores métodos de ensino, pesquisa e consultoria, saber trabalhar cooperativamente, além de negociar e se comunicar de forma eficaz, inclusive na língua inglesa". Sobre a competência macro do Eixo 6, temos:

> Construir (criar, reusar e/ou integrar) software considerando o projeto e o uso de tecnologias e ambientes de desenvolvimento de software. O profissional de Engenharia de Software também deve ser capaz de realizar a avaliação (teste) do produto de software construído. (ZORZO et al., 2017, p.71)

Podemos perceber que o Eixo 3 espera competências mais voltadas para aspectos sociotécnicos, enquanto o Eixo 6 espera competências desenvolvidas com teor mais técnico da área da ES. Quando percebemos uma relação relevante das competências derivadas de ambos os eixos com as competências gerais das DCNs, fica latente a importância do desenvolvimento técnico alinhado ao desenvolvimento sociotécnico. Da exploração do RF-ES também podemos perceber que vários conteúdos de teor sociotécnico são indicados

para alcance de várias competências específicas aos Egressos de Bacharelados em ES. Para continuar a análise, partimos para o agrupamento das competências e conteúdos associados como mostra a Tabela 4.9.

Tabela 4.9 Lista de Unidades de Registros do RF-ES

Unidades de Registros extraídas do Referencial de Formação para Graduação Engenharia de Software			
1	Competência Geral III	34	Competência Derivada 6.3
2	Competência Geral IV	35	Competência Derivada 6.6
3	Competência Geral V	36	Competência Derivada 6.7
4	Competência Geral VI	37	Competência Derivada 6.9
5	Competência Geral VII	38	Competência Derivada 7.2
6	Competência Geral IX	39	Tomada de Decisão
7	Competência Geral XII	40	Empreendedorismo
8	Competência Específica I	41	Leis acordos e Instruções Normativas dobre ES
9	Competência Específica II	42	Métodos de Pesquisa e Experimentação em ES
10	Competência Específica III	43	Relato de estudos experimentais de ES
11	Competência Específica V	44	Técnicas de Comunicação
12	Competência Específica VI	45	Técnicas de Treinamento
13	Competência Específica VII	46	Técnicas de Consultoria
14	Competência Específica IX	47	Técnicas de Negociação
15	Competência Específica X	48	Competências Competitivas
16	Competência Específica XI	49	Técnicas de Ideação
16	Competência Específica XIII	50	Métodos e Técnicas de Especificação e modelagem da Interação com Usuários
18	Competência Específica XIV	51	Projeto de Arq e Reutilização de Software
19	Competência Derivada C.1.5	52	Técnicas de Revisão e Artefatos de Software
20	Competência Derivada 2.2	53	Ferramentas e Frameworks de Desenvolvimento e Gerenciamento de Software
21	Competência Derivada 2.3	54	Software como serviço
22	Competência Derivada 3.1	55	Princípios e Aplicações de Padrões em ES
23	Competência Derivada 3.2	56	Técnicas de verificação e análise de artefatos de software
24	Competência Derivada 3.3	57	Projeto (desing) de Interface com usuários
25	Competência Derivada 3.4	58	Sistemas de Sistemas
26	Competência Derivada 3.5	59	Melhoria Contínua e Gestão do Conhecimento
27	Competência Derivada 3.6	60	Técnicas de Avaliação de Produto
28	Competência Derivada 3.7	61	Noções Básicas de Direito
29	Competência Derivada 4.7	62	Conhecimento Científico
30	Competência Derivada 5.1	63	Análise Qualitativa
31	Competência Derivada 5.7		
32	Competência Derivada 6.1		
33	Competência Derivada 6.2		

Fonte: Elaboração Própria

Podemos observar que das doze competências gerais das DCNs, sete delas podem ser estimuladas por competências desenvolvidas por conhecimentos em Engenharia de software. Isso pode confirmar o potencial que o conhecimento em ES tem para estimular a aquisição de competências de Computação.

Das quatorze competências específicas aos egressos de Bacharelado em Engenharia de Software, doze delas são estimuladas por competências desenvolvidas por conhecimentos sociotécnicos concomitante com conhecimento de ES e afins. Isso pode ser um indicador do potencial que a área de Engenharia de software tem para se agregar a práticas e conhecimentos sociotécnicos nos processos educacionais. Além disso, tal aspecto mostra a relação direta de importância que o desenvolvimento sociotécnico do estudante tem para as próprias competências em ES a serem desenvolvidas. As relações de coocorrências entre as unidades de registro são mostradas na tabela 4.10.

Tabela 4.10 Tabela de Coocorrência do RF-ES

Competências	Eixos Formadores, Competências e Conteúdos Associados						
CG e CE	Eixo 1	Eixo 2	Eixo 3	Eixo 4	Eixo 5	Eixo 6	Eixo 7
das DCNs CG III						Proj de Arq e Reutilização de Software (C.6.1); Téc de Rev e Ana. de Artefatos de Software(C.6.2); Fer. e Frameworks de Des. e Ger. de Software(C.6.3); Software como Serviço(C.6.6); Princípios e Aplicações de Padrões em ES.(C.6.7); Técnicas de Verificação e análise de Artefatos de Software(C.6.9);	
CG IV			Leis Acor. Inst. Normativas Sobre ES (C.3.1)				
CG V							
CG VI			Leis Acor. Inst. Normativas Sobre ES (C.3.1); Met Pesq e Experimentação em ES (C.3.2); Relat Est Experimentais de ES (C.3.3)				
CG VII			Relat Est Experimentais de ES (C.3.3)				
CG IX			Leis Acor. Inst. Normativas Sobre ES (C.3.1); Met Pesq e Experimentação em ES (C.3.2); Relat Est Experimentais de ES (C.3.3)				
CG XII			Leis Acor. Inst. Normativas Sobre ES (C.3.1); Met Pesq e Experimentação em ES (C.3.2); Relat Est Experimentais de ES (C.3.3)				
CE I	Tomada de Decição (C.1.5)				Técnicas de Ideação (C.5.1)		
CE II						Projeto (design) de interface com usuários (C.6.1);	
CE IV			Noções Básicas de Direito (C.3.1)				
CE V							Técnicas de Avaliação de Produto (C.7.2)
CE VI						Software como Serviço(C.6.6);	
CE VII				Competências Competitivas (C.4.7)			
CE IX			Conhecimento Científico (C.3.2); Análise Qualitativa(C.3.3)				
CE X			Técnicas de Comunicação (C.3.4) Técnicas de Treinamento (C.3.5) Técnicas de Consultoria (C.3.6) Técnicas de Negociação(C.3.7)				
CE XI						Melhoria Contínua e Gestão do Conhecimento(C.6.7)	
CE XIII		Empreendedorismo (C.2.2) (C.2.3)					
CE XIV					Métodos, Tec. esp de Modelagem e Interação com Usuários (C.5.7)		

Fonte: Elaboração Própria

A tabela de coocorrência mostra as relações entre os conteúdos e as competências. Como já observado, uma quantidade maior de conhecimentos em ES é indicada para promover Competências Gerais (DCN) no Eixo Três. Isso pode indicar a importância dos conteúdos técnicos para uma formação consciente com a necessidade do aprendizado sociotécnico para o desenvolvimento dos egressos do curso superior de ES e para a Computação com um todo.

Observamos uma relação forte de conteúdos com a Competência Geral III (DCN) que indica "Resolver problemas usando ambientes de programação" (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22, p.4). Isso mostra uma possível vocação da ES e suas práticas como mola-mestra de desenvolvimento de competências técnicas de desenvolvimento usando ambientes de programação, independente do curso na qual esteja sendo aplicada.

Ao observar as competências específicas esperadas aos Bacharéis em Engenharia de Software, podemos constatar vários conteúdos sociotécnicos que são indicados como conteúdo promovedor da mesma forma que conteúdos técnicos da área. Podemos observar também a distribuição dessas relações em vários eixos.

Isso pode trazer luz ao potencial de desenvolvimento sociotécnico que o ensino de ES pode adquirir, mostrando-se uma área norteadora para a formação de graduados em Computação segundo as prerrogativas esperadas pelas DCNs, inclusive como versa o seu *Artigo Quarto*:

> Os cursos de bacharelado e de licenciatura da área de Computação devem assegurar a formação de profissionais dotados:
>
> I - de conhecimento das questões sociais, profissionais, legais, éticas, políticas e humanísticas;
>
> II - da compreensão do impacto da Computação e suas tecnologias na sociedade no que concerne ao atendimento e à antecipação estratégica das necessidades da sociedade;
>
> III - de visão crítica e criativa na identificação e resolução de problemas contribuindo para o desenvolvimento de sua área;
>
> IV - da capacidade de atuar de forma empreendedora, abrangente e cooperativa no atendimento às demandas sociais da região onde atua, do Brasil e do mundo;
>
> V - de utilizar racionalmente os recursos disponíveis de forma transdisciplinar;
>
> VI - da compreensão das necessidades da contínua atualização e aprimoramento de suas competências e habilidades;
>
> VII - da capacidade de reconhecer a importância do pensamento computacional na vida cotidiana, como também sua aplicação em outros domínios e ser capaz de aplicá-lo em circunstâncias apropriadas; e
>
> VIII - da capacidade de atuar em um mundo de trabalho globalizado[...] (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22, p.4)

Ou seja, é indubitável a importância do pensamento sociotécnico quando se planeja a Educação Superior em Computação, de atitudes profissionais à perspectiva de transformação social e de formação cidadã.

4.1.6 Cruzamento das Unidades de Registros Extraídas dos RFs (Competências e Conteúdos).

Para ter uma visão geral de todas as relações entre os conteúdos de ES ou afins com as competências que estas estimulam, vamos apresentar os cruzamentos de competências gerais e específicas (DCN) e conteúdos de ES mapeados de todos os referenciais por meio das coocorrências apresentadas na Tabela 4.11. A estrutura da tabela de coocorrência é organizada em competências gerais e específicas das DCNs, os referenciais mapeados e o código das competências derivadas (C.Número do eixo do RF. Numero da competência) que são desenvolvidas por conteúdos de Engenharia de software e conteúdos sociotécnicos e afins que estão marcados com (*). Para cada Unidade de Registro referente a conteúdos de ES e Sociotécnico relacionados a competência identificada será inserido um símbolo (+) ao lado do código da competência derivada.

Por meio da análise da Tabela 4.11, podemos obter uma visão do cruzamento das competências. Ao analisar a tabela, observamos que todas as competências Gerais esperadas aos Egressos de cursos superiores em Computação associam ao menos uma competência derivada que é desenvolvida por conhecimentos da ES em um dos RFs. Essa observação pode constatar o poder que a educação em Engenharia de software pode exercer na formação superior em Computação. Podemos observar associações de conteúdos a competências gerais variando de acordo com o RF, isso pode demonstrar a visão diversificada que cada curso pode associar para o desenvolvimento da formação dos estudantes.

Das doze competências gerais das DCNs, o RF de Ciência da Computação relaciona conteúdos de ES a cinco destas. O RF de Engenharia da Computação associa conteúdos de ES a seis competências gerais. O RF de Sistemas de Informação associa três competências gerais a conteúdos de ES. Já o RF de Licenciatura em Computação associa conteúdos de ES a seis competências gerais e o RF de Engenharia de Software associa conteúdos técnicos de ES a seis competências gerais.

Ao analisar as coocorrências é possível também observar as Competências Específicas das DCNs esperadas aos egressos de cada um dos cursos analisados. Das treze Competências Específicas das DCNs para egressos de Bacharelados em Ciência da Computação, quatro delas tem competências derivadas desenvolvidas por Engenharia de Software associada. O RF de Engenharia da Computação indica conteúdos de ES como promovedor de oito das doze competências específicas.

No RF de Sistemas de Informação, podemos observar conteúdos de ES sendo associados a treze das dezesseis competências específicas esperadas a egressos de Bacharelados em Sistemas de Informação. Este é o RF que mais associou ES a competências específicas de seu curso e mais teve conteúdos diversificados da Área de ES relacionados as competências. Das nove Competências Específicas relativas a Licenciatura em Computação, sete delas tem conteúdos de ES associados. Isto pode corroborar com a relevância da Engenharia de Software não somente como um promovedor de competências a atividades práticas e técnicas mas também com a relevância da ES para formar educadores na área da Computação.

Observemos agora as competências específicas para os egressos de Bacharelados em Engenharia de Software. Das quatorze competências indicadas pelas DCNs, onze são al-

Tabela 4.11 Tabela de Coocorrência Entre Competências e Conteúdos dos RFs e Competências das DCNs

Competências	Referenciais de Formação e Código de Competências estimuladas por conteúdos de ES ou Sociotécnicos* associados				
CG e CE das DCNs	RF-CC	RF-EC	RF-SI	RF -LC	RF- ES
CG I			(C3.3)++++++;(C.3.4)+++++	(C.2.1)+	
CG II			(C.3.4)+++++		
CGIII			(C.3.4)+++++;(C.4.3)+	(C.2.1)+	C.6.1)+;(C.6.2)+; (C.6.3)+;(C.6.6)+; (C.6.7)+;(C.6.9)+;
CG IV	(C.2.2)+		(C.3.1)+		(C.3.1)+
CG V				(C.2.1)+	
CG VI				(C.6.6)+;(C.6.7)+; (C.6.9)+	(C.3.1)+;(C.3.2)+; (C.3.3)+;
CG VII	(C.4.3)+	(C 3.4)++			(C.3.3)+;
CG VIII	(C.2.3)+	(C 3.4)++; (C.3.3)+			
CG IX	(C.3.8)+;(C.3.9)+; (C.7.6)+;	(C 3.4)++; (C.3.3)+			(C.3.1)+;(C.3.2)+; (C.3.3)+;
CG X		(C 3.4)++; (C.3.3)+			
CG XI		(C 3.4)++		(C.6.6)+;(C.6.7)+; (C.6.9)+	
CG XII	(C.6.5)+	(C 3.4)++; (C.3.3)+		(C.6.6)+;(C.6.7)+; (C.6.9)+	(C.3.1)+;(C.3.2)+; (C.3.3)+;
CE I		(C.1.4)+; (C.3.4)++;	(C.3.1)+;(C.3.2)+++;(C.3.5)++; (C.3.6)++++++;	(C.5.2)+	*(C.1.5)+;*(C.5.1)+;
CE II		(C.2.2)+	(C.3.1)+;(C.3.2)+++;(C.3.3)++++++; (C.3.4)+++++; (C.3.5)++; (C.3.6)++++++;(C.4.3)+	(C.5.2)+	*C.6.1)+;
CE III		(C.3.1)+;(C 3.4)++	(C.3.1)+; (C.3.3)++++++; (C.3.4)+++++;(C.4.3)+;	(C.5.2)+	
CE IV	(C.2.7)+;(C.3.8)+; (C.5.8)+(C.7.6)+		(C.3.1)+;(C.4.3)+;		*(C.3.1)+;
CE V	(C.2.2)+; (C.2.3)+;(C.2.7)+; (C.2.8)+;(C.2.9)+; (C.7.6)+;	(C.1.4)+; (C 3.4)++; (C.3.3)+	(C.3.5)++;	(C.5.2)+	*(C.7.2)+;
CE VI			(C.3.2)+++; (C.3.3)++++++; (C.3.4)+++++;(C.4.3)+;	(C.5.2)+	*(C.6.1)+;
CE VII	(C.2.8)+; (C.4.8)+;	(C.2.2)+		(C.5.2)+	*(C.4.7)+;
CE VIII	(C.2.9)+; (C.3.9)+;	(C.1.4)+;	(C.3.1)+;(C.3.2)+++;(C.3.3)++++++; (C.3.4)+++++;(C.3.7);+++++++		
CE IX		(C.1.4)+;		(C.5.2)+	*(C.3.2)+;*(C.3.3)+;
CE X		(C 3.4)++; (C.3.3)+	(C.3.2)+++;(C.3.5)++;(C.4.3)+		*(C.3.4)+;*(C.3.6)+; *(C.3.5)++;*(C.3.7)+;
CE XI			(C.3.1)+;(C.3.2)+++; (C.3.3)++++++;(C.3.4)+++++; (C.3.5)++;(C.3.7)+++++++;		*(C.6.7)+;
CE XII					
CE XIII			(C.3.2)+++;		*(C.2.2)+;*(C.2.3)+;
CE XIV			(C.3.1)+;(C.3.2)+++; (C.3.3)++++++;(C.3.4)+++++;		*(C.5.7)+;
CE XV			(C.3.1)+;(C.3.7)+++++++;		
CE XVI			(C.3.6)++++++;		

Fonte:Elaboração Própria

cançadas por competências desenvolvidas por conteúdos sociotécnicos ou complementares em concomitância a conteúdos específicos de Engenharia de software. Isso é norteador para a compreensão do aprendizado sociotécnico como mola mestra do desenvolvimento de competências esperadas aos futuros engenheiros de software. Tal observação, dialoga com a perspectiva de indissociabilidade entre o pensamento técnico e sociotécnico na educação em Engenharia de software, onde a relevância da formação nesta é reconhecida para o desenvolvimento profissional em Computação alinhado com as necessidades da indústria de software e do mundo do trabalho.

Após a observação das relações entre as unidades de registros extraídas partimos para a articulação e contextualização destas em eixos temáticos, agrupando-as e rotulando-as de acordo com as confluências de sentido, proximidade semântica ou congruência funcional, a fim de compreender as interações e articulações conceituais entre os temas.

4.1.7 Agrupamentos e Articulação das Unidades de Registros e Categorização nos RFs

Após mapearmos os RFs, extrairmos as unidades de contexto e as unidades de Registros, partimos para o processo de codificação. Por meio da análise temática, partimos para o refinamento das unidades de Registros extraídos em Eixos temáticos, onde os temas podem exercer relações e interações diretas e indiretas entre si, de acordo com a noção de coocorrência usada como enumeração para que seja possível a definição das categorias de análise. Para procedermos com a classificação das unidades de registros em eixos temáticos, precisamos organizar e agrupar as unidades com semelhança semântica ou terminológica. Para compreender os critérios dos agrupamentos é necessário a contextualização teórica que torna o agrupamento compreensível.

As competências determinadas pelas Diretrizes Curriculares Nacionais são referência para os RFs, que indica a necessidade de cada RF estar alinhado com as DCNs e num modelo baseado em competências. Dessa forma, agrupamos todas as unidades de registros provenientes das Competências Gerais (CGs) a qualquer egresso de cursos superiores em Computação e das Competências Específicas de cada um dos cursos superiores que tratam as DCNs e os RFs. O contexto desse agrupamento se dá pela característica temática na qual as mesmas estão inseridas. Para compreender esse contexto a as DCNs versam em seu quinto artigo que "Os cursos de bacharelado e licenciatura da área de Computação devem formar egressos que revelem pelo menos as competências e habilidades comuns[...]" (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22). Essa afirmação mostra que o objetivo de formação dos cursos é ter egressos que apresentem as competências Gerais e Específicas descritas no decorrer do referido artigo. Quando partimos para o princípio dos RFs estarem alinhados com as DCNs, percebemos que os RFs tem como alvo as competências descritas por estas. Dessa forma, chamaremos o tema que qualifica as unidades de registros relacionadas com as Competências Gerais e Específicas das DCNs de **"Competências Alvo"**.

Após agruparmos as unidades de Registros referentes a Competências das DCNs partimos para o agrupamento das unidades de Registros referentes a Competências Derivadas do eixos de cada um dos Referenciais Mapeados. As competências derivadas são aquelas que devem ser alcançadas para prover a competência do eixo que determina o perfil dos egressos de cada curso. Competências de eixo no RF formam o perfil do egresso e estão relacionadas diretamente com as necessidades de formação segundo o Artigo Quarto das Diretrizes Curriculares Nacionais (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22). Por sua vez as competências derivadas no RFs estão diretamente relacionadas com as competências das DCNs. Segundo (ZORZO et al., 2017, p.8), "Uma competência derivada corresponde, na prática, a uma competência que o egresso deve ter para atuar profissionalmente".

Quando partimos para a compreensão dessa afirmação podemos observar a visão de necessidade de formação segundo (ALVES, 2019, p.83) quando afirma que esta "Mais não é do que uma necessidade que se presume possa ser satisfeita por via de formação". Ora, se as Diretrizes Curriculares Nacionais versam da necessidade de profissionais com determinadas formações garantidas e com determinadas competências e estas estão diretamente relacionadas com uma ou várias competências derivadas segundo cada um

dos Referenciais de Formação, podemos perceber as competências derivadas como necessidades de formação, estas por sua vez auxiliam as competências alvo e compõem as competências que formam o perfil do egresso de cada curso. Sendo assim, agruparemos todas as Unidades de Registros provenientes de competências derivadas no eixo temático **"Necessidades de Formação"**.

Agora partimos para o agrupamento temático das Unidades de Registros referentes a Conteúdos de Engenharia de Software e Conteúdos Sociotécnicos ou Transversais. Para compreender a relação temática entre os conteúdos vamos analisar o Inciso primeiro do Sexto Artigo das Diretrizes Curriculares Nacionais que versa que "§ 1º Estes conteúdos não consistem em disciplinas obrigatórias, mas no conjunto substantivo de conhecimentos que poderão ser selecionados pelas Instituições de Educação Superior para compor a formação dos egressos em cada curso em questão" (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22, p.9). Para compreender a relação dos conteúdos com a formação dos egressos salienta-se que "[...] ter competência é a capacidade de um indivíduo em mobilizar recursos, tais como conhecimentos, habilidades, atitudes e valores [...]" (ZORZO et al., 2017, p.10).

Sobre essa perspectiva o RF-CC versa "[...]um conteúdo associado a certa competência do Bacharel em Ciência da Computação corresponde, intrinsecamente, a um ou mais desses recursos"(ZORZO et al., 2017, p.10). Ou seja, os conteúdos são conhecimentos selecionados e associados a competências e habilidades necessárias a determinados objetivos de formação. Segundo (ZORZO et al., 2017) os conteúdos específicos são necessários para prover as competências derivadas. Sendo assim, agrupamos as Unidades de Registros referentes a conteúdos de Engenharia de Software em um eixo temático com o nome **"Conhecimentos de Engenharia de Software"** e agrupamos as Unidades de Registros referentes a conteúdos Sociotécnicos no eixo temático **"Conhecimentos Sociotécnico"**. As Unidades de Registros extraídas de todos os RFs e agrupadas nos eixos temáticos são exibidas na Tabela 4.12.

A partir do agrupamento das Unidades de Registros em Eixos Temáticos, é possível continuar com a Categorização. Na Análise de Conteúdo Temática e Categorial no qual esta análise se debruça, a confluência dos temas e suas relações são determinantes para a concepção das conjecturas que aproximam ou afastam as características temáticas das Unidades de Registros extraídas de acordo com o contexto de análise. Segundo (BARDIN, 2011, p.201) a análise de categorias:

> Funciona por operações de desmembramento do texto em unidades, em categorias segundo reagrupamentos analógicos. Entre as diferentes possibilidades de categorização, a investigação dos temas, ou análise temática, é rápida e eficaz na condição de se aplicar a discursos diretos (significações manifestas) e simples.

Ou seja, a partir dos temas que emergem das Unidades de Registros, é possível exercer a análise das características que permitem a categorização destas. Sendo assim, partimos para a análise das relações contextuais dos eixos temáticos percebidos dos RFs.

O primeiro eixo temático (Competências Alvo) qualifica as Competências Gerais e Específicas das DCNs como foco de desenvolvimento profissional, afinal, as DCNs versam

Tabela 4.12 Tabela de Agrupamento de Unidades de Registros em Eixos Temáticos

Agrupamento das Unidades de Registros de Todos os RFs					
Unidades de Registro					Eixos Temáticos
RF-CC	RF-EC	RF-SI	RF-LC	RF-ES	
Competência Geral IV Competência Geral VII Competência Geral VIII	Competência Geral VII Competência Geral VIII Competência Geral IX	Competência Geral I Competência Geral II Competência Geral III	Competência Geral I Competência Geral III Competência Geral III	Competência Geral III Competência Geral IV Competência Geral V	
Competência Geral IX Competência Geral XII Competência Específica IV	Competência Geral X Competência Geral XI Competência Geral XII	Competência Geral IV Competência Específica I Competência Específica II	Competência Geral VI Competência Geral XI Competência Geral XII	Competência Geral VI Competência Geral VII Competência Geral IX	
Competência Específica V Competência Específica VII Competência Específica VIII	Competência Específica I Competência Específica II Competência Específica III	Competência Específica III Competência Específica IV Competência Específica V	Competência Específica I Competência Específica II Competência Específica III	Competência Geral XII Competência Específica I Competência Específica II	Competências Alvo
	Competência Específica V Competência Específica VII	Competência Específica VI Competência Específica VIII	Competência Específica V Competência Específica VI	Competência Específica III Competência Específica V	
	Competência Específica VIII Competência Específica IX Competência Específica X	Competência Específica X Competência Específica XI Competência Específica XIII	Competência Específica VII Competência Específica IX	Competência Específica VI Competência Específica VII Competência Específica IX	
		Competência Específica XIV Competência Específica XV Competência Específica XVI		Competência Específica X Competência Específica XI Competência Específica XIII	
				Competência Específica XIV	
	Competência Derivada C.1.4	Competência Derivada C.3.1	Competência Derivada C.2.1	Competência Derivada C.1.5	
	Competência Derivada C.2.2	Competência Derivada C.3.2	Competência Derivada C.5.2	Competência Derivada C.1.5	
	Competência Derivada C.3.4	Competência Derivada C.3.3	Competência Derivada C.6.6	Competência Derivada 2.3	
	Competência Derivada C.3.3	Competência Derivada C.3.4	Competência Derivada C.6.7	Competência Derivada 3.1	
	Competência Derivada C.3.1	Competência Derivada C.3.5	Competência Derivada C.6.9	Competência Derivada 3.2	
		Competência Derivada C.3.6		Competência Derivada 3.3	
		Competência Derivada C.3.7		Competência Derivada 3.4	
		Competência Derivada C.3.4		Competência Derivada 3.5	
		Competência Derivada C.4.3		Competência Derivada 3.6	
				Competência Derivada 3.7	Necessidades de Formação
				Competência Derivada 4.7	
				Competência Derivada 5.1	
				Competência Derivada 5.7	
				Competência Derivada 6.1	
				Competência Derivada 6.2	
				Competência Derivada 6.3	
				Competência Derivada 6.6	
				Competência Derivada 6.7	
				Competência Derivada 6.9	
				Competência Derivada 7.2	
Engenharia de Software Conteúdo Aplicado por Práticas Colaborativas	Requisitos de Sistema Engenharia de Software	Engenharia de Requisitos Arquitetura de Software	Engenharia de Software Todos os Conteúdos	Leis acordos e Instruções Normativas sobre ES Métodos de Pesquisa e Experimentação em ES	
	Ciclo de vida de produtos de software e Hardware Técnicas para especificação de Requisitos Documentação técnica em projetos de Hardware e Software Erros, motivos de fracasso e riscos envolvidos no projeto de sistemas de software e Hardware.	Projeto de Software Verificação e Validação de Software Qualidade de Software Gerência de Configuração de Software Tesde software		Relato de estudos experimentais de ES Projeto de Arquitetura e Reutilização de Software Técnicas de Revisão e Artefatos de Software Ferramentas e Frameworks de Desenvolvimento e Gerenciamento de Software Princípios e Aplicações de Padrões em ES	Conhecimentos de Engenharia de Software
		Engenharia de Requisitos de Sistemas		Técnicas de verificação e análise de artefatos de software	
		Manutenção de Software Engenharia de Requisitos Construção de Software			
				Empreendedorismo	
				Tomada de Decisão	
				Técnicas de Comunicação	
				Técnicas de Treinamento	
				Técnicas de Consultoria	
				Técnicas de Negociação	
				Competências Competitivas	
				Técnicas de Ideação	
				Métodos e Técnicas de Especificação e modelagem da Interação com Usuários	
				Software como serviço	
				Projeto (desing) de Interface com usuários	Conhecimentos Sociotécnicos
				Sistemas de Sistemas	
				Melhoria Contínua e Gestão do Conhecimento	
				Técnicas de Avaliação de Produto	
				Noções Básicas de Direito	
				Conhecimento Científico	
				Análise Qualitativa	

Fonte: Elaboração Própria

tais competências como aquelas esperadas aos egressos dos cursos. Quando analisamos o eixo temático (Necessidades de Formação), tratamos das Competências Derivadas descritas nos Eixos de Formação dos Referenciais de Formação. Segundo (ZORZO et al., 2017, p.11), "Uma competência derivada corresponde, na prática, a uma competência que o egresso deve ter para atuar profissionalmente". Para que o perfil dos egressos sejam alcançados, é necessário o alcance destas competências de forma articulada.

Além desta afirmação, (ZORZO et al., 2017, p.16) também versa que as competências derivadas são uma "lista de competências, oriundas das vinte e cinco competências e habilidades, gerais e específicas, definidas pelas DCN16, necessárias para construir a competência de eixo." Ora, se as necessidades de formação são as competências oriundas daquelas descritas pelas DCNs e que são articuladas para desenvolver o perfil dos egressos (também segundo o que preconizam as DCNs), podemos afirmar que as competências das DCNs são alcançadas quando as necessidades de formação são satisfeitas, ou seja, quando as competências derivadas são alcançadas.

Então, percebemos que Competências Alvo são alcançadas pelo desenvolvimento de formação necessária a seu alcance (Competências Derivadas). Uma competência Derivada (necessidade de formação) é alcançada por meio de conteúdos. Segundo (ZORZO et al., 2017, p.), cada competência derivada, por sua vez, requer a mobilização de um conjunto de recursos, o que é materializado pelos conteúdos associados. O documento também define os conteúdos como competências a serem mobilizadas em forma de recursos como conhecimentos, habilidades atitudes e valores. As diretrizes curriculares versam que um conteúdo é um conjunto de conhecimentos selecionados.
Os RFs versam sobre definição de competência segundo a Taxonomia de Bloom. Para eles uma Competência "[...] pode expressar o conhecimento, as habilidades ou as atitudes esperadas do egresso do curso, sob a perspectiva de objetivos de aprendizagem (o que o aluno será capaz de)". Ora, se as DCNs afirmam que conteúdos são conhecimentos e os RFs afirmam que conhecimentos e atitudes são competências, podemos perceber que, no contexto dos RFs, conteúdos são conhecimentos que, por sua vez, são competências que auxiliam o alcance de competências derivadas (Necessidades de Formação). Ou seja, os Conhecimentos em Engenharia de Software e Conhecimentos Sociotécnicos são competências que devem ser desenvolvidas para suprir as necessidades de formação.

A partir do momento que o efetivo conhecimento em ES depende da articulação não apenas de competências técnicas mas também de competências sociotécnicas (CUKIERMAN; TEIXEIRA; PRIKLADNICKI, 2007), vemos uma relação entre o conhecimento sociotécnico com o efetivo desenvolvimento de Competências em Engenharia de Software. Ao observar todas essas confluências, percebemos que todos os eixos temáticos nada mais são que competências que vão se estruturando de forma ascendente até alcançar as competências das DCNs. *Então, podemos relacionar os eixos temáticos extraídos numa mesma categoria de análise, a Categoria* ***(COMPETÊNCIAS)***. Essa classificação pode ser observada na Tabela 4.13.

Tabela 4.13 Agrupamento de Eixos Temáticos extraídos em Categorias.

Eixos Temáticos	Categorias
Competências Alvo	Competências
Necessidades de Formação	
Conhecimentos em Engenharia de Software	
Conhecimentos Sociotécnicos	

Fonte: Elaboração Própria

Finalmente, chegamos à Categoria de Análise, Competências. Esta Categoria resume todas as relações entre os fragmentos de informação relevantes para a respostas às nossas investigações extraídas dos Referenciais de Formação. Segundo (BARDIN, 2011, p.61) "[...] na análise convém classificar as unidades de significação criando categorias, introduzindo uma ordem suplementar reveladora de uma estrutura interna". Ou seja, ao determinarmos Competências como uma categoria de análise, exercemos uma temática geral de classificação de toda a estrutura de competências relacionadas com as práticas educativas em Engenharia de Software em convergência com a metodologia de construção dos referenciais.

Essa classificação temática responde nossa primeira Questão de Pesquisa, ou seja, a articulação entre o conhecimento técnico e sociotécnico da Engenharia de Software alinhados com as necessidades de formação de cada curso estruturam as doze Competências Gerais comuns a todos os graduados em computação e as várias Competências Específicas esperadas pelos egressos de cada Curso. A representação dessa articulação de competências é apresentada na Figura 4.5.

Figura 4.5 Articulação de Competências da Engenharia de Software
Fonte: Elaboração Própria

A partir desta categorização, partimos para a análise da relevância que a Educação em Engenharia de Software (EES) em seus aspectos técnicos e aspectos transversais pode exercer para a efetiva formação superior em Computação por meio da análise de conteúdo dos principais guias curriculares internacionais.

4.2 MAPEAMENTO DAS COMPETÊNCIAS DA ES SEGUNDO GUIAS CURRICULARES INTERNACIONAIS

Ao mapear as competências relacionadas com a Educação em Engenharia de Software nos principais Referenciais de Formação da SBC, percebemos *a relação entre conteúdos específicos de Engenharia de Software e conteúdos Sociotécnicos com as competências esperadas aos egressos dos cursos superiores em Computação segundo as Diretrizes Curriculares Brasileiras*. Porém, ao analisarmos as competências das DCNs nos RFs, percebemos que não apenas conteúdos de ES podem ser usados para alcance destas competências, mas sim, conteúdos de outras áreas de Computação, como Redes, Sistemas de Informação Arquitetura de Computadores entre outras (ZORZO et al., 2017). Sobre isso, o Artigo Sexto das Diretrizes Curriculares Nacionais versa:

> Os currículos dos cursos de bacharelado e licenciatura da área da Computação deverão incluir conteúdos básicos e tecnológicos referentes à área da Computação, comuns a todos os cursos, bem como conteúdos básicos e tecnológicos específicos para cada curso, todos selecionados em grau de abrangência e de profundidade de forma consistente com o perfil, as competências e as habilidades especificadas para os egressos.

Ou seja, a seleção dos conteúdos em abrangência e profundidade determinam o perfil e as competências dos egressos. Tendo na ES uma subárea da Computação com reconhecida relevância para o mundo do trabalho, é importante mensurar também sua importância para a organização curricular e planejamento da formação em Computação.

Já é perceptível a influência da ES na educação em Computação segundo os Referenciais Nacionais, quando estes atribuem algum conhecimento da área de ES como promotor de todas as Competências Gerais de Formação das DCNs. Torna-se importante analisar qual a relevância que os principais guias internacionais indicam aos conteúdos de ES e aos conteúdos sociotécnicos na formação dos estudantes dos principais cursos superiores em Computação internacionais.

Para a nossa análise, vamos nos debruçar sobre os principais Currículos da Série Curricula da ACM, o Currículo de Computação de 2013, o Currículo de Engenharia da Computação de 2016, o Currículo de Sistemas de Informação de 2010, o Currículo de Tecnologia da Informação de 2017 e o Currículo de Engenharia de Software de 2014. Esses Guias foram desenvolvidos sobre as bases do Currículo de Computação (2005).

O *Computing Curricula 2005* é um relatório de visão geral que tem como um de seus princípios auxiliar no desenvolvimento da identidade da Computação por meio do reconhecimento de uma identidade para cada área ou disciplina da Computação (SHACKELFORD et al., 2005). O documento traz princípios norteadores para o desenvolvimento curricular de referenciais sobre cinco cursos da área da Computação, sendo elas, a área de Ciência da Computação, Engenharia da Computação, Sistemas de Informação e Tecnologia da Informação e Engenharia de Software.

O *Curricula Computing* versa sobre a importância da Engenharia de Software como subárea relevante para a Computação. O documento versa a importância da ES como disciplina norteadora para a aplicação de métodos e processos rígidos provenientes das

Engenharias, mas preocupada com o desenvolvimento de ciência e conhecimento que torne o desenvolvimento de software complexo compatível com as demandas da Sociedade (SHACKELFORD et al., 2005).

Para compreender a importância das principais áreas da Computação em cada um dos cinco cursos abordados, o Relatório geral organiza uma visão tabular que mostra a ênfase de tópicos em Computação e de tópicos não relacionados a Computação, como Negócios e Eletrônica. Foram organizados cerca de quarenta tópicos diversos e relacionados com cada um dos cursos por meio de pesos atribuídos a uma coluna de valor mínimo e máximo. Os pesos são escalonados em valores de zero a cinco. O valor mínimo representa a ênfase mínima que deve ser atribuída por esse tópico a cada curso, e valor máximo representa a ênfase pode ser atribuída ao tópico segundo o relatório. Desta tabela foram retirados os tópicos ou conhecimentos íntimos com área de ES e organizados os pesos mínimos e máximos que cada tópico tem em cada curso segundo o Guia CC-2005. Os tópicos e pesos comparativos são apresentados na Tabela 4.14. A representação gráfica da tabela de pesos de relevância de tópicos em ES é exibida na Figura 4.6.

Tabela 4.14 Pesos Comparativos de Tópicos da Engenharia de Software (CC-2005)

Peso comparativo dos tópicos de Computação relacionados com a Engenharia de Software nos cinco tipos de programas de graduação										
Área de Conhecimento	**EC**		**CC**		**SI**		**TI**		**ES**	
	EC - min	**EC - Máx**	**CC - min**	**CC - Máx**	**SI - min**	**SI - Máx**	**TI - min**	**TI - Máx**	**ES - min**	**ES - Máx**
Modelagem e Análise de Software	2	4	2	3	3	3	1	3	4	5
Design de Software	2	4	3	5	1	3	1	2	5	5
Verificação e Validação de Software	1	3	1	2	1	2	1	2	4	5
Evolução do Software (Manutenção)	1	3	1	1	1	2	1	2	2	4
Processo de Software	1	1	1	2	1	2	1	1	2	5
Qualidade de Software	1	2	1	2	1	2	1	2	2	4

Fonte:(SHACKELFORD et al., 2005, p.24)

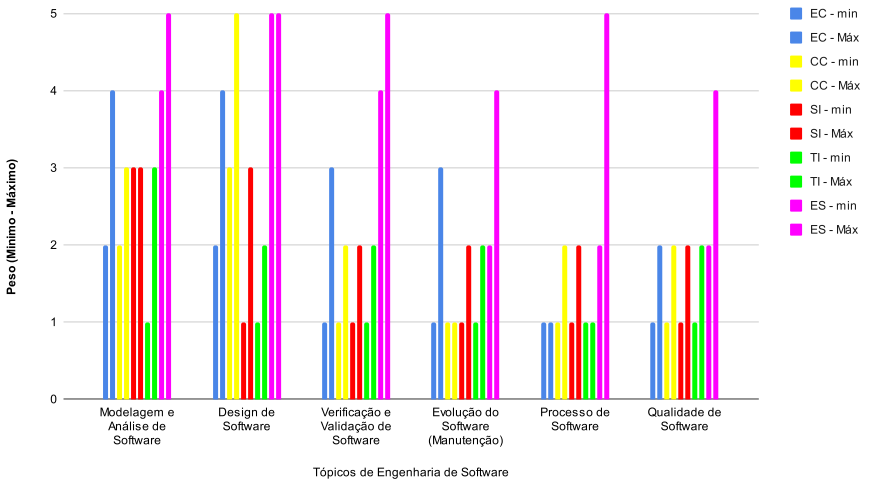

Figura 4.6 Pesos de Relevância dos Tópico de ES em cada Curso de Computação
Fonte:(SHACKELFORD et al., 2005, p.24)

Podemos perceber que todos os tópicos da área de ES tem peso mínimo acima de zero em ênfase a todos os cursos abordados. Isso mostra que todos os tópicos de conhecimentos em ES tendem a ser essenciais para a boa formação dos egressos destes cursos e para a modelagem curricular nos mesmos. Os tópicos selecionados foram analisados segundo o Glossário do CC-2005 e estão próximos aos tópicos elencados pelo SWEBOK (BOURQUE; FAIRLEY et al., 2014).

Percebendo a ênfase relevante que o CC-2005 instrui sobre a ES, partimos para a compreensão de tópicos transversais ou sociotécnicos. O Relatório mostra também uma visão tabular de tópicos que não estão relacionadas com a Computação; dentre eles, o tópico *Comunicação Interpessoal* é relevante para o desenvolvimento sociotécnico do estudante pois se alinha com objetivos profissionais aguardados pela indústria de software e do mundo do trabalho como um todo (CUKIERMAN; TEIXEIRA; PRIKLADNICKI, 2007; WANGENHEIM; SILVA, 2009). Sobre este tópico, o Glossário do documento versa:

> Uma área de estudo que ajuda os alunos de Computação a melhorar suas habilidades de comunicação oral e escrita para trabalho em equipe, apresentações, interação com clientes e outros informantes, documentação, atividades de vendas e marketing, etc. (SHACKELFORD et al., 2005, p.54)

Podemos perceber que o objetivo do conhecimento abordado no tópico é aprimorar habilidades de comunicação para diversos âmbitos profissionais de sua área de atuação. Um dos tópicos descritos no quadro tabular de conhecimentos de Computação chamada *Legal/Profissional/Ética/Sociedade* impressiona pela sua descrição no Glossário do Relatório:

> As áreas de prática e estudo dentro das disciplinas de Computação que ajudam os profissionais de Computação a tomar decisões eticamente informadas, que estão dentro dos limites dos sistemas legais relevantes, e códigos de conduta profissional (SHACKELFORD et al., 2005, p.54)(Tradução Nossa).

Quando observamos a descrição "áreas de prática e estudo dentro das disciplinas", percebemos clara indicação de interdisciplinaridade entre conhecimento técnico e sociotécnico. Isso é norteador por corroborar com a perspectiva de complementariedade do conhecimento técnico em Computação por meio de conhecimentos transversais ou sociotécnicos. A Tabela 4.15 apresenta os conteúdos sociotécnicos indicados com certa relevância para todos os cursos abordados pelo Currículo de Computação. A representação gráfica da tabela de pesos de relevância de tópicos sociotécnicos é exibida na Figura 4.7.

Tabela 4.15 Pesos comparativos de conteúdos sociotécnicos de relevância para os cursos superiores de Computação

Peso comparativo dos tópicos relacionados com o desenvolvimento sócio-técnico nos cinco tipos de programas de graduação										
Área de Conhecimento	**EC**		**CC**		**SI**		**TI**		**ES**	
	EC - min	**EC - Máx**	**CC - min**	**CC - Máx**	**SI - min**	**SI - Máx**	**TI - min**	**TI - Máx**	**ES - min**	**ES - Máx**
Legal/Profissional/ Ética/Sociedade	2	5	2	4	2	5	2	4	2	5
Comunicação interpessoal	3	4	1	4	3	5	3	4	3	4

Fonte:(SHACKELFORD et al., 2005, p.24)

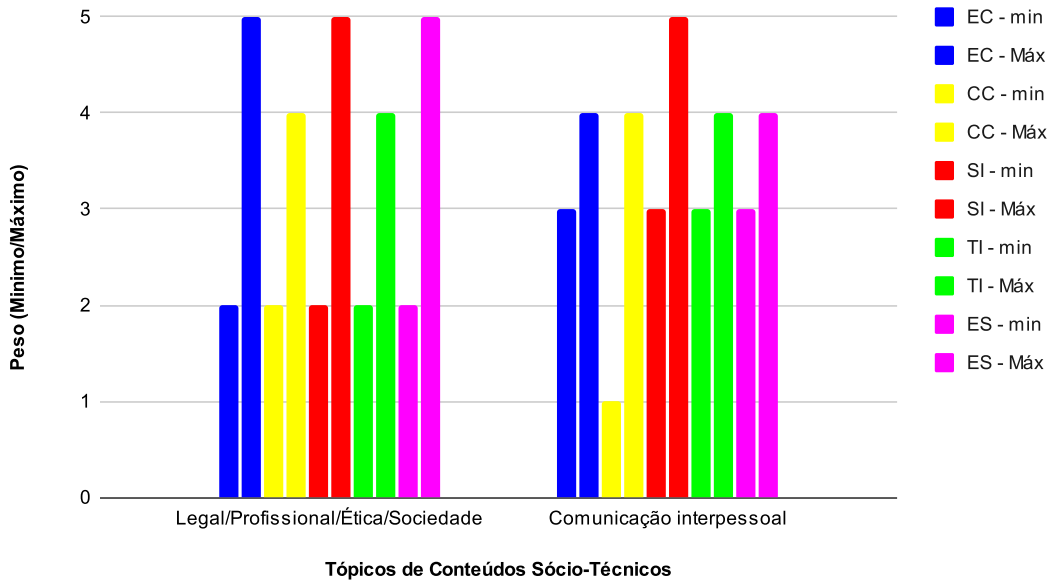

Figura 4.7 Pesos de Relevância de Tópicos Sociotécnicos em cada Curso de Computação
Fonte:(SHACKELFORD et al., 2005)

Podemos perceber que os tópicos de aspecto sociotécnicos referenciados aos cursos possuem ênfase importante. Todos os cursos têm peso mínimo acima de 2,0 (dois) para o tópico `Legal/Profissional/Ética/Sociedade`, indicando uma alta necessidade de aproximação de aspectos éticos, profissionais e sociais na formação técnica dos egressos. Todos os cursos indicaram pesos acima de zero no tópico `Comunicação Interpessoal`, o que indica a necessidade de formação que aproxime o egresso com a dinâmica de interação social diversa do mundo do trabalho.

O Relatório CC-2005 também instrui sobre *expectativa de carreira* para os egressos dos cursos de graduação abordados. O documento tabula requisitos de carreira em categorias de formação. Para cada requisito, é estabelecido um peso de 0(zero) a 5(cinco) de importância na formação dos egressos, sendo zero atribuído para nenhuma expectativa e cinco para a maior expectativa (ROCHA et al., 2005).

Vários requisitos diversos de carreira referentes a programação, desenvolvimento, gerenciamento e projeto de software tiveram expectativas de formação acima de zero. São alguns dos exemplos de requisitos de formação esperados aos egressos dos cursos que têm expectativa relevante: Programe em pequena escala, Programe em larga escala, Desenvolva novos sistemas de software, Crie interface de uso de software, e Implemente Software de recuperação de informações. Além de tabular expectativas de carreiras especificamente para cada um dos cursos abordados, o *Curricula Computing 2005* estabelece cerca de dez requisitos de carreira gerais, esperados para a formação em qualquer um dos cursos abordados.

Em resumo, os requisitos gerais versam sobre: Dominar as bases essenciais e fundamentais de sua disciplina da Computação; Uma base no que tange a habilidades de

programação de computadores; Compreender possibilidades e limitações de alcance de tecnologias de computador; Compreender o conceito de ciclo de vida e suas fases; Compreender o conceito essencial de processo; Estudar tópicos de Computação avançada que revele os limites de sua área de formação; Identificar e adquirir habilidades além do aspecto técnicos; Ser exposto a uma gama de situações que conectem habilidades aprendidas na academia com ocorrências do mundo real; Atenção a questões profissionais, legais e éticas; Integrar vários elementos da experiência de graduação.

Percebe-se uma convergência relevante entre os aspectos de formação determinados pelo *Curricula Computing 2005* com o que versam as Diretrizes Curriculares de formação superior em Computação no Brasil e também os Referenciais de Formação da SBC. Dentre os aspectos que chamam atenção, temos a relevância da área de ES e da formação sociotécnica como fator preponderante na expectativa de atuação dos egressos dos cursos e no planejamento pedagógico e curricular de acordo com questões próprias de cada um dos cursos. Assim, partimos para a análise da relevância da EES e de tópicos transversais ou sociotécnicos *em cada um dos guias mais atuais de cada curso* provenientes da Série *Curricula Computing 2005*.

4.2.1 Mapeamento do Currículo de Ciência da Computação 2013 da ACM/IEEE.

O Currículo de Computação 2013 é o principal Guia internacional para o planejamento curricular de cursos de graduação em Ciência da Computação. O documento conta com uma gama de capítulos cujo objetivo é instruir no desenvolvimento de cursos compatíveis com a identidade dá área, respeitando as perspectivas regionais e vocacionais específicas em cada contexto. O Currículos de Ciência da Computação da ACM/IEEE (CS-2013) tem como estrutura a descrição do processo de desenvolvimento, os princípios norteadores, as características dos graduados, o corpo de conhecimentos, instrução para o planejamento dos cursos e questões de desafios institucionais (SHACKELFORD et al., 2005).

Em seus princípios norteadores, o CS-2013 destaca a importância dos currículos *permitir flexibilidade aos alunos* para atuação de trabalho em diversas áreas. O Guia também versa sobre a importância de se fornecer orientação para o nível esperado de domínio dos tópicos pelos egressos, com *recomendações realistas que orientem e flexibilizem os projetos curriculares*. O CS-2013 indica que os currículos devem estimular a preparação dos graduados para serem bem-sucedidos mesmo em um campo com rápidas mudanças. O Guia conclui indicando a necessidade identificar conhecimentos e habilidades fundamentais que todo graduado em Ciência da Computação deve ter, fornecendo organização de tópicos de forma flexível. Quando partimos para a análise das competências indicadas pelo Guia, observamos o que o mesmo afirma:

> Os graduados em programas de Ciência da Computação devem ter competência fundamental nas áreas descritos pelo Corpo de Conhecimento (consulte o Capítulo 4), particularmente os tópicos centrais contidos lá. No entanto, também existem competências que os graduados dos programas de CS devem ter que são não listados explicitamente no Corpo de Conhecimento (DRAFT, 2013, p.23)(Tradução Nossa).

Por meio desta afirmação, podemos perceber a indicação do corpo de conhecimentos como a referência de conhecimentos esperados pelos egressos dos cursos de Ciência da Computação. Os conhecimentos diversos abordados pelo Guia são relacionados com as noções de temas comuns da Computação, *valorização da interação com teoria e prática*, visão sistêmica, habilidades para resolver problemas, *experiência com projetos*, noção de aprendizagem contínua, habilidades e responsabilidades profissionais, habilidades de comunicação e organização, e conhecimento do alcance da Computação e de domínios específicos diversos (DRAFT, 2013).

Conhecendo as características esperadas pelos egressos de graduação em Ciência da Computação, podemos identificar as competências e habilidades indicadas no corpo de conhecimento. Segundo (DRAFT, 2013, p.22), "Os graduados devem ter domínio da Ciência da Computação, conforme descrito pelo núcleo do Corpo de Conhecimento.". Sendo assim, é importante analisar a relevância do conhecimento em ES e de conhecimentos transversais segundo o que indica o corpo de conhecimento do Currículo de Ciência da Computação (2013).

Em seu capítulo quarto, o Guia descreve a estrutura lógica do corpo de conhecimento, e esta é vista como uma especificação do conteúdo a ser abordado. Algumas considerações do Guia indicam que os conteúdos não correspondem diretamente a cursos (componentes), mas que os mesmos podem ser indicados para a concepção e combinação dos componentes próximos a realidade de cada graduação.

O Guia define a organização e aspectos lógicos dos conteúdos, dividindo-os em *tópicos nucleares, ou núcleos*, e *tópicos eletivos*. Os conteúdos do núcleo são divididos em dois subtipos, *Núcleo-1* e *Núcleo-2*. Por questões de expansão e flexibilidade na área ao decorrer do tempo, o Guia optou por dividir o núcleo em dois, indicando que todos os tópicos contidos no Núcleo-1 devem ser cobertos nos currículos, e que todos ou quase todos os tópicos contidos no Núcleo-2 devem ser abordados. Os conteúdos eletivos são aqueles que não têm grau de importância para estar no Núcleo-1 ou Núcleo-2. Estes conteúdos são indicados para que o estudante possa aprofundar sua compreensão em várias áreas de acordo com aspectos específicos da graduação ou de um componente curricular. O relatório também estipula a quantidade de carga horária mínima como medida de abrangência em cada tópico de acordo com a sua classificação (SHACKELFORD et al., 2005).

Na sequência, partimos para a contextualização da análise dos tópicos referentes ao conhecimento em ES e conhecimentos sociotécnicos. O corpo de conhecimento apresenta os tópicos em um conjunto de Áreas de Conhecimento (AC) organizadas por temas atuais. Cada Área de Conhecimento organiza um conjunto de Unidades de Conhecimento (UC). São essas unidades de conhecimento que dão base aos tópicos organizados nos cursos/componentes curriculares. Assim, a etapa de exploração da análise de conteúdo do CS-2013 inclui a análise dos quadros de Áreas de Conhecimento determinados no Apêndice A do Guia, identificando Unidades de Conhecimento de ES e Unidades de Conhecimento de temas sociotécnicos que tenham carga horária atribuída como Núcleo-1 ou Núcleo-2, ou que possam ser incluídas como Unidade de Conhecimento eletivas.

A inclusão de uma unidade de conhecimento como tópico de ES ou como tópico sociotécnico será por meio da identificação de equivalência ou aproximação da terminologia dos conteúdos com as competências e expectativas apresentadas pelo Guia, e com refe-

rências externas, como por exemplo, o Corpo de Conhecimento SWEBOK. A Figura 4.8 mostra a organização da exploração do CS-2013.

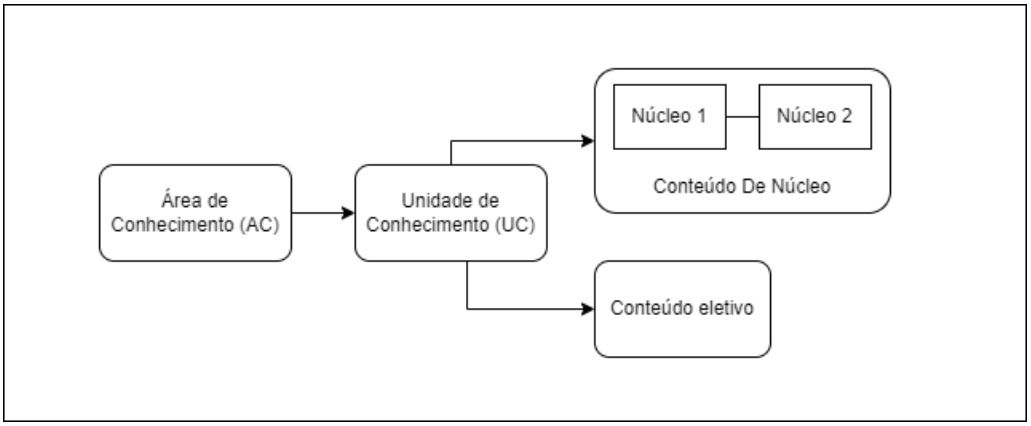

Figura 4.8 Metodologia de Exploração do Curriculo de Ciência da Computação 2013 (ACM)
Fonte:(DRAFT, 2013)

Por meio das análise de cada uma das Áreas de Conhecimento (AC), podemos identificar as Unidades de Conhecimento (UC). Cada Área de Conhecimento é identificada por um termo e uma sigla; por exemplo, a Área de Conhecimento em Algoritmos e Complexidade é identificada por AL. Cada Unidade de Conhecimento correspondente a Área de Conhecimento é descrita pela sigla da *AC/nome da (UC)*, por exemplo, a Unidade de Conhecimento `AL/Análise Básica`. A exploração dos quadros de conhecimento do CS-2013 é exibida na Tabela B.1 (Apêndice B).

Podemos observar que várias unidades de registros são definidas de acordo com a seleção das Unidades de Conhecimento em ES contidas, principalmente, na área de Conhecimento `SE.Engenharia de Software` e Unidades de Conhecimentos sociotécnicos contidas na Área de Conhecimento. A Tabela 4.16 mostra as Unidades de Registros encontradas.

Foram extraídos vinte e uma (21) Unidades de Registros, diretamente associadas a Unidades de Conhecimentos na área de ES e em áreas transversais. Todas as unidades de registros foram indicadas como conteúdos nucleares ou eletivos na ênfase de planejamento curricular para suprir as expectativas básicas de formação dos graduados em Ciência da Computação. *Isto converge para a compreensão do conhecimento sociotécnico e do conhecimento em ES como competências relevantes para a formação superior em Computação.*

Para continuar a investigação sobre a relevância das competências técnicas em ES e competências sociotécnicas na formação em Computação, apresentamos a Análise de Conteúdo do Currículo de Engenharia da Computação 2016 da ACM/IEEE.

Tabela 4.16 Lista de Unidades de Registros do CS-2013

Unidades de Registros extraídas do Currículo de Ciência da Computação 2013 da ACM			
1	Engenharia de Software Seguro	12	Contexto Social
2	Processo de Software	13	Ferramentas Analíticas em Ética
3	Gerenciamento de Projetos de Software	14	Ética Profissional
4	Ferramentas e Ambientes em Engenharia de Software	15	Propriedade Intelectual
5	Engenharia de Requisitos	16	Privacidade e Liberdades Civis
6	Desing de Software	16	Comunicação Profissional
7	Construção de Software	18	Sustentabilidade
8	Validação e Verificação de Software	19	História
9	Evolução de Software	20	Economias de Computação
10	Confiabilidade de Software	21	Políticas de Segurança, leis e Crimes Informáticos
11	Métodos Formais em Engenharia de Software		

Fonte: Elaboração Própria

4.2.2 Mapeamento do Currículo de Engenharia da Computação 2016-ACM/IEEE

O Currículo de Engenharia da Computação 2016 (ACM em conjunto com a IEEE) é mais um dos relatórios da Série *Curricula Computing 2005*. Este relatório apresenta todas as diretrizes curriculares para programas de graduação em Engenharia da Computação. A Engenharia da Computação é uma importante área de formação em Computação e, por isso, vamos abordar suas visões e concepções sobre o uso da Engenharia de Software como competência necessária para o desenvolvimento esperado dos egressos. Como em qualquer processo de formação em Computação, não podemos deixar de analisar a relevância do conhecimento sociotécnico segundo o relatório. Sobre a Engenharia de Computação como uma disciplina da Computação, o Currículos de Engenharia da Computação da ACM/IEEE (CE-2016) versa:

> A Engenharia da Computação é uma disciplina que incorpora a ciência e a tecnologia de projeto, construção, implementação e manutenção de componentes de software e hardware de sistemas de Computação modernos e equipamentos controlados por computador. (IMPAGLIAZZO et al., 2016, p.9)

Podemos mais uma vez observar a importância que o efetivo aprendizado em desenvolvimento e gerenciamento de projetos de software tem para a formação efetiva do graduado em Computação. Dessa forma, partimos para o teor do documento e sua estrutura.

O CE-2016 se estrutura por meio de uma descrição sobre a visão geral do relatório, os princípios subjacentes, a abordagem da Engenharia de Computação como disciplina (área) da Computação, o corpo de conhecimento, questões curriculares, a atuação profissional, e considerações sobre implementação dos currículos.

Sobre as características dos graduados, o Guia descreve as diferenças básicas das demais Engenharias para a Engenharia da Computação, descreve a importância do profissionalismo, da capacidade de projetar, de explorar a amplitude de conhecimento na área e a preparação para o exercício profissional.

Sobre a importância do profissionalismo, o relatório diz que "[...] os graduados devem entender as responsabilidades associadas à prática de Engenharia, incluindo o contexto profissional, social e ético em que realizam seu trabalho" (IMPAGLIAZZO et al., 2016, p.18). Podemos observar nessa afirmação a importância da formação sociotécnica associada com a formação prática na Engenharia da Computação. Já que a ES é uma área essencial para a formação em Computação, esse conceito converge com a noção da necessidade de uma formação em Engenharia de Software que assegure alinhamento com o aprendizado técnico e sociotécnico.

Para abordar especificamente a relevância da ES e dos temas sociotécnicos no planejamento curricular de cursos de Engenharia da Computação, partimos para o mapeamento destes no CE-2016 por meio da Análise de Conteúdo de seu corpo de conhecimento. Segundo (IMPAGLIAZZO et al., 2016, p.9), "A base para este relatório é um corpo de conhecimento fundamental a partir do qual uma instituição pode desenvolver ou modificar um currículo para atender às suas necessidades". A metodologia de desenvolvimento do corpo de conhecimento é próxima da usada no Currículo de Ciência da Computação 2013 e contem amplas Áreas de Conhecimento (AC) aplicáveis a todos os programas de Engenharia da Computação.

Cada área de conhecimento conta com Unidades de Conhecimento (UC) que por sua vez definem os resultados de aprendizagem esperados. Unidades de conhecimento são classificadas como "Essenciais" (devem constar em todos os currículos implementados) ou "Complementares", ambos com indicação de carga horária básica para o planejamento curricular. Quando uma unidade de conhecimento tiver carga horária associada, o componente é essencial, quando não apresenta a carga horária mínima significa que a Unidade é complementar.

Dessa forma, o método do mapeamento será próximo ao definido no CS-2013, onde cada área de conhecimento será analisada e cada unidade de conhecimento pertencente a tópicos de Engenharia de Software ou tópicos de desenvolvimento sociotécnico serão extraídos de acordo com sua classificação em Unidade de Conhecimento essencial ou complementar. A metodologia do mapeamento do CE-2016 é apresentada na Figura 4.9.

Analogamente à análise de Áreas de Conhecimentos (AC) do Guia CS-2013, podemos identificar as Unidades de Conhecimento (UC). Cada Área de Conhecimento é identificada por um termo e uma sigla. A Unidade de Conhecimento (UC) correspondente a cada Área de Conhecimento (AC) é descrita pela sigla da *AC/nome da (UC)*.

O critério para seleção de uma unidade de conhecimento como pertencente a área de ES ou pertencente a área sociotécnica, é a proximidade terminológica ou temática com tópicos abordados pelo SWEBOK, *Curricula Computing 2005* e descrições da aprendi-

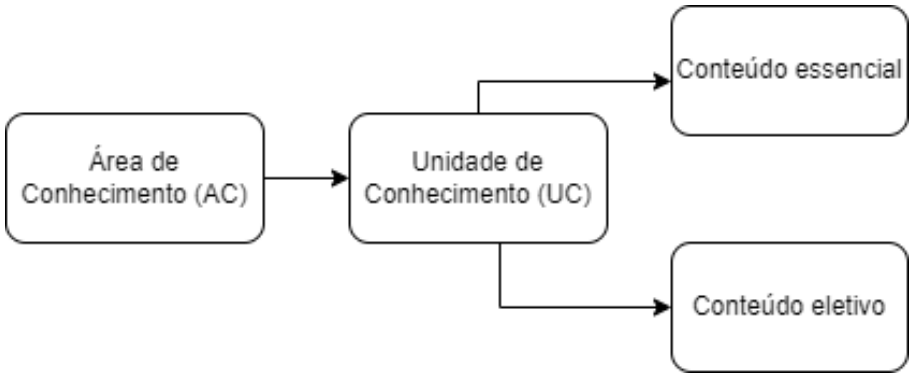

Figura 4.9 Metodologia de Exploração do Currículo de Engenharia da Computação 2016 (ACM)

Fonte:(IMPAGLIAZZO et al., 2016)

zagem pertencentes a cada Unidade de Conhecimento do próprio corpo de conhecimento. As Unidades de Contexto e de Registros são apresentadas na Tabela B.2 (Apêndice B).

Na exploração dos corpos do Guia relevantes para nosso objetivo, conseguimos extrair *dezessete (17) unidades de registros*, sendo onze referentes a Unidades de Conhecimento na área de preparação para a prática profissional, e seis unidades referentes a Unidades de Conhecimento relacionadas áreas de ES. A Tabela 4.17 mostra as Unidades de Registros encontradas.

Todas as unidades de registros definidas foram indicadas como conteúdos principais ou complementares para o planejamento curricular básico de formação dos graduados em ES. Assim como no CS-2013, observamos que o conhecimento em ES e o conhecimento sociotécnico colocados como referência para as necessidades de aprendizagem para a graduação.

Continuaremos a investigação sobre a relevância das competências técnicas em ES e competências Sociotécnicas na formação em Computação na Análise de Conteúdo do Currículo de Sistemas de Informação 2010 da ACM.

4.2.3 Mapeamento do Currículo de Sistemas de Informação 2010 da ACM.

A ACM e a Associação de Sistemas de Informação apresentam o Currículo de Sistemas de Informação de 2010. Este Currículo é o guia mais recente da ACM para composição de Curriculos de Graduação em Sistemas de Informação proveniente do *Curricula Computing 2005*. Assim como os demais guias apresentados até aqui, o SI-2010 é estruturado em princípios norteadores, expectativas dos Graduados, a visão sumária da área de SI, relação entre os componentes ou tópicos de conhecimento abordados entre outros elementos úteis.

Sobre os princípios orientadores, o SI-2010 trata de aspectos próximos ao que versam o CS-2013 e CE-2016. O Relatório orienta o modelo apresentado como um consenso para a comunidade de Sistemas de Informação, também versa da necessidade de projetos curriculares que ajudem os programas de SI a produzir graduados competentes e confiantes. O guia também fala sobre flexibilidade e domínios curriculares (TOPI et al., 2010).

Tabela 4.17 Lista de Unidades de Registros do CE-2016

Unidades de Registros extraídas do Currículo de Engenharia da Computação 2016 da ACM			
1	Histórico e Visão Geral para prática Profissional	10	Questões de Negócios e Gerenciamento
2	Ferramentas, padrões e/ou restrições de Engenharia relevantes em Prática Profissiona	11	Escolhas na Prática profissional
3	Estratégia de Comunicação Eficazes	12	Processos de Software
4	Abordagens de equipe Interdisciplinar	13	Análise e elicitação de requisitos.
5	Enquadramentos Filosóficos e Questões Culturais	14	Testes Integração e validação de sistemas de software
6	Soluções de Engenharia e efeitos sociais	15	Histórico e Visão Geral do Design de Software
7	Responsabilidades profissionais e Éticas	16	Ferramentas, padrões e/ou restrições de Engenharia de software relevantes
8	Questões Contemporâneas	17	Teste e qualidade de software

Fonte: Elaboração Própria

Um dos princípios importantes a serem abordados é que "O modelo curricular tem metas de carreira que exigem conteúdos básicos e eletivos" (TOPI et al., 2010, p.6). Esse princípio indica uma classificação de competências e conhecimentos próximas ao que já foi abordado nos currículos de Ciência da Computação e Engenharia da Computação.

Sobre os princípios formadores na Graduação em Sistemas de Informação, o SI-2010 versa sobre a necessidade do profissional atuar em vários domínios, ter habilidades de pensamento analítico e crítico, solucionar problemas, ter visão da interação da Computação com pessoas, ambientes e dados, ter ética e boa comunicação e conduta profissional, implementando soluções de tecnologia da informação e dominando os princípios de sistemas de informação (TOPI et al., 2010). Podemos perceber a relevância da aprendizagem técnica e sociotécnica também na formação do Graduado em Sistemas de Informação.

Para analisarmos a importância dessas duas áreas no SI-2010 precisamos conhecer a relação dos conteúdos ou conhecimentos na estrutura de formação segundo o currículo. O Currículo de Sistemas de Informação 2010 da ACM/AIS adota uma estrutura conceitual com três elementos: Curso, Objetivo de Aprendizagem e Hierarquia de Níveis de Área de Conhecimento. A hierarquia de Conhecimento é estruturada em Áreas de Conhecimento que organizam Unidades de Conhecimento que, por sua vez, organizam tópicos. Tópicos também podem ser compostos por conjuntos de tópicos. A estrutura conceitual do SI-

2010 é mostrada na Figura 4.10.

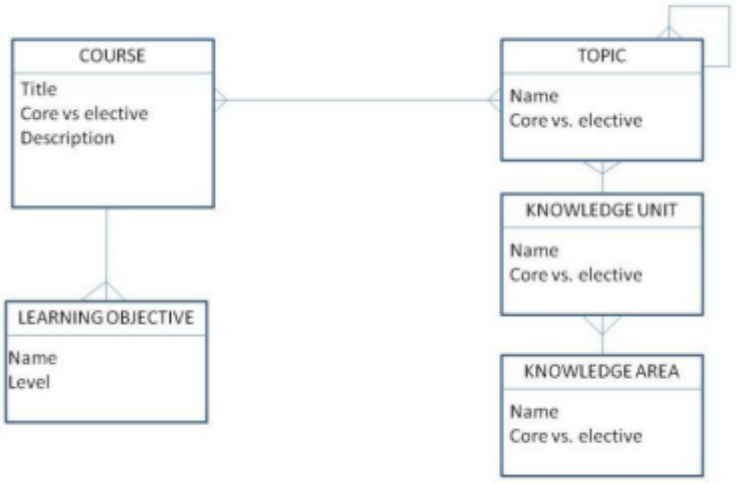

Figura 4.10 Estrutura Conceitual do SI-2010
Fonte:(TOPI et al., 2010, p.24)

Seguindo a lógica da estrutura apresentada, partimos para o mapeamento da relevância das competências de ES do SI-2010, competências essas determinadas por tópicos ou Unidades de Conhecimento definidos como Nucleares ou Eletivos. O Currículo de Sistemas de Informação separa os Conhecimentos e Competências necessários ao desenvolvimento dos graduados em três categorias: Conhecimentos e habilidades específicos de Sistemas de Informação; Conhecimentos e Habilidades Fundamentais; e Conhecimentos e habilidades relacionadas aos fundamentos do domínio.

A Figura 4.11 mostra como será o mapeamento dos conhecimentos ou tópicos de ES exclusivamente na categoria de Conhecimentos e Habilidades específicas de SI. Os tópicos serão selecionados conforme a proximidade temática ou textual com os conhecimentos de ES segundo o Guide to the Software Engineering Body of Knowledge (SWEBOK) e as unidades de conhecimento do Currículo de Engenharia de Software 2014 (BOURQUE et al., 1999; ARDIS et al., 2014b).

No mapeamento da ES, percorremos cada conjunto de conhecimentos e competências específicas da área de Sistemas de Informação, identificando cada tópico e subtópico equivalente relacionado com a área de ES. Cada curso é identificado pelo título, classificação do curso (Nuclear, Eletivo) e o tópico (Nome). A análise de conteúdo dos tópicos técnicos de Engenharia de Software é listada na tabela B.3 (Apêndice B).

Podemos perceber que *seis unidades de Registros* foram extraídas das unidades de contexto. Mesmo adotando um referencial curricular que separa tópicos gerais da Computação dos tópicos específicos da área de Sistemas de Informação, podemos perceber uma intimidade de conteúdos selecionadas com atividades de ES. Para compreender a importância do conhecimento sociotécnico no Guia SI-2010, é importante observar como o mesmo destaca tais conhecimentos na estrutura do documento. Sobre a importância do conhecimento transversal para o currículo SI-2010, vamos observar o que o mesmo versa.

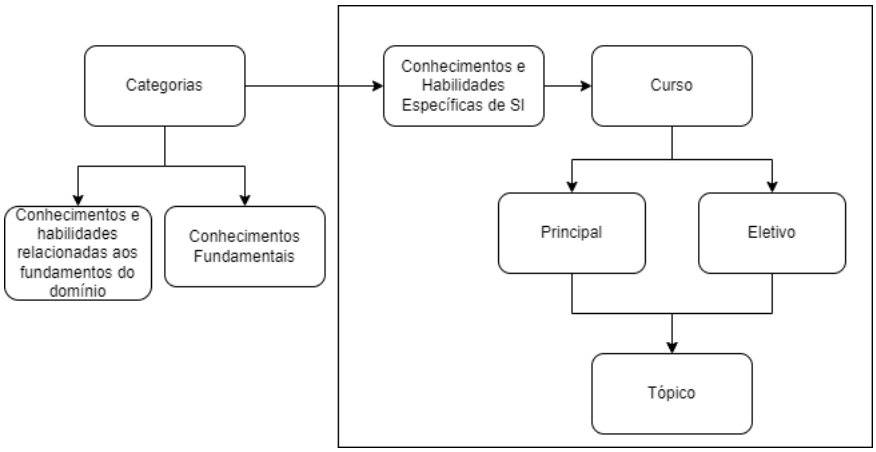

Figura 4.11 Método do Mapeamento do Currículo de SI-2010
Fonte: (TOPI et al., 2010)

> O conhecimento e as habilidades fundamentais não são exclusivos de Sistemas de Informação como disciplina. Em vez disso, a maioria dos programas que educam profissionais pretendem desenvolver algumas ou todas essas habilidades e capacidades (TOPI et al., 2010, p.21).

Dessa forma, precisamos mapear os conhecimentos ou conteúdos transversais/sociotécnicos indicados pelo Guia nessa categoria de conhecimentos. O Guia versa sobre os conhecimentos e habilidades fundamentais, indicando a habilidade, descrevendo sobre a habilidade e estruturando capacidades que devem ser estimuladas formar a habilidade. Dessa forma, a análise do conteúdo para o mapeamento sociotécnico observa:

> **Conhecimentos e habilidades dos graduados em SI > Conhecimentos e Habilidades Fundamentais > Rótulo do Conhecimento**.

A análise de conteúdo dos tópicos sociotécnicos é listada na tabela B.4 (Apêndice B). Extraímos *quatro unidades de registros* referentes a conhecimentos ou tópicos sociotécnicos. Estes tópicos são próximos a aspectos sociotécnicos citados por trabalhos de referência (CUKIERMAN; TEIXEIRA; PRIKLADNICKI, 2007; WANGENHEIM; SILVA, 2009). Ou seja, o conhecimento sociotécnico é imprescindível para a formação efetiva em qualquer área de formação em Computação. O próprio Currículo de Sistemas de Informação converge com esse raciocínio ao afirmar que "[...] é impossível para graduados em SI exibir as capacidades de SI de alto nível necessárias sem esses conhecimentos e habilidades fundamentais" (TOPI et al., 2010, p.21).

As unidades de registros extraídas do SI-2010 foram agrupadas e são exibidas na Tabela 4.18. Assim temos as unidades de registros referentes aos conhecimentos em ES e conhecimentos Sociotécnicos segundo o referencial da ACM para os cursos de graduação em Sistemas de Informação (2010). Mesmo separando as áreas de conhecimento em categorias de formação, o SI-2010 mostra como os tópicos em Engenharia de Software e como a formação sociotécnica são importantes não só para a Computação como um todo, mas também para a formação específica em Sistemas de Informação.

Tabela 4.18 Lista de Unidades de Registros do SI-2010

Unidades de Registros extraídas do Currículo de Sistemas de Informação 2010 da ACM			
1	Design de Software	6	Teste de Software
2	Ciclo de Vida de Desenvolvimento de software	7	Capacidade de Liderança e Colaboração
3	Análise de Requisitos	8	Comunicação
4	Técnicas para Modelar Estruturas do Software	9	Negociação
5	Desing de Software Centrado no usuário	10	Criatividade e Capacidade de Análise crítica e ética

Fonte: Elaboração Própria

4.2.4 Mapeamento do Currículo de Tecnologia da Informação 2017 da ACM.

O Currículo de Tecnologia da Informação 2017 da ACM/IEEE é o guia para curricular para os programas de Bacharelado em Tecnologia da Informação a ser analisado. Sua estrutura tem vários aspectos semelhantes aos currículos já abordados até aqui. O documento descreve a visão geral do Guia, descreve a área de Tecnologia da Informação(TI) como área ou disciplina da Computação, as noções de competência, as perspectivas da indústria, questões de implementação do currículo e corpo do conhecimento.

Em termos gerais, o Guia destaca que o relatório deve refletir necessidades acadêmicas e da indústria, e competências de Tecnologia da Informação que formam a estrutura curricular de TI. Diretrizes devem refletir aspectos que diferenciem a TI das demais áreas da Computação, mas que também mostrem as relações entre elas (SABIN et al., 2017).

Sobre o perfil do graduado em Tecnologia da Informação, o relatório versa sobre a capacidade de analisar problemas do mundo real e aplicar requisitos computacionais para solucioná-los, projetar, implementar e avaliar soluções, se comunicar efetivamente com diversos públicos, fazer julgamentos sobre perspectivas da Computação com base nas leis e na ética e atuar em equipe (SABIN et al., 2017).

Sobre a visão de competências, o IT-2017 estrutura um *modelo embasado em conhecimentos, habilidades e disposições*. *Conhecimento* aborda proficiência em conteúdos de TI e sua aplicação. *Habilidades* referem-se a capacidades e estratégias desenvolvidas com práticas (pensamento de ordem superior) e interações com outros e com o mundo. *Disposições* abrangem questões socio-emocionais, comportamentos e atitudes que indicam a sensibilidade e inclinação para saber quando e como executar as tarefas (SABIN et al., 2017). Podemos perceber um modelo inter-relacionado onde competências técnicas e sociotécnicas são estruturadas para serem estimuladas em determinados contextos de aprendizagem e aplicações na realidade como mostra a Figura 4.12.

Conhecendo as relações de competências entre os conhecimentos técnicos e sociotécnicos, partimos para a compreensão destas na estrutura curricular do Guia. O Guia estabelece o corpo de conhecimentos em TI como um conjunto de domínios que, por sua vez, surgem das competências. O Guia versa que um domínio não é um curso mas sim

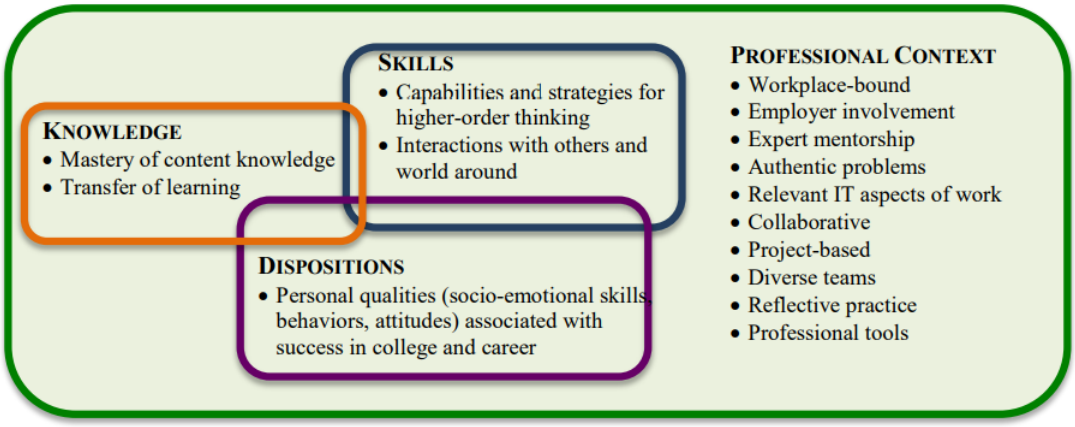

Figura 4.12 Método de mapeamento do IT2017
Fonte: (SABIN et al., 2017, p.30)

requisitos dos cursos a serem implementados conforme a estrutura do mesmo. Assim, os domínios são os conhecimentos, habilidades e disposições de áreas que compõem o escopo da TI. Esses domínios são classificados em *essenciais e suplementares*. Os domínios essenciais se referem a todas as competências devem ser alcançadas por todos os alunos de Graduação em TI. Os domínios suplementares são competências que refletem metas específicas de programas individuais de TI de acordo com suas missões e vocações (SABIN et al., 2017). Os domínios de TI são compostos por subdomínios. Numa estrutura curricular, um curso pode ter vários subdomínios de um mesmo domínio ou pode ter vários subdomínios de domínios diferentes.

Dessa forma, partimos para o mapeamento dos subdomínios que se relacionam a área de Engenharia de software ou a áreas de conhecimento sociotécnico. Cada domínio é identificado por um código de Prefixo Inicial ITE para domínios essenciais e ITS para domínios suplementares. Cada subdomínio tem o prefixo do domínio no qual faz parte e um número no final, e será selecionado de acordo com a proximidade temática ou descritiva das áreas de conhecimento em ES, conforme o SWEBOK e o Guia da ACM para graduação em ES. Os subdomínios de abordagem sociotécnica serão selecionados de acordo com a proximidade temática e descritiva do próprio TI-2017 e dos demais guias Curriculares ACM e Referenciais da SBC. O mapeamento das competências da TI no TI-2017 para exploração via Análise de Conteúdo é descrita na Figura 4.13.

Com o método de exploração explicitado, partimos para as unidades de contexto e de registros extraídas do mapeamento. A Tabela B.5 (Apêndice B) mostra o resultado da exploração do IT-2017. Foram *dezenove (19) unidades de registros* que representam subdomínios essenciais e suplementares para a formação específica em Tecnologia da Informação. Os subdomínios estão classificados em L1, L2, L3, que representam níveis de engajamento para a aprendizagem e que podem determinar a ordem que os subdomínios sejam abordados no currículo. Essa classificação foi listada nas unidades de contexto mas não entrará na nossa análise. Das dezenove unidades de registros, oito delas estão relacionadas a aspectos sociotécnicos e onze unidades são relacionadas às competências

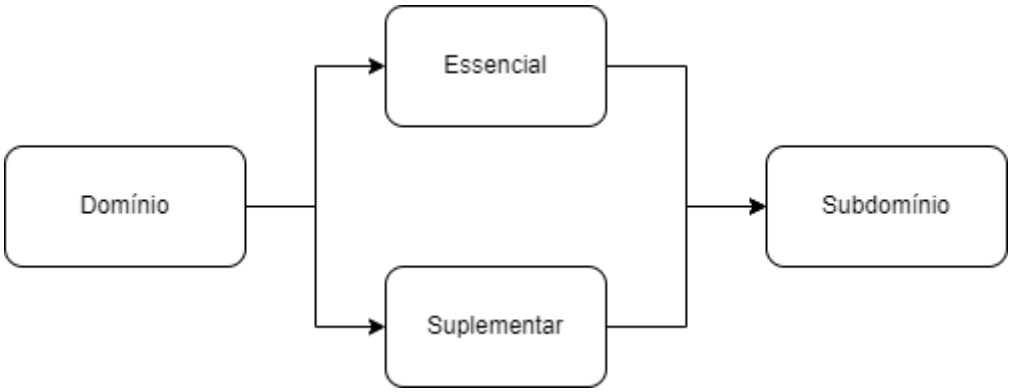

Figura 4.13 Modelo de competências do IT2017
Fonte:(SABIN et al., 2017)

em ES. A lista de unidades de registros é apresentada na Tabela 4.19.

Com base nas competências extraídas, nota-se que o Guia de Tecnologia da Informação da ACM também observa a relevância dos aspectos sociotécnicos para a aprendizagem em Computação e especificamente em TI. O Guia também indica aspectos da ES como norteadores para o desenvolvimento direcionado e específico das práticas de TI.

4.2.5 Mapeamento do Currículo de Engenharia de Software 2014 da ACM

O Currículos de Engenharia de Software da ACM/IEEE (SE-2014) é o principal Guia Curricular internacional para a graduação em Engenharia de Software. O SE-2014 versa sobre a área de Engenharia de Software como disciplina da Computação, fala sobre os princípios orientadores, mostra o corpo de conhecimento e a visão geral deste com competências a serem desenvolvidas, além de indicar diretrizes para o projeto curricular e abordar ambientes e estratégias de implementação curricular (ARDIS et al., 2014b).

O Guia observa as visões da ES como área, tanto da Computação como da Engenharia. Fundamentada na Computação, a Engenharia de Software "[...] busca desenvolver e usar modelos sistemáticos e técnicas confiáveis para produzir software de alta qualidade." (ARDIS et al., 2014b, p.14). Essa concepção evidencia a proximidade da ES com a Engenharia tradicional, considerando a Ciência, o uso da Matemática e as entidades lógicas ao invés de artefatos físicos. O documento também destaca a importância da Matemática e Estatística, da Psicologia e Ciências Sociais e da Ciência da Administração como áreas fundamentais para o bom desenvolvimento da formação do Graduado em Engenharia de Software.

O Guia ES aborda a importância de aspectos da prática profissional na formação dos egressos. Segundo (ARDIS et al., 2014b, p.17), "um dos principais objetivos de qualquer programa de Engenharia é fornecer aos graduados as ferramentas necessárias para iniciar a prática profissional de Engenharia." Isso mostra que um curso de graduação deve fornecer aos estudantes as condições para que tal objetivo seja alcançado. Neste contexto, o Guia aconselha princípios orientadores para a formação efetiva do Graduado em ES.

Tabela 4.19 Lista de Unidades de Registros do TI-2017

Unidades de Registros extraídas do Currículo de Tecnologia da Informação 2017 da ACM			
1	Perspectivas e Impactos da prática profissional	14	Cadeia de Suprimentos e Garantia de Software
2	Questões e responsabilidades profissionais	15	Modelo de Processos e Atividades de desenvolvimento e Gerenciamento de Software
3	Questões Ambientais	16	Desenvolvimento de Software Baseado em plataforma
4	Questões éticas, legais e de privacidade	17	Ferramentas e Serviços
5	Propriedade Intelectual	18	Gerenciamento de software e de projetos de software
6	Comunicações	19	Implantação, operações Manutenção de Software
7	Trabalho em equipe e gestão de conflitos	20	Contexto Social da Computação
8	Habilidades de empregabilidade e carreiras em TI	21	Objetivos, planos, tarefaz, prazos e riscos
9	Arquitetura de Sistemas	22	Papel e regulamentos do Governos
10	Teste e garantia de qualidade	23	Desafios e Abordagens Globais
11	Fatores Humanos no Desing	24	Gerenciamento de Riscos
12	Conceitos de Aplicativos	25	Computação Sustentável
13	Frameworks de Desenvolvimento		

Fonte: Elaboração Própria

Um dos princípios aborda a capacidade do estudante desenvolver qualidades como o conhecimento profissional, o conhecimento técnico, trabalho em equipe, negociação e liderança, boa comunicação e resolução de problemas, atentando-se a questões éticas, legais e econômicas (ARDIS et al., 2014b). Sobre indicação das competências, o Guia ES-2014 versa:

> Para que os engenheiros de software tenham competência técnica para desenvolver software de alta qualidade, eles devem ter uma formação sólida e profunda nos fundamentos da Ciência da Computação, conforme descrito no Capítulo 4. (ARDIS et al., 2014b, p.42) (Tradução nossa)

O capítulo 4 do Guia SE-2014 versa sobre os conhecimentos em ES indicados como competências a serem inseridas no currículo visando o desenvolvimento do egresso. O

Guia também organiza o corpo de conhecimento em Áreas e Unidades de Conhecimento. Cada Área, representada por uma abreviatura, é dividida em Unidades que, por sua vez, representam *Módulos Temáticos*. Cada Unidade organiza um conjunto de tópicos e tem um sufixo associado. O Guia afirma que as áreas de conhecimento abordadas não representam todo o universo de conhecimento da área de ES, mas buscam concentrar o conjunto mínimo de conhecimentos necessários de acordo com o consenso de educadores na área (ARDIS et al., 2014b). O ES-2014 também instrui sobre *unidade de tempo* como referência de organização do tempo de instrução. Também são abordados *níveis cognitivos de aprendizagem* como referência de objetivo educacional do tópico.

Os tópicos abordados são definidos como *Essenciais* ou *Desejáveis*. Tópicos essenciais devem estar incluídos no núcleo do currículo. Tópicos desejáveis podem não fazer parte do núcleo curricular, mas podem ser incluídos no núcleo de determinados programas. A Figura 4.14 apresenta o método de mapeamento dos conhecimentos no SE-2014.

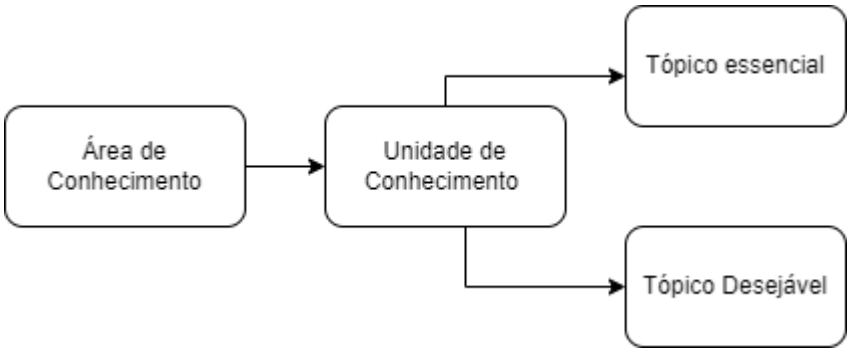

Figura 4.14 Método de mapeamento do SE-2014
Fonte:(ARDIS et al., 2014b)

Assim, a exploração do material no Guia ES-2014 se dará no mapeamento das unidades de conhecimento de Engenharia de Software e de aspectos sociotécnicos que tenham tópicos essenciais associados. Os critérios de mapeamento se aplicam na busca temática ou descritiva segundo o próprio ES-2014, o Currículo de Engenharia de Software de 2004, o Corpo de Conhecimentos em Engenharia de Software (SWEBOK), trabalhos e referencias em abordagem sociotécnica, e o Curricula Computing 2005 (LEBLANC et al., 2006; BOURQUE; FAIRLEY et al., 2014; CUKIERMAN; TEIXEIRA; PRIKLADNICKI, 2007; WANGENHEIM; SILVA, 2009). A exploração das unidades de contextos e unidades de registros provenientes do mapeamento é exibida na Tabela B.6 (Apêndice B).

Foram extraídos *vinte e seis (26) unidades de registros* referentes a cada Unidade de Conhecimento encontrada nas unidades de contexto. Das vinte e seis unidades, *três se referem a conhecimentos sociotécnicos*. Podemos perceber, ao analisar as unidades de contexto na exploração, uma grande quantidade de tópicos essenciais e desejáveis associados a conhecimentos em ES e em aspectos sociotécnicos. Isso inclina o raciocínio à percepção que os currículos abordados respeitam a relevância da ES e do conhecimento sociotécnico para o desenvolvimento profissional em ES, independente do curso.

As unidades de registros agrupadas são exibidas na Tabela 4.20. Todas as unidades de registros extraídas foram indicadas como Unidades de Conhecimento com carga horária

mínima a ser considerada na formação dos currículos, o que evidencia a sua relevância na formação profissional da prática de ES. Na sequência, partimos para análise das unidades de registros para que assim possamos extrair os eixos temáticos de acordo com o contexto da pesquisa.

Tabela 4.20 Lista de Unidades de Registros do SE-2014

Unidades de Registros extraídas do Currículo de Engenharia de Software 2014 da ACM			
1	Dinâmica de Grupo e Psicologia	14	Design Detalhado
2	Habilidades de Comunicação	15	- V&V terminologia e fundamentos
3	Profissionalismo	16	Comentários e análise estática teste
4	Fundamentos de Modelagem de Software	17	Análise de problemas e relatórios
5	Tipos de Modelos	18	Conceitos e cultura de qualidade de software
6	Fundamentos da Análise	19	Garantia de processo
7	Fundamentos de Requisitos	20	Garantia do produto
8	Elicitação de Requisitos	21	Conceitos de processo
9	Validação de Requisitos	22	Implementação do processo
10	Conceitos de Design de Software	23	- Planejamento e acompanhamento de projetos
11	Estratégias de Design de Software	24	Gestão de configuração de software
12	Design Arquitetural	25	- Evolution processos e atividade
13	Design de Interação Humano-Computador	26	Desenvolvimento de Software Seguro

Fonte: Elaboração Própria

4.2.6 Agrupamentos e Articulação das Unidades de Registros e Categorização nos Currículos da ACM

Passamos por todo o processo de exploração dos principais guias curriculares da ACM. Podemos perceber uma coerência no teor dos documentos no que tange sua finalidade, sua metodologia de desenvolvimento e sua estrutura de acordo com algumas especificidades para cada área abordada. Dessa forma, foi possível explorar os documentos para investigar a relações de conhecimentos em Engenharia de Software e conhecimentos Sociotécnicos com a relevância curricular e necessidade de formação dos egressos.

Em cada um dos currículos abordados, percebe-se na metodologia a indicação de conhecimentos como competências necessárias a serem estimuladas aos egressos. Estes conhecimentos foram abordados em grau de carga horária, de profundidade e de relevância. Sendo assim o foco da exploração dos documentos foi identificar conhecimentos em ES e conhecimentos sociotécnicos indicados como conteúdo relevante. A relevância de um conteúdo ou conhecimento foi comumente dividida em níveis por cada currículo. A nomenclatura dos níveis de relevância muda um pouco para cada Currículo, mas a lógica que os permeia é convergente. O CS-2013 classifica os níveis entre conteúdos de Núcleo e

Eletivo, o CE-2016 classifica entre Essencial e Eletivo, o SI-2010 classifica níveis em Principal e Eletivo, o TI-2017 classifica em Essencial e Suplementar, e o SE-2014 classifica os níveis em Essencial e Desejável. Assim, extraímos as unidades de registros de todos os conhecimentos pertencentes a qualquer destes níveis de classificação segundo o currículo. Após a extração, agrupamos todos os conhecimentos com proximidade temática a conhecimentos de Engenharia de Software e conhecimentos Sociotécnicos de cada curso. A Tabela 4.21 mostra o agrupamento das unidades de registros extraídas dos Currículos da ACM; também mostra a concentração das unidades de registros em todas as unidades de acordo com sua característica temática. Todos os Guias abordaram conhecimentos específicos de ES em Unidades de Conhecimento ou tópicos, como já discutido. Assim, vamos concentrar todos os tópicos de ES em um único eixo temático: *Conhecimento em Engenharia de Software*. Da mesma forma, todas as unidades de conhecimento ou tópicos de conhecimento transversal e sociotécnico foram concentrados de acordo com a temática que os caracteriza. Assim, concentraremos estas no mesmo eixo temático *Conhecimento Sociotécnico*.

Após a definição da relação temática das competências investigadas no contexto dos currículos da Série Curricula, partimos para a compreensão da relevância destes para o desenvolvimento profissional na área da Computação. Para isso, as unidades de registros foram reagrupadas de acordo com sua posição de relevância nos currículos. As *posições de relevância* foram definidas de acordo com as classificações dos níveis dos conteúdos ou tópicos abordados no currículo. Os níveis foram classificados em dois tipos: *nuclear/essencial/principal*, e *eletivo/desejável*. Dessa forma, as unidades de registros pertencentes ao eixo temático *Conhecimento em Engenharia de Software* foram reagrupadas em níveis de relevância – como mostra a Tabela 4.22.

Após extraídos e identificados os níveis de relevância dos conhecimentos em ES, seguimos para a análise das unidades de registros do eixo temático *Conhecimento Sociotécnico*. O agrupamento destes nos respectivos níveis é mostrado na Tabela 4.23.

Podemos perceber uma variedade de unidades de conhecimento ou tópicos distribuídos nos Guias curriculares, classificados como essenciais ou eletivos[1]. Foram mapeados *quarenta e um (41) tópicos específicos da Engenharia de Software nos Guias explorados*. Os conteúdos eletivos ou desejáveis também foram referenciados como aqueles que não são obrigatórios no currículo mas que são importantes para a formações específicas dentro da própria formação geral de acordo com vocações institucionais ou regionais. Foram mapeados *quatorze tópicos (14) específicos de ES nos guias*.

[1]Lembrando que os guias instruem que conteúdos essenciais ou nucleares devem ser incluídos nos currículos dos cursos.

Tabela 4.21 Tabela de Agrupamento de Unidades de Registros da Série Curricula em Eixos Temáticos

Agrupamento das Unidades de Registros de Todos os Guias da Série Currícula					
Unidades de Registro					Eixos Temáticos
CS-2013	**CE-2016**	**SI-2010**	**TI-2017**	**SE-2014**	
Engenharia de Software Seguro	Processos de Software	Design de Software	Arquitetura de Sistemas	Fundamentos de Modelagem de Software	Conhecimento em Engenharia de Software
Processo de Software	Análise e elicitação de requisitos.	Ciclo de Vida de Desenvolvimento de software	Teste e garantia de qualidade	Tipos de Modelos	
Gerenciamento de Projetos de Software	Testes Integração e validação de Sistemas de Software	Análise de Requisitos	Fatores Humanos no Desing	Fundamentos da Análise	
Ferramentas e Ambientes em Engenharia de Software	Histórico e Visão Geral do Desig de Software	Técnicas para Modelar Estruturas do Software	Conceitos de Aplicativos	Fundamentos de Requisitos	
Engenharia de Requisitos	Ferramentas, padrões e/ou restrições de Engenharia de software relevantes	Desing de Software Centrado no usuário	Frameworks de Desenvolvimento	Elicitação de Requisitos	
Desing de Software	Teste e qualidade de software	Teste de Software	Cadeia de Suprimentos e Garantia de Software	Validação de Requisitos	
Construção de Software			Modelo de Processos e Atividades de desenvolvimento e Gerenciamento de Software	Conceitos de Design de Software	
Validação e Verificação de Software			Desenvolvimento de Software Baseado em plataforma	Estratégias de Design de Software	
Evolução de Software Confiabilidade de Software			Ferramentas e Serviços Gerenciamento de software e de projetos de software	Design Arquitetural Design de Interação Humano-Computador	
Métodos Formais em Engenharia de Software			Implantação, operações Manutenção de Software	Design Detalhado	
				V&V terminologia e fundamentos Comentários e análise estática teste Análise de problemas e relatórios	
				Conceitos e cultura de qualidade de software	
				Garantia de processo	
				Garantia do produto	
				Conceitos de processo	
				Implementação do processo - Planejamento e acompanhamento de projetos Gestão de configuração de software	
				- Evolution processos e atividade	
				Desenvolvimento de Software Seguro	
Contexto Social	Histórico e Visão Geral para prática Profissional.	Capacidade de Liderança e Colaboração	Perspectivas e Impactos da prática profissional	Dinâmica de Grupo e Psicologia	Conhecimento Sociotécnico
Ferramentas Analíticas em Ética	Ferramentas, padrões e/ou restrições de Engenharia relevantes em Prática Profissional	Comunicação	Questões e responsabilidades profissionais	Habilidades de Comunicação	
Ética Profissional	Estratégia de Comunicação Eficazes	Negociação	Questões Ambientais	Profissionalismo	
Propriedade Intelectual	Abordagens de equipe Interdisciplinar	Criatividade e Capacidade de Análise crítica e ética	Questões éticas, legais e de privacidade		
Privacidade e Liberdades Civis	Enquadramentos Filosóficos e Questões Culturais		Propriedade Intelectual		
Comunicação Profissional	Soluções de Engenharia e efeitos sociais		Comunicações		
Sustentabilidade	Responsabilidades profissionais e Éticas		Trabalho em equipe e gestão de conflitos		
História	Propriedade Intelectual e questões legais		Habilidades de empregabilidade e carreiras em TI		
Economias de Computação	Questões Contemporâneas		Contexto Social da Computação		
Políticas de Segurança, leis e Crimes Informáticos	Questões de Negócios e Gerenciamento		Objetivos, planos, tarefas, prazos e riscos		
	Escolhas na Prática profissional		Papel e regulamentos do Governos		
			Desafios e Abordagens Globais		
			Gerenciamento de Riscos		
			Computação Sustentável		

Fonte: Elaboração Própria

Tabela 4.22 Tabela de Relevância dos Conhecimentos de ES extraídos da análise.

Relevância dos Conhecimentos em Engenharia de Software em cada Guia de Graduação						
Unidades de Conhecimento ou Tópicos Específicos de ES						Relevância do Tópico/ Unidade de Conhecimento
CS-2013	CE-2016	SI-2010	TI-2017	SE-2014		
Processo de Software	Processos de Software		Arquitetura de Sistemas	Fundamentos de Modelagem de Software	Comentários e análise estática teste	Nuclear/ Essencial/ Principal
Processo de Software	Análise e elicitação de requisitos.		Teste e garantia de qualidade	Tipos de Modelos	Análise de problemas e relatórios	
Gerenciamento de Projetos de Software	Histórico e Visão Geral do Desig de Software		Fatores Humanos no Desing	Fundamentos da Análise	Conceitos e cultura de qualidade de software	
Ferramentas e Ambientes em Engenharia de Software			Conceitos de Aplicativos	Fundamentos de Requisitos	Garantia de processo	
Engenharia de Requisitos			Frameworks de Desenvolvimento	Elicitação de Requisitos	Garantia do produto	
Desing de Software				Validação de Requisitos	Conceitos de processo	
Construção de Software				Conceitos de Design de Software	Implementação do processo	
Validação e Verificação de Software				Estratégias de Design de Software	- Planejamento e acompanhamento de projetos	
Evolução de Software				Design Arquitetural	Gestão de configuração de software	
Confiabilidade de Software				Design de Interação Humano-Computador	- Evolution processos e atividade	
				Design Detalhado	Desenvolvimento de Software Seguro	
				V&V terminologia e fundamentos		
Engenharia de Software Seguro	Testes Integração e validação de Sistemas de Software	Design de Software	Cadeia de Suprimentos e Garantia de Software			Eletivo/Desejável
Métodos Formais em Engenharia de Software	Ferramentas, padrões e/ou restrições de Engenharia de software relevantes	Ciclo de Vida de Desenvolvimento de software	Modelo de Processos e Atividades de desenvolvimento e Gerenciamento de Software			
	Teste e qualidade de software	Análise de Requisitos	Desenvolvimento de Software Baseado em plataforma			
		Técnicas para Modelar Estruturas do Software	Ferramentas e Serviços			
		Desing de Software Centrado no usuário	Gerenciamento de software e de projetos de software			
		Teste de Software	Implantação, operações Manutenção de Software			

Fonte: Elaboração Própria

O gráfico da relação de quantidade entre os conteúdos de *ES* é exibido na Figura 4.15.

Na Tabela 4.23, podemos observar que há várias Unidades de Conhecimentos sociotécnicos essenciais e eletivas nos guias curriculares. Foram mapeados *trinta e dois (32) tópicos sociotécnicos essenciais nos Guias explorados e dez (10) tópicos sociotécnicos eletivos*. O gráfico da relação de quantidade entre os conteúdos Sociotécnicos é exibido na Figura 4.16.

Analisando de forma criteriosa as relações de presença das unidades de registros (Unidades de Conhecimento) nas posições de relevância nos currículos (Essencial/Eletivo), podemos perceber que a Educação em Engenharia de Software é norteadora para o desenvolvimento das necessidades de formação dos graduados em Computação. A importância do pensamento sociotécnico se mostra relevante não apenas para a formação dos egressos, mas também para a formação específica em cada uma das áreas ou disciplinas da Computação.

Os guias curriculares abordam o corpo de conhecimento como um apanhado de competências necessárias para a formação do graduado que devem ser estruturadas nos currículos dos cursos. Sendo o conhecimento de determinadas áreas da Computação como uma competência a ser considerada nos currículos, podemos definir a categoria *Competência* como o tema geral que resume as características dos dois eixos temáticos extraídos das unidades de registros (Tabela 4.24).

Assim, podemos observar a relevância do conhecimento em ES e do conhecimento sociotécnico como competências essenciais para a formação em qualquer graduação em Computação. O conhecimento sociotécnico assume um papel preponderante não apenas

Tabela 4.23 Tabela de Relevância dos Conhecimentos Sociotécnicos extraídos da análise.

Tópicos Relacionados com os Conhecimentos Sociotécnicos em cada Guia de Graduação					
Unidades de Conhecimento ou Tópicos Sociotécnicos					Relevância do Tópico/ Unidade de Conhecimento
CS-2013	**CE-2016**	**SI-2010**	**TI-2017**	**SE-2014**	
Contexto Social	Histórico e Visão Geral para prática Profissional.	Capacidade de Liderança e Colaboração	Perspectivas e Impactos da prática profissional	Dinâmica de Grupo e Psicologia	Nuclear/Essencial/Principal
Ferramentas Analíticas em Ética	Ferramentas, padrões e/ou restrições de Engenharia relevantes em Prática Profissional	Comunicação	Questões e responsabilidades profissionais	Habilidades de Comunicação	
Ética Profissional	Estratégia de Comunicação Eficazes	Negociação	Questões Ambientais	Profissionalismo	
Propriedade Intelectual	Abordagens de equipe Interdisciplinar	Criatividade e Capacidade de Análise crítica e ética	Questões éticas, legais e de privacidade		
Privacidade e Liberdades Civis	Enquadramentos Filosóficos e Questões Culturais		Propriedade Intelectual		
Comunicação Profissional	Soluções de Engenharia e efeitos sociais		Comunicações		
Sustentabilidade	Responsabilidades profissionais e Éticas		Trabalho em equipe e gestão de conflitos		
	Propriedade Intelectual e questões legais		Habilidades de empregabilidade e carreiras em TI		
	Questões Contemporâneas				
	Questões de Negócios e Gerenciamento				
História	Escolhas na Prática profissional		Contexto Social da Computação		Eletivo/Desejável
Economias de Computação			Objetivos, planos, tarefas, prazos e riscos		
Políticas de Segurança, leis e Crimes Informáticos			Papel e regulamentos do Governos		
			Desafios e Abordagens Globais		
			Gerenciamento de Riscos		
			Computação Sustentável		

Fonte: Elaboração Própria

Tabela 4.24 Agrupamento de Eixos Temáticos extraídos dos Guias ACM em Categoria.

Eixos Temáticos	Categorias
Conhecimentos em Engenharia de Software	Competências
Conhecimentos Sociotécnicos	

Fonte: Elaboração Própria

na concepção do currículo, mas como competência essencial para a efetiva formação técnica específica alinhada com as necessidades da indústria e do mundo do trabalho. O conhecimento em ES se mostra como competência-chave para o desenvolvimento profissional em Computação alinhado com a necessidade de desenvolvimento de soluções de software complexos e imprescindíveis para o desenvolvimento da Sociedade.

O alcance das competências em ES e desenvolvimento sociotécnico para atender às necessidades esperadas de formação de egressos de cursos superiores em Computação, passa por um planejamento de práticas de ensino que aproximem o desenvolvimento destas competências com as necessidades da indústria. As práticas de ensino devem ter um alinhamento do objetivo pedagógico que o docente ou instrutor almeja com os recursos que o mesmo escolhe para atingir tais objetivos. Considerando que projetos FLOSS são um recurso com reconhecida relevância para práticas de ensino em ES que aproximam as atividades acadêmicas com as necessidades da indústria, é importante que o docente tenha conhecimento dos aspectos que possam interferir nos seus objetivos de instrução ou aprendizagem. A Análise de Conteúdo será conduzida para a investigação

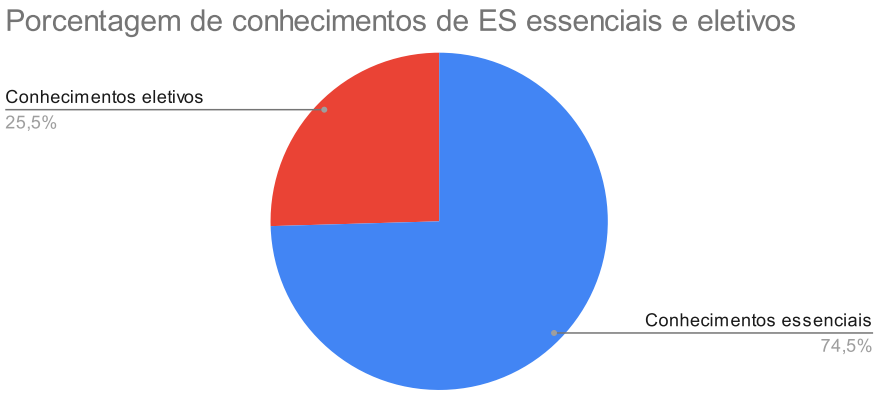

Figura 4.15 Relação quantitativa: tópicos de ES essenciais e eletivos nos Currículos da ACM
Fonte: Elaboração Própria

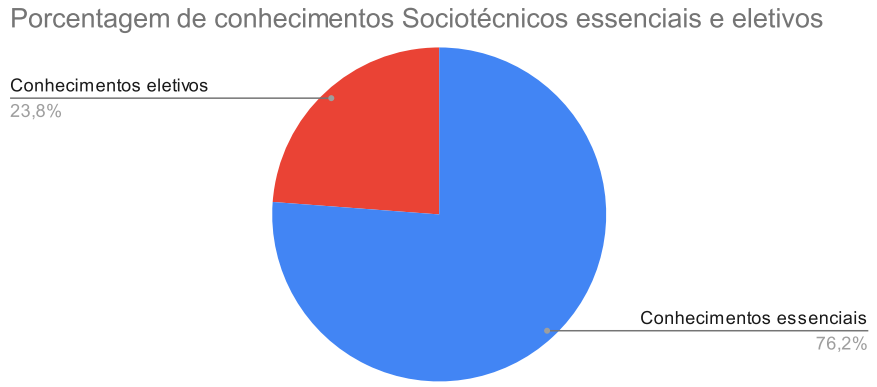

Figura 4.16 Relação quantitativa: tópicos Sociotécnicos essenciais e eletivos nos Currículos da ACM
Fonte: Elaboração Própria

destes aspectos.

4.3 MAPEAMENTO DE INFLUÊNCIA NO ENSINO/APRENDIZAGEM EM ES POR MEIO DE PROJETOS FLOSS

Conhecendo as competências relacionadas a ES e ao comportamento sociotécnico, partimos para a análise dos estudos que abordam o uso de Projetos FLOSS no ensino de ES. Segundo o (ARDIS et al., 2014b), os objetivos pedagógicos para o desenvolvimento das competências dos egressos passam pela escolha de estratégias que auxiliem no alcance das mesmas. As prerrogativas de formação perpassam pela aproximação do estudante com as práticas e conteúdos que permeiam a realidade da indústria e do mundo do trabalho (SABIN et al., 2017).

Projetos FLOSS assumem características interessantes para a abordagem educacional. Seu processo de desenvolvimento distribuído, a complexidade das soluções em desenvol-

vimento, a dinâmica das comunidades e a acessibilidade dos projetos para um amplo público interessado em contribuir com a comunidade, sugerem que Projetos FLOSS podem ser considerados um laboratório dinâmico que traz para dentro do meio acadêmico a rotina de atividades complexas do mundo profissional de desenvolvimento de software (STAMELOS, 2011; NASCIMENTO, 2017).

Quando percebemos a necessidade de formação superior em Computação condizente com a realidade da indústria e do mundo do trabalho, percebemos o potencial dos Projetos FLOSS para a efetiva formação na área de desenvolvimento e Engenharia de software. Nesse aspecto, os docentes podem ver os Projetos FLOSS como uma ferramenta poderosa para seus objetivos educacionais. Porém, como em qualquer prática educativa, os objetivos propostos podem não ser alcançados em parte ou até mesmo em sua totalidade por diversos aspectos. As questões discentes, acadêmicas e institucionais podem interferir no processo. Além destas, os recursos pedagógicos ou objetos de ensino podem não ser condizentes com a proposta educacional ansiada pelo docente. É sobre os aspectos de interferência nos processos de ensino e aprendizagem mediadas por uso de FLOSS que vamos nos debruçar.

Para realizar uma análise de conteúdo sobre os aspectos que estimulam ou limitam a aprendizagem em ES usando FLOSS como elemento prático de estudo, é necessário compreender os elementos temáticos desses aspectos indicados em estudos que são referência no assunto. Esses elementos são supostos problemas, limitações ou vantagens diretamente relacionados ao desenvolvimento da aprendizagem. Sobre alguns desses aspectos, (NASCIMENTO, 2017; NASCIMENTO; BITTENCOURT; CHAVEZ, 2015; FILHO; FF, 2017) versam sobre a necessidade de alinhar o tempo e a cobertura do componente acadêmico e uso do projeto FLOSS, manter a estrutura tecnológica compatível, observar o tamanho e a complexidade do projeto escolhido.

A importância do interesse do estudante nos projetos é citada por (SOWE; STAMELOS, 2007). Os autores também versam sobre o tamanho do projeto e da comunidade de desenvolvimento, o domínio da linguagem de programação e o *status* do desenvolvimento. Além destes aspectos, a participação e envolvimento da comunidade são citados por (STAMELOS, 2011). O autor também versa sobre a confiança do estudante no contato com projetos FLOSS. Stamelos também versa sobre as regras próprias que os projetos possuem. Questões como personalidade e temperamento, domínio do idioma também são abordados pelos autores. (KON et al., 2011) versam sobre a importância de contribuir para causas relevantes, enquanto (CHAVEZ et al., 2011) destacam a importância de participação docente e discente em grandes projetos e a familiaridade prévia com tópicos e ferramentas.

Esses critérios são índices de aspectos que podem interferir na aprendizagem dos estudantes. A escolha do projeto, ou dos projetos, a liberdade de interação e participação instruídas aos estudantes nos projetos escolhidos são fatores preponderantes para o alcance dos objetivos de aprendizagem esperadas pelo professor (MUSSOI; FLORES; BEHAR, 2010; NASCIMENTO, 2017; FILHO; FF, 2017). Portanto, conhecer os aspectos comuns na prática educativa de ES mediada por projetos FLOSS é relevante para avaliar um projeto escolhido como eficaz para determinado objetivo de aprendizagem.

A estratégia escolhida para compreender os aspectos de interferência no uso de FLOSS

para ensinar é analisar as experiências ou relatos de professores e pesquisadores sobre o assunto. Segundo (PUGLISI; FRANCO, 2005, p.54), "O índice pode ser a menção explicita, ou subjacente, de um tema em uma mensagem". Conhecendo os elementos básicos de investigação (índices) citados por autores na área, partimos para a Análise de Conteúdo de trabalhos que versam sobre o uso de Projetos FLOSS na EES.

O trabalho de (BRITO et al., 2018), um mapeamento sistemático da literatura sobre uso de Projetos FLOSS na Educação em Engenharia de Software, traz a base para a seleção da amostra dos estudos sobre uso de FLOSS na EES. Ao analisarmos o protocolo de busca e investigação, e realizarmos a leitura flutuante dos trabalhos selecionados, percebemos que os critérios temáticos obedecem aos nossos propósitos de busca. As bases utilizadas, os critérios de inclusão e exclusão e palavras-chave são coerentes com o universo de estudos que suprem o objetivo desse estudo. O estudo selecionou trinta e três trabalhos que formaram o *corpus* de nossa análise. Analisando os critérios de busca e seleção, é possível notar que tais trabalhos não fazem parte de todo o universo de análise que aborda a temática a ser investigada, mas representam uma amostra pertinente desse universo de estudo no período analisado. Bardin versa sobre o uso de amostras para a análise de conteúdo:

> A análise pode efetuar-se numa amostra desde que o material a isso se preste. A amostragem diz-se rigorosa se a amostra for uma parte representativa do universo inicial. (BARDIN, 2011, p.127)

Sendo assim, partimos para a análise de conteúdo dos trabalhos selecionados. As fichas de Unidades de Contexto e Unidades de Registro dos trabalhos selecionados têm identificação idêntica ao fichamento do trabalho de origem: Código, Autores e Título. A análise do conteúdo se dá nos resumos, fundamentação, desenvolvimento e resultados dos trabalhos. Os critérios para seleção dos aspectos se dará na identificação das temática próximas aos elementos (*índices*) citados pelos autores de referência já abordados. As unidades de registros extraídas das unidades de contextos (*excertos*) retirados dos textos são referentes a citações diretas e indiretas dos autores, resultados de questionários, resumos e percepções indicadas no corpo do texto. Ou seja, qualquer modo de percepção direta dos próprios autores/participantes do estudo ou de percepção indireta (citações ou resultados de outras pesquisas indicadas no trabalho) sobre interferência em resultados de aprendizagem ou influência em questões pedagógicas, didáticas, técnicas ou comportamentais.

Dos trinta e três trabalhos analisados, trinta destes apresentaram unidades de registros referentes a aspectos de influência no uso de Projetos FLOSS no ensino de ES. Após análise da exploração de cada trabalho, notam-se unidades de registros frequentes sobre temáticas equivalentes ou próximas a aspectos citados aqui segundo trabalhos referência. É por meio das frequências de cada tema em comum nos trabalhos, que os eixos temáticos podem ser retirados.

Após análise e organização das unidades de registros de cada trabalho (Apêndice C) foram extraídos *quatrocentos e vinte e sete (427)* unidades de registros. Cada Unidade de Registro se refere a um tema sobre aspecto de influência técnica ou pedagógica no

ensino de ES usando FLOSS. Cerca de dezenove (19) aspectos/fatores de influência na aprendizagem foram mapeados das unidades de registros:

A1-Tamanho e complexidade do projeto. Projetos devem ter tamanho e complexidade compatíveis com o objetivo do instrutor. Projetos muito ou pouco complexos podem gerar insatisfação, frustração, fadiga e incompatibilidade com o que os alunos esperam.

A2-Interesse, motivação e atração. Alunos podem sentir interesse em participar ou iniciar atividades em projetos Open Sorce. Frustrações, expectativas, senso de orgulho ou falta de confiança podem interferir no engajamento e na aprendizagem dos alunos.

A3-Adaptação ao currículo ou curso. Professores precisam se atentar na busca de projetos ou desenvolver atividades que sejam compatíveis com os objetivos curriculares ou com o planejamento do curso que leciona.

A4-Nível de formação e experiência do estudante. Estudantes novatos ou experientes em FLOSS e Estudantes concluintes ou iniciantes na sua formação acadêmica podem responder de formas diferentes a atividades ou interações no projeto ou na comunidade além de ter motivação ou resistência para ingressar no envolvimento profissional em Projetos FLOSS.

A5-Tempo, Calendário ou Cronograma. É essencial ao docente se atentar ao tempo gasto no planejamento e execução de suas atividades de ensino num projeto, atividades da comunidade podem ser incompatíveis com os prazos dos cursos, o esforço nos projetos devem respeitar o cronograma dos próprios objetivos de ensino da atividade ou componente lecionado.

A6-Uso ou conhecimento de ferramentas e ambientes de desenvolvimento. Aprender novas ferramentas pode ser positivo ou interferir na aprendizagem e estimular ou frustrar estudantes, a necessidade de domínio prévio de ferramentas pode desafiar a administração da aprendizagem pelo professor.

A7-Suporte, acompanhamento e avaliação do estudante. Alunos podem enfrentar dificuldade em participar e se manter nas atividades do Projeto e o suporte docente é importante. O docente deve se atentar a formas de acompanhamento e avaliação das participações dos alunos nas atividades planejadas.

A8-Capacidade e Habilidade de Programação. O nível de domínio de programação ou de uma linguagem de programação pode atrair ou intimidar a participação dos estudantes, pode inviabilizar a participação ou inserir o aluno em atividades relevantes na comunidade do projeto, pode causar frustração ou senso de orgulho ao aluno nas atividades.

A9-Suporte ou viabilidade acadêmica. As instituições de ensino devem auxiliar na participação dos alunos e dos professores nas dinâmicas de aprendizagem usando

projetos FLOSS, fornecendo ferramentas, infraestrutura e apoio técnico compatível com as demandas técnicas ou pedagógicas de participação nas atividades.

A10-Percepção ou Autopercepção das Habilidades de Computação. É importante que os alunos se percebam nas atividades e tenham consciência da sua evolução e dos colegas, conhecendo suas qualidades e pontos de melhorias tanto no aspecto técnico como sociotécnico.

A11-Dinâmica, rotina e cultura de Projetos FLOSS. A rotina de contribuição, as divisões de tarefas, o modelo de desenvolvimento distribuído e a disposição de elementos de aprendizagem não estruturados são aspectos que o docente deve se atentar ao planejar as atividades usando projetos FLOSS.

A12-Relação com a comunidade, Receptividade do Projeto. A interação dos alunos com a comunidade deve ser observada, membros da comunidade podem demorar para responder alunos, não dar suporte nas atividades, pode haver choque entre a expectativa da participação do aluno na visão da comunidade e do professor. Contribuições dos alunos podem exigir planejamento e negociação prévia com líderes.

A13-Restrições, Viabilidade e Adequação do Projeto. Alguns Projetos podem apresentar restrições técnicas, legais, éticas e culturais para uma atividade a ser desenvolvida. Leis acordos e regras internas podem inviabilizar atividades e o alcance de objetivos de ensino usando o projeto.

A14-Estado atual do projeto. Projetos podem estar inacabados ou maduros, com comunidade ativa ou abandonado, com pouca ou muita documentação e suporte. Cada um desses fatores podem ser usados nas atividades de acordo com o perfil dos alunos ou objetivos do professor, podendo também interferir na confiança e expectativa dos alunos e gerar esforço extra ao planejamento docente.

A15-Assunto ou Domínio do Projeto. O planejamento da aprendizagem e da participação do aluno nas atividades pode ser influenciada pelo assunto do projeto. Alunos tendem a se interessar por projetos com assuntos ou domínios que tenham interesse como música, jogos, sustentabilidade e altruísmo. Alunos podem participar de atividades de acordo com o conhecimento do domínio do projeto. Mulheres se atraem por projetos altruístas.

A16-Estratégia para atividades e participações dos estudantes. A forma de participação do aluno no projeto, a liberdade de escolha do projeto, a função do estudante na interação com a comunidade e com os colegas, a organização dos grupos e os critérios de metas de participação podem ser decisivos no objetivo de ensino.

A17-Curva de Aprendizagem. Alunos podem enfrentar Curva de aprendizagem para desenvolver atividades. Curva de aprendizagem acentuada pode demandar esforço

docente e do aluno para a continuação de atividades. Alunos podem desenvolver curva de aprendizagem acentuada ao trabalhar em projetos.

A18-Questões sociais, pessoais e legais. Alunos podem ter problemas de comunicação, dificuldade com o idioma, questões religiosas, culturais e de gênero podem estimular ou intimidar a participação dos alunos nas atividades.

A19-Desgaste, Entusiasmo e Experiência do Docente. Docentes inexperientes podem sofrer desgaste na seleção de projetos e na escolha das atividades. Professores Entusiasmados estimulam a participação e confiança do estudante. Professores podem sofrer desgaste e ficarem sobrecarregados no acompanhamento de muitos projetos ou projetos muito complexos.

Após a descrição dos aspectos em temas comuns citados em todos os trabalhos analisados, os mesmos foram classificados e quantificados de acordo com o número de unidades de registros retirados em cada unidade de contexto extraídos dos trabalhos. A Tabela 4.25 mostra a relação da quantidade de Unidades Registros de cada Tema (Aspectos) extraídas de cada trabalho.

Tabela 4.25 Relação de Unidades de Registros em cada trabalho analisado

Trabalho Analisado	Quantidade de Unidades de Registros referente a cada Eixo temático (Apectos)																		
ID - Tabela - Apêndice	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17	A18	A19
89 - Tabela C1(Apêndice C)	2	2	1		1	2	1									2			
93 - Tabela C2(Apêndice C)	1						1	1	1										
106 - Tabela C3(Apêndice C)	1	2				1				1	2								
115 - Tabela C4(Apêndice C)						1				1									
135 - Tabela C5(Apêndice C)	4	4			1	1		3		1	1		1	8	1	1			
1088 - Tabela C6(Apêndice C)		2												1	1	1			
1097 - Tabela C7(Apêndice C)	1									1		1			1	3			1
1192 - Tabela C8(Apêndice C)		3		2			1									1			
1193 - Tabela C9(Apêndice C)		1	1	1	2	2	1		2		1	1	2	1	1	3	1	1	3
4503 - Tabela C10(Apêndice C)	2			3	1	4	3				1		1	1		2		1	1
4663 - Tabela C11(Apêndice C)		1					1			1	1				2	2			
4811 - Tabela C12(Apêndice C)				1		1	1					1				1			
4815 - Tabela C13(Apêndice C)		4		3		1		3		6	1	7				3			
4966 - Tabela C14(Apêndice C)	3	4			2	7	1		1	3	6	8	1	1	6	5	2	2	3
5147 - Tabela C15(Apêndice C)	3	2		2	1	4	1	2		2	2	2	1		1		1		
5328 - Tabela C16(Apêndice C)	2	1			2	3		1							3	5		1	
5329 - Tabela C17(Apêndice C)	4	4		3	1	2	1	1			1	4		4		5	1		
5335 - Tabela C18(Apêndice C)	1	2				2				1	2	1	1	1			2		
5343 - Tabela C19(Apêndice C)	1	2			1		1							1		1			1
5353 - Tabela C20(Apêndice C)	1				1	1				1									
5357 - Tabela C21(Apêndice C)	1	3	1	1			1			3	1				2	2			
5546 - Tabela C22(Apêndice C)		1				1		2		4	1	3				2	2		
5676 - Tabela C23(Apêndice C)		2				2	4	1		3		1				1			2
17796 - Tabela C24(Apêndice C)	5					1		2		2		2		1	1	1			
17800 - Tabela C25(Apêndice C)																			1
17805 - Tabela C26(Apêndice C)											1			1					
17830 - Tabela C27(Apêndice C)	1	6	2		1	2	1			2	3	3	1		2		2	2	
17882 - Tabela C28(Apêndice C)		1	2		1	1		1		2		1						1	
18359 - Tabela C29(Apêndice C)	1	2				1	1			1	1	2				1			
18433 - Tabela C30(Apêndice C)		2	1		1					1									1
Total de Unidades de Registros	34	51	8	16	16	40	20	17	4	36	25	37	8	20	21	42	11	8	13

Fonte: Elaboração Própria

Após a seleção, quantificação e organização das Unidades de Registros em temas, torna-se possível fazer uma análise quantitativa de frequência dos temas mapeados. Segundo (PUGLISI; FRANCO, 2005, p.54):

> Neste caso, o indicador correspondente seria frequência observada acerca do tema em questão. Para tal, deve-se recorrer a uma análise quantitativa sistemática para que seja possível identificar a frequência relativa ou absoluta do tema escolhido e a proporcionalidade de sua menção em relação a outros temas igualmente presentes.

A frequência absoluta mostra o total de aspectos referente a cada um dos temas encontrados no mapeamento do *corpus*. A frequência relativa é a quantidade de vezes que o aspecto foi encontrado referente ao total de aspectos mapeados. A relação de frequência absoluta e relativa dos aspectos mapeados é apresentada na Tabela 4.26.

Tabela 4.26 Tabela de análise quantitativa de citações dos aspectos mapeados.

Análise da Frequência do aspectos extraídos da análise de conteúdo no corpus da pesquisa			
Temas de aspectos citados	**Frequência Absoluta**	**Frequência Relativa**	**Frequência Relativa (%)**
A1- Tamanho e complexidade do projeto	**34**	**0,07962529274**	**7,9625293%**
A2- Interesse, motivação e atração	**51**	**0,1194379391**	**11,9437939%**
A3- Adaptação ao currículo ou curso	**8**	**0,018735363**	**1,8735363%**
A4- Nível de formação e experiência do estudante:	**16**	**0,037470726**	**3,7470726%**
A5- Tempo, Calendário ou Cronograma	**16**	**0,037470726**	**3,7470726%**
A6- Uso ou conhecimento de ferramentas e ambientes de desenvolvimento	**40**	**0,09367681499**	**9,3676815%**
A7- Suporte, acompanhamento e avaliação do estudante	**20**	**0,04683840749**	**4,6838407%**
A8- Capacidade e Habilidade de Programação	**17**	**0,03981264637**	**3,9812646%**
A9- Suporte ou viabilidade acadêmica	**4**	**0,009367681499**	**0,9367681%**
A10- Percepção ou Autopercepção das Habilidade de Computação	**36**	**0,08430913349**	**8,4309133%**
A11- Dinâmica, rotina e cultura de Projetos FLOSS	**25**	**0,05854800937**	**5,8548009%**
A12- Relação com a comunidade, Receptividade do Projeto	**37**	**0,08665105386**	**8,6651054%**
A13- Restrições, Viabilidade e Adequação do Projeto	**8**	**0,018735363**	**1,8735363%**
A14- Estado atual do projeto	**20**	**0,04683840749**	**4,6838407%**
A15- Assunto ou Domínio do Projeto	**21**	**0,04918032787**	**4,9180328%**
A16- Estratégia para atividades e participações dos estudantes	**42**	**0,09836065574**	**9,8360656%**
A17- Curva de Aprendizagem	**11**	**0,02576112412**	**2,5761124%**
A18- Questões sociais, pessoais e legais	**8**	**0,018735363**	**1,8735363%**
A19- Desgaste, Entusiasmo e Experiência do Docente	**13**	**0,03044496487**	**3,0444965%**
Total	**427**	**1**	**100%**

Fonte: Elaboração Própria

Ao compararmos as quantidades de citações referente ao total, é possível definir os aspectos encontrados no geral de citações como mostra a Tabela 4.27. Ao analisarmos apenas o quantitativo de unidades de registros comuns a cada tema, temos uma relevância de citações do aspecto A2 (Interesse, motivação e atração) como o primeiro mais citado, com 11,9437939 % de citações, seguido pelo aspecto A16 (Estratégia para atividades e participações dos estudantes) com 9,8360656% de citações, e pelo Uso e conhecimento de ferramentas e ambientes de desenvolvimento (A6) com 9,3676815% das citações. Além

destes, a Relação com a Comunidade, a percepção e autopercepção de habilidades, e tamanho e complexidade do projeto estão entre os seis aspectos mais mapeados.

Tabela 4.27 Aspectos mais citados segundo ordem de frequência absoluta e relativa

Temas de aspectos citados	Frequência Absoluta	Frequência Relativa (%)
A2	51	11,9437939%
A16	42	9,8360656%
A6	40	9,3676815%
A12	37	8,6651054%
A10	36	8,4309133%
A1	34	7,9625293%
A11	25	5,8548009%
A15	21	4,9180328%
A7	20	4,6838407%
A14	20	4,6838407%
A8	17	3,9812646%
A4	16	3,7470726%
A5	16	3,7470726%
A19	13	3,0444965%
A17	11	2,5761124%
A3	8	1,8735363%
A13	8	1,8735363%
A18	8	1,8735363%
A9	4	0,9367681%

Fonte:Elaboração Própria

Podemos perceber também a relevância da Cultura e dinâmica do FLOSS (A11), do Assunto ou domínio do projeto (A15), do Suporte, acompanhamento e avaliação do estudante e do Estado do projeto. Do total de registros, os aspectos menos citados foram: Suporte ou Viabilidade acadêmica (A9); Questões sociais, pessoais e legais; e as restrições, viabilidade e adequação do projeto (A13), todos com menos de 10 citações.

Quando a análise permite o mapeamento em cada trabalho, é possível também analisar em quantos trabalhos o aspecto foi citado. A Tabela 4.28 informa em quais dos trinta trabalhos analisados os aspectos foram identificados. A Figura 4.17 mostra a representação gráfica da Tabela 4.28.

Os aspectos foram ordenados de acordo com o número de trabalhos no qual foram mapeados como mostra a Tabela 4.29. Podemos perceber uma pequena variação na ordem quantitativa de citações de cada um dos aspectos mapeados na análise de conteúdo em todos os trinta trabalhos analisados.

O aspecto A2 foi o mais citado em relação à quantidade de trabalhos analisados e aparece em setenta por cento dos trabalhos analisados. O aspecto A6 foi o terceiro mais citado em relação à frequência absoluta e aparece como o segundo mais presente em relação ao total de trabalhos. O aspecto A16 que foi o segundo mais citado em relação a

Tabela 4.28 Quantidade de trabalhos que citam cada aspecto.

Trabalho Analisado	Relação de Aspecto a cada trabalho analisado																		
ID - Tabela - Apêndice	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17	A18	A19
89 - Tabela C1(Apêndice C)	X	X	X		X	X	X									X			
93 - Tabela C2(Apêndice C)	X						X	X	X										
106 - Tabela C3(Apêndice C)	X	X				X				X	X								
115 - Tabela C4(Apêndice C)						X				X									
135 - Tabela C5(Apêndice C)	X	X			X	X		X		X	X		X	X	X	X			
1088 - Tabela C6(Apêndice C)		X												X	X	X			
1097 - Tabela C7(Apêndice C)	X									X		X			X	X			X
1192 - Tabela C8(Apêndice C)		X		X			X									X			
1193 - Tabela C9(Apêndice C)		X	X	X	X	X	X		X		X	X	X	X	X	X	X	X	X
4503 - Tabela C10(Apêndice C)	X			X	X	X	X				X		X	X		X		X	X
4663 - Tabela C11(Apêndice C)		X					X			X	X				X	X			
4811 - Tabela C12(Apêndice C)				X		X	X					X				X			
4815 - Tabela C13(Apêndice C)		X		X		X		X		X	X	X				X			
4966 - Tabela C14(Apêndice C)	X	X			X	X	X		X	X	X	X	X	X	X	X	X	X	X
5147 - Tabela C15(Apêndice C)	X	X		X	X	X	X	X		X	X	X	X		X		X		
5328 - Tabela C16(Apêndice C)	X	X			X	X		X							X	X		X	
5329 - Tabela C17(Apêndice C)	X	X		X	X	X	X	X			X	X		X		X	X		
5335 - Tabela C18(Apêndice C)	X	X				X				X	X	X	X	X			X		
5343 - Tabela C19(Apêndice C)	X	X			X		X							X		X			X
5353 - Tabela C20(Apêndice C)	X				X	X				X									
5357 - Tabela C21(Apêndice C)	X	X	X	X			X			X	X				X	X			
5546 - Tabela C22(Apêndice C)		X				X		X		X	X	X				X	X		
5676 - Tabela C23(Apêndice C)		X				X	X	X		X		X				X			X
17796 - Tabela C24(Apêndice C)	X					X		X		X		X		X	X	X			
17800 - Tabela C25(Apêndice C)																			X
17805 - Tabela C26(Apêndice C)											X			X					
17830 - Tabela C27(Apêndice C)	X	X	X		X	X	X			X	X	X	X		X		X	X	
17882 - Tabela C28(Apêndice C)		X	X		X	X		X		X		X						X	
18359 - Tabela C29(Apêndice C)	X	X				X	X			X	X	X				X			
18433 - Tabela C30(Apêndice C)		X	X		X					X									X
Quantidade de trabalhos por Aspecto	17	21	6	8	13	20	15	10	3	18	15	14	7	10	11	19	7	6	8

Fonte:Elaboração Própria

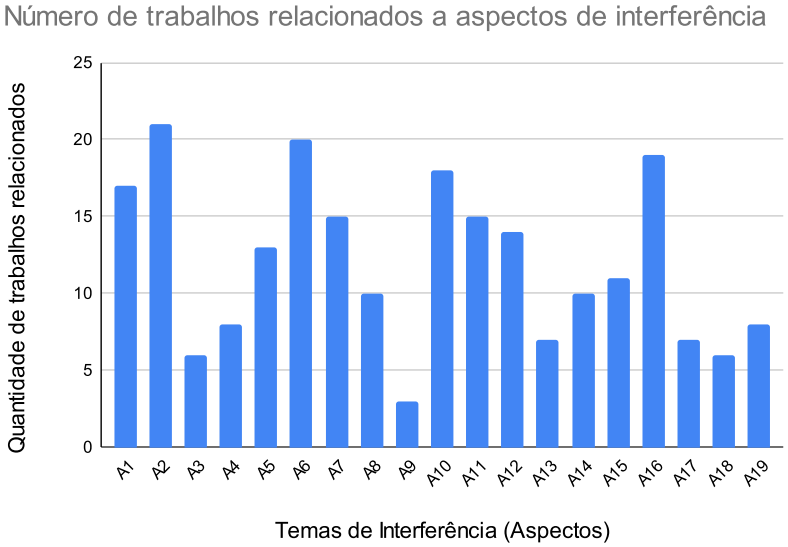

Figura 4.17 Relação entre trabalhos e aspectos de interferência na EES com uso de FLOSS
Fonte:Elaboração Própria

todas as citações, aparece em terceiro em relação aos trabalhos analisados.

Tabela 4.29 Ordem quantitativa de trabalhos no qual o aspecto foi mapeado.

Ordem quantitativa de trabalhos no qual o aspecto foi mapeado.																	
A2	A6	A16	A10	A1	A7	A11	A12	A5	A15	A8	A14	A4	A19	A13	A17	A3	A18	A9
21	20	19	18	17	15	15	14	13	11	10	10	8	8	7	7	6	6	3
70,00%	66,67%	63,33%	60,00%	56,67%	50,00%	50,00%	46,67%	43,33%	36,67%	33,33%	33,33%	26,67%	26,67%	23,33%	23,33%	20,00%	20,00%	10,00%

Fonte:Elaboração Própria

Ao analisarmos a ordem quantitativa de aspectos tanto em relação ao valor absoluto de citações (Unidades de Registro) quanto em relação ao total de trabalhos, podemos *perceber a relevância que os aspectos têm no ensino de Engenharia de Software mediado por Projetos FLOSS do mundo real.*

Todo o processo de análise de conteúdo for direcionado aos trabalhos sobre uma perspectiva direta ou indireta de fichamento das Unidades de Contexto (Excertos do Texto). Os trabalhos de (DING et al., 2014), (FILHO; FF, 2017) e de (CICIRELLO, 2017) não entraram na análise, pois não se enquadraram no princípio de pertinência da análise segundo Bardin. O modo de percepção direta foram os trabalhos com unidades de contexto extraídas da própria percepção dos autores nos resultados ou no desenvolvimento de seus estudos aplicados. O modo de percepção indireta foram os trabalhos com unidades de contextos extraídos da percepção teórica dos pesquisadores usada para fundamentar seus estudos ou de bases teóricas extraídas de pesquisas puramente bibliográficas. O modo de análise e percepção em cada trabalho são descritos na Tabela 4.30.

Tabela 4.30 Modo de percepção para extração das Unidades de Contexto e de Registros (Citações)

Id	Modelo de Análise	Modo de Percepção	Id	Modelo de Análise	Modo de Percepção
89	Estudo Aplicado	Direta	5328	Estudo Aplicado	Direta e Indireta
93	Estudo Aplicado	Indireta	5329	Estudo Aplicado	Direta e Indireta
106	Estudo Aplicado	Direta/Indireta	5335	Estudo Aplicado	Direta e Indireta
115	Estudo Aplicado	Direta	5343	Estudo Aplicado	Direta
134	Estudo Aplicado	Não se Aplica	5353	Estudo Aplicado	Indireta
135	Estudo/Aplicado	Direta e Indireta	5357	Estudo Aplicado	Direta e Indireta
1088	Estudo Aplicado	Direta e Indireta	5546	Estudo Aplicado	Direta
1097	Bibliográfico	Indireta	5676	Estudo Aplicado	Direta e Indireta
1192	Estudo Aplicado	Indireta	17796	Estudo Aplicado	Direta
1193	Estudo Aplicado	Indireta	17800	Estudo Aplicado	Direta
4503	Estudo Aplicado	Direta e Indireta	17805	Bibliográfico	Indireto
4663	Estudo Aplicado	Indireta	17830	Estudo Aplicado	Indireta e Direta
4811	Estudo Aplicado	Direta	17845	Não Compatível	Não se Aplica
4815	Estudo Aplicado	Direta e Indireta	17882	Estudo Aplicado	Direta
4966	Estudo Aplicado	Indireta e Direta	18359	Estudo Aplicado	Direta e Indireta
5147	Estudo Aplicado	Direta e Indireta	18433	Estudo Aplicado	Direta e Indireta

Fonte:Elaboração Própria

Independentemente da quantidade maior ou menor de citação de cada aspecto analisado, devemos nos atentar que os mesmos se aproximam dos alertas sobre fatores relevantes de influência na prática docente e na aprendizagem discente segundo os autores referência na área.

Todos os fatores de interferência mapeados podem ser decisivos no sucesso ou no fracasso na proposta educativa do professor. A escolha de um projeto eficaz para suas intenções educacionais pode ter resultados imprevisíveis de acordo com: contexto de ensino, perfil da turma e cada aluno no contexto de ensino e aprendizagem, características do projeto e de sua comunidade de desenvolvimento, questões técnicas e emocionais do professor, entre outros, e podem ser determinantes para que o projeto FLOSS escolhido pelo professor não seja eficaz o suficiente para que o mesmo perceba o desenvolvimento das competências que ele objetiva desenvolver. Compreendendo as relações que cada um dos aspectos tem para o processo de ensino e aprendizagem em ES usando Projetos FLOSS, podemos definir que *cada aspecto mapeado é um eixo temático retirado das unidades de registros mapeadas*.

Todos os eixos temáticos têm as mesmas características temáticas levantadas em (CHAVEZ et al., 2011; KON et al., 2011; NASCIMENTO, 2017; STAMELOS, 2011; SOWE; STAMELOS, 2007), ou seja, *são aspectos que podem exercer influência nos processos de ensino e aprendizagem na Educação em Engenharia de Software mediada pelo uso de Projetos FLOSS*.

Dessa forma, é possível definir a Categoria de análise que resume e define toda a perspectiva temática e teórica abordada até aqui, a Categoria *ASPECTO DE INFLUÊNCIA*. Essa representação de temas definidos em uma mesma categoria é demonstrada na Tabela 4.31.

Tabela 4.31 Resumo da Categorização da Exploração dos Trabalhos sobre FLOSS e EES

Resumo da Categorização na Análise de Conteúdo dos Estudos sobre uso de FLOSS em EES			
Unidades Contextos e Unidades de Registros		**Eixos Temáticos**	**Categoria de Análise**
89 - Tabela C1(Apêndice C)	427 Unidades de Registros	A1- Tamanho e complexidade do projeto.	**Aspecto de Interferência em EES usando FLOSS**
93 - Tabela C2(Apêndice C)		A2- Interesse, motivação e atração.	
106 - Tabela C3(Apêndice C)		A3- Adaptação ao currículo ou curso.	
115 - Tabela C4(Apêndice C)		A4- Nível de formação e experiência do estudante.	
135 - Tabela C5(Apêndice C)		A5- Tempo, Calendário ou Cronograma.	
1088 - Tabela C6(Apêndice C)		A6- Uso ou conhecimento de ferramentas e ambientes de desenvolvimento.	
1097 - Tabela C7(Apêndice C)		A7- Suporte, acompanhamento e avaliação do estudante.	
1192 - Tabela C8(Apêndice C)		A8- Capacidade e Habilidade de Programação.	
1193 - Tabela C9(Apêndice C)		A9- Suporte ou viabilidade acadêmica	
4503 - Tabela C10(Apêndice C)		A10- Percepção ou Autopercepção das Habilidade de Computação.	
4663 - Tabela C11(Apêndice C)		A11- Dinâmica, rotina e cultura de Projetos FLOSSA.	
4811 - Tabela C12(Apêndice C)		12- Relação com a comunidade, Receptividade do Projeto.	
4815 - Tabela C13(Apêndice C)		A13- Restrições, Viabilidade e Adequação do Projeto.	
4966 - Tabela C14(Apêndice C)		A14- Estado atual do projeto.	
5147 - Tabela C15(Apêndice C)		A15- Assunto ou Domínio do Projeto.	
5328 - Tabela C16(Apêndice C)		A16- Estratégia para atividades e participações dos estudantes.	
5329 - Tabela C17(Apêndice C)		A17- Curva de Aprendizagem.	
5335 - Tabela C18(Apêndice C)		A18- Questões sociais, pessoais e legais.	
5343 - Tabela C19(Apêndice C)		A19- Desgaste, Entusiasmo e Experiência do Docente	
5353 - Tabela C20(Apêndice C)			
5357 - Tabela C21(Apêndice C)			
5546 - Tabela C22(Apêndice C)			
5676 - Tabela C23(Apêndice C)			
17796 - Tabela C24(Apêndice C)			
17800 - Tabela C25(Apêndice C)			
17805 - Tabela C26(Apêndice C)			
17830 - Tabela C27(Apêndice C)			
17882 - Tabela C28(Apêndice C)			
18359 - Tabela C29(Apêndice C)			
18433 - Tabela C30(Apêndice C)			

Fonte: Elaboração Própria.

Assim, conhecemos os aspectos de interferência nos processos de ensino e aprendizagem em Engenharia de Software mediada por Projetos FLOSS mais citados segundo estudos na área. Aqui encerra-se a exploração do material e a categorização nos três *corpus* de análise. O próximo passo é desenvolver as atividades de tratamento dos resultados e a interpretação. Essa fase se estrutura no movimento dialógico das categorias de análise, nas inferências e interpretação dos resultados. É através desse diálogo teórico e interpretação que o `Modelo` para avaliar a eficácia do uso de Projetos FLOSS no ensino de Engenharia de Software poderá ser alicerçado.

4.4 UM MODELO PARA AVALIAR A EFICÁCIA DO USO DE PROJETOS FLOSS NO ENSINO DE ES

Aqui nos debruçamos sobre a última etapa da Análise de Conteúdo nos três *corpus* da pesquisa, o movimento dialógico entre as Categorias de análise e as interpretações que permitirão definir quais os critérios para a avaliação da eficácia de um projeto FLOSS como ferramenta de aprendizagem em Engenharia de Software.

Sobre os preparativos para a terceira fase da Análise de Conteúdo (BARDIN, 2011, p.131) diz que "Operações estatísticas simples (percentagens), ou mais complexas (análise fatorial), permitem estabelecer quadros de resultados, diagramas, figuras e modelos, os quais condensam e põem em relevo as informações fornecidas pela análise.".

Foi através das análises qualitativas e quantitativas no corpus que conseguimos definir as categorias. Estas serão a base para estabelecermos as relações causais e conexões teóricas que permitirá modelar os critérios de avaliação da eficácia proposta como objetivo de pesquisa. Segundo (BARDIN, 2011, p.44), "o interesse não está na descrição dos conteúdos, mas sim no que estes nos poderão ensinar após serem tratados". Para aprendermos o que todos os mapeamentos desenvolvidos até aqui têm a nos ensinar, é necessário observar o diálogo dos resultados com as perspectivas teóricas que embasarão nossas descobertas.

4.4.1 Movimento dialógico das categorias de análise

Competência. O primeiro diálogo será sobre a Competência. É necessário validar a Competência na perspectiva de ensino da Engenharia de Software, na perspectiva de ensino sociotécnico e na perspectiva de necessidade de formação em Computação. Esses foram os eixos temáticos relacionados a Competências de acordo com a análise desenvolvida.

Sobre o termo "Competência" (ZORZO et al., 2017, p.7) diz que "[...] uma competência pode expressar o conhecimento, as habilidades ou as atitudes esperadas do egresso do curso, sob a perspectiva de objetivos de aprendizagem (o que o aluno será capaz de)". Essa perspectiva é baseada na taxonomia de Bloom. Já o SWECOM versa que "Conhecimento, nesse modelo de competência, é diferente de habilidade: conhecimento é o que se sabe, enquanto habilidade é o que se pode fazer" (ARDIS et al., 2014a, p.1). Podemos perceber que apesar de apresentar perspectivas diferentes, ambas os autores direcionam a perspectiva da Competência como um movimento ascendente de habilidades. Onde os

níveis de exigência da prática profissional dependem de Conhecimentos e sua aplicação prática.

Quando observamos o artigo quinto das DCNs "Os cursos de bacharelado e licenciatura da área de Computação devem formar egressos que revelem pelo menos as competências e habilidades comuns para [...]" (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22, p.4), podemos compreender que o objetivo de formação é lançar profissionais que revelem os resultados de sua formação. Nessa perspectiva podemos perceber o alinhamento temático dos conhecimentos como base para o desenvolvimento das habilidades e atitudes esperadas pelos egressos dos cursos. Na perspectiva de Bloom (TAVARES; CARVALHO, 2010; FERRAZ; BELHOT et al., 2010), podemos observar que tanto os conhecimentos, as habilidades e as atitudes podem ser concebidas como Competências, onde estas se revelam em níveis cognitivos ou afetivos.

Dessa forma, no contexto de formação superior em Computação e na formação técnica ou sociotécnica em Engenharia de Software, as competências podem ser nada mais que objetivos de aprendizagem que devem ser almejadas pelas práticas educativas.

Objetivos de aprendizagem determinam as competências a serem desenvolvidas através das práticas educativas. Sobre isso (NASCIMENTO, 2017, p.33) versa que os Objetivos de Aprendizagem "[...]devem descrever o conhecimento ou as habilidades que os estudantes devem apresentar como resultado da atividade".

Desenvolver ensino baseado em competências diversas em um mundo globalizado e em constante evolução tecnológica demanda esforço docente de técnicas e métodos que aproximem o aluno da realidade do mundo do trabalho. É nessa perspectiva que surgem as ferramentas de ensino e aprendizagem. Estas tem o poder de levar o estudante a uma realidade de prática educacional a um contexto próximo ao que o professor desejaria. Segundo (LOBO; MAIA, 2015), "No processo de ensino-aprendizagem (EA), é importante destacar a importância do aprender fazendo, do aprender a aprender, do interesse, da experiência e da participação como base para a vida em uma democracia".

Analisando essa afirmação, temos o conceito de Objeto de Aprendizagem segundo (AGUIAR; FLÔRES, 2014) "O Objeto de Aprendizagem (OA) apresenta-se como uma vantajosa ferramenta de aprendizagem e instrução, a qual pode ser utilizada para o ensino de diversos conteúdos e revisão de conceitos".

Quando o professor usa um Projeto FLOSS como uma ferramenta de apoio para ensinar Engenharia de Software, ele assume características comuns a um Objeto de Aprendizagem. É importante salientar que um Projeto FLOSS usado para ensinar ES deixa de ser apenas um recurso tecnológico e técnico e passa a ser um instrumento pedagógico na perspectiva docente. Dessa forma, o professor deve se atentar aos limites do que é técnico e do que é pedagógico na forma de avaliar um Projeto usado para ensinar. Professores pouco experientes no uso de FLOSS podem encontrar dificuldade em estabelecer esse diálogo entre aspecto técnico e pedagógico.

Assim, conhecer alguns fatores de avaliação de Objetos de Aprendizagens no geral pode gerar um aporte intelectual inicial sobre como avaliar um projeto a ser usado não apenas antes de desenvolver uma atividade, mas também, após a atividade ser concluída. Sobre essa perspectiva (ALMEIDA; CHAVES; JR, 2015, p.5):

> Alguns questionamentos são levantados acerca do que torna um recurso adequado ou não ao processo ensino-aprendizagem. Ele é projetado de modo a garantir o sucesso de todos os objetivos educacionais a que se propõe ou simplesmente um produto que desperta interesse no aluno e que pode gerar uma maior motivação em sala de aula?

Sobre os aspectos que devem ser observados em Objetos de Aprendizagem, (ALMEIDA; CHAVES; JR, 2015) versa sobre a qualidade do conteúdo como aprofundamento na temática, se é contextualizado e se facilita os conteúdos a serem assimilados. Almeida ainda comenta a importância de "investigar se a utilização daquele recurso é ou não eficaz no processo de ensino e aprendizagem." (ALMEIDA; CHAVES; JR, 2015, p.5).

Assim, temos uma observação de convergência direta entre os desafios no uso de Projetos FLOSS na EES e os desafios de qualquer Objeto de Aprendizagem ou recurso educacional usado no processo de ensino e aprendizagem.

Considerando que, o processo de desenvolvimento de competências dá-se nos objetivos de aprendizagem (ZORZO et al., 2017), pode-se dizer que um recurso educacional usado para atingir tais objetivos interfere no alcance de competências planejadas pelo docente (AGUIAR; FLÔRES, 2014; MUSSOI; FLORES; BEHAR, 2010). Sendo o projeto FLOSS um recurso educacional no contexto da ES, este passa a ter influência nas competências de Engenharia de Software planejadas pelo professor.

Portanto, chega-se a relação causal das duas categorias de análise pesquisadas como mostra a Figura 4.18.

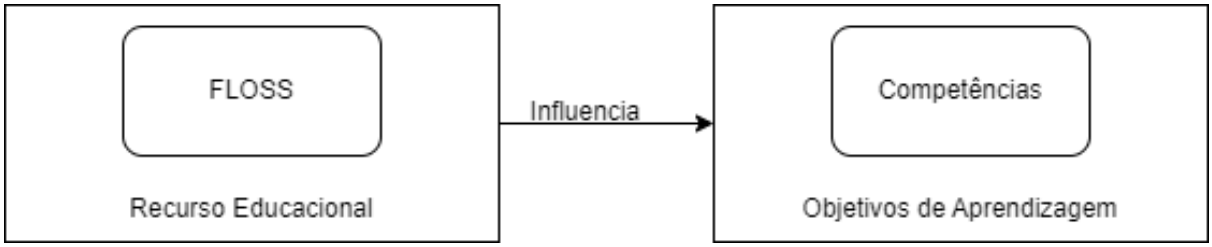

Figura 4.18 Relação entre Floss e Competências da ES
Fonte: Elaboração Própria

4.4.2 Inferência e Interpretação da Análise

Aqui partimos para a inferência e interpretação da nossa análise que permitirá formar a base para a concepção do modelo conceitual. Percebe-se, através da própria análise de conteúdo, que a hipótese de competência como critério de objetivo educacional é moderna e pertinente ao que versam as referências científicas e profissionais da área da formação e educação em Computação. Percebe-se também que essas competências são elementos independentes. Qualquer competência se molda por estímulos de conhecimentos e habilidades diversos praticados no contexto principal, a formação em Computação.

Vimos também que, dentro da própria área da Computação, a relevância das suas áreas ou disciplinas varia de acordo com a profundidade e abrangência na formação em

Computação. A Engenharia de Software mostra-se com relevância considerável para a formação em qualquer área da Computação. A formação em Computação se dá pelo alcance de objetivos de formação que gerem profissionais preparados para o mundo do trabalho e para a indústria. Com a formação em Engenharia de Software não é diferente: como área da Computação, ela herda desta os requisitos fundamentais para uma boa formação profissional.

Sobre essa perspectiva, o Artigo quarto das Diretrizes Curriculares Nacionais abordam as expectativas de formação em Computação. Uma das expectativas indicadas pelas DCNs é que os profissionais tenham conhecimento de questões legais, sociais, profissionais e éticas (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22). Nesse sentido, a formação sociotécnica assume papel essencial (TOPI et al., 2010). As DCNs também abordam a importância do aluno compreender o impacto da Computação na Sociedade, de ter visão crítica e agindo de forma empreendedora (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22). Ou seja, o processo de avaliação de objetivos educacionais deve incorporar macro-objetivos comuns a qualquer profissional. A incorporação de tais objetivos de forma interdisciplinar e transdisciplinar pode ser uma estratégia pertinente.

Sobre avaliação profissional, (SABIN et al., 2017) versa que os ambientes de aprendizagem apoiadores de aquisição por competências devem incluir a avaliação baseada em competências, a inclusão da prática profissional, o acompanhamento do desenvolvimento do aluno e com processo de avaliação que encoraje o aluno a aplicar boas práticas técnicas. Tendo nas competências a representação interligada das dimensões de conhecimento, habilidades e disposições, chegamos nos aspectos que influenciam a aprendizagem em ES sobre essas três dimensões. (VILLARRUBIA; KIM, 2015) Versa que a complexidade do projeto pode afetar a atração do aluno pelas atividades, ou seja, inibindo sua atitude na dinâmica educativa. A estratégia de organizar tarefas que diminua a rejeição do estudante a participação é mencionada por (DINIZ et al., 2017). Já (ELLIS et al., 2013) versa sobre a curva de aprendizado que o aluno possa enfrentar ao se deparar com novas ferramentas e ambientes.

Podemos perceber que vários aspectos não interferem apenas no alcance de uma competência, mas interferem também em outros fatores condicionantes de forma relacionada. Podemos perceber também que as competências desenvolvidas e os aspectos de interferência nelas ao usar um FLOSS para ensinar ES são critérios chave para avaliar um projeto escolhido pelo professor para atingir um objetivo de aprendizagem. Observamos que apesar de o professor ter o olhar para seus objetivos de ensino, o objetivo de aprendizagem perpassa pela realização do aluno na prática e evolução profissional e cidadã (SABIN et al., 2017).

Sendo assim, um modelo que estruture as competências de ES e os critérios que podem auxiliar ou inibir seu alcance pode ser estipulado.

4.4.2.1 Estrutura de Competências segundo SWECOM O SWECOM é referência na modelagem de competências para Engenheiros de Software no mundo. O guia estrutura seus elementos em Habilidades técnicas. Esta se forma através da relação dos requisitos de conhecimento na área, do domínio de áreas relacionadas, das habilidades cognitivas, e habilidades comportamentais como mostra a figura 4.19.

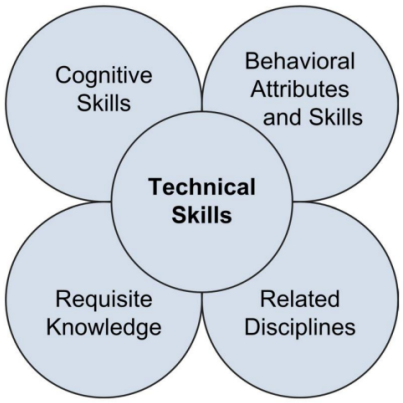

Figura 4.19 Elementos do SWECOM
Fonte: (ARDIS et al., 2014a, p.5)

As habilidades cognitivas são a base de todas as habilidades (ARDIS et al., 2014a). São o conhecimento adquirido e aplicado em atividades de habilidades técnicas. Resumem-se em formas de raciocínio e abstração, habilidades analíticas, resolução de problemas e inovação.

Nas habilidades comportamentais é indicada a capacidade de aplicar o conhecimento e as demais habilidades. São necessárias a qualquer área de formação. São listadas como Aptidão, iniciativa, Entusiasmo, Ética, Disposição, confiabilidade, Liderança.(ARDIS et al., 2014a)

O conhecimento necessário é a base intelectual para a profissão de Engenharia de software (ARDIS et al., 2014a, p.5). O mesmo usa como base o corpo de conhecimento do SEWBOK. As habilidades técnicas do SWECOM são todas as áreas de conhecimento que devem ser aplicados pelo engenheiro de software. Estas são o foco do Modelo de Competências em questão. O SWECOM agrupa habilidades técnicas em habilidades de ciclo de vida e habilidades transversais. As habilidades de ciclo de vida são aquelas vistas como essenciais para o perfil profissional de um engenheiro de software. Já as habilidades transversais são aquelas que aplicam nas habilidades de ciclo de vida.

As áreas de habilidades do ciclo de vida do SWECOM são próximas das áreas de conhecimento técnico mapeadas na Análise de Conteúdo. Como o SWECOM versa que as habilidades técnicas são uma estrutura de diversas habilidades combinadas, podemos perceber as relações de concomitância de competências sociotécnicas mapeadas.

Sendo assim, podemos interpretar que competência em Engenharia de Software é a soma das competências técnicas (Conhecimentos e práticas de ES) e Competências Sociotécnicas (Comportamentais e cognitivos ou disposições). Ao aproximar os eixos temáticos mapeados com os eixos de competência indicados pelo SWECOM, a representação lógica das Competências no modelo pode ser como na Figura 4.20.

Essa representação visa simbolizar a temática de competência sobre uma visão de organizações profissionais, onde, por meio da análise de conteúdo abordada mostra que pode ser pertinente alinhar objetivos educacionais a objetivos profissionais.

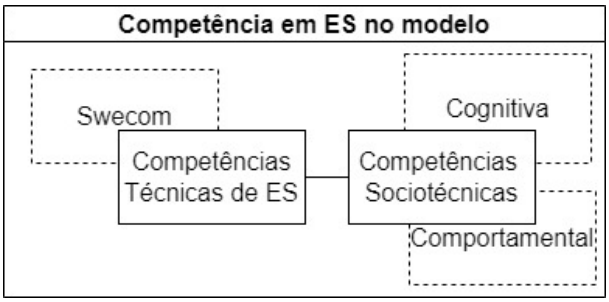

Figura 4.20 Simbologia de Competências do Modelo Conceitual proposto (Baseado no Modelo SWECOM)

Fonte: Elaboração Própria

4.4.2.2 Objetivos de Aprendizagem e Taxonomia de Bloom. A abordagem de aprendizagem por competências visa o desenvolvimento de uma forma de ensinar não centrada nas ações didáticas do professor, mas em atividades que reflitam realmente o desenvolvimento do aluno. Sobre isso, (ALMEIDA; CHAVES; JR, 2015) diz:

> A avaliação educacional tem sido, historicamente, a via pela qual a sociedade se vale para conhecer tendências, responsabilidades, resultados e coerências entre teorias e práticas na área. A avaliação pode gerar transformações, justificativas ou descrédito sobre o que se avalia, dependendo dos múltiplos fatores que a influenciam. Avalia-se para agir, tomar decisões, sustentar argumentos. E, especialmente no caso educacional, para guiar indicadores da qualidade.

Ou seja, a avaliação educacional é o processo de retorno sobre o alcance dos objetivos de instruções do professor. Os instrumentos de apoio didático e pedagógicos são poderosos no apoio ao planejamento docente. Segundo (FERRAZ; BELHOT et al., 2010), a estruturação do processo educacional se dá por meio da escolha do conteúdo, dos procedimentos de atividades e recursos disponíveis, além da metodologia e instrumentos de avaliação a ser adotada. Sendo assim, a escolha e avaliação dos instrumentos de apoio didático passam pela estruturação do processo educacional.

A avaliação de recursos educacionais se direciona aos objetivos educacionais estruturados e que também possam ser avaliados. Um contribuinte relevante sobre instrumentos de avaliação da aprendizagem foi (BLOOM, 1956) ao criar uma Taxonomia que auxilie a avaliação de objetivos instrucionais. Ela pode estimular os educadores no acompanhamento da aprendizagem do aluno e usar estratégias diferenciadas para a avaliar o desempenho deste (FERRAZ; BELHOT et al., 2010).

A Taxonomia de Bloom auxilia ao professor no acompanhamento das competências a serem desenvolvidas em níveis cognitivos de aprendizagem. O uso de recursos de apoio a aprendizagem podem auxiliar o professor no alcance dessas competências. Portanto, a Taxonomia de Bloom será a referência de instrumentação da avaliação da aprendizagem e alcance das competências no Modelo conceitual proposto por ser uma das referências de avaliação dos Referenciais de Formação para cursos de Graduação em Computação(SBC)

e dos Currículos da série Curricula Computing 2005 da ACM. A Taxonomia de Bloom usa seis níveis cognitivos para demonstrar o meio de alcance dos objetivos de aprendizagem.

Segundo (BLOOM, 1956; TAVARES; CARVALHO, 2010; FERRAZ; BELHOT et al., 2010), os níveis do processo cognitivo são em geral: Lembrar, relacionado a reprodução de conteúdos; Entender, representando a necessidade do aluno reter a informação e reproduzi-la; Aplicar, relacionado ao uso de procedimentos em determinada situação; Analisar, relacionado a manipular a informação em partes e entender suas relações através de diferenciação, organização atribuição e conclusão; Avaliar é fazer julgamentos de aspectos qualitativos e quantitativos, de eficácia e eficiência; e Criar é desenvolver novas ideias e métodos, relacionando elementos para criar novas soluções, estruturas ou modelos.

A partir do momento que a aprendizagem em ES é mediada por projetos FLOSS, este está envolvido como elemento de auxílio aos processos cognitivos que o professor deve planejar para serem desenvolvidos pelas atividades aplicadas. Sendo assim, os Projetos FLOSS podem ter influência direta nos objetivos de aprendizagem. Essa relação de influência é mostrada na Figura 4.21.

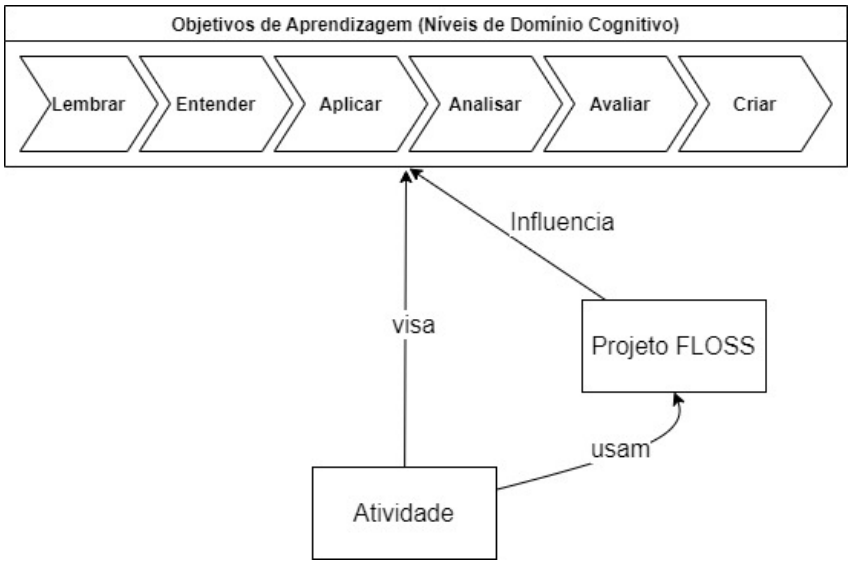

Figura 4.21 Influência de Projetos FLOSS nos Objetivos de Aprendizagem baseado na Taxonomia de Bloom

Fonte: Elaboração Própria

Assim, percebe-se que aspectos de influência mais citados no uso de Projetos FLOSS para o ensino de Engenharia de Software, conforme mapeamento, são critérios que podem estimular ou limitar os objetivos de aprendizagem planejados.

Sabendo que os Objetivos de Aprendizagem visam o desenvolvimento das competências planejadas pelo professor e que instrumentos de auxílio a aprendizagem exercem influência no alcance desses objetivos, podemos relacionar conceitualmente as duas categorias mapeadas para a concepção do Modelo de avaliação.

4.4.2.3 Estrutura do Modelo para Avaliar a Eficácia do uso de Projetos FLOSS no ensino de ES. A formação superior em Computação passa por aquisição de várias competências que gerem profissionais capazes de suprir as necessidades e anseios da sociedade e da indústria. A Engenharia de Software como ciência base da Computação visa garantir profissionais de Software capacitados para projetar, desenvolver e gerenciar softwares que atendam as demandas das organizações e da sociedade.

Essa concepção sobre a formação em ES e em Computação como um todo é perseguida com seriedade pelos principais guias e referenciais curriculares nacionais e internacionais como mostrou o mapeamento realizado até aqui. Várias áreas de conhecimento, metodologia, diretrizes e aspectos institucionais e técnicos de educação foram indicados para auxiliar os profissionais de educação no planejamento de currículos convergente com as necessidades de formação concebidas.

Foi sobre toda a estrutura de indicação de conhecimentos e competências que o *corpus* do mapeamento de competências foi efetuado. Por meio do mapeamento realizado através da Análise de Conteúdo de Bardin, toda a temática de competências técnicas e sociotécnicas foi mapeada e estruturada para responder às questões de pesquisa deste trabalho.

Através do mapeamento dos Referenciais de Formação nacionais, as competências foram mapeadas segundo a metodologia própria de construção destes. A categoria competências foi primeiramente organizada em Competências Alvo (Competências Gerais e Específicas das DCNs) estas são o nível mais alto de necessidades de formação que qualquer currículo e qualquer profissional da educação em Computação deve objetivar. Estas competências Alvo são estruturadas em competências derivadas, que por sua vez, define as necessidades de formação de cada eixo específico dos cursos. Essas competências derivadas são alcançadas pelos conteúdos (conhecimentos). Os conhecimentos adquiridos e articulados também são considerados competências (SHACKELFORD et al., 2005; EDUCAÇÃO-MEC, 2016. Seção 1. p. 22; NUNES; YARMAGUTI; NUNES, 2016; CAMARGO; FABRI, 2006).

O conhecimento específico da ES foi relacionado a todas as Competências Gerais (Competências Alvo) e o Conhecimento Sociotécnico foi relacionado a praticamente todas as Competências Específicas a Graduados em ES (Competências Alvo). O resultado dessas relações é a indicação da relevância das competências técnicas de ES para a formação em Computação e das competências sociotécnicas para a formação em ES. Essa relação intrínseca entre as competências sociotécnicas e as competências é referenciado na figura 4.22, no qual mostra áreas de conhecimento como Processo (PRO), Requisitos(RES), Qualidade(QUA), e Design (DES) de software tendo dependencia da área de conhecimentos profissionais (PRF).

Portanto, podemos definir as *Competências em Engenharia de Software* como a união das competências técnicas específicos da área de ES com as competências sociotécnicas ou profissionais. Estas, por sua vez, têm relevância considerável na formação em Computação, ou seja, no alcance das competências-alvo. A relevância da ES e do desenvolvimento sociotécnico na formação em Computação foi interpretada após o mapeamento dos Guias curriculares internacionais. A representação visual das competências é exibida na Figura 4.23.

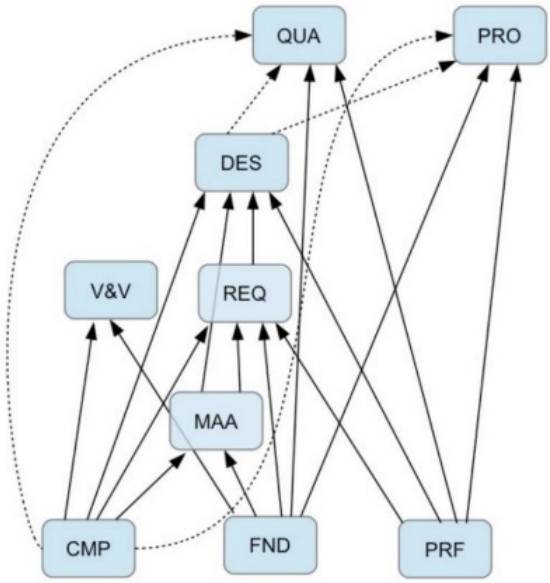

Figura 4.22 Dependências de Áreas de conhecimento em Engenharia de Software
Fonte: (ARDIS et al., 2014b, p.53)

As competências em ES são alcançadas por meio de Objetivos de Aprendizagem, sendo que estas são desenvolvidas pelas intervenções didáticas do professor (atividades) e são auxiliadas por uso de Projetos FLOSS. O projeto FLOSS, por sua vez, pode interferir nos objetivos de aprendizagem de forma positiva ou limitante. Os aspectos que mais influenciam nos objetivos de aprendizagem em ES auxiliados por uso de FLOSS foram mapeados e ordenados. A representação dessa interação entre atividade, FLOSS, Objetivos de Aprendizagem e Competências é exibida na Figura 4.24.

Avaliar Projetos FLOSS para ensinar Engenharia de Software tende a ser uma tarefa custosa. Nesse tema (ELLIS; PURCELL; HISLOP, 2012) é uma das referências. Ellis aborda um modelo para avaliar adequação de projetos para uso nas turmas. O autor cita diversos aspectos que foram mapeados na análise de conteúdo dessa pesquisa, o mesmo separou grupos de critérios em Viabilidade, Acessibilidade e Adequação além de outras considerações.

Ellis define critérios bem definidos e de fácil identificação para estruturar o modelo. Porém, tais critérios podem influenciar outros critérios que podem ser determinantes para a eficácia do projeto escolhido. Questões pedagógicas além das técnicas são de análise mais complexas, pois tem influência no emocional do aluno e do professor. Ao usar critérios mais técnicos de avaliação, o professor pode encontrar projetos que se adéque a uma proposta didática num determinado contexto e se sentir seguro para usá-lo em outras atividades, mas pode se deparar com questões pedagógicas que tornem a atratividade do projeto diferente em outras turmas, como a cultura, experiência do aluno com a área de formação e com a própria dinâmica de Projetos FLOSS. (STAMELOS, 2011).

Sendo assim, o modelo aqui proposto organiza os aspectos tanto no aspecto técnico de seleção e introdução do aluno como nas questões pedagógicas que podem interferir nos

Figura 4.23 Representação visual de competências no modelo
Fonte: Elaboração Própria

Objetivos de Aprendizagem determinados pelo professor. Cada um dos aspectos podem exercer influência positiva ou negativa nos objetivos de aprendizagem, essa influência pode ser concomitante, por exemplo, projetos muito complexos podem afetar a confiança de alunos inexperientes sobre sua capacidade de Computação e inibir sua livre iniciativa na atividade e na interação com a comunidade. O mesmo projeto pode ser positivo em alunos avançados e experientes na dinâmica de desenvolvimento em FLOSS.

Então, analisar eficácia de projetos para ensinar ES tende a ser sobre análise de dupla perspectiva, os aspectos técnicos que o docente idealiza para desenvolver as atividades e os aspectos pedagógicos, ou seja, as respostas que os alunos fornecem ao projeto e o contexto de aprendizagem, mostrando que não é importante apenas avaliar se o projeto é eficaz, mas a eficácia da forma que é usado para ensinar(NASCIMENTO, 2017; STAMELOS, 2011; CHAVEZ et al., 2011).

A representação visual dos aspectos, das competências contextualizadas segundo SWECOM e os Objetivos de Aprendizagem (organizados em níveis segundo taxonomia de Bloom) é exibida na Figura 4.25.

Podemos observar um mapa conceitual que organiza os temas e suas interações, resultantes da análise de conteúdo e sua aplicação no contexto de avaliação de material didático, objeto de aprendizagem ou recurso educacional. É importante abordar os aspectos pedagógicos que norteiam as práticas educativas do professor/instrutor. Todo o processo de avaliação deve passar pela análise dos resultados planejados previamente e acompanhar o desenvolvimento do que foi proposto. Ou seja, *o processo de avaliação do projeto escolhido pelo professor passa pela gestão da aprendizagem.*

Ao adaptarmos os princípios gerais de gestão à gestão de aprendizagem, percebemos que esta também passa por planejamento dos objetivos e práticas educativas, pela execução do que foi planejado e pelo controle dos resultados (BALBINOT; FILHO, 2005). O planejamento serve para decidir onde se quer chegar e quais objetivos devem ser atingidos, a execução define como os objetivos vão ser atingidos e as tarefas realizadas para tal, e o controle é o instrumento que serve para mensurar ou avaliar a execução do que

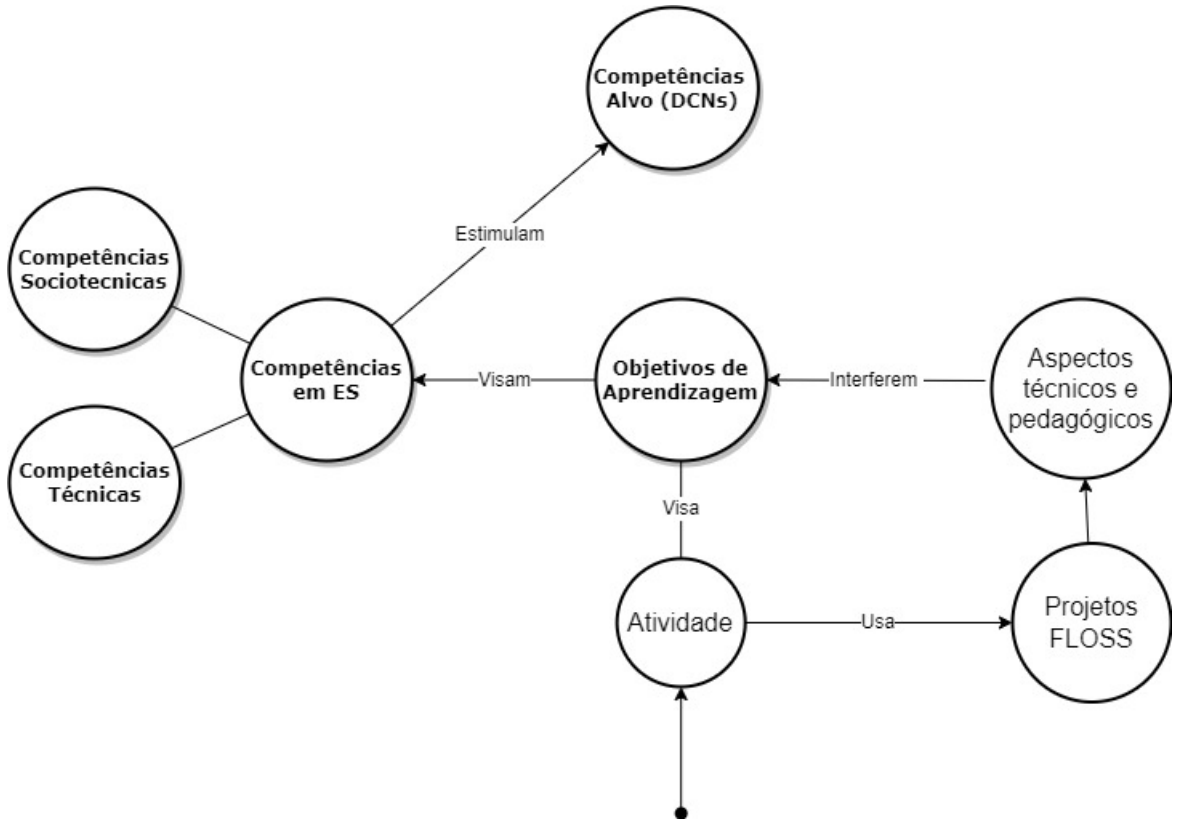

Figura 4.24 Representação do modelo simplificado de acordo com os resultados da Análise de Conteúdo (Interpretação).

Fonte: Elaboração Própria

foi planejado e/ou os objetivos alcançados(BALBINOT; FILHO, 2005).

Sendo assim, *o modelo proposto pode ser um instrumento valioso para o professor controlar o planejamento, a execução e a avaliação das atividades usando um Projeto FLOSS*. Os aspectos identificados na análise de conteúdo mostram uma tendência de influência nas diferentes fases da gestão de aprendizagem em Engenharia de Software exercida pelo professor ou instrutor.

Alguns aspectos do projeto já podem ser mensurados antes mesmo da execução das atividades, ou seja, na fase de planejamento. Alguns podem ser identificados na execução das atividades e outros podem ser identificados na análise dos objetivos (competências desenvolvidas).

É importante que o professor conheça seus alunos, sua cultura, seu nível de formação, suas habilidades de programação, em Computação e em ES, além da experiência na dinâmica e ferramentas de desenvolvimento em Projetos FLOSS (SILVA, 2003; STAMELOS, 2011; CHAVEZ et al., 2011).

É importante o professor analisar as restrições do projeto, a viabilidade e adequação deste na prática das atividades. O estado do projeto e o assunto ou domínio já podem ser mensurados no planejamento das atividades. Essas questões já podem ser pensadas para a Estratégia de atividades e participação dos estudantes.

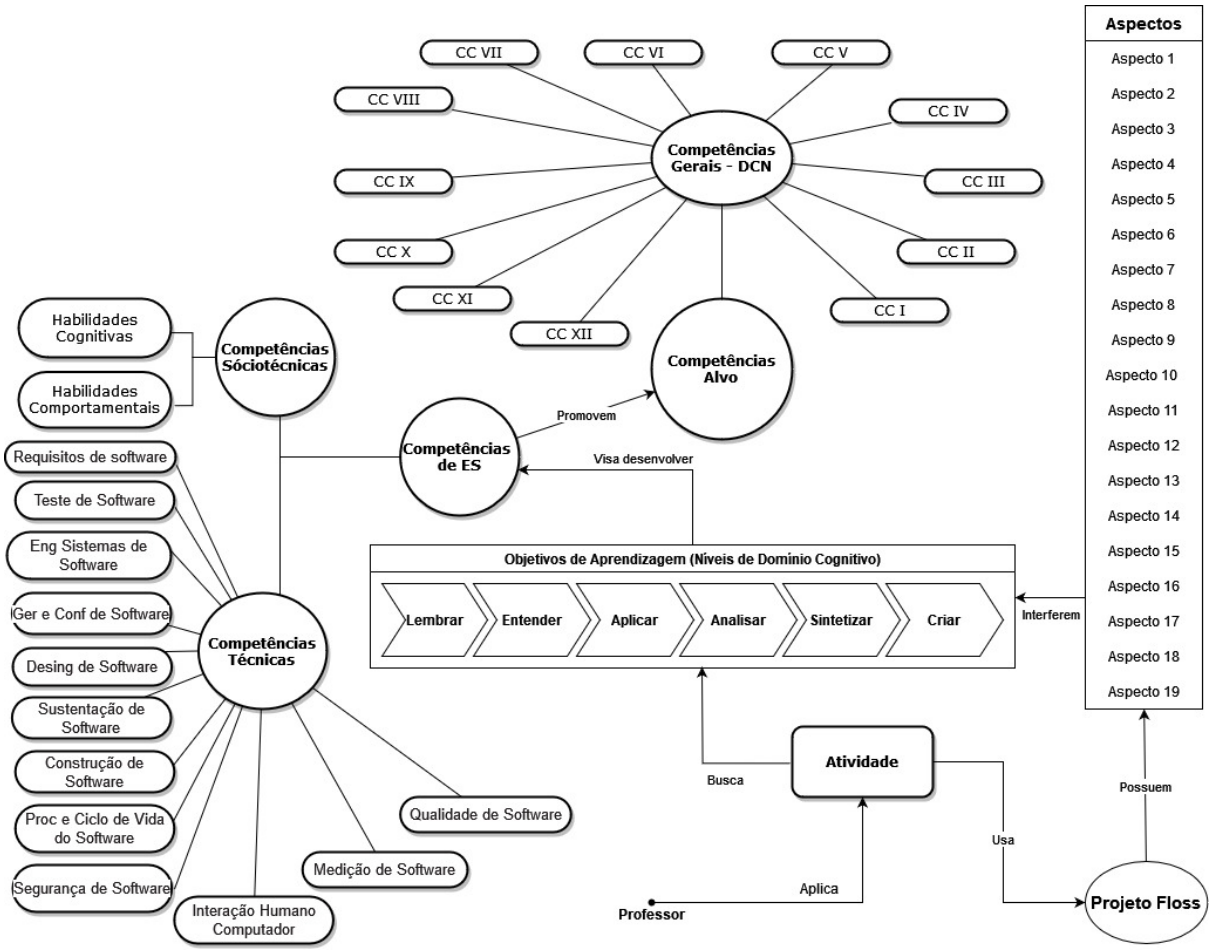

Figura 4.25 Representação do modelo após análise e contextualização com Swecom e Taxonomia de Bloom

Fonte: Elaboração Própria

Já alguns aspectos podem ser identificados na execução das atividades. O uso de ferramentas e ambientes de desenvolvimento, percepção e autopercepção das habilidades de Computação, a relação com a comunidade e recepção do projeto a novos alunos, a curva de aprendizagem e desgaste do professor são fatores que podem interferir nos resultados de aprendizagem que o professor planejou.

A Tabela 4.32 *propõe uma classificação de aspectos a serem analisados nas fases de planejamento, de execução e de controle das atividades usando o projeto.*

Essa proposta de uso do modelo não é uma regra, já que o professor experiente pode perceber vários aspectos já no planejamento das atividades, porém, a análise de conteúdo usada no mapeamento e a abordagem de (STAMELOS, 2011; ELLIS; PURCELL; HISLOP, 2012; NASCIMENTO, 2017) mostram que as relações entre atividades, competências e aspectos de uso de FLOSS tendem a ser coerentes. De acordo com o nível de conhecimento e experiência do docente e dos alunos em atividades de ES usando FLOSS, os aspectos podem ter maior ou menor influência positiva/negativa no objetivo

Tabela 4.32 Proposta de uso do Modelo Conceitual para avaliar a eficácia do uso de Projetos FLOSS no Ensino de ES

Proposta de uso do modelo na Gestão da Aprendizagem em ES mediada por uso de FLOSS		
Controle		
Planejamento	Execução	Avaliação
Atividade executada usando FLOSS	Atividade executada usando FLOSS	Atividade executada usando FLOSS
Competências de ES a serem desenvolvidas (Objetivos de aprendizagem)	Competências em desenvolvimento (Objetivos de aprendizagem)	Objetivos alcançados e competências desenvolvidas de acordo com os níveis Cognitivos (Objetivos de aprendizagem)
Competências alvo a serem estimuladas (Necessidades de Formação)	Competências gerais em estímulo (Necessidades de Formação)	Competências Gerais percebidas nas atividades (Necessidades de Formação)
Aspectos críticos no panejamento	Aspectos críticos na execução	Aspectos críticos na avaliação
A16. Estratégia para atividades e participações dos estudantes	A1- Tamanho e complexidade do projeto	Todos os Aspectos
A18- Questões sociais, pessoais e legais	A2- Interesse, motivação e atração	
A1- Tamanho e complexidade do projeto	A5- Tempo, Calendário ou Cronograma	
A3- Adaptação ao currículo ou curso	A7- Suporte, acompanhamento e avaliação do estudante	
A4- Nível de formação e experiência do estudante	A10- Percepção ou Autopercepção das Habilidade de Computação	
A13- Restrições, Viabilidade e Adequação do Projeto	A11- Dinâmica, rotina e cultura de Projetos FLOSS	
A14- Estado atual do projeto	A12- Relação com a comunidade, Receptividade do Projeto	
A15- Assunto ou Domínio do Projeto	A17- Curva de Aprendizagem	
	A19- Desgaste, Entusiasmo e Experiência do Docente	

Fonte: Elaboração Própria

de aprendizagem.

Utilizando as contribuições de (ELLIS; PURCELL; HISLOP, 2012; STAMELOS, 2011; NASCIMENTO, 2017; LESSA; CHAVEZ, 2020; GALHARDI; AZEVEDO, 2013), o *Modelo Conceitual* pode ser usado para o desenvolvimento de uma ficha de avaliação sobre a eficácia do uso de Projeto FLOSS no Ensino de ES. Nessa ficha, o professor pode relatar cada atividade planejada, indicar os níveis de processo cognitivo a serem desenvolvidos em cada etapa da atividade, e associar a influência positiva, negativa e/ou nula de cada aspecto no processo cognitivo associado a aprendizagem. A proposta de ficha de avaliação do `Modelo Conceitual` é exibida na Tabela 4.33.

Como os níveis cognitivos são classificados, segundo (BLOOM, 1956), em complexidades ascendentes, podemos deduzir que aspectos que exercem influência positiva em níveis mais altos são relevantes para indicar a eficácia do Projeto escolhido na atividade planejada. O inverso também ocorre, caso um aspecto tenha influência negativa em atividades com processos de níveis cognitivos mais baixos, podemos subtender que existe uma relevância negativa desse aspecto no uso do projeto para ensinar.

O objetivo deste Modelo não é avaliar o projeto FLOSS, ou descarta-lo do portfólio de uso do professor, mas sim avaliar a eficácia de seu uso para Ensinar ES. O mesmo projeto pode ser altamente eficaz ou pode ser totalmente limitante na aprendizagem a depender da atividade planejada, o contexto de ensino, o perfil da turma ou dos alunos entre outras questões.

Portanto, a aprendizagem do próprio uso de FLOSS no ensino pode ser estimulada

Tabela 4.33 Proposta de ficha de avaliação da eficácia de uso do Projeto FLOSS em EES

Atividade Planejada:																		
PROJETO FLOSS ESCOLHIDO	**Objetivo de Aprendizagem:**																	
	Processo cognitivo usado na atividade para atingir objetivo																	
	Lembrar			**Entender**			**Aplicar**			**Analisar**			**Avaliar**			**Criar**		
Aspecto que influencia na aprendizagem em ES encontrado no uso de FLOSS	**P**	**NA**	**N**	**P**	**NA**	**N**	**P**	**NA**	**N**	**P**	**NA**	**N**	**P**	**NA**	**N**	**P**	**NA**	**N**
A1-Tamanho e complexidade do projeto:																		
A2-Interesse, motivação e atração:																		
A3-Adaptação ao currículo ou curso:																		
A4-Nível de formação e experiência do estudante:																		
A5-Tempo, Calendário ou Cronograma																		
A6-Uso ou conhecimento de ferramentas e ambientes de desenvolvimento																		
A7-Suporte, acompanhamento e avaliação do estudante																		
A8-Capacidade e Habilidade de Programação																		
A9-Suporte ou viabilidade acadêmica																		
A10-Percepção ou Autopercepção das Habilidade de Computação																		
A11-Dinâmica, rotina e cultura de Projetos FLOSS																		
A12-Relação com a comunidade e Receptividade do Projeto																		
A13-Restrições, Viabilidade e Adequação do Projeto																		
A14-Estado atual do projeto																		
A15-Assunto ou Domínio do Projeto																		
A16-Estratégia para atividades e participações dos estudantes																		
A17-Curva de Aprendizagem																		
A18-Questões Legais Sociais e Éticas																		
A19-Desgaste, Entusiasmo e Experiência do Docente																		
Considerações do professor sobre a eficácia de uso do projeto na atividade de ensino em ES:																		
Legenda: P = Influência Positiva; NA = Não associado; N= Influência negativa																		

Fonte: Elaboração Própria

ao professor que conhece os aspectos que podem ser identificados por meio de seu uso. Porém, o professor necessita adquirir a consciência de seu papel como gestor e mediador do ensino, analisando não apenas aspectos técnicos no uso de FLOSS, mas também, aspectos pedagógicos relevantes que interfiram, inclusive, na observação e percepção da aprendizagem e do uso do FLOSS escolhido como material didático.

É necessário mais estudos e pesquisas que investiguem os aspectos mapeados e suas relações com as competências em ES em um contexto real e específico. As evidências indicadas pela análise de conteúdo podem ser relevantes, mas nos processos educacionais, a própria aprendizagem dos processos e decisões tomadas são dinâmicas e podem mudar.

É reconhecido o poder de aprendizagem ativa que os Projetos FLOSS proporcionam como recurso didático, mas, por ser um ambiente de ensino e aprendizagem não estruturado e expansível, os Projetos FLOSS devem ser investigados e seus critérios de uso classificados. Essa classificação permite que, cada vez mais, o uso de projetos FLOSS assuma características implícitas de estrutura didática.

Mapear os critérios de uso do FLOSS para ensinar ES pode auxiliar a catalogação dos projetos e seus artefatos no contexto educacional, mostrando qual aspecto interfere em

cada intervenção, e com a catalogação dos critérios mapeados, um projeto FLOSS pode assumir características de Objetos de Aprendizagem rastreáveis e avaliáveis, contribuindo para o crescimento dessa abordagem em escala relevante no Ensino de ES mundialmente.

4.4.3 Exemplo de aplicação do modelo no ensino de ES.

Para ilustrar como o modelo proposto pode auxiliar os professores no processo de avaliação da eficácia do uso de um projeto FLOSS escolhido para ensinar Engenharia de Software, vamos usar um exemplo de aplicação do mesmo aproveitando dados e resultados de Estudo de Caso proveniente do estudo de (NASCIMENTO, 2017).

Objetivo. O estudo de caso teve como objetivo explorar e investigar consequências e objetivos da aprendizagem mediadas por uso de projetos OSS, conexões entre teoria e prática, a percepção da experiência de mundo real e tratamento de lacunas de formação identificadas na indústria (NASCIMENTO, 2017).

Contexto. Aplicação de um projeto OSS no ensino de Teste de Software em uma disciplina de um Curso de Graduação em Ciência da Computação.

Cenário. O estudo foi realizado em atividades na disciplina Desenvolvimento de Software III, do Curso de Bacharelado em Ciência da Computação na Universidade Federal de Sergipe (UFS) no segundo semestre de 2015. A exposição do conteúdo foi aplicada pelo professor de forma tradicional (expositiva e com exercícios). As atividades práticas no projeto OSS foram ofertadas de forma **opcional**, sendo que as notas das provas do estudante poderiam ser substituídas pelas notas obtidas nas atividades com o projeto OSS. O estudo foi dividido em duas etapas, sendo a primeira com estudantes **em equipes** para implementar **individualmente** testes de unidade e integração para grupos de funções atribuídos para a equipe. Na segunda etapa, os estudantes deveriam implementar e analisar testes e elaborar planos de testes e responder questionários sobre a aplicação abordada no projeto.

População. A população do estudo foi composta pelo docente e 15 alunos que participaram das duas etapas do projeto. A disciplina se encaixa no sétimo período, ou seja, *os alunos estão se formando e, portanto, possuem razoável maturidade acadêmica.* Dos 15 estudantes que participaram efetivamente *sete deles não tinham nenhuma experiência com projetos reais, dois participaram de até dois projetos e seis alunos tinham participado de mais projetos.* Todos os estudantes mais experientes *aceitaram o desafio e participaram do projeto.* O professor, graduado, especialista e mestre na área de Computação, trabalhou com desenvolvimento, com experiência anterior no ensino de programação e ES. O professor admitiu usar o método tradicional de ensino. Um aluno assistente que já havia cursado a disciplina foi selecionado para dar suporte aos estudantes (NASCIMENTO, 2017).

Contexto de aplicação da atividade com projeto OSS. A disciplina apresentou e fez a exposição teórica de conteúdos de Teste de Software. A atividade prática com o projeto OSS visou a aplicação de testes automatizados em um projeto real. O projeto OSS utilizado foi o `JabRef`, um gerenciador de referências bibliográficas, de médio porte, desenvolvido em Java para ambiente *desktop* (NASCIMENTO, 2017).

O programa extenso da disciplina foi um problema na aplicação da abordagem, onde

a quantidade de avaliações e o tempo para realizar as atividades também exigiram mudanças no foco inicial das atividades (escopo do estudo). Devido a carga de avaliações da disciplina e a natureza voluntária da participação no projeto, as atividades planejadas tiveram que ter prazos de entrega estendidos. Isso gerou diminuição de tempo das demais atividades e por sua vez gerou mudanças nas necessidades de entrega do trabalho e prejuízos na discussão final.

A coleta e análise dos dados provenientes da abordagem foi feita por questionários, entrevistas e análise das entregas dos estudantes.

Objetivos de Aprendizagem. Os objetivos de aprendizagem relacionados a testes de software e a práticas profissionais beneficiadas pela aplicação da abordagem foram selecionados, ou seja, os resultados de aprendizagem foram divididos em aspecto técnico e aspecto da prática profissional (sociotécnico).

Os objetivos de aprendizagem associados ao aspecto técnico de teste de software foram selecionados de acordo com a capacidade de descrever conceitos e práticas de teste, seus tipos, níveis e técnicas; Entender diferenças de porte de testes, ferramentas de integração e testes; Aplicar testes e ferramentas de testes; Avaliar suíte de testes e métricas de testes; e Criar Testes de Unidades não redundantes, planejar e gerar casos de testes e criar e documentar conjunto de testes. Os objetivos de aprendizagem sociotécnicos foram analisados sobre aspectos de respostas a crítica de terceiros; Ter experiencia e avaliar trabalhos de terceiros; Ser proativo e criativo; Solucionar problemas e gerar soluções alternativas; Ter visão holística e; Compreender, sumarizar e escrever materiais técnicos.

Estrutura para Atividade. Em geral, as atividades propostas visavam criar e executar testes e plano de testes em um projeto real "elaborar e executar ao menos uma vez um plano de testes de regressão, incluindo os testes implementados pela equipe" (NASCIMENTO, 2017, p.144).

Objetivo de Aprendizagem da Atividade. O Objetivo de Aprendizagem analisado no contexto de aplicação do modelo E3S é o objetivo *LO104 - Capacidade de planejar e gerar casos de testes para softwares de médio porte.* Este objetivo demanda o alcance de outros objetivos ou requisitos na área de Testes (conceitos e ferramentas e técnicas necessárias para a execução e documentação dos testes a serem realizados). Como os estudantes têm que saber e saber fazer para o alcance deste objetivo, unindo elementos e gerando algo original, este objetivo foi avaliado baseado na dimensão *Criar* da Taxonomia de Bloom.

Competências de ES. A Competência Técnica em Teste de Software, segundo (ARDIS et al., 2014a), resume competências em planejar testes, infraestrutura de testes, técnicas de teste, medição de testes e rastreamento de defeitos. As competências sociotécnicas associadas são abordadas como competências da prática profissional (NASCIMENTO, 2017).

É importante destacar que os estudantes *realizaram as atividades propostas* e implementaram testes automatizados para testar um software de médio porte, envolvendo complexidade similar a encontrada em software do mundo do trabalho.

Aspectos de influência relevantes no uso do projeto `JabRef`:

A16-Estratégia para atividades e participações dos estudantes: Já no cenário

do estudo em questão, foi descrito que o professor da disciplina fez uma abordagem teórica e expositiva dos conceitos abordados, oferecendo a participação OPCIONAL do estudante nas atividades no Projeto de Código Aberto e, "Caso o estudante participasse, as notas de suas provas poderiam ser substituídas pelas notas obtidas com a realização das atividades" (NASCIMENTO, 2017, p.144). Ou seja, devido a natureza opcional da participação nas atividades com o projeto, a quantidade de alunos participantes tende a não ser a totalidade da turma. Porém, a participação voluntária dos que se propõe a trabalhar nas atividades e a oportunidade de aproveitar a pontuação oriunda desta podem influenciar em alunos mais interessados, atraídos e motivados nas atividades com o projeto, o que pode indicar uma influência no aspecto ***A2-Interesse, motivação e atração.*** Segundo (NASCIMENTO, 2017, p.144):

> Dos 34 matriculados, inicialmente 33 estudantes aceitaram participar do projeto, porém com o andamento da disciplina e a participação sendo voluntária, somente 19 participaram da primeira parte do projeto e 15 participaram da etapa final do projeto.

Portanto, a estratégia adotada na participação dos estudantes pode ter influência na motivação dos mesmos, já que, dos 33 alunos que se interessaram na participação da atividade, apenas 15 permaneceram até o final.

A3-Adaptação ao currículo ou curso: As atividades no projeto foram desenvolvidas de acordo com o programa e os objetivos da disciplina, como versa (NASCIMENTO, 2017, p.145) "Para a aplicação da abordagem, o primeiro passo foi ajustar os objetivos da pesquisa ao programa da disciplina. Como mencionamos anteriormente, escolhemos trabalhar testes de software.".

A4-Nível de formação e experiência do estudante: A disciplina na qual a atividade foi desenvolvida pertencia ao sétimo período do curso, ou seja, os estudantes já estavam em um estágio avançado de formação no mesmo. Dos 15 estudantes com a atuação efetiva nas atividades, sete não tinha experiência em projetos reais e oito tinham experiência em dois ou mais projetos. Sobre isso "Cabe ressaltar que todos os estudantes mais experientes da turma aceitaram o desafio e participaram do projeto." (NASCIMENTO, 2017, p.145). Ou seja, a escolha do projeto, das atividades, e dos Objetivos de aprendizagem podem ser definidos de acordo com o perfil da turma.

A7-Suporte, acompanhamento e avaliação do estudante: Um estudante que já havia cursado a disciplina e com bastante experiência profissional foi escolhido para ser assistente nas atividades para dar suporte no uso de ferramentas e na manipulação do código. Isso pode ser positivo por manter os alunos próximos e engajados nas atividades, além de que os alunos podem se sentir mais confortáveis em tirar as dúvidas ou demais comunicação com um colega de curso.

A8-Capacidade e Habilidade de Programação: Um dos motivos pela escolha do JabRef para as atividades é: "o software é implementado em Java, linguagem que faz parte

do currículo do curso e é empregada em outras disciplinas do curso" (NASCIMENTO, 2017, p.145). Isso é benéfico, pois os estudantes vão analisar um projeto e seu código em uma linguagem nos quais terão familiaridade.

A15-Assunto ou Domínio do Projeto: O JabRef "representa um sistema de informação, com um domínio fácil de ser compreendido" (NASCIMENTO, 2017, p.145), sendo também útil a interesses acadêmicos do estudante.

A5-Tempo, Calendário ou Cronograma: Uma dificuldade encontrada na abordagem foi adaptar as atividades no projeto dentro do cronograma da disciplina, devido esta ter um programa extenso e com muitas atividades de avaliação, gerando concorrência do tempo disponível para a realização da disciplina. Sobre isso (NASCIMENTO, 2017, p.147) descreve:

> Essa problemática acarretou na redução do escopo do estudo de caso, uma vez que retiramos a área de qualidade de software, como mencionamos no início da descrição desse estudo. Mesmo assim, vários estudantes desistiram e para conseguirmos a participação dos demais, precisamos adiar diversas vezes as entregas periódicas dos produtos das atividades, anteriormente planejadas. Em consequência desses adiamentos, o tempo para execução das demais atividades tornou-se cada vez mais exíguo, fazendo com que, não exigíssemos nenhuma entrega individualizada na segunda parte do trabalho e que a discussão final sobre os trabalhos fosse prejudicada.

Portanto, essas dificuldades acabaram tendo impacto em outro aspecto importante, o aspecto ***A7-Suporte, acompanhamento e avaliação do estudante***, tendo adiamento e mudanças em metas e prazos para avaliações e atividades.

Outra dificuldade percebida na aplicação da abordagem foi a quantidade de material adicional que foi necessário preparar e apresentar ao estudante para que as atividades fossem desenvolvidas (NASCIMENTO, 2017). Essa dificuldade foi percebida devido a necessidade de apresentação de ferramentas necessárias como AssertJ e EclEmma, a apresentação de métricas e conceitos necessários do conteúdo abordado e do projeto envolvido, e a apresentação do passo a passo das atividades a serem desenvolvidas e seus objetivos. Ou seja, tais materiais envolvem aspectos como ***A6-Uso ou conhecimento de ferramentas e ambientes de desenvolvimento*** necessários ao desenvolvimento das ativides no JabRef, a ***A17-Curva de Aprendizagem*** dos conceitos, métricas e ferramentas necessárias à participação dos estudantes, e a apresentação do plano e das ***A16-Estratégias para atividades e participações dos estudantes***.

A19-Desgaste, Entusiasmo e Experiência do Docente: Sobre esse aspecto vamos observar o que diz (NASCIMENTO, 2017, p.147).

> Por último, relatamos a dificuldade específica de utilização de um software sobre o qual o professor não tem o domínio, fato comum quando utilizamos

> projetos de código aberto. Como a visão que se tem do software é superficial, torna-se incerto que as funcionalidades distribuídas entre as equipes sejam de complexidade semelhantes, ou seja, o professor não tem a medida correta da complexidade da atividade solicitada.

Ou seja, a pouca experiência do docente com atividades em Projetos Floss e o pouco domínio do Software utilizado pode interferir na coerência da complexidade das atribuições e atividades entre os estudantes.

A16-Estratégia para atividades e participações dos estudantes: Os módulos do projeto foram divididos em equipes no qual as atividades eram desenvolvidas. Isso pode ser benéfico para que cada equipe entregue os produtos de suas atividades e aproximem a realidade de trabalhos colaborados e, ao mesmo tempo divididos em Projetos de Software.

A1-Tamanho e complexidade do projeto: Os estudantes julgaram o projeto com tamanho e complexidade que aproximasse a experiência nas atividades com a realidade de projetos reais. Segundo (NASCIMENTO, 2017, p.153) "todos os estudantes enxergaram o projeto como uma experiência de mundo real. 93% consideraram que o software utilizado representa um exemplo de tamanho e complexidade semelhantes aos que são trabalhados na indústria de software.". Em geral os estudantes julgaram o JabRef como um projeto bastante complexo.

A10-Percepção ou Autopercepção das Habilidades de Computação. Os alunos perceberam suas habilidades aplicadas a complexidade e dificuldade de um projeto real, ou seja, ao questionar os estudantes:

"93% acreditaram que foi possível compreender melhor as habilidades necessárias ao profissional que trabalha no desenvolvimento e manutenção de software." (NASCIMENTO, 2017, 151).

A14-Estado atual do projeto: Problemas na estrutura do código e baixa documentação fizeram os estudantes julgarem o JabRef como um projeto complexo e difícil de entender, porém, mostrou a proximidade com o estado de projetos que existem nas empresas:

"O estudante ainda destacou que a complexidade é agravada pela falta ou escassez de documentação." (NASCIMENTO, 2017, p.154)

A11-Dinâmica, rotina e cultura de Projetos FLOSS. O grande número de envolvidos no projeto e a necessidade de aprender códigos de terceiros pode ter influenciado na percepção de grande complexidade do projeto por alguns alunos.

A2-Interesse, motivação e atração: "um terceiro estudante salientou que alguns estudantes ficaram descontentes porque precisaram ser mais autônomos, enquanto prefeririam algo menos independente" (NASCIMENTO, 2017, p.156).

A1-Tamanho e complexidade do projeto. Alguns alunos sentiram dificuldade para trabalhar no JabRef devido ao tamanho do código.

"Um dos estudantes reportou justamente que teve muita dificuldade para trabalhar com o código grande, dada a dificuldade em entendê-lo." (NASCIMENTO, 2017, p.157).

A4-Nível de formação e experiência do estudante: A falta de experiência de alguns estudantes com projetos mais complexos podem dificultar a realização da atividade solicitada.

"Paralelamente, um estudante justificou a dificuldade em realizar a atividade solicitada, devido à falta de experiência com projetos complexos." (NASCIMENTO, 2017, p.157).

A12-Relação com a comunidade, Receptividade do Projeto: Por ser projeto real, o JabRef pode apresentar dificuldades comuns a projetos desse tipo. Porém, em projetos de código aberto como o JabRef é comum não existir suporte dos desenvolvedores já estabelecidos e os alunos podem ficar receosos em solicitar suporte e comunicação com os desenvolvedores.

> Um dos estudantes relatou exatamente essa dificuldade. Segundo ele, a complexidade dos projetos de código aberto e dos projetos existentes nas empresas podem até ser similares, porém o suporte nas empresas está disponível. Haveria sempre alguém para esclarecer dúvidas ou indicar algum caminho. Quando questionamos que ele poderia ter contatado os desenvolvedores da comunidade, ele assumiu que não o fez por receio deles não terem paciência para respondê-lo. (NASCIMENTO, 2017, p.158).

A17-Curva de Aprendizagem: Para executar os testes no projeto, os estudantes tiveram que enfrentar curva aprendizagem para entender o software a ser testado e identificar no software o que deveria ser testado, suas classes e métodos.

"Um dos estudantes descreveu que para entender como os módulos funcionavam foi preciso fazer algo muito próximo a uma reengenharia." (NASCIMENTO, 2017, p.159).

A4-Nível de formação e experiência do estudante: A facilidade da realização dos testes foi influenciada pela experiência anterior do estudante.

"Enquanto o primeiro estudante imputou a facilidade em decidir o que testar à sua experiência anterior, o segundo estudante admitiu que a falta de experiência anterior prejudicou a realização da atividade."(NASCIMENTO, 2017, p.160).

A11 - Dinâmica, rotina e cultura de Projetos FLOSS: Os alunos tiveram que enfrentar limitações para desenvolver as atividades em um projeto já existente e que não foi projetado com a intenção para as atividades planejadas:

> Como esse é o caso de muitos projetos, incluindo o JabRef, os estudantes deparam-se com tais situações, como por exemplo: componentes não nomeados, que dificultou a utilização do AssertJ; atributos finais, que não poderiam ter seus valores alterados; métodos estáticos, que não poderiam ser

> simulados (“mockados”);métodos que recebiam expressões regulares como parâmetro, entre outros. Finalmente, muitas vezes, era preciso refatorar o código.(NASCIMENTO, 2017, p.160).

A10-Percepção ou Autopercepção das Habilidades de Computação: Os alunos perceberam aspectos de execução de testes em projetos reais e a importância de saber o que fazer e onde fazer para facilitar essa atividade.

> A percepção final dos estudantes sobre a execução dos testes em projetos reais que capturamos foi: não é tão simples implementar testes, porque é complexo achar o que deve ser feito e onde deve ser feito; descobertas as respostas a estas questões, a implementação do teste não é complicada. (NASCIMENTO, 2017, p.160).

A10-Percepção ou Autopercepção das Habilidades de Computação: A atividade no projeto também permitiu que o estudante percebesse a conexão teoria e prática na realização desta e também a importância da teoria para a prática.

“Segundo os estudantes, por meio da prática eles conseguiram ‘entender melhor os conceitos’ de testes caixa-branca, caixa-preta, etc. Exatamente por permitir visualizar através dos testes elaborados, as distinções entre os diferentes tipos de testes.”(NASCIMENTO, 2017, p.163).

“Os estudantes reconheceram a necessidade da teoria para realizar a prática, inclusive para planejar o que deve ser feito.” (NASCIMENTO, 2017, p.165)

A16 - Estratégia para atividades e participações dos estudantes: O uso de poucos exemplos e pouca significação dos exemplos prévios sobre alguns aspectos estudados pode ter influenciado na baixa conexão teoria e prática destes em alguns estudantes.

> Convém ressaltar que estes problemas acarretaram em dificuldades na realização dos testes, especificamente, categorizamo-las em ‘Dificuldades de conexão da teoria com a prática’, divididas em ‘Diferenciação entre teste unitário e teste de integração’ e ‘Diferenciação entre testes caixa-branca e testes caixa-preta’. Essas dificuldades geraram prejuízos no alcance de alguns objetivos de aprendizagem, conforme apresentaremos a seguir. (NASCIMENTO, 2017, p.168).

A6-Uso ou conhecimento de ferramentas e ambientes de desenvolvimento. A atividade propiciou o conhecimento e uso de diversas ferramentas necessárias para a realização desta.

“Todos os estudantes entrevistados destacaram a aprendizagem sobre ferramentas e de como usar essas ferramentas, ao serem questionados sobre sua experiência de aprendizagem realizando o trabalho.” (NASCIMENTO, 2017, p.172).

A10-Percepção ou Autopercepção das Habilidades de Computação: Estudantes e professor observaram a aprendizagem e o conhecimento desenvolvidos na abordagem.

> Os envolvidos também reconheceram esse aprendizado. Enquanto um dos estudantes experientes atestou a aprendizagem de seus colegas, o professor confessou que os estudantes que participaram dessa abordagem terminaram a disciplina com conhecimento muito maior que outros, que fizeram a disciplina em uma oferta anterior.(NASCIMENTO, 2017, p.177).

A12-Relação com a comunidade, Receptividade do Projeto: A comunidade do JabRef se interessou pelo resultado das atividades dos estudantes.

"Outro ponto positivo bastante relevante para a abordagem foi o interesse da comunidade do projeto JabRef pelos testes implementados."(NASCIMENTO, 2017, p.177).

A10 - Percepção ou Autopercepção das Habilidades de Computação: Através da participação na atividade os alunos observaram evolução nas suas habilidades sociotécnicas.

"Tendo uma visão global, houve concordância para todas as habilidades listadas." (NASCIMENTO, 2017, p.181).

A2-Interesse, motivação e atração: Devido a algumas desistências de se manter na atividade até o final por parte de alguns alunos, algumas equipes da atividade acabaram sendo desfeitas restando apenas um aluno para a realização das intervenções planejadas para a equipe, prejudicando a percepção da importância do trabalho em equipe no projeto.

"O trabalho em equipe não funcionou realmente para as equipes ET1 e ET3, já que nesses casos apenas um estudante terminou participando em cada uma dessas equipes." (NASCIMENTO, 2017, p.184).

"Para as equipes onde as atividades terminaram sendo individualizadas, a relevância do trabalho em equipe foi ao menos reconhecida pelos estudantes, que foram, de algum modo, prejudicados." (NASCIMENTO, 2017, p.185).

A10 - Percepção ou Autopercepção das Habilidades de Computação: Foi observada aumento da confiança dos estudantes para a realização de atividades em outros projetos e na sensação de melhora do seu desempenho profissional futuramente.

> Observamos que enquanto 80% dos estudantes estão mais confiantes que serão capazes de realizar atividades semelhantes em outro projeto (QSP1) e 93,33% concordam que as atividades no projeto contribuíam para a melhoria do seu desempenho profissional futuramente (QSP2).(NASCIMENTO, 2017, p.185).

A17-Curva de Aprendizagem e A6-Uso ou conhecimento de ferramentas e ambientes de desenvolvimento. Os alunos enfrentaram curva de aprendizagem íngreme com o domínio de ferramentas diversas.

> Os estudantes apontaram uma curva de aprendizagem mais acentuada para o AssertJ e Mockito, sendo que um deles acrescentou a dificuldade de entender como o Gradle (ferramenta de automação de Build utilizada pelo JabRef) funcionava e a equipe ET6 adicionou que foi necessário estudar mais sobre JUnit, para entender como o framework tratava o lançamento de exceções. Já a equipe ET2, descrevendo as dificuldades que enfrentou, queixou-se que o uso do Mockito dificultou ao invés de ajudar.(NASCIMENTO, 2017, p.197).

A5-Tempo, Calendário ou Cronograma: O tempo curto para aprender todas as ferramentas, estudar o projeto e aplicar a abordagem foi um problema relevante apontado pelo professor e pelos alunos.

> No que diz respeito à aplicação da abordagem, a dificuldade mais citada pelos envolvidos, professor e estudantes, foi exatamente esta: a questão do pouco tempo disponível para a aprendizagem das ferramentas (como mencionado), para o entendimento do projeto de código aberto, para a implementação dos testes e por fim, para a discussão do conteúdo associado. (NASCIMENTO, 2017, p.197).

A15-Assunto ou Domínio do Projeto: Os alunos consideraram positivo o domínio do Projeto JabRef, já que o mesmo apresenta um assunto dentro do contexto de uso dos mesmos.

> Particularmente sobre o seu domínio, os estudantes consideraram uma aplicação importante, já que várias pessoas utilizam-na; que ela está dentro do seu contexto (acadêmico) e é bastante útil para quem faz pesquisa. Desse modo, os estudantes consideraram interessante trabalhar com a aplicação.(NASCIMENTO, 2017, p.198).

A8- Capacidade e Habilidade de Programação: Apesar da Linguagem do Projeto ser comum ao longo do curso, alguns alunos não se motivaram com a linguagem.

"Sobre a tecnologia empregada, apesar de um estudante valorizar a linguagem de construção da aplicação por ser utilizada ao longo do curso, dois estudantes não se sentiram muito motivados." (NASCIMENTO, 2017, p.199).

A2-Interesse, motivação e atração: Apesar de boa parte da turma participar das atividades até o final, outra parte da mesma não se sentiu motivada para participar das atividades devido ao contexto de aplicação da abordagem, sua natureza opcional, com necessidade de tempo e curva de aprendizagem acentuada para o domínio de ferramentas em um projeto grande e com estado do projeto pouco organizado e documentado.

"Para um dos estudantes, o projeto grande, com o código "bagunçado", e não conhecer as ferramentas foram as principais razões para que os demais estudantes não participassem do projeto, uma vez que essa participação era opcional." (NASCIMENTO, 2017, p.199).

"O professor da disciplina confirmou essa mesma percepção, de modo que, os estudantes preferiram dedicarem-se às atividades que eram obrigatórias, para assim conseguirem boas notas."(NASCIMENTO, 2017, p.199).

"Entretanto, o professor ainda salientou que os bons alunos justificaram a não participação em razão da falta de tempo para fazer as atividades."(NASCIMENTO, 2017, p.199).

Discussões sobre o resultado do uso do Projeto JabRef na abordagem.

Podemos perceber que desde o planejamento da abordagem até a sua aplicação e posterior análise foi possível observar diversos aspectos que podem exercer alguma influência

na atividade e no alcance dos objetivos de aprendizagem. Estes aspectos por sua vez podem ser visto como influentes direta ou indiretamente em outros aspectos observados, mostrando que uma abordagem de aprendizagem desenvolvida por atividades mediadas por FLOSS deve ser planejada e acompanhada de forma multifatorial e inter-relacionada.

A1-Tamanho e complexidade do projeto: Foi considerado no planejamento da atividade e na escolha do JabRef, sendo visto como um dos causadores da Curva de Aprendizagem (A17) para a compreensão do projeto e do que deveria ser explorado. Foi observado nos resultados da abordagem como um dos fatores que desestimularam a participação de parte da turma, porém foi positivo ao proporcionar a percepção dos estudantes da atividade com a realidade de projetos reais e com a curva de aprendizagem desenvolvidas por eles.

A2-Interesse, motivação e atração: Esse aspecto foi diretamente associado a não participação de parte da turma, principalmente por influência de outros aspectos como tempo reduzido(A5), carga de trabalho(A7), curva de aprendizagem necessária(A17), complexidade do projeto(A1) e seu estado(A14), e principalmente a natureza opcional da abordagem associada aos fatores anteriores(A16). Porém a parte da turma que decidiu participar da abordagem até o final se sentiram motivados e interessados com a sua percepção de aprendizagem (A10), a Curva de aprendizagem acentuada sobre diversos aspectos da disciplina (A17) e o domínio e aprendizagem de diversas ferramentas (A6).

A3-Adaptação ao currículo ou curso: Aspecto considerado no planejamento da abordagem e responsável por basear o planejamento das atividades de acordo com o planejamento da disciplina, influenciando no aspecto (A16). Provavelmente esse aspecto, mesmo não estando diretamente associado ao sucesso ou fracasso da atividade e dos objetivos alcançados foi um fator relevante no planejamento destes devido a carga de conteúdo e conhecimento necessários ser expostos pela disciplina.

A4 - Nıvel de formação e experiência do estudante: Analisado no planejamento da abordagem, onde o perfil acadêmico e técnico foi analisado também sobre a percepção de habilidades e problemas solucionados pelos estudantes no decorrer da atividade.

A5-Tempo, Calendário ou Cronograma: Foi o aspecto que mais limitou a abordagem, o tempo curto demandou alterações no Aspecto 16-Estratégia para atividades e participações dos estudantes, com mudança no escopo da abordagem, mudanças nos objetivos cobertos pela abordagem e na entrega das atividades pelos estudantes, interferindo nas entregas seguintes, alterando prazos e sobrecarregando os organizadores da atividade. Segundo (NASCIMENTO, 2017, 199) “Convém ressaltar que a questão do tempo disponível reduzido surge novamente como principal problema para a abordagem, conforme discutimos anteriormente.”

A6-Uso ou conhecimento de ferramentas e ambientes de desenvolvimento: Foi analisada na fase de planejamento da atividade e na execução da abordagem. Foi

um dos principais aspectos que causou curva de aprendizagem acentuada dos estudantes. O aprendizado dessas ferramentas para a participação da abordagem foi necessário e os estudantes observaram os resultados de sua aprendizagem. Esse aspecto possivelmente foi um dos que mais interferiram no tempo necessário para a entrega das atividades pelos estudantes e na atração reduzida de parte da turma para a participação da abordagem, porém, foi relatada como positiva para a percepção da aprendizagem dos alunos e da confiança que os mesmos obtiveram em participar de atividades semelhantes posteriormente.

A7-Suporte, acompanhamento e avaliação do estudante: Foi observado esse aspecto de forma positiva no planejamento ao destinar um aluno experiente e que já trabalhou na disciplina para acompanhar os estudantes.

A8- Capacidade e Habilidade de Programação: Foi considerado no planejamento da abordagem ao usar um projeto com linguagem já conhecida dos estudantes (Java). Apesar de alguns alunos não sentirem motivação com a linguagem do projeto, o mesmo não apresentou empecilho para a participação dos alunos ou necessitou de curva de aprendizagem acentuada na linguagem escolhida.

A9-Suporte ou viabilidade acadêmica: Apesar da abordagem ter sido aplicada em uma disciplina importante do curso, com participação não obrigatória dos estudantes, o professor da disciplina foi importante no decorrer do planejamento e execução da abordagem. Porém não foi observado claramente o suporte acadêmico de aspectos técnicos como adaptação de laboratórios e apoio técnico. Portanto, tal aspecto parece não ter sido relevante na aplicação da abordagem e no alcance de seus objetivos.

A10-Percepção ou Autopercepção das Habilidades de Computação: O aspecto mais positivo observado na abordagem, os alunos perceberam suas habilidades aplicadas, perceberam curva de aprendizagem acentuada desenvolvida, perceberam evolução dos colegas, perceberam a conexão entre teoria e prática e perceberam sua evolução na aprendizagem tanto nos aspectos técnicos como nos aspectos sociotécnicos. Essa percepção acurada dos estudantes em diversos aspectos de sua aprendizagem pode ter sido auxiliada pela Estratégia de Aplicação da Abordagem(A16), com um conjunto de entregas bastante ativas, que exigiu esforço autônomo pelos estudantes.

A11- Dinâmica, rotina e cultura de Projetos FLOSS: A Quantidade de envolvidos no projeto e a rotina de avaliar projetos complexos pode ser importante para o estudante observar a dinâmica de desenvolvimento de projetos reais e a perspectiva das atividades no âmbito profissional.

A12-Relação com a comunidade, Receptividade do Projeto: A relação com a comunidade não foi obrigatória(A16), porém, as atividades desenvolvidas pelos estudantes foram bem aceitas pela comunidade. Alguns estudantes tiveram receio de interagir com os desenvolvedores da comunidade para perguntas e suporte. Possivelmente o aluno assistente acompanhando e dando suporte aos estudantes(A7) seja um fator que dimi-

nuiu a necessidade de interação dos alunos com a comunidade para auxílio nas atividades.

A13-Restrições, Viabilidade e Adequação do Projeto: Não observado.

A14-Estado atual do projeto: Considerado no planejamento e na execução da abordagem, O projeto com erros na estrutura do código e falta na documentação foi considerada relevante para a Curva de Aprendizagem Acentuada desenvolvida pelos estudantes(A17), para a noção de complexidade alta do projeto(A1), e para inibição dos estudantes que decidiram não participar da abordagem(A2), visto que a curva de aprendizagem desprendida na compreensão do projeto e das atividades nele desenvolvidas estaria diretamente relacionada com o tempo reduzido que os estudantes tinham para a conclusão de suas obrigações acadêmicas. Devido ao projeto ainda necessitar de várias abordagens de testes e correções em sua estrutura, o estado do JabRef foi satisfatório para a cobertura dos objetivos a serem estimulados pela atividade.

A15-Assunto ou Domínio do Projeto: Por ser de um domínio importante para os estudantes e com possível aproveitamento dos mesmos em atividades acadêmicas posteriores, o JabRef foi considerado um projeto importante para os estudantes e com uma regra de negócio atrativa a ser compreendida pelos mesmos.

A16- A Estratégia para atividades e participações dos estudantes: foi relevante para definir o escopo da atividade, as equipes e planejamento dos resultados. Foi fortemente influenciada pelo aspecto pouco tempo disponível. O aspecto A5- Tempo, Calendário ou Cronograma, foi a temática que mais limitou o desenvolvimento da abordagem, primeiro que esse fator tempo já obrigou mudanças em outras temáticas do aspecto A16, como cobertura dos objetivos, entregas e prazos. O tempo também exerceu forte influência no Aspecto A2-Interesse, motivação e atração, já que o calendário da disciplina e o tempo gasto para realizar a atividade foi um dos problemas indicados pelo professor e pelo aluno. A cobertura de uma gama de ferramentas necessárias para a participação nas atividades definidas pela estratégia foi relevante na Curva de Aprendizagem desenvolvida pelos aluno nesse tema, mas inibiu a participação de outros estudantes que se assustaram com a carga de trabalho que desenvolveriam. A natureza opcional da atividade e os objetivos alcançados por esta pelos estudantes que aceitaram participar mostram que esse aspecto foi satisfatório no que tange a proposta planejada.

A17- Curva de Aprendizagem: A curva de aprendizagem foi um dos aspectos mais demandaram tempo para a realização das atividades segundo os estudantes. Os alunos tiveram que enfrentar curva de aprendizagem para dominar os módulos do projeto e sua codificação, enfrentaram curva de aprendizagem acentuada principalmente sobre o uso de ferramentas e ambientes de controle de versões (A6). Essa curva de aprendizagem prévia com ferramentas demandou tempo adicional para a entrega das atividades, porém, essa essa curva de aprendizagem enfrentada e desenvolvida pelos estudantes foram fundamentais para a percepção positiva sobre o desenvolvimento de suas habilidades em computação(A10), tanto nas competências técnicas planejadas e no domínio das ferra-

mentas necessárias quanto nas competências sociotécnicas desenvolvidas.

A18-Questões sociais, pessoais e legais: não observado.

A19-Desgaste, Entusiasmo e Experiência do Docente: A experiência do docente foi observada no planejamento da atividade, a baixa experiência do docente no uso de FLOSS para ensinar foi uma oportunidade para avaliação dos objetivos da pesquisa visada pelo estudo de caso, portanto, a curva de aprendizagem do docente no que tange a dinâmica de aprendizagem mediada por FLOSS pode ser um fator relevante, apesar do professor ter que demandar esforço tanto na aplicação da abordagem como na aplicação da sua prática comum com os alunos que decidiram não participar até o final das atividades. Portanto, a equipe que aplicou o estudo de caso e a abordagem foram importantes para o acompanhamento e suporte de aprendizagem dos estudantes que participaram das atividades.

Podemos observar várias temáticas de aspectos que influenciaram direta ou indiretamente o planejamento e execução das atividades na abordagem bem como o objetivo por ela proposto e aqui analisado.

Vários aspectos tiveram grande interferência no esforço desprendido pelos alunos e pelo corpo docente/aplicadores da abordagem. Porém, tais esforços também estimularam a transposição dos desafios na abordagem, trazendo aprendizados não apenas para os alunos nas atividades, mas também ao professor e pesquisador que aplicou a abordagem, auxiliando os mesmos a observar aspectos que podem surgir na aplicação de uma proposta educativa ativa e próxima a realidade da indústria.

Os aspectos críticos observados no planejamento da abordagem foram importantes para diminuir o impacto destes na execução da mesma e nos resultados da aprendizagem. A Atenção com a Linguagem, com o perfil e experiência da turma e com a adaptação ao currículo ou curso, evitaram Curva de Aprendizagem desnecessária à proposta da disciplina. Designar aluno para auxílio dos estudantes pode auxiliar no processo de suporte mais próximo nas atividades.

A carga de competências necessárias a serem estimuladas pela disciplina foi um fator desafiador para definir o escopo do que seria coberto pela proposta. Esse fator foi determinante para a influência do tempo de forma negativa para as decisões na estratégia de atividades e na execução da abordagem, forçando ao professor/aplicador a extensão de prazos de entrega de atividades. A natureza opcional da participação foi relevante para baixa atração de parte da turma e para a motivação dos que decidiram participar até o final.

O Aspecto de influência mais limitadora na abordagem foi o Tempo reduzido para o desenvolvimento da proposta, influenciando na estratégia e planejamento da atividade, na atração dos estudantes e no cronograma de entrega das atividades, demandando es-

força docente e discente para o alcance dos objetivos.

A curva de aprendizagem foi o aspecto mais presente na execução da abordagem. Foi necessário o domínio de diversas ferramentas previamente, o domínio do código e de como participar ativamente da abordagem. O Resultado de tanto esforço foi o alcance dos objetivos propostos, com reconhecida aprendizagem das ferramentas e dos aspectos técnicos de testes objetivados.

A percepção da aprendizagem por parte dos estudantes e instrutores foi o aspecto mais positivo observado na abordagem. A curva de aprendizagem necessária e desenvolvida nas atividades estimulou o estudante a observar positivamente sua aprendizagem tanto nos aspectos técnicos como nos aspectos sociotécnicos, aumentando a confiança sobre suas habilidades e perspectivas profissionais.

A estratégia de aplicação da abordagem foi o aspecto mais relevante no resultado de aprendizagem, apesar de todas as dificuldades e desafios apresentados no decorrer da proposta, a condução da aprendizagem foi reconhecidamente positiva pelos estudantes e pelo professor. Segundo (NASCIMENTO, 2017, p.170):

> Numa análise descritiva, percebemos que os estudantes, em sua grande maioria, concordaram com a contribuição da execução do projeto para a sua aprendizagem, para todos os objetivos de aprendizagem tratados pela abordagem.

Por sua vez o alcance do objetivo de aprendizagem ao nível cognitivo mais alto, estimulou o alcance das competências técnicas na área de testes de software e sociotécnicas da Engenharia de Software como um todo. Estas por sua vez podem ser relevantes para o alcance das Competências Gerais esperadas aos egressos de graduação em Computação segundo a (EDUCAÇÃO-MEC, 2016. Seção 1. p. 22), sendo elas especificamente:

I - identificar problemas que tenham solução algorítmica: A necessidade de testes automatizados no JabReb e o desenvolvimento destes pelos alunos estimula o alcance desta necessidade de formação;

III - resolver problemas usando ambientes de programação: As diversas atividades desenvolvidas pelos alunos usando as ferramentas e ambientes de desenvolvimento na solução dos problemas propostos se mostram relevantes no estímulo desta competência;

IV - 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: A descoberta do que fazer e como fazer no projeto, quais intervenções desenvolver e como executar as atividades de forma a auxiliar o desenvolvimento do JabRef pode ser visto como estimulante para esta competência geral;

VI - gerir a sua própria aprendizagem e desenvolvimento, incluindo a gestão de tempo e competências organizacionais: A natureza ativa e a imersão na realidade de desenvolvimento de projetos reais, alinhados com as obrigações da disciplina e da necessidade de alcance dos objetivos demonstra o estimulo desta competência;

VII - preparar e apresentar seus trabalhos e problemas técnicos e suas soluções para audiências diversas, em formatos apropriados (oral e escrito): A apresentação dos resultados da atividade desenvolvidas pelos estudantes e o interesse da comunidade do JabRef nestas mostram que essa competência pode ter sido estimulada;

VIII - avaliar criticamente projetos de sistemas de computação: O principal foco da abordagem foi exatamente a avaliação e testagem do JabRef e de seus elementos, mostrando grande estímulo nesta competência;

IX - adequar-se rapidamente às mudanças tecnológicas e aos novos ambientes de trabalho: A abordagem aqui analisada tem grande influência nesta competência. Os estudantes observaram a possibilidade de atuar em novos métodos de desenvolvimento e ganhar confiança sobre suas habilidades;

X - ler textos técnicos na língua inglesa: A codificação do JabRef e sua documentação majoritariamente na língua inglesa (NASCIMENTO, 2017) estimulou o estudante a desenvolver esta competência;

XII - ser capaz de realizar trabalho cooperativo e entender os benefícios que este pode produzir: Os estudantes contribuíram para uma comunidade de desenvolvimento com suas atividades, trabalharam em equipe, apesar de alguns alunos terem executado atividades individualmente (Esse problema pode ser resolvido com a obrigatoriedade dos grupos colaborarem nos resultados entre si), os mesmos foram estimulados a contribuir e compreender contribuições na comunidade, estimulando esta competência.

Após toda essa discussão podemos perceber os benefícios da abordagem para a aprendizagem dos estudantes. Para visualizar a relevância dos aspectos na abordagem nas diferentes fases da mesma podemos usar a ficha de listagem dos aspectos críticos aqui proposto para listar os aspectos mais relevantes na abordagem do Estudo de Caso 2 como mostra a tabela 4.34

Para contextualizar a avaliação de cada aspecto encontrado de acordo com os relatos e percepções advindas da abordagem podemos usar a Proposta de ficha de avaliação da eficácia de uso do Projeto FLOSS em EES apresentada anteriormente. Um exemplo de fichamento de avaliação da eficácia do uso do JabRef pode ser observada na tabela 4.35

Analisando os relatos referente aos aspectos que tiveram potos positivos e limitadores, podemos analisar que os aspectos que tiveram influência negativa em algum momento da abordagem não foram suficientes para superar a influência positiva, pois, os resultados de aprendizagem tanto observados pelo instrutor quanto pelos alunos mostram que a influência na aprendizagem foi mais positiva que limitadora.

Tabela 4.34 Proposta de uso do Modelo Conceitual para avaliar a eficácia do uso do Projeto JabRef no Ensino de Teste de Software no Estudo de Caso

Fichamento dos Aspectos observados na aplicação da Abordagem		
Controle		
Planejamento	Execução	Avaliação
Atividade: Elaborar e executar ao menos uma vez um plano de testes de regressão, incluindo os testes implementados pela equipe		
Competências de ES a ser desenvolvida (Objetivo de aprendizagem) : Capacidade de planejar e gerar casos de testes para softwares de médio porte; Competências Sociotécnicas de ES	Competências em desenvolvimento (Objetivos de aprendizagem): Capacidade de planejar e gerar casos de testes para softwares de médio porte; Competências Sociotécnicas de ES	Objetivos alcançados e competências desenvolvidas de acordo com os níveis Cognitivos (Objetivos de aprendizagem): CRIAR- Capacidade de planejar e gerar casos de testes para softwares de médio porte.
Competências alvo a serem estimuladas (Necessidades de Formação) Não observado no planejamento.	Competências gerais em estímulo (Necessidades de Formação) Não mensurado na Execução	Competências Gerais percebidas na atividade (Necessidades de Formação) CGI; CGIII;CGVI;CGVII; CGVIII;CGIX; CGX; CGXII
Aspectos críticos no panejamento	Aspectos críticos na execução	Aspectos críticos na avaliação
A16- Estratégia para atividades e participações dos estudantes	A1- Tamanho e complexidade do projeto	A16- Estratégia para atividades e participações dos estudantes
A1- Tamanho e complexidade do projeto	A2- Interesse, motivação e atração	A5- Tempo, Calendário ou Cronograma
A3- Adaptação ao currículo ou curso	A5- Tempo, Calendário ou Cronograma	A10- Percepção ou Autopercepção das Habilidade de Computação
A4- Nível de formação e experiência do estudante	A7- Suporte, acompanhamento e avaliação do estudante	A6-Uso ou conhecimento de ferramentas e ambientes de desenvolvimento.
A7- Suporte, acompanhamento e avaliação do estudante	A10- Percepção ou Autopercepção das Habilidade de Computação	A17- Curva de Aprendizagem
A14- Estado atual do projeto	A6-Uso ou conhecimento de ferramentas e ambientes de desenvolvimento.	
A15- Assunto ou Domínio do Projeto	A17- Curva de Aprendizagem	
A5- Tempo, Calendário ou Cronograma	A16. Estratégia para atividades e participações dos estudantes	
A8- Capacidade e Habilidade de Programação		

Fonte: Elaboração Própria

Como observado em 4.35 apenas o aspecto A5- Tempo, Calendário ou Cronograma foi constantemente avaliada sobre uma ótica limitadora na abordagem. Os aspectos A9, A13 e A18 não foram associadas a interferências na aprendizagem.

Quando se observa a análise dos resultados positivos do estudo e das competências desenvolvidas pelos estudantes que optaram participar do projeto, considerando o cenário e o perfil dos alunos é pertinente considerar o uso do JabRef como eficaz na abordagem proposta. Sobre isso (NASCIMENTO, 2017, p.177) versa:

> Por fim, concluímos que os estudantes cumpriram o que foi proposto: implementaram testes automatizados para testar um software de médio porte envolvendo complexidade similar a encontrada nos softwares existentes nas empresas.

Além do cumprimento do que foi proposto, podemos observar que as competências necessárias na formação do específica em teste de software alinhado com o aprendizado sociotécnico foi relevante para o estímulos das competências gerais necessárias a qualquer graduado com computação segundo as DCNs.

Tabela 4.35 Proposta de ficha de avaliação da eficácia de uso do JabRef no ensino de Teste de Software

Atividade Planejada: **elaborar e executar ao menos uma vez um plano de testes de regressão,incluindo os testes implementados pela equipe**																		
PROJETO FLOSS ESCOLHIDO:	**Objetivo de Aprendizagem:** Capacidade de planejar e gerar casos de testes para softwares de médio porte																	
	Processo cognitivo usado na atividade para atingir objetivo																	
JABREF	**Lembrar**			**Entender**			**Aplicar**			**Analisar**			**Avaliar**			**Criar**		
Aspecto que influencia na aprendizagem em ES encontrado no uso de FLOSS	**P**	**NA**	**N**	**P**	**NA**	**N**	**P**	**NA**	**N**	**P**	**NA**	**N**	**P**	**NA**	**N**	**P**	**NA**	**N**
A1- Tamanho e complexidade do projeto:																X		X
A2- Interesse, motivação e atração:																X		X
A3- Adaptação ao currículo ou curso:																X		
A4- Nível de formação e experiência do estudante:																X		
A5- Tempo, Calendário ou Cronograma																		X
A6- Uso ou conhecimento de ferramentas e ambientes de desenvolvimento																X		
A7- Suporte, acompanhamento e avaliação do estudante																X		
A8- Capacidade e Habilidade de Programação																X		
A9- Suporte ou viabilidade acadêmica																	X	
A10- Percepção ou Autopercepção das Habilidade de Computação																X		
A11- Dinâmica, rotina e cultura de Projetos FLOSS																X		
A12- Relação com a comunidade e Receptividade do Projeto																X		
A13- Restrições, Viabilidade e Adequação do Projeto																	X	
A14- Estado atual do projeto																X		X
A15- Assunto ou Domínio do Projeto																X		
A16- Estratégia para atividades e participações dos estudantes																X		X
A17- Curva de Aprendizagem																X		X
A18- Questões Sociais, Éticas e Legais																	X	
A19- Desgaste, Entusiasmo e Experiência do Docente																X		
Considerações do professor sobre a eficácia de uso do projeto na atividade de ensino em ES: **O resultado de aprendizagem e dos objetivos planejados indicam que a influência positiva nas temáticas dos aspectos foi maior que a influência negativa.** **Portanto o uso do JabRef foi eficaz na abordagem.**																		
Legenda: P = Influência Positiva; NA = Não associado; N= Influência negativa																		

Fonte: Elaboração Própria

A apresentação dessa abordagem na exemplificação do modelo deve ser vista apenas como um exemplo hipotético. Devido a natureza investigativa da abordagem, muitos aspectos puderam ser vistos sob uma ótica mais científica, e a análise dos resultados de aprendizagem também.

O modelo proposto não deve ser visto com um processo que deve ser seguido de forma inflexível, mas sim um agrupamento de conceitos e temas que se relacionam e que pode servir ao professor para desenvolver um olhar mais direcionado aos fatores de interferência na aprendizagem que podem ser planejados, controlados e avaliados, contribuindo para o ganho de experiência do docente no uso do Projeto FLOSS não apenas sobre aspectos técnicos, mas também pedagógicos.

4.4.4 Usando a metodologia de construção do `Modelo Conceitual` proposto para instruir novos modelos de avaliação de ferramentas ou metodologias de aprendizagem em Computação.

- Defina a área da Computação no qual a ferramenta/recurso ou metodologia de ensino será aplicada, as questões e objetivos da pesquisa. Use a Análise de Conteúdo para mapear competências, conteúdos, conhecimentos, regras metodológicas ou demais requisitos críticos de avaliação segundo os objetivos de pesquisa.
- Faça a leitura flutuante para identificar possíveis hipóteses, índices e indicadores. Defina o *corpus* da análise, podendo ser guias curriculares, currículos ou referenciais de formação, para mapear os requisitos de avaliação objetivados pelo uso da ferramenta/recurso ou metodologia. Defina o *corpus* de análise para mapear aspectos de interferência no ensino da área da Computação encontrados na ferramenta/recurso ou metodologia.
- Faça os mapeamentos através da exploração do material para identificar os requisitos de avaliação e os aspectos encontrados nos estudos sobre uso da ferramenta/-recurso ou metodologia relacionados aos requisitos de avaliação.
- Encontre as categorias de análise provenientes da exploração e faça movimento dialógico entre elas para analisar as inter-relações. Compare as relações entre as categorias com os critérios de avaliação abordados nos objetivos de pesquisa.
- Contextualize os critérios de avaliação com modelos ou guias profissionais. Contextualize os objetivos educacionais em modelos ou taxonomias de aprendizagem. Modele visualmente as relações entre cada critério ou categoria mapeado pela análise de conteúdo. A descrição visual das atividades é exibida na Figura 4.26.

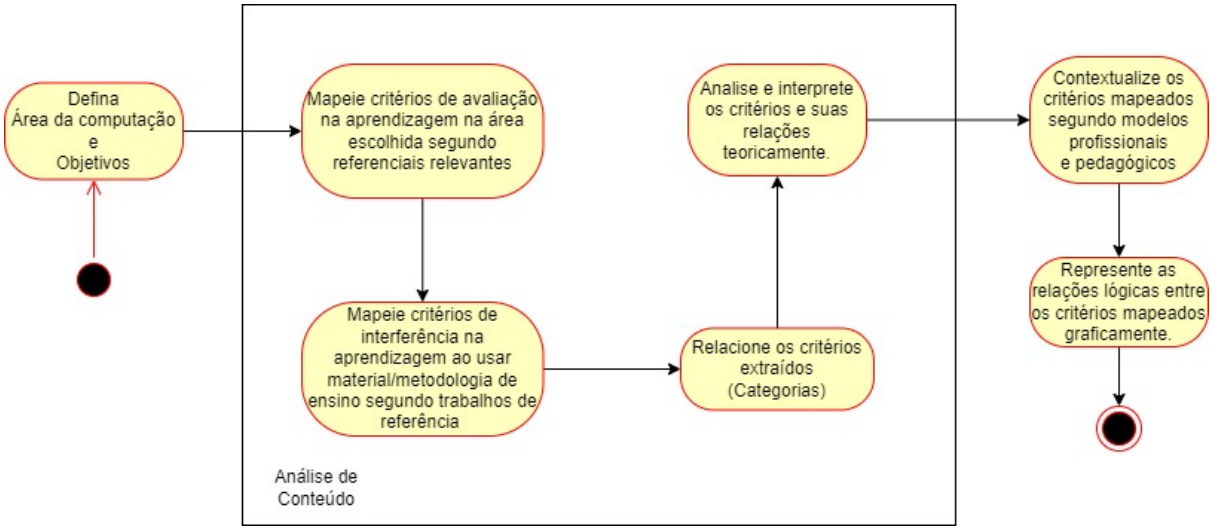

Figura 4.26 Contextualização metodológica de adaptação do modelo proposto para outras áreas da computação e demais ferramentas de aprendizagem
Fonte: Elaboração Própria

A análise de conteúdo é uma técnica poderosa para compreender o que está explícito ou implícito sobre objetivos educacionais e sobre uso de recursos didáticos para o alcance destes. A análise de conteúdo também pode ser usada para mapear competências e aspectos de interferência na aprendizagem segundo a vivência de alunos e professores nas práticas educativas. Modelar relações temáticas diversas de interferência na aprendizagem pode auxiliar os profissionais da educação a prever situações no qual o risco de falha ou de sucesso nos processos de ensino e aprendizagem se apresente.

Capítulo 5

CONSIDERAÇÕES FINAIS E CONCLUSÃO

Este Capítulo apresenta discussões sobre ameaças à validade da pesquisa, indicações de trabalhos futuros e as conclusões provenientes do estudo.

5.1 AMEAÇAS À VALIDADE

Para o desenvolvimento da pesquisa, limitações do estudo e ameaças a validade possíveis foram analisados. Nesta pesquisa, a preocupação com a validade é dupla, pois, por ser uma pesquisa de método misto, deve ser analisada sobre o olhar de validade considerada em pesquisas qualitativas e pesquisas quantitativas. Considerando que a pesquisa se delineia por método qualitativo seguido posteriormente pelo quantitativo, a validade da pesquisa no método quantitativo é diretamente relacionada com a validade no método qualitativo.

Uma das ameaças a validade nessa pesquisa é o viés proveniente da subjetividade do pesquisador. Para mitigar essa ameaça, a estratégia de triangulação foi adotada. Sobre ela (CRESWELL, 2007, p.200) versa que o pesquisador "Faça uma triangulação de diferentes fontes de informações de dados, examinando as evidências das fontes e usando-as para criar uma justificativa coesa para os tema". A base teórica e o *corpus* da pesquisa foi delineado com base em diferentes trabalhos, referenciais curriculares, leis e guias internacionais. Essa base extensa permitiu amplas perspectivas sobre o delineamento teórico e serviu para dar consistência às relações temáticas que emergiram do estudo.

Outra possível limitação é a quantidade reduzida de codificadores na análise de conteúdo proposta. Para mitigar essa ameaça todo o processo de exploração e codificação foi fichado passo a passo e organizado em linha sequencial próxima ao recomendado pela técnica, permitindo que qualquer pesquisador possa percorrer o processo de codificação e avaliar o procedimento. Sobre isso (BARDIN, 2011, p.48) versa que o pesquisador "Pode utilizar uma ou várias operações, em complementaridade, de modo a enriquecer os resultados, ou aumentar a sua validade, aspirando assim a uma interpretação final fundamentada". Nesse estudo, foram desenvolvidas operações diversas usando enumeração de coocorrência, presença e frequência nos diversos elementos do *corpus* da pesquisa, a fim de confrontar as temáticas e suas relações em cada análise.

Outra limitação importante a ser analisada é a natureza dinâmica dos princípios teóricos e definições na visão de competência na área de Engenharia de Software. De acordo com o desenvolvimento de novos conceitos e compreensão de novos aspectos que abordam a área da ES, a concepção ou visão da abordagem por competências pode variar com o tempo, tornando o que se compreende de competências e conhecimentos abordados neste estudo algo a ser reanalisado e contextualizado a novos conceitos e abordagens. A codificação contextualizada e o diálogo dos diversos conceitos e definições segundo os autores abordados dentro do universo de estudo podem ser úteis para mitigar essa limitação.

Sobre as ameaças à validade da pesquisa na etapa quantitativa, a quantificação e a instrumentação enviesada pelas crenças e concepções do pesquisador podem gerar resultados inapropriados para o estudo. A triangulação constante na etapa qualitativa e a sequência registrada passo a passo na análise de conteúdo com descrição de cada decisão na análise permite quantificação e interpretação dos resultados quantitativos que podem ser avaliados ao se repetir a codificação.

5.2 TRABALHOS FUTUROS

Algumas oportunidades de pesquisas e estudos surgem após os resultados deste trabalho. O próximo passo, em continuidade a nosso estudo, é fazer testes e avaliações do modelo proposto em situações reais de ensino, por meio de novos procedimentos e abordagens, como estudos de caso com professores e demais profissionais em educação em Engenharia de Software que usam Projetos FLOSS como material de ensino.

Outras oportunidades de pesquisa são o mapeamento de competências e suas relações com as necessidades de formação em Computação para cada uma de suas subáreas, e o mapeamento de competências em ES nos referenciais de formação ou guias curriculares em Cursos de Pós-Graduação.

Também, pesquisas que envolvam o mapeamento de competências no Curricula Computing 2020 podem trazer resultados que norteiam o desenvolvimento de novos Referenciais Curriculares para cada área da computação abordado pelo relatório. A análise de conteúdo usada para mapear guias curriculares mostra a possibilidade do uso da técnica em pesquisas sobre vários tipos de análises como comparar currículos, Projetos Pedagógicos de Cursos, planos de disciplinas e cursos entre outros elementos bibliográficos. A análise de conteúdo nos trabalhos de relevância mostram que essa abordagem pode ser usada em pesquisas para investigar qualquer material ou metodologia usado no ensino de Computação.

5.3 CONCLUSÃO

A Educação em Engenharia de Software apresenta-se com extrema relevância para o desenvolvimento profissional da área de Engenharia de Software e da Computação. A formação de profissionais de software preparados para o mundo do trabalho passa pela aquisição de competências e habilidades que garantam tal preparação. Estas dependem do domínio de técnicas, conceitos e métodos da Engenharia de Software que são essenciais

para enfrentar os problemas ou desafios nas atividades profissionais. Porém, ensinar tais técnicas e métodos de forma a aproximar o estudante à realidade da industria de software ainda é um desafio. Os Projetos Free/Libre Open Source Software (FLOSS) mostram-se como uma ferramenta viável e promissora para inserir o estudante em Engenharia de Software na dinâmica profissional da indústria de software.

Contudo, a escolha de um Projeto FLOSS eficaz para o ensino de ES é um desafio devido aos vários projetos disponíveis nas comunidades de desenvolvimento. O uso de Projetos FLOSS como recurso ou Objeto de Aprendizagem de forma efetiva no contexto de ensino planejado pelo professor passa pela a análise e avaliação destes nas práticas de ensino. Esta avaliação permite ao docente adquirir experiência sobre o projeto escolhido, o contexto de ensino aplicado e sobre sua própria prática educativa. Portanto, o problema alvo dessa pesquisa foi *avaliar a eficácia de uso de um projeto FLOSS no ensino de Engenharia de Software*.

Para a resolução deste problema é relevante a perspectiva de que a educação em ES se torna efetiva quando os objetivos de aprendizagem delineados pelo professor são alcançados. Tais objetivos tratam do desenvolvimento de competências técnicas e sociotécnicas(comportamentais, atitudinais) em engenharia de software que garantam profissionais prontos para as demandas da indústria e da sociedade. O uso dos Projetos FLOSS como ferramenta, recurso ou Objeto de Aprendizagem em ES devem objetivar o desenvolvimento destas competências. Porém, alguns fatores ou aspectos técnicos e pedagógicos podem interferir no alcance destas competências. Assim, estruturar um modelo para o conhecimento das competências atreladas ao ensino de engenharia de software e o conhecimento dos principais aspectos de interferências no alcance de tais competências ao usar Projetos FLOSS no ensino é a hipótese que guia o estudo.

Para validar ou refutar a hipótese, a pesquisa teve como objetivos elucidar quais as competências mais atreladas ao ensino de engenharia de software e a relevância das competências técnicas e sociotécnicas da engenharia de software na formação em computação, elucidar os principais aspectos de interferência na aprendizagem em ES por meio do uso de Projetos FLOSS, e relacionar as competências e aspectos elucidados para propor um modelo para avaliar a eficácia do uso de Projetos FLOSS no ensino de Engenharia de Software.

Para atingir os objetivos, o método utilizado foi o uso da Técnica de Análise de conteúdo para: mapear as competências mais associadas a Engenharia de Software segundo Referenciais de Formação em computação nacionais; mapear a relevância de competências técnicas e sociotécnicas de ES na formação em computação segundo Referenciais de Formação em computação internacionais; mapear aspectos técnicos ou pedagógicos mais relatados em estudos que versam sobre o uso de Projetos FLOSS no ensino de Engenharia de Software; e assim dialogar os resultados do mapeamento para propor um Modelo para avaliar a eficácia do uso de Projetos FLOSS no ensino de ES.

A análise de conteúdo indicou a relação de competências sociotécnicas com os objetivos de formação em Engenharia de Software, a relação das competências técnicas em Engenharia de Software com objetivos de formação em computação, a relevância das competências técnicas de ES e competências sociotécnicas para a formação em computação, a relação de aspectos que podem influenciar a aprendizagem em ES ao se usar Projetos

FLOSS como recurso didático e, assim, permitiu relacionar as competências com os aspectos para dialogar suas relações e propor um Modelo para avaliar a eficácia do uso de Projetos FLOSS no ensino de ES planejado pelo professor.

Essa pesquisa trouxe algumas contribuições. A descrição das competências em engenharia de software mostra que é essencial desenvolver um ensino técnico específico atrelado ao desenvolvimento sociotécnico. O estudo mostrou como o conhecimento em engenharia de software e o conhecimento sociotécnico têm relevância central no planejamento da formação superior em Computação. O estudo trouxe luz a aspectos que são críticos na interferência da aprendizagem em engenharia de software por meio de Projetos FLOSS. Toda a análise desenvolvida mostra que é possível relacionar aspectos técnicos e pedagógicos a competências em ES a serem desenvolvidas e auxiliam o professor a tomar decisões antes, durante e após as atividades desenvolvidas.

O `Modelo` proposto auxilia professores experientes no uso de FLOSS em suas disciplinas a ter mais eficiência em suas análises e avaliações do projeto escolhido. O `Modelo` também auxilia professores inexperientes a conhecerem os aspectos que podem interferir no seu trabalho e se planejar para mitigar possíveis erros comuns a professores iniciantes nas dinâmicas de ensino por meio do uso de Projetos FLOSS. Além do mais, toda a análise desenvolvida sugere investigações mais profundas que permitam contribuir para pesquisadores, educadores e profissionais da Computação no mapeamento de competências e de fatores de interferência pedagógica em Computação, auxiliando na investida por novas metodologias e práticas de ensino que aproximem o estudante de computação da realidade da indústria e do mundo do trabalho como um todo.

REFERÊNCIAS BIBLIOGRÁFICAS

ADINARAYANAN, V. et al. An open source approach to enhance industry preparedness of students. In: IEEE. *2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. [S.l.], 2014. p. 194–200.

AGUIAR, E. V. B.; FLÔRES, M. L. P. Objetos de aprendizagem: conceitos básicos. *Objetos de aprendizagem: teoria e prática. Porto Alegre: Evangraf*, p. 12–28, 2014.

ALMEIDA, R. R.; CHAVES, A. C. L.; JR, C. F. de A. Avaliação de objetos de aprendizagem: aspectos a serem considerados neste processo. *Revista Educação & Tecnologia*, n. 13, 2015.

ALVES, A. G.; BENITTI, F. B. Processo de desenvolvimento integrando disciplinas de engenharia de software. In: *XIV Workshop sobre Educação em Computação–Anais do XXVI Congresso da Sociedade Brasileira de Computação–Anais do*. [S.l.: s.n.], 2006. p. 206–215.

ALVES, M. O tempo e o espaço da formação contínua de professores: Diagnóstico, processo e perspetivas. Edições Universitárias Lusófonas, 2019.

ANDRADE, R. M. de C.; SANTOS, I. de S.; LINHARES, I. Uma metodologia para o ensino teórico e prático da engenharia de software. *FEES*, p. 60, 2015.

ARDIS, M. et al. The software engineering competency model (swecom). *IEEE Computer Society, Los Alamitos, CA, USA*, 2014.

ARDIS, M. et al. *Software Engineering 2014: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering*. 2014. Disponível em: <https://www.acm.org/binaries/content/assets/education/se2014.pdf>. Acesso em: 04 de fevereiro de 2017.

BAILEY, M.; LUNT, B.; ROMNEY, G. Computing curricula: The history and current status of 4 year computing programs. In: *2006 Annual Conference & Exposition*. [S.l.: s.n.], 2006. p. 11–346.

BALBINOT, E. L.; FILHO, A. R. Controles de gestão: um estudo teórico. *Revista Eletrônica de Contabilidade*, v. 2, n. 2, p. 114–114, 2005.

BARBOSA, M. W.; NELSON, M. A. V. Desafios do desenvolvimento de atividades práticas de engenharia de software em grupo em cursos a distância. *FEES 2015*, p. 1, 2015.

BARDIN, L. Análise de conteúdo/laurence bardin; tradução luís antero reto, augusto pinheiro. *São Paulo: Edições*, v. 70, 2011.

BEGOSSO, L. R. et al. Programa de residência em software. In: *XIX Workshop de Educaç ao em Informática., Natal, Brasil*. [S.l.: s.n.], 2011.

BLOOM, B. *Bloom's taxonomy*. 1956.

BOURQUE, P. et al. The guide to the software engineering body of knowledge. *IEEE software*, IEEE, v. 16, n. 6, p. 35–44, 1999.

BOURQUE, P.; FAIRLEY, R. E. et al. *Guide to the software engineering body of knowledge (SWEBOK (R)): Version 3.0*. [S.l.]: IEEE Computer Society Press, 2014.

BOWRING, J.; BURKE, Q. Shaping software engineering curricula using open source communities. *Journal of Interactive Learning Research*, Association for the Advancement of Computing in Education (AACE), v. 27, n. 1, p. 5–26, 2016.

BRAGA, J. Objetos de aprendizagem, volume 1: Introdução e fundamentos. santo andré: Ufabc. *2015a). http://pesquisa. ufabc. edu. br/intera/wp-content/uploads/2015/11/ObjetosDeAprendizagemVol1_Braga. pdf*, p. 157, 2014.

BRANDÃO, H. P.; BAHRY, C. P. Gestão por competências: métodos e técnicas para mapeamento de competências. *Revista do Serviço Público*, Escola Nacional de Administração Pública-ENAP, v. 56, n. 2, p. 179, 2005.

BRASIL, G. do. Decreto n 9.005, de 14 de março de 2017. *Diário Oficial, Brasília, DF, 14 de Mar.*, 2017. Seção1. p.1.

BRITO, M. S. et al. Floss in software engineering education: an update of a systematic mapping study. In: *Proceedings of the XXXII Brazilian Symposium on Software Engineering*. [S.l.: s.n.], 2018. p. 250–259.

BUCHTA, J. et al. Teaching evolution of open-source projects in software engineering courses. In: IEEE. *Software Maintenance, 2006. ICSM'06. 22nd IEEE International Conference on*. [S.l.], 2006. p. 136–144.

BUFFARDI, K. Localized open source collaboration in software engineering education. In: IEEE. *2015 IEEE Frontiers in Education Conference (FIE)*. [S.l.], 2015. p. 1–5.

BUFFARDI, K. Localized open source software projects: Exploring realism and motivation. In: IEEE. *2016 11th International Conference on Computer Science & Education (ICCSE)*. [S.l.], 2016. p. 382–387.

CÂMARA, R. H. Análise de conteúdo: da teoria à prática em pesquisas sociais aplicadas às organizações. *Gerais: Revista Interinstitucional de Psicologia*, Universidade Federal de Minas Gerais, v. 6, n. 2, p. 179–191, 2013.

CAMARGO, V. L. S.; FABRI, J. A. *Competências de estudantes de Engenharia de software mapeadas através de projeto piloto de Fábrica Acadêmica de Software-FAS*. 2006.

CASTELLUCCIA, D.; VISAGGIO, G. Teaching evidence-based software engineering: learning by a collaborative mapping study of open source software. *ACM SIGSOFT Software Engineering Notes*, ACM New York, NY, USA, v. 38, n. 6, p. 1–4, 2013.

CERONE, A.; SOWE, S. K. Using free/libre open source software projects as e-learning tools. *Electronic Communications of the EASST*, v. 33, 2010.

CHAVEZ, C. et al. Free/libre/open source software development in software engineering education: Opportunities and experiences. *Fórum de Educaçao em Engenharia de Software (CBSoft'11-SBES-FEES)*, 2011.

CHEN, Z.; MEMON, A.; LUO, B. Combining research and education of software testing: a preliminary study. In: *Proceedings of the 29th Annual ACM Symposium on Applied Computing*. [S.l.: s.n.], 2014. p. 1179–1180.

CICIRELLO, V. A. Student developed computer science educational tools as software engineering course projects. *Journal of Computing Sciences in Colleges*, v. 32, n. 3, p. 55–61, 2017.

COMMISSION, I. O. for S. E. et al. Iso/iec 12207 systems and software engineering–software life cycle processes. *Geneve: ISO*, 2008.

CRESWELL, J. Projeto de pesquisa: métodos quali, quanti e misto. *POA: Bookman*, 2007.

CUKIERMAN, H. L.; TEIXEIRA, C.; PRIKLADNICKI, R. Um olhar sociotécnico sobre a engenharia de software. *Revista de Informática Teórica e Aplicada*, v. 14, n. 2, p. 199–219, 2007.

CURRICULA, C. Computer science, final report, the joint task force on computing curricula. *IEEE Computer Society and Association for Computing Machinery, IEEE Computer Society*, 2001.

DAL-FARRA, R. A.; LOPES, P. T. C. Métodos mistos de pesquisa em educação: pressupostos teóricos. *Nuances: estudos sobre Educação*, v. 24, n. 3, p. 67–80, 2013.

DERMEVAL, D.; COELHO, J.; BITTENCOURT, I. I. Mapeamento sistemático e revisão sistemática da literatura em informática na educação. *JAQUES, Patrícia Augustin; SIQUEIRA; Sean; BITTENCOURT, Ig; PIMENTEL, Mariano.(Org.) Metodologia de Pesquisa Científica em Informática na Educação: Abordagem Quantitativa. Porto Alegre: SBC*, 2020.

DEURSEN, A. V. et al. A collaborative approach to teaching software architecture. In: *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*. [S.l.: s.n.], 2017. p. 591–596.

DING, Y. et al. Application of software visualization in programming teaching. In: IEEE. *2014 9th International Conference on Computer Science & Education*. [S.l.], 2014. p. 803–806.

DINIZ, G. C. et al. Using gamification to orient and motivate students to contribute to oss projects. In: IEEE. *2017 IEEE/ACM 10th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)*. [S.l.], 2017. p. 36–42.

DORODCHI, M.; DEHBOZORGI, N. Utilizing open source software in teaching practice-based software engineering courses. In: IEEE. *2016 IEEE frontiers in education conference (FIE)*. [S.l.], 2016. p. 1–5.

DRAFT, S. *Computer Science Curricula 2013*. [S.l.]: ACM and IEEE Computer Society, Incorporated: New York, NY, USA, 2013.

DUARTE-FILHO, N. F. et al. Semes: Um sistema educacional móvel para o ensino de engenharia de software. *RENOTE*, v. 13, n. 1, 2015.

EDUCAÇÃO-MEC, C. N. de. Resolução 5 de 16 de novembro de 2016. *Diário Oficial, Brasília, DF.17 de Nov.*, 2016. Seção 1. p. 22.

ELLIS, H. J. et al. Towards a model of faculty development for foss in education. In: IEEE. *2013 26th International Conference on Software Engineering Education and Training (CSEE&T)*. [S.l.], 2013. p. 269–273.

ELLIS, H. J. et al. Team project experiences in humanitarian free and open source software (hfoss). *ACM Transactions on Computing Education (TOCE)*, ACM New York, NY, USA, v. 15, n. 4, p. 1–23, 2015.

ELLIS, H. J. et al. Software engineering learning in hfoss: A multi-institutional study. In: *2015 ASEE Annual Conference & Exposition*. [S.l.: s.n.], 2015. p. 26–1379.

ELLIS, H. J. et al. Learning within a professional environment: shared ownership of an hfoss project. In: *Proceedings of the 15th Annual Conference on Information technology education*. [S.l.: s.n.], 2014. p. 95–100.

ELLIS, H. J.; PURCELL, M.; HISLOP, G. W. An approach for evaluating foss projects for student participation. In: ACM. *Proceedings of the 43rd ACM technical symposium on Computer Science Education*. [S.l.], 2012. p. 415–420.

FALBO, R. de A. Mapeamento sistemático. *Retrieved October*, v. 7, 2018.

FERINO, S. et al. *Overcoming Challenges in DevOps Education through Teaching Methods*. 2023. Disponível em: <https://doi.org/10.48550/arXiv.2302.05564>.

FERNANDES, S.; BARBOSA, L. S. Collaborative environments in software engineering teaching: A floss approach. In: ACADEMIC CONFERENCES INTERNATIONAL LIMITED. *European Conference on e-Learning*. [S.l.], 2016. p. 201.

FERNANDES, S. et al. Integrating formal and informal learning through a floss-based innovative approach. In: SPRINGER. *International Conference on Collaboration and Technology*. [S.l.], 2013. p. 208–214.

FERRAZ, A.; BELHOT, R. V. et al. Taxonomia de bloom: revisão teórica e apresentação das adequações do instrumento para definição de objetivos instrucionais. *Gest. Prod., São Carlos*, SciELO Brasil, v. 17, n. 2, p. 421–431, 2010.

FERREIRA, M. et al. Introdução e condução dos métodos mistos de pesquisa em educação física. *Pensar a Prática*, v. 23, 2020.

FIGUEIREDO, R. M. d. C. et al. Graduação em engenharia de software: uma proposta de flexibilização e interdisciplinaridade. *III Fórum de Educação em Engenharia de Software (FEES)*, 2010.

FILHO, G. P.; FF, S. I., gerosa, ma: Training software engineers using open-source software: the professors' perspective. In: *2017 IEEE 30th Conference on Software Engineering Education and Training (CSEET)*. [S.l.: s.n.], 2017. p. 117–121.

FRANCO, M. L. P. B. *Análise de conteúdo*. [S.l.]: Autores Associados, 2020.

GALHARDI, A. C.; AZEVEDO, M. d. Avaliações de aprendizagem: o uso da taxonomia de bloom. In: *Workshop de Pós-Graduaçao e Pesquisa do Centro Paula Souza*. [S.l.: s.n.], 2013. v. 8.

GIL, A. C. et al. *Como elaborar projetos de pesquisa*. [S.l.]: Atlas São Paulo, 2002.

GOKHALE, S.; SMITH, T.; MCCARTNEY, R. Teaching software engineering from a maintenance-centric view using open-source software. *Journal of Computing Sciences in Colleges*, Consortium for Computing Sciences in Colleges, v. 28, n. 6, p. 189–191, 2013.

GOKHALE, S.; SMITH, T.; MCCARTNEY, R. Teaching software maintenance with open source software: Experiences and lessons. In: IEEE. *2013 IEEE Frontiers in Education Conference (FIE)*. [S.l.], 2013. p. 1664–1670.

GORGONE, J. et al. *Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems*. 2002. Disponível em: <https://www.acm.org/binaries/content/assets/education/curricula-recommendations/is2002.pdf>. Acesso em: 04 de fevereiro de 2017.

HISLOP, G. W.; ELLIS, H. J. Humanitarian open source software in computing education. *Computer*, IEEE, v. 50, n. 10, p. 98–101, 2017.

HISLOP, G. W. et al. A multi-institutional study of learning via student involvement in humanitarian free and open source software projects. In: *Proceedings of the eleventh annual International Conference on International Computing Education Research*. [S.l.: s.n.], 2015. p. 199–206.

IEEE. *Instituto de Engenheiros Eletricistas e Eletrônicos*. 2019. Disponível em: <https://www.IEEE.org/>. Acesso em: 15 de agosto de 2019.

IMPAGLIAZZO, J. et al. *Computer Curricula 2016:Curriculum Guidelines for UNdergraduate Degree Programas in Computer*. 2016. Disponível em: <https://www.acm.org/binaries/content/assets/education/ce2016-final-report.pdf>. Acesso em: 04 de fevereiro de 2017.

INFOCOMP(2012). *Information Technology Competency Model, Employment and Training Administration, United States Department of Labor*. 2012. Disponível em:<https://www.careeronestop.org/competencymodel/pyramid_download.aspx?IT=Y>. Acesso em: 10 de fevereiro de 2019.

JACCHERI, L.; OSTERLIE, T. Open source software: A source of possibilities for software engineering education and empirical software engineering. In: IEEE. *Emerging Trends in FLOSS Research and Development, 2007. FLOSS'07. First International Workshop on*. [S.l.], 2007. p. 5–5.

JONATHAN, M. Um breve histórico da formação em computação no brasil. *Anais doScientiarum História VI: filosofia, ciências e artes, conexões interdisciplinares. Rio de Janeiro: UFRJ*, 2013.

KON, F. et al. Free and open source software development and research: Opportunities for software engineering. In: IEEE. *2011 25th Brazilian Symposium on Software Engineering*. [S.l.], 2011. p. 82–91.

KRISHNAMOORTHY, V.; APPASAMY, B.; SCAFFIDI, C. Using intelligent tutors to teach students how apis are used for software engineering in practice. *IEEE Transactions on Education*, IEEE, v. 56, n. 3, p. 355–363, 2013.

KRUTZ, D. E.; MALACHOWSKY, S. A.; REICHLMAYR, T. Using a real world project in a software testing course. In: *Proceedings of the 45th ACM technical symposium on Computer science education*. [S.l.: s.n.], 2014. p. 49–54.

KUSSMAUL, C. Experience report: Guiding faculty & students to participate in humanitarian foss communities. In: IEEE. *2016 IEEE Eighth International Conference on Technology for Education (T4E)*. [S.l.], 2016. p. 224–227.

KUSSMAUL, C. L. et al. Board# 77: Helping faculty & students to participate in humanitarian free & open source software: The openfe & openpath projects. In: *2017 ASEE Annual Conference & Exposition*. [S.l.: s.n.], 2017.

LEBLANC, R. J. et al. *Software Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering*. [S.l.]: IEEE Computer Society, 2006.

LESSA, M. S. B.; CHAVEZ, C. von F. G. An approach for selecting floss projects for education. In: *Proceedings of the 34th Brazilian Symposium on Software Engineering*. [S.l.: s.n.], 2020. p. 463–472.

LESSA, R. O.; JUNIOR, E. O. lessa. Modelos de processos de engenharia de software. *Link para o PDF: http://xps-project. googlecode. com/svn-history/r43/trunk/outros/02_-Artigo. pdf*, 2009.

LOBO, A. S. M.; MAIA, L. C. G. O uso das tics como ferramenta de ensino-aprendizagem no ensino superior. *Caderno de Geografia*, Pontifícia Universidade Católica de Minas Gerais, v. 25, n. 44, p. 16–26, 2015.

LUNT, B. M. et al. Curriculum guidelines for undergraduate degree programs in information technology. *Retrieved March*, v. 2, n. 2009, 2008.

MACKELLAR, B. K.; SABIN, M.; TUCKER, A. B. Bridging the academia-industry gap in software engineering: A client-oriented open source software projects course. In: *Open Source Technology: Concepts, Methodologies, Tools, and Applications*. [S.l.]: IGI Global, 2015. p. 1927–1950.

MATOS, E. de S. Identidade profissional docente e o papel da interdisciplinaridade no currículo de licenciatura em computação. *Revista Espaço Acadêmico*, v. 13, n. 148, p. 26–34, 2013.

MEC. Conselho nacional de educação. RESOLUÇÃO Nº 2, DE 1º DE JULHO DE 2015.Define as Diretrizes Curriculares Nacionais para a formação inicial em nível superior (cursos de licenciatura, cursos de formação pedagógica para graduados e cursos de segunda licenciatura) e para a formação continuada. *Diário Oficial, Brasília, DF.02 de Jul.*, 2015. Seção 1. p. 8.

MENDES, R. M.; MISKULIN, R. G. S. A análise de conteúdo como uma metodologia. *Cadernos de Pesquisa*, SciELO Brasil, v. 47, p. 1044–1066, 2017.

MISHRA, D.; HACALOGLU, T.; MISHRA, A. Teaching software verification and validation course: A case study. *International Journal of Engineering Education*, v. 30, n. 6, 2014.

MORGAN, B.; JENSEN, C. Lessons learned from teaching open source software development. In: SPRINGER. *IFIP International Conference on Open Source Systems*. [S.l.], 2014. p. 133–142.

MUSSOI, E. M.; FLORES, M. L. P.; BEHAR, P. A. Avaliação de objetos de aprendizagem. In: *Congresso Iberoamericano de Informática Educativa, Santiago, Chile. Anais.[Google Scholar]*. [S.l.: s.n.], 2010.

NASCIMENTO, D. M.; BITTENCOURT, R. A.; CHAVEZ, C. Open source projects in software engineering education: a mapping study. *Computer Science Education*, Taylor & Francis, v. 25, n. 1, p. 67–114, 2015.

NASCIMENTO, D. M. C. Educacão em engenharia de software com a adoção de projetos de código aberto: uma análise detalhada. Universidade Federal da Bahia, 2017.

NUNES, P.; YARMAGUTI, M.; NUNES, I. Refinamento de competências do egresso do curso de engenharia de software. *Fórum de Educação em Engenharia de Software,*, FEES, v. 2016, n. IX, p. 143–155, 2016.

OLIVEIRA, M.; OLIVEIRA, S. R. B.; MEIRA, S. Condução de uma fábrica de software e o processo de aprendizagem em cursos de graduação de ti: Uma aplicação de um survey sobre a percepção da importância. In: *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*. [S.l.: s.n.], 2017. v. 28, n. 1, p. 92.

PAPADOPOULOS, P.; STAMELOS, I.; MEISZNER, A. *Enhancing software engineering education through open source projects: Four years of students' perspectives. EAIT 18 (2), 381 397 (2013)*. 2013.

PORTELA, C. S.; VASCONCELOS, A. M.; OLIVEIRA, S. R. Análise da relevância dos tópicos e da efetividade das abordagens para o ensino de engenharia de software. In: *Fórum de Educação em Engenharia de Software (FEES). In VI Congresso Brasileiro de Software: Teoria e Prática (CBSoft)*. [S.l.: s.n.], 2015.

PORTELA, C. S.; VASCONCELOS, A. M.; OLIVEIRA, S. R. Frames: Um framework para o ensino-aprendizagem dos tópicos de engenharia de software dos currículos de referência da acm/ieee e sbc. In: *Fórum de Educação em Engenharia de Software (FEES). In VII Congresso Brasileiro de Software: Teoria e Prática (CBSoft)*. [S.l.: s.n.], 2016.

PRESSMAN, R.; MAXIM, B. *Engenharia de Software-8^a Edição*. [S.l.]: McGraw Hill Brasil, 2016.

PRIKLADNICKI, R. et al. Ensino de engenharia de software: desafios, estratégias de ensino e lições aprendidas. *Anais do II Fórum de Educação em Engenharia de Software*, 2009.

PUGLISI, M. L.; FRANCO, B. Análise de conteúdo. Brasília: Líber Livro, 2005.

REATEGUI, E.; FINCO, M. D. Proposta de diretrizes para avaliação de objetos de aprendizagem considerando aspectos pedagógicos e técnicos. *RENOTE*, v. 8, n. 3, 2010.

ROCHA, F. G.; SABINO, R. F.; ACIPRESTE, R. H. L. A metodologia scrum como mobilizadora da prática pedagógica: Um olhar sobre a engenharia de software. *FEES 2015*, p. 13, 2015.

ROCHA, M. d. G. B. et al. Currículo de referência da sbc para cursos de graduação em bacharelado em ciência da computação e engenharia de computação. *Relatório Técnico. Acesso em 04 de Abril de 2017*, v. 8, 2005.

RODRIGUES, M. U. Potencialidades do pibid como espaço formativo para professores de matemática no brasil. Universidade Estadual Paulista (UNESP), 2016.

SABIN, M. et al. *Curriculum guidelines for undergraduate degree programs in information technology*. 2017. Disponível em: <https://www.acm.org/binaries/content/assets/education/curricula-recommendations/it2017.pdf>. Acesso em: 04 de fevereiro de 2017.

SANTOS, R. et al. *Uma Estratégia para Apoiar a Pesquisa em Educação em Engenharia de Software no Brasil*. 2009.

SANTOS, R. et al. Portal edues brasil: Um ambiente para apoiar a pesquisa em educação em engenharia de software no brasil. *Anais do II Fórum de Educação em Engenharia de Software*, p. 33–40, 2009.

SANTOS, R. E. et al. Ferramentas, métodos e experiências no ensino de engenharia de software: um mapeamento sistemático. In: *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*. [S.l.: s.n.], 2014. v. 25, n. 1, p. 544.

SANTOS, S. C. dos et al. Usando pbl na qualificação de profissionais em engenharia de software. *Anais do FEES08-Fórum de Educação em Engenharia de Software*, 2008.

SAVI, R.; WANGENHEIM, C. G. von; BORGATTO, A. F. A model for the evaluation of educational games for teaching software engineering. In: IEEE. *Software Engineering (SBES), 2011 25th Brazilian Symposium on*. [S.l.], 2011. p. 194–203.

SBC. *Currículo de Referência da SBC para Cursos de Graduação Plena em Computação 1991. Diretoria de Educação, Sociedade Brasileira de Computação, junho de 1991*. 1991. Disponível em: <https://homepages.dcc.ufmg.br/~bigonha/Cr/cr91.html>. Acesso em: 15 de agosto de 2019.

SBC. *Curriculo de referencia da Sociedade Brasileira de Computação - SBC para Cursos de Graduacao em Computacao, versao 1999*. 1999.

SHACKELFORD, R. et al. *Computing Curricula 2005:The Overview Report*. 2005. Disponível em: <https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2005-march06final.pdf>. Acesso em: 04 de fevereiro de 2019.

SHACKELFORD, R. et al. Computing curricula 2005: The overview report. In: ACM. *ACM SIGCSE Bulletin*. [S.l.], 2006. v. 38, n. 1, p. 456–457.

SILVA, J. F. da. Avaliação do ensino e da aprendizagem numa perspectiva formativa reguladora. 2003.

SMITH, T.; GOKHALE, S.; MCCARTNEY, R. Understanding students' preferences of software engineering projects. In: *Proceedings of the 2014 conference on Innovation & technology in computer science education*. [S.l.: s.n.], 2014. p. 135–140.

SMITH, T. M. et al. Selecting open source software projects to teach software engineering. In: *Proceedings of the 45th ACM technical symposium on Computer science education*. [S.l.: s.n.], 2014. p. 397–402.

SOUZA, R. S. d.; DIESEL, V. Metodologia da pesquisa. Brasil, 2008.

SOWE, S. K.; STAMELOS, I. G. Involving software engineering students in open source software projects: Experiences from a pilot study. *Journal of Information Systems Education*, EDSIG, v. 18, n. 4, p. 425, 2007.

SOWMYA, B.; HIRIYANNAIAH, S.; SRINIVASA, K. A case study based software engineering education using open source tools. In: *QuASoQ/WAWSE/CMCE@ APSEC*. [S.l.: s.n.], 2015. p. 79–88.

STAMELOS, I. G. Teaching software engineering with free/libre open source projects. *Multi-disciplinary advancement in open source software and processes*, IGI Global, p. 67–84, 2011.

STANDARDIZATION, I. *ISO/IEC 15288, Systems and software engineering-System life cycle processes*. [S.l.]: IEEE, 2008.

TAVARES, R.; CARVALHO, C. O mapa conceitual hierárquico e a taxonomia de bloom modificada. *Encontro Internacional Nacional de Aprendizagem Significativa*, v. 6, 2010.

TEIXEIRA, C. A. N.; CUKIERMAN, H. Apontamentos para enriquecer o perfil do engenheiro de software. In: *Congresso da Sociedade Brasileira de Computação*. [S.l.: s.n.], 2005. v. 25, p. 2005.

TOPI, H. et al. Curriculum guidelines for undergraduate degree programs in information systems. *ACM/AIS task force*, 2010.

VILLARRUBIA, A.; KIM, H. Building a community system to teach collaborative software development. In: IEEE. *2015 10th International Conference on Computer Science & Education (ICCSE)*. [S.l.], 2015. p. 829–833.

WANGENHEIM, C. G. von; SILVA, D. Qual conhecimento de engenharia de software é importante para um profissional de software? *Proceedings of the Fórum de Educação em Engenharia de Software*, v. 2, p. 1–8, 2009.

YAMAMOTO, F. S. et al. Interdisciplinaridade no ensino de ciência da computação. In: *Anais do XXV Congresso da SBC, Unisinos, São Leopoldo, RS*. [S.l.: s.n.], 2005.

ZORZO et al. *Referenciais de Formação para os Cursos de Graduação em Computação*. [S.l.]: Sociedade Brasileira de Computação (SBC). 153p, 2017. ISBN 9788576694243.

Apêndice

A

Aqui são demonstradas todas as unidades de contexto e unidades de registro que estruturam a codificação dos Referenciais Nacionais

ANÁLISE DE CONTEÚDO DOS RFS DA SBC

Tabela A.1 Tabela descritiva de Exploração do RF-CC

Eixo de Formação	Unidades de Contexto (Excertos extraídos do Referencial de Formação do RF-CC)	Unidades de Registro
Eixo - 2	(C.2.2 .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 causados (CG-IV).) >Criar >Engenharia de Software	- Competência Geral IV - - Engenharia de Software
	C.2.3 Avaliar criticamente projetos de sistemas de computação (CG-VIII) >Avaliar >Engenharia de Software	- Competência Geral VIII - - Engenharia de Software
	C.2.7. Identificar e analisar requisitos e especificações para problemas específicos e planejar estratégias para suas soluções (CE-IV) >Criar >Engenharia de Software	- Competência Específica IV - - Engenharia de Sofware
	C.2.8 Empregar metodologias que visem garantir critérios de qualidade ao longo de todas as etapas de desenvolvimento de uma solução computacional CE-VII > Aplicar >Engenharia de Software	- Competência Específica VII - - Engenharia de Software
	C.2.9. Analisar quanto um sistema baseado em computadores atende os critérios definidos para seu uso corrente e futuro (adequabilidade) (CE-VIII) > >Avaliar >Engenharia de Software	- Competência Específica VIII - - Engenharia de Software
	CE-V. Especificar, projetar, implementar, manter e avaliar sistemas de computação, empregando teorias, práticas e ferramentas adequadas. >Competência de Eixo 2 >CG-IV, CG-VIII, CE-IV, CE-VII, CE-VIII > Engenharia de Software.	- Competência Específica V - - Competências Gerais VIII e IV - - Competências Específicas IV, CE VIII e CE VII - - Engenharia de Software
Eixo - 3	C.3.8. Identificar e analisar requisitos e especificações para problemas específicos e planejar estratégias para suas soluções (CE-IV) >Analisar >Engenharia de Software	- Competência Específica IV - - Engenharia de Software
	C.3.9. Analisar quanto um sistema baseado em computadores atende os critérios definidos para seu uso corrente e futuro (adequabilidade) (CE-VIII) >Analisar > Engenharia de Software	- Competência Específica VIII - - Engenharia de Software
	CG-IX. Adequar-se rapidamente às mudanças tecnológicas e aos novos ambientes de trabalho >Competências de Eixo 3>CE-IV, CE-VIII > Engenharia de Software	- Competência Geral IX - - Competência Específica IV e VIII - - Engenharia de Software
Eixo 4	C.4.3. Preparar e apresentar seus trabalhos e problemas técnicos e suas soluções para audiências diversas, em formatos apropriados (oral e escrito) (CG-VII) >Aplicar > Engenharia de Software	- Competência Geral VII - Engenharia de Software
	C.4.8. Empregar metodologias que visem garantir critérios de qualidade ao longo de todas as etapas de desenvolvimento de uma solução computacional (CE-VII) > Aplicar>Engenharia de Software	- Competência Específica VII - - Engenharia de Software
Eixo 5	C.5.8. Identificar e analisar requisitos e especificações para problemas específicos e planejar estratégias para suas soluções (CE-IV) >Criar > Engenharia de Software	- Competência Específica IV - - Engenharia de Software
Eixo 6	C.6.5. Ser capaz de realizar trabalho cooperativo e entender os benefícios que este pode produzir (CG-XII) >Aplicar > Aplicável a todos os conteúdos, utilizando práticas pedagógicas colaborativas.	- Competência Geral XII - - Conteúdo Aplicado por práticas Colaborativas.
Eixo 7	C.7.6. Identificar e analisar requisitos e especificações para problemas específicos e planejar estratégias para suas soluções (CE-IV) >Aplicar > Engenharia de Software.	- Competência Específica IV - - Engenharia de Software
	CG-IX. Adequar-se rapidamente às mudanças tecnológicas e aos novos ambientes de trabalho >Competência de Eixo 7>CE-IV >Engenharia de Software.	- Competência Geral IX - - Engenharia de Software

Fonte: Elaboração Própria

Tabela A.2 Tabela descritiva de Exploração do RF-EC

Eixo de Formação	Unidades de Contexto (Excertos extraídos do Referencial de Formação do RF-EC)	Unidades de Registro
Eixo - 1	Competência Específica I (DCN) >C.1.4. Criticar e escolher sistemas operacionais para contextos específicos, considerando como funcionam os principais componentes de cada sistema e os requisitos do contexto de aplicação. >Requisitos de Sistemas.	- Competência Específica I - Competência Derivada C.1.4 - Requisitos de Sistemas
	Competência Específica V (DCN) >C.1.4 Criticar e escolher sistemas operacionais para contextos específicos, considerando como funcionam os principais componentes de cada sistema e os requisitos do contexto de aplicação. > Requisitos de Sistemas.	- Competência Específica V - Competência Derivada C.1.4 - Requisitos de Sistemas
	Competência Específica VIII (DCN) >C.1.4 Criticar e escolher sistemas operacionais para contextos específicos, considerando como funcionam os principais componentes de cada sistema e os requisitos do contexto de aplicação. > Requisitos de Sistemas.	- Competência Específica VIII - Competência Derivada C.1.4 - Requisitos de Sistemas
	Competência Específica IX (DCN) >C.1.4 Criticar e escolher sistemas operacionais para contextos específicos, considerando como funcionam os principais componentes de cada sistema e os requisitos do contexto de aplicação. > Requisitos de Sistemas.	- Competência Específica IX - Competência Derivada C.1.4 - Requisitos de Sistemas
Eixo 2	Competência Específica II (DCN) >C.2.2 Especificar e validar os requisitos, projetar, implementar, verificar, implantar e documentar soluções de software baseadas no conhecimento apropriado de teorias, modelos e técnicas. >Engenharia de Software	- Competência Específica II - Competência Derivada C.2.2 - Engenharia de Software
	Competência Específica VII (DCN) >C.2.2 Especificar e validar os requisitos, projetar, implementar, verificar, implantar e documentar soluções de software baseadas no conhecimento apropriado de teorias, modelos e técnicas. >Engenharia de Software	- Competência Específica VII - Competência Derivada C.2.2 - Engenharia de Software
Eixo - 3	Competência Geral VII (DCN) >C.3.4. Aplicar metodologias de gestão de projetos, serviços e experimentos de engenharia na área de computação > Ciclo de Vida de produtos de Software e Hardware, Técnicas para Especificação de Requisitos	- Competência Geral VII - Competência Derivada C.3.4 - Ciclo de vida de produtos de software e Hardware - Técnicas para especificação de Requisitos
	Competência Geral VIII (DCN) >C.3.4. Aplicar metodologias de gestão de projetos, serviços e experimentos de engenharia na área de computação > Ciclo de Vida de produtos de Software e Hardware, Técnicas para Especificação de Requisitos	- Competência Geral VIII - Competência Derivada C.3.4 - Ciclo de vida de produtos de software e Hardware - Técnicas para especificação de Requisitos
	Competência Geral VIII >C.3.3. Identificar normas e documentações técnicas necessárias em projetos, serviços e experimentos de Engenharia de Computação > Documentação técnica em projetos de hardware e software	- Competência Geral VIII - Competência Derivada C.3.3 - Documentação técnica em projetos de Hardware e Software
	Competência Geral IX (DCN) >C.3.4. Aplicar metodologias de gestão de projetos, serviços e experimentos de engenharia na área de computação > Ciclo de Vida de produtos de Software e Hardware, Técnicas para Especificação de Requisitos	- Competência Geral VIII - Competência Derivada C.3.4 - Ciclo de vida de produtos de software e Hardware - Técnicas para especificação de Requisitos
	Competência Geral IX >C.3.3. Identificar normas e documentações técnicas necessárias em projetos, serviços e experimentos de Engenharia de Computação > Documentação técnica em projetos de hardware e software	- Competência Geral IX - Competência Derivada C.3.3 - Documentação técnica em projetos de Hardware e Software

Eixo de Formação	Unidades de Contexto (Excertos extraídos do Referencial de Formação do RF-EC) Continuação	Unidades de Registro
Eixo - 3	Competência Geral X (DCN) >C.3.4. Aplicar metodologias de gestão de projetos, serviços e experimentos de engenharia na área de computação > Ciclo de Vida de produtos de Software e Hardware, Técnicas para Especificação de Requisitos	- Competência Geral X - Competência Derivada C.3.4 - Ciclo de vida de produtos de software e Hardware - Técnicas para especificação de Requisitos
	Competência Geral X >C.3.3. Identificar normas e documentações técnicas necessárias em projetos, serviços e experimentos de Engenharia de Computação > Documentação técnica em projetos de hardware e software	- Competência Geral X - Competência Derivada C.3.3 - Documentação técnica em projetos de Hardware e Software
	Competência Geral XI (DCN) >C.3.4. Aplicar metodologias de gestão de projetos, serviços e experimentos de engenharia na área de computação > Ciclo de Vida de produtos de Software e Hardware, Técnicas para Especificação de Requisitos	- Competência Geral XI - Competência Derivada C.3.4 - Ciclo de vida de produtos de software e Hardware - Técnicas para especificação de Requisitos
	Competência Geral XII (DCN) >C.3.4. Aplicar metodologias de gestão de projetos, serviços e experimentos de engenharia na área de computação > Ciclo de Vida de produtos de Software e Hardware, Técnicas para Especificação de Requisitos	- Competência Geral XII - Competência Derivada C.3.4 - Ciclo de vida de produtos de software e Hardware - Técnicas para especificação de Requisitos
	Competência Geral XII >C.3.3. Identificar normas e documentações técnicas necessárias em projetos, serviços e experimentos de Engenharia de Computação > Documentação técnica em projetos de hardware e software	- Competência Geral XII - Competência Derivada C.3.3 - Documentação técnica em projetos de Hardware e Software
	Competência Específica I (DCN) >C.3.4. Aplicar metodologias de gestão de projetos, serviços e experimentos de engenharia na área de computação > Ciclo de Vida de produtos de Software e Hardware, Técnicas para Especificação de Requisitos	- Competência Específica I - Competência Derivada C.3.4 - Ciclo de vida de produtos de software e Hardware - Técnicas para especificação de Requisitos
	Competência Específica III (DCN) >C.3.4. Aplicar metodologias de gestão de projetos, serviços e experimentos de engenharia na área de computação > Ciclo de Vida de produtos de Software e Hardware, Técnicas para Especificação de Requisitos	- Competência Específica III - Competência Derivada C.3.4 - Ciclo de vida de produtos de software e Hardware - Técnicas para especificação de Requisitos
	Competência Específica III (DCN) >C.3.1. Compreender conceitos relevantes sobre projetos, serviços e experimentos de engenharia na área de computação.>Erros , Motivos de fracasso e Riscos envolvidos no projeto de sistemas de software e hardware	- Competência Específica III - Competência Derivada C.3.1 - Erros, Motivos de fracasso e Riscos envolvidos no projeto de sistemas de software/hardware.
	Competência Específica V (DCN >C.3.3. Identificar normas e documentações técnicas necessárias em projetos, serviços e experimentos de Engenharia de Computação > Documentação técnica em projetos de hardware e software	- Competência Específica V - Competência Derivada C.3.3 - Documentação técnica em projetos de Hardware e Software
	Competência Específica V (DCN) >C.3.4. Aplicar metodologias de gestão de projetos, serviços e experimentos de engenharia na área de computação > Ciclo de Vida de produtos de Software e Hardware, Técnicas para Especificação de Requisitos	- Competência Específica V - Competência Derivada C.3.4 - Ciclo de vida de produtos de software e Hardware - Técnicas para especificação de Requisitos

Eixo de Formação	**Unidades de Contexto (Excertos extraídos do Referencial de Formação do RF-EC) Continuação**	**Unidades de Registro**
Eixo - 3	Competência Específica VII (DCN >C.3.3. Identificar normas e documentações técnicas necessárias em projetos, serviços e experimentos de Engenharia de Computação > Documentação técnica em projetos de hardware e software	- Competência Específica VII - Competência Derivada C.3.3 - Documentação técnica em projetos de Hardware e Software
	Competência Específica VII (DCN) >C.3.4. Aplicar metodologias de gestão de projetos, serviços e experimentos de engenharia na área de computação > Ciclo de Vida de produtos de Software e Hardware, Técnicas para Especificação de Requisitos	- Competência Específica VII - Competência Derivada C.3.4 - Ciclo de vida de produtos de software e Hardware - Técnicas para especificação de Requisitos
	Competência Específica X (DCN >C.3.3. Identificar normas e documentações técnicas necessárias em projetos, serviços e experimentos de Engenharia de Computação > Documentação técnica em projetos de hardware e software	- Competência Específica X - Competência Derivada C.3.3 - Documentação técnica em projetos de Hardware e Software
	Competência Específica X (DCN) >C.3.4. Aplicar metodologias de gestão de projetos, serviços e experimentos de engenharia na área de computação > Ciclo de Vida de produtos de Software e Hardware, Técnicas para Especificação de Requisitos	- Competência Específica X - Competência Derivada C.3.4 - Ciclo de vida de produtos de software e Hardware - Técnicas para especificação de Requisitos

Fonte: Elaboração Própria

Tabela A.3 Tabela descritiva de Exploração do RF-SI

Eixo de Formação	**Unidades de Contexto (Excertos extraídos do Referencial de Formação do RF-SI)**	Unidades de Registro
Eixo - 3	Competência Geral I (DCN) > C.3.3. Projetar software para informatização de sistemas, determinando sua arquitetura, garantindo sua qualidade técnica e validando seu potencial de eficácia, eficiência, efetividade e sustentabilidade. >Criar > Engenharia de Requisitos, Arquitetura de Software, Projeto de Software, Verificação e Validação de Software, Qualidade de Software, Gerência de Configuração de Software.	-Competência Geral I; Competência Derivada C3.3; - Engenharia de Requisitos; - Arquitetura de Software; - Projeto de Software; Verificação e Validação de Software; - Qualidade de Software; - Gerência de Configuração de Software;
	Competência Geral I (DCN)>C.3.4. Construir software para informatização de sistemas avaliando sua qualidade técnica, testando o seu funcionamento, e validando seu atendimento às necessidades de eficácia, eficiência, efetividade e sustentabilidade desses sistemas. >Criar > Projeto de Software, Verificação e Validação de Software, Qualidade de Software, Gerência de Configuração de Software, Teste de Software.	- Competência Geral I - Competência Derivada C.3.4 - Projeto de Software - Verificação e Validação de Software - Qualidade de Software - Gerência de Configuração de Software, - Teste de Software.
	Competência Geral II (DCN)>C.3.4. Construir software para informatização de sistemas avaliando sua qualidade técnica, testando o seu funcionamento, e validando seu atendimento às necessidades de eficácia, eficiência, efetividade e sustentabilidade desses sistemas. >Criar > Projeto de Software, Verificação e Validação de Software, Qualidade de Software, Gerência de Configuração de Software, Teste de Software.	- Competência Geral II - Competência Derivada C.3.4 - Projeto de Software - Verificação e Validação de Software - Qualidade de Software - Gerência de Configuração de Software, - Teste de Software.
	Competência Geral III (DCN) >C.3.4. Construir software para informatização de sistemas avaliando sua qualidade técnica, testando o seu funcionamento, e validando seu atendimento às necessidades de eficácia, eficiência, efetividade e sustentabilidade desses sistemas. >Criar > Projeto de Software, Verificação e Validação de Software, Qualidade de Software, Gerência de Configuração de Software, Teste de Software.	- Competência Geral III - Competência Derivada C.3.4 - Projeto de Software - Verificação e Validação de Software - Qualidade de Software - Gerência de Configuração de Software, - Teste de Software.
	Competência Geral IV (DCN) >C.3.1. Avaliar as necessidades de informatizar sistemas, articulando visões individuais e organizacionais, e apreciando oportunidades de melhorias e/ou mudanças em processos, com o uso ou evolução do software. > Avaliar >Engenharia de Requisitos de Sistemas	- Competência Geral IV - Competência Derivada C.3.1 - Engenharia de Requisitos de Sistemas.
	Competência Específica I (DCN) >C.3.1. Avaliar as necessidades de informatizar sistemas, articulando visões individuais e organizacionais, e apreciando oportunidades de melhorias e/ou mudanças em processos, com o uso ou evolução do software. > Avaliar >Engenharia de Requisitos de Sistemas	- Competência Específica I; - Competência Derivada C.3.1; - Engenharia de Requisitos de Sistemas;
	Competência Específica I (DCN) >C.3.2. Especificar software para informatização de sistemas, elicitando os requisitos do software em conformidade com os requisitos do produto, dos processos e das partes interessadas, analisando e especificando seus requisitos funcionais e não funcionais e validando o seu potencial de solução das necessidades de sistemas de informação. >Criar > Engenharia de Requisitos de Sistemas, Qualidade de Software , Gerência de Configuração de Software.	- Competência Específica I; - Competência Derivada C.3.2; - Engenharia de Requisitos de Sistemas; - Qualidade de Software ; - Gerência de Configuração de Software.
	Competência Geral I (DCN) >C.3.5. Implantar software para informatização de sistemas, avaliando o impacto de seu uso. >Aplicar > Qualidade de Software, Verificação e Validação de Software.	- Competência Específica I; - Competência Derivada C.3.5; - Qualidade de Software; - Verificação e Validação de Software.
	Competência Geral I (DCN) >C.3.6 Manter software, corrigindo falhas, adaptandoo ao seu contexto, identificando e implementando melhorias, migrando softwares legados e retirando software. >Avaliar > Qualidade de Software, Gerência de Configuração de Software, Verificação e Validação de Software, Manutenção de software, Engenharia de Requisitos, Projeto de Software	- Competência Específica I; - Competência Derivada C.3.6; - Qualidade de Software; - Gerência de Configuração de Software; - Verificação e Validação de Software; - Manutenção de software; - Engenharia de Requisitos, - Projeto de Software
	Competência Específica II (DCN) >C.3.1. Avaliar as necessidades de informatizar sistemas, articulando visões individuais e organizacionais, e apreciando oportunidades de melhorias e/ou mudanças em processos, com o uso ou evolução do software. > Avaliar >Engenharia de Requisitos de Sistemas	- Competência Específica II; - Competência Derivada C.3.1; - Engenharia de Requisitos de Sistemas;
	Competência Específica II (DCN) >C.3.2. Especificar software para informatização de sistemas, elicitando os requisitos do software em conformidade com os requisitos do produto, dos processos e das partes interessadas, analisando e especificando seus requisitos funcionais e não funcionais e validando o seu potencial de solução das necessidades de sistemas de informação. >Criar > Engenharia de Requisitos de Sistemas, Qualidade de Software , Gerência de Configuração de Software.	- Competência Específica II; - Competência Derivada C.3.2; - Engenharia de Requisitos de Sistemas; - Qualidade de Software ; - Gerência de Configuração de Software.

Eixo de Formação	Unidades de Contexto (Excertos extraídos do Referencial de Formação do RF-SI) (Continuação)	Unidades de Registro
Eixo - 3	Competência Específica II (DCN) > C.3.3. Projetar software para informatização de sistemas, determinando sua arquitetura, garantindo sua qualidade técnica e validando seu potencial de eficácia, eficiência, efetividade e sustentabilidade. >Criar > Engenharia de Requisitos, Arquitetura de Software, Projeto de Software, Verificação e Validação de Software, Qualidade de Software, Gerência de Configuração de Software.	- Competência Específica II; - Competência Derivada C.3.3; - Engenharia de Requisitos; - Arquitetura de Software; - Projeto de Software; Verificação e Validação de Software; - Qualidade de Software; - Gerência de Configuração de Software;
	Competência Específica II (DCN)>C.3.4. Construir software para informatização de sistemas avaliando sua qualidade técnica, testando o seu funcionamento, e validando seu atendimento às necessidades de eficácia, eficiência, efetividade e sustentabilidade desses sistemas. >Criar > Projeto de Software, Verificação e Validação de Software, Qualidade de Software, Gerência de Configuração de Software, Teste de Software.	- Competência Específica II - Competência Derivada C.3.4 - Projeto de Software - Verificação e Validação de Software - Qualidade de Software - Gerência de Configuração de Software, - Teste de Software.
	Competência Específica II (DCN) >C.3.5. Implantar software para informatização de sistemas, avaliando o impacto de seu uso. >Aplicar > Qualidade de Software, Verificação e Validação de Software.	- Competência Específica II; - Competência Derivada C.3.5; - Qualidade de Software; - Verificação e Validação de Software.
	Competência Específica II (DCN) >C.3.6 Manter software, corrigindo falhas, adaptandoo ao seu contexto, identificando e implementando melhorias, migrando softwares legados e retirando software. >Avaliar > Qualidade de Software, Gerência de Configuração de Software, Verificação e Validação de Software, Manutenção de software, Engenharia de Requisitos, Projeto de Software	- Competência Específica II; - Competência Derivada C.3.6; - Qualidade de Software; - Gerência de Configuração de Software; - Verificação e Validação de Software; - Manutenção de software; - Engenharia de Requisitos, - Projeto de Software
	Competência Específica III (DCN) >C.3.1. Avaliar as necessidades de informatizar sistemas, articulando visões individuais e organizacionais, e apreciando oportunidades de melhorias e/ou mudanças em processos, com o uso ou evolução do software. > Avaliar >Engenharia de Requisitos de Sistemas	- Competência Específica III - Competência Derivada C.3.1 - Engenharia de Requisitos de Sistemas.
	Competência Específica III (DCN) > C.3.3. Projetar software para informatização de sistemas, determinando sua arquitetura, garantindo sua qualidade técnica e validando seu potencial de eficácia, eficiência, efetividade e sustentabilidade. >Criar > Engenharia de Requisitos, Arquitetura de Software, Projeto de Software, Verificação e Validação de Software, Qualidade de Software, Gerência de Configuração de Software.	- Competência Específica III; - Competência Derivada C.3.3; - Engenharia de Requisitos; - Arquitetura de Software; - Projeto de Software; Verificação e Validação de Software; - Qualidade de Software; - Gerência de Configuração de Software;
	Competência Específica III (DCN)>C.3.4. Construir software para informatização de sistemas avaliando sua qualidade técnica, testando o seu funcionamento, e validando seu atendimento às necessidades de eficácia, eficiência, efetividade e sustentabilidade desses sistemas. >Criar > Projeto de Software, Verificação e Validação de Software, Qualidade de Software, Gerência de Configuração de Software, Teste de Software.	- Competência Específica III - Competência Derivada C.3.4 - Projeto de Software - Verificação e Validação de Software - Qualidade de Software - Gerência de Configuração de Software, - Teste de Software.
	Competência Específica IV (DCN) >C.3.1. Avaliar as necessidades de informatizar sistemas, articulando visões individuais e organizacionais, e apreciando oportunidades de melhorias e/ou mudanças em processos, com o uso ou evolução do software. > Avaliar >Engenharia de Requisitos de Sistemas	- Competência Específica IV - Competência Derivada C.3.1 - Engenharia de Requisitos de Sistemas.
	Competência Específica V (DCN) >C.3.5. Implantar software para informatização de sistemas, avaliando o impacto de seu uso. >Aplicar > Qualidade de Software, Verificação e Validação de Software.	- Competência Específica V; - Competência Derivada C.3.5; - Qualidade de Software; - Verificação e Validação de Software.
	Competência Específica VI (DCN) >C.3.2. Especificar software para informatização de sistemas, elicitando os requisitos do software em conformidade com os requisitos do produto, dos processos e das partes interessadas, analisando e especificando seus requisitos funcionais e não funcionais e validando o seu potencial de solução das necessidades de sistemas de informação. >Criar > Engenharia de Requisitos de Sistemas, Qualidade de Software , Gerência de Configuração de Software.	- Competência Específica VI; - Competência Derivada C.3.2; - Engenharia de Requisitos de Sistemas; - Qualidade de Software ; - Gerência de Configuração de Software.
	Competência Específica VI (DCN) > C.3.3. Projetar software para informatização de sistemas, determinando sua arquitetura, garantindo sua qualidade técnica e validando seu potencial de eficácia, eficiência, efetividade e sustentabilidade. >Criar > Engenharia de Requisitos, Arquitetura de Software, Projeto de Software, Verificação e Validação de Software, Qualidade de Software, Gerência de Configuração de Software.	- Competência Específica VI; - Competência Derivada C.3.3; - Engenharia de Requisitos; - Arquitetura de Software; - Projeto de Software; Verificação e Validação de Software; - Qualidade de Software; - Gerência de Configuração de Software;

Eixo de Formação	Unidades de Contexto (Excertos extraídos do Referencial de Formação do RF-SI) (Continuação)	Unidades de Registro
Eixo - 3	Competência Específica VI (DCN)>C.3.4. Construir software para informatização de sistemas avaliando sua qualidade técnica, testando o seu funcionamento, e validando seu atendimento às necessidades de eficácia, eficiência, efetividade e sustentabilidade desses sistemas. >Criar > Projeto de Software, Verificação e Validação de Software, Qualidade de Software, Gerência de Configuração de Software, Teste de Software.	- Competência Específica VI - Competência Derivada C.3.4 - Projeto de Software - Verificação e Validação de Software - Qualidade de Software - Gerência de Configuração de Software, - Teste de Software.
	Competência Específica VIII (DCN) >C.3.1. Avaliar as necessidades de informatizar sistemas, articulando visões individuais e organizacionais, e apreciando oportunidades de melhorias e/ou mudanças em processos, com o uso ou evolução do software. > Avaliar >Engenharia de Requisitos de Sistemas	- Competência Específica VIII - Competência Derivada C.3.1 - Engenharia de Requisitos de Sistemas.
	Competência Específica VIII (DCN) >C.3.2. Especificar software para informatização de sistemas, elicitando os requisitos do software em conformidade com os requisitos do produto, dos processos e das partes interessadas, analisando e especificando seus requisitos funcionais e não funcionais e validando o seu potencial de solução das necessidades de sistemas de informação. >Criar > Engenharia de Requisitos de Sistemas, Qualidade de Software , Gerência de Configuração de Software.	- Competência Específica VIII; - Competência Derivada C.3.2; - Engenharia de Requisitos de Sistemas; - Qualidade de Software ; - Gerência de Configuração de Software.
	Competência Específica VIII (DCN) > C.3.3. Projetar software para informatização de sistemas, determinando sua arquitetura, garantindo sua qualidade técnica e validando seu potencial de eficácia, eficiência, efetividade e sustentabilidade. >Criar > Engenharia de Requisitos, Arquitetura de Software, Projeto de Software, Verificação e Validação de Software, Qualidade de Software, Gerência de Configuração de Software.	- Competência Específica VIII; - Competência Derivada C.3.3; - Engenharia de Requisitos; - Arquitetura de Software; - Projeto de Software; Verificação e Validação de Software; - Qualidade de Software; - Gerência de Configuração de Software;
	Competência Específica VIII (DCN)>C.3.4. Construir software para informatização de sistemas avaliando sua qualidade técnica, testando o seu funcionamento, e validando seu atendimento às necessidades de eficácia, eficiência, efetividade e sustentabilidade desses sistemas. >Criar > Projeto de Software, Verificação e Validação de Software, Qualidade de Software, Gerência de Configuração de Software, Teste de Software.	- Competência Específica VIII - Competência Derivada C.3.4 - Projeto de Software - Verificação e Validação de Software - Qualidade de Software - Gerência de Configuração de Software, - Teste de Software.
	Competência Específica VIII >C.3.7. Gerenciar projetos de produção de software para informatizar sistemas aplicando processos, técnicas e ferramentas de engenharia de software. >Aplicar > Qualidade de Software, Gerência de Configuração de Software, Engenharia de Requisitos Projeto de Software, Teste de Software; Construção de Software, Manutenção de Software,	- Competência Específica VIII - Competência Derivada C.3.7 - Qualidade de Software; - Gerência de Configuração de Software; - Engenharia de Requisitos; Projeto de Software; Teste de Software; Construção de Software; Manutenção de Software;
	Competência Específica X (DCN) >C.3.2. Especificar software para informatização de sistemas, elicitando os requisitos do software em conformidade com os requisitos do produto, dos processos e das partes interessadas, analisando e especificando seus requisitos funcionais e não funcionais e validando o seu potencial de solução das necessidades de sistemas de informação. >Criar > Engenharia de Requisitos de Sistemas, Qualidade de Software , Gerência de Configuração de Software.	- Competência Específica X; - Competência Derivada C.3.2; - Engenharia de Requisitos de Sistemas; - Qualidade de Software ; - Gerência de Configuração de Software.
	Competência Específica X (DCN) >C.3.5. Implantar software para informatização de sistemas, avaliando o impacto de seu uso. >Aplicar > Qualidade de Software, Verificação e Validação de Software.	- Competência Específica X; - Competência Derivada C.3.5; - Qualidade de Software; - Verificação e Validação de Software.
	Competência Específica XI (DCN) >C.3.1. Avaliar as necessidades de informatizar sistemas, articulando visões individuais e organizacionais, e apreciando oportunidades de melhorias e/ou mudanças em processos, com o uso ou evolução do software. > Avaliar >Engenharia de Requisitos de Sistemas	- Competência Específica XI; - Competência Derivada C.3.1 - Engenharia de Requisitos de Sistemas.
	Competência Específica XI (DCN) >C.3.2. Especificar software para informatização de sistemas, elicitando os requisitos do software em conformidade com os requisitos do produto, dos processos e das partes interessadas, analisando e especificando seus requisitos funcionais e não funcionais e validando o seu potencial de solução das necessidades de sistemas de informação. >Criar > Engenharia de Requisitos de Sistemas, Qualidade de Software , Gerência de Configuração de Software.	- Competência Específica XI; - Competência Derivada C.3.2; - Engenharia de Requisitos de Sistemas; - Qualidade de Software ; - Gerência de Configuração de Software.
	Competência Específica XI (DCN) > C.3.3. Projetar software para informatização de sistemas, determinando sua arquitetura, garantindo sua qualidade técnica e validando seu potencial de eficácia, eficiência, efetividade e sustentabilidade. >Criar > Engenharia de Requisitos, Arquitetura de Software, Projeto de Software, Verificação e Validação de Software, Qualidade de Software, Gerência de Configuração de Software.	- Competência Específica XI; - Competência Derivada C.3.3; - Engenharia de Requisitos; - Arquitetura de Software; - Projeto de Software; Verificação e Validação de Software; - Qualidade de Software; - Gerência de Configuração de Software;

Eixo de Formação	Unidades de Contexto (Excertos extraídos do Referencial de Formação do RF-SI) (Continuação)	Unidades de Registro
Eixo - 3	Competência Específica XI (DCN)>C.3.4. Construir software para informatização de sistemas avaliando sua qualidade técnica, testando o seu funcionamento, e validando seu atendimento às necessidades de eficácia, eficiência, efetividade e sustentabilidade desses sistemas. >Criar > Projeto de Software, Verificação e Validação de Software, Qualidade de Software, Gerência de Configuração de Software, Teste de Software.	- Competência Específica XI - Competência Derivada C.3.4 - Projeto de Software - Verificação e Validação de Software - Qualidade de Software - Gerência de Configuração de Software, - Teste de Software.
	Competência Específica XI (DCN) >C.3.5. Implantar software para informatização de sistemas, avaliando o impacto de seu uso. >Aplicar > Qualidade de Software, Verificação e Validação de Software.	- Competência Específica XI; - Competência Derivada C.3.5; - Qualidade de Software; - Verificação e Validação de Software.
	Competência Específica XI >C.3.7. Gerenciar projetos de produção de software para informatizar sistemas aplicando processos, técnicas e ferramentas de engenharia de software. >Aplicar > Qualidade de Software, Gerência de Configuração de Software, Engenharia de Requisitos Projeto de Software, Teste de Software; Construção de Software, Manutenção de Software,	- Competência Específica XI - Competência Derivada C.3.7 - Qualidade de Software; - Gerência de Configuração de Software; - Engenharia de Requisitos; Projeto de Software; Teste de Software; Construção de Software; Manutenção de Software;
	Competência Específica XIII (DCN) >C.3.2. Especificar software para informatização de sistemas, elicitando os requisitos do software em conformidade com os requisitos do produto, dos processos e das partes interessadas, analisando e especificando seus requisitos funcionais e não funcionais e validando o seu potencial de solução das necessidades de sistemas de informação. >Criar > Engenharia de Requisitos de Sistemas, Qualidade de Software , Gerência de Configuração de Software.	- Competência Específica XIII; - Competência Derivada C.3.2; - Engenharia de Requisitos de Sistemas; - Qualidade de Software ; - Gerência de Configuração de Software.
	Competência Específica XIV (DCN) >C.3.1. Avaliar as necessidades de informatizar sistemas, articulando visões individuais e organizacionais, e apreciando oportunidades de melhorias e/ou mudanças em processos, com o uso ou evolução do software. > Avaliar >Engenharia de Requisitos de Sistemas	- Competência Específica XIV - Competência Derivada C.3.1 - Engenharia de Requisitos de Sistemas.
	Competência Específica XIV(DCN) >C.3.2. Especificar software para informatização de sistemas, elicitando os requisitos do software em conformidade com os requisitos do produto, dos processos e das partes interessadas, analisando e especificando seus requisitos funcionais e não funcionais e validando o seu potencial de solução das necessidades de sistemas de informação. >Criar > Engenharia de Requisitos de Sistemas, Qualidade de Software , Gerência de Configuração de Software.	- Competência Específica XIV; - Competência Derivada C.3.2; - Engenharia de Requisitos de Sistemas; - Qualidade de Software ; - Gerência de Configuração de Software.
	Competência Específica XIV (DCN) > C.3.3. Projetar software para informatização de sistemas, determinando sua arquitetura, garantindo sua qualidade técnica e validando seu potencial de eficácia, eficiência, efetividade e sustentabilidade. >Criar > Engenharia de Requisitos, Arquitetura de Software, Projeto de Software, Verificação e Validação de Software, Qualidade de Software, Gerência de Configuração de Software.	- Competência Específica XIV; - Competência Derivada C.3.3; - Engenharia de Requisitos; - Arquitetura de Software; - Projeto de Software; Verificação e Validação de Software; - Qualidade de Software; - Gerência de Configuração de Software;
	Competência Específica XIV (DCN) > C.3.3. Projetar software para informatização de sistemas, determinando sua arquitetura, garantindo sua qualidade técnica e validando seu potencial de eficácia, eficiência, efetividade e sustentabilidade. >Criar > Engenharia de Requisitos, Arquitetura de Software, Projeto de Software, Verificação e Validação de Software, Qualidade de Software, Gerência de Configuração de Software.	- Competência Específica XIV; - Competência Derivada C.3.3; - Engenharia de Requisitos; - Arquitetura de Software; - Projeto de Software; Verificação e Validação de Software; - Qualidade de Software; - Gerência de Configuração de Software;
	Competência Específica XV (DCN) >C.3.1. Avaliar as necessidades de informatizar sistemas, articulando visões individuais e organizacionais, e apreciando oportunidades de melhorias e/ou mudanças em processos, com o uso ou evolução do software. > Avaliar >Engenharia de Requisitos de Sistemas	- Competência Específica XV - Competência Derivada C.3.1 - Engenharia de Requisitos de Sistemas.
	Competência Específica XV >C.3.7. Gerenciar projetos de produção de software para informatizar sistemas aplicando processos, técnicas e ferramentas de engenharia de software. >Aplicar > Qualidade de Software, Gerência de Configuração de Software, Engenharia de Requisitos Projeto de Software, Teste de Software; Construção de Software, Manutenção de Software,	- Competência Específica XV - Competência Derivada C.3.7 - Qualidade de Software; - Gerência de Configuração de Software; - Engenharia de Requisitos; Projeto de Software; Teste de Software; Construção de Software; Manutenção de Software;
	Competência Específica XVI (DCN) >C.3.6 Manter software, corrigindo falhas, adaptandoo ao seu contexto, identificando e implementando melhorias, migrando softwares legados e retirando software. >Avaliar > Qualidade de Software, Gerência de Configuração de Software, Verificação e Validação de Software, Manutenção de software, Engenharia de Requisitos, Projeto de Software	- Competência Específica XVI; - Competência Derivada C.3.6; - Qualidade de Software; - Gerência de Configuração de Software; - Verificação e Validação de Software; - Manutenção de software; - Engenharia de Requisitos, - Projeto de Software

Eixo de Formação	Unidades de Contexto (Excertos extraídos do Referencial de Formação do RF-SI) (Continuação)	Unidades de Registro
Eixo - 4	Competência Geral III (DCN) >C.4.3. Especificar modelos conceituais de banco de dados, analisando aspectos do mundo real a serem tratados pelos sistemas de informação e representando os corretamente de acordo com o metamodelo selecionado e integrando-os com as diretrizes de administração de dados da organização.>Criar >Engenharia de Requisitos	- Competência Geral III - Competência Derivada C.4.3 - Engenharia de Requisitos
	Competência Específica II (DCN) >C.4.3. Especificar modelos conceituais de banco de dados, analisando aspectos do mundo real a serem tratados pelos sistemas de informação e representando os corretamente de acordo com o metamodelo selecionado e integrando-os com as diretrizes de administração de dados da organização.>Criar >Engenharia de Requisitos	- Competência Específica II - Competência Derivada C.4.3 - Engenharia de Requisitos
	Competência Específica III (DCN) >C.4.3. Especificar modelos conceituais de banco de dados, analisando aspectos do mundo real a serem tratados pelos sistemas de informação e representando os corretamente de acordo com o metamodelo selecionado e integrando-os com as diretrizes de administração de dados da organização.>Criar >Engenharia de Requisitos	- Competência Específica III - Competência Derivada C.4.3 - Engenharia de Requisitos
	Competência Específica IV (DCN) >C.4.3. Especificar modelos conceituais de banco de dados, analisando aspectos do mundo real a serem tratados pelos sistemas de informação e representando os corretamente de acordo com o metamodelo selecionado e integrando-os com as diretrizes de administração de dados da organização.>Criar >Engenharia de Requisitos	- Competência Específica IV - Competência Derivada C.4.3 - Engenharia de Requisitos
	Competência Específica VI (DCN) >C.4.3. Especificar modelos conceituais de banco de dados, analisando aspectos do mundo real a serem tratados pelos sistemas de informação e representando os corretamente de acordo com o metamodelo selecionado e integrando-os com as diretrizes de administração de dados da organização.>Criar >Engenharia de Requisitos	- Competência Específica VI - Competência Derivada C.4.3 - Engenharia de Requisitos
	Competência Específica X (DCN) >C.4.3. Especificar modelos conceituais de banco de dados, analisando aspectos do mundo real a serem tratados pelos sistemas de informação e representando os corretamente de acordo com o metamodelo selecionado e integrando-os com as diretrizes de administração de dados da organização.>Criar >Engenharia de Requisitos	- Competência Específica X - Competência Derivada C.4.3 - Engenharia de Requisitos

Fonte: Elaboração Própria

Tabela A.4 Tabela descritiva de Exploração do RF-LC

Eixo de Formação	Unidades de Contexto (Excertos extraídos do Referencial de Formação do RF-LC	Unidades de Registro
Eixo 2	Competência Geral I (DCN) > C.2.1 Formular e resolver problemas com a aplicação do raciocínio lógico, matemático e computacional> Engenharia de Software.	- Competência Geral I - Competência Derivada C.2.1 - Engenharia de Software
	Competência Geral III (DCN) > C.2.1 Formular e resolver problemas com a aplicação do raciocínio lógico, matemático e computacional> Engenharia de Software.	- Competência Geral I - Competência Derivada C.2.1 - Engenharia de Software
	Competência Geral V(DCN) > C.2.1 Formular e resolver problemas com a aplicação do raciocínio lógico, matemático e computacional> Engenharia de Software.	- Competência Geral I - Competência Derivada C.2.1 - Engenharia de Software
	Competência Geral V(DCN) > C.2.1 Formular e resolver problemas com a aplicação do raciocínio lógico, matemático e computacional> Engenharia de Software.	- Competência Geral I - Competência Derivada C.2.1 - Engenharia de Software
Eixo 5	Competência Específica I (DCN) > C.5.2 Desenvolver recursos tecnológicos para fins educacionais> Engenharia de Software	- Competência Específica I - Competência Derivada C.5.2 - Engenharia de Software
	Competência Específica II (DCN) > C.5.2 Desenvolver recursos tecnológicos para fins educacionais> Engenharia de Software	- Competência Específica II - Competência Derivada C.5.2 - Engenharia de Software
	Competência Específica III (DCN) > C.5.2 Desenvolver recursos tecnológicos para fins educacionais> Engenharia de Software	- Competência Específica III - Competência Derivada C.5.2 - Engenharia de Software
	Competência Específica V (DCN) > C.5.2 Desenvolver recursos tecnológicos para fins educacionais> Engenharia de Software	- Competência Específica V - Competência Derivada C.5.2 - Engenharia de Software
	Competência Específica VI (DCN) > C.5.2 Desenvolver recursos tecnológicos para fins educacionais> Engenharia de Software	- Competência Específica VI - Competência Derivada C.5.2 - Engenharia de Software
	Competência Específica VII (DCN) > C.5.2 Desenvolver recursos tecnológicos para fins educacionais> Engenharia de Software	- Competência Específica VII - Competência Derivada C.5.2 - Engenharia de Software
	Competência Específica IX (DCN) > C.5.2 Desenvolver recursos tecnológicos para fins educacionais> Engenharia de Software	- Competência Específica IX - Competência Derivada C.5.2 - Engenharia de Software
Eixo 6	Competência Geral VI (DCN) > C.6.6 Pesquisar, compreender e avaliar criticamente informações> Todos os conteúdos	- Competência Geral VI - Competência Derivada C.6.6 - Todos os Conteúdos
	Competência Geral VI (DCN) > C.6.7 Produzir novos conhecimentos e gerir a própria aprendizagem> Todos os conteúdos	- Competência Geral VI - Competência Derivada C.6.7 - Todos os Conteúdos
	Competência Geral VI (DCN) > C.6.9 Realizar trabalho cooperativo e compreender a sua importância> Todos os conteúdos	- Competência Geral VI - Competência Derivada C.6.9 - Todos os Conteúdos
	Competência Geral XI (DCN) > C.6.6 Pesquisar, compreender e avaliar criticamente informações> Todos os conteúdos	- Competência Geral XI - Competência Derivada C.6.6 - Todos os Conteúdos
	Competência Geral XI (DCN) > C.6.7 Pesquisar, compreender e avaliar criticamente informações> Todos os conteúdos	- Competência Geral XI - Competência Derivada C.6.7 - Todos os Conteúdos
	Competência Geral XI (DCN) > C.6.9 Pesquisar, compreender e avaliar criticamente informações> Todos os conteúdos	- Competência Geral XI - Competência Derivada C.6.9 - Todos os Conteúdos
	Competência Geral XII (DCN) > C.6.6 Pesquisar, compreender e avaliar criticamente informações> Todos os conteúdos	- Competência Geral XII - Competência Derivada C.6.6 - Todos os Conteúdos
	Competência Geral XII (DCN) > C.6.7 Pesquisar, compreender e avaliar criticamente informações> Todos os conteúdos	- Competência Geral XII - Competência Derivada C.6.7 - Todos os Conteúdos
	Competência Geral XII (DCN) > C.6.9 Pesquisar, compreender e avaliar criticamente informações> Todos os conteúdos	- Competência Geral XII - Competência Derivada C.6.9 - Todos os Conteúdos

Fonte:Elaboração Própria

Tabela A.5 Tabela descritiva de Exploração do RF-ES

Eixo de Formação	Unidades de Contexto (Excertos extraídos do Referencial de Formação do RF-ES	Unidades de Registro
Eixo I	Competência Específica I (DCN) > C.1.5. Otimizar processos e produtos considerando aspectos econômicos e de qualidade.>Entender > Engenharia de Produto>Tomada de Decisão	- Competência Específica I - Competência Derivada C.1.5 - Tomada de Decisão
Eixo II	Competência Específica XIII (DCN) >C.2.2. Criar modelos de negócios, transformando ideias em produtos ou serviços>Aplicar >Frameworks para construção de modelos de negócio >Empreendedorismo	- Competência Específica XIII - Competência Derivada C.2.2 - Empreendedorismo
	Competência Específica XIII (DCN) >C.2.3. Planejar Empreendimentos Inovadores serviços>Aplicar >Planejamento de Negócios >Empreendedorismo	- Competência Específica XIII - Competência Derivada C.2.3 - Empreendedorismo
Eixo III	Competência Geral IV (DCN) >C.3.1. Conhecer os direitos e deveres dos criadores, comercializadores, compradores e usuários de software>Conhecer> Registro de Software>Leis acordos e Instruções Normativas Sobre Engenharia de Software.	- Competência Geral IV - Competência Derivada C. 3.1 - Leis acórdãos e instruções Normativas sobre ES;
	Competência Geral VI (DCN) >C.3.1. Conhecer os direitos e deveres dos criadores, comercializadores, compradores e usuários de software>Conhecer> Registro de Software>Leis acordos e Instruções Normativas Sobre Engenharia de Software.	- Competência Geral VI - Competência Derivada C. 3.1 - Leis acórdãos e instruções Normativas sobre ES;
	Competência Geral VI (DCN) >C.3.2. Aplicar métodos de pesquisa em Engenharia de Software >Aplicar >Métodos de pesquisa e experimentação em Engenharia de Software	- Competência Geral VI - Competência Derivada C. 3.2 - Métodos de Pesquisa e Experimentação em ES
	Competência Geral VI (DCN) >C.3.3. Aplicar métodos de pesquisa em Engenharia de Software >Entender >Métodos de pesquisa e experimentação em Engenharia de Software >Relato de estudos experimentais de Engenharia de Software	- Competência Geral VI - Competência Derivada C. 3.3 - Relato de estudos experimentais de ES;
	Competência Geral VII (DCN) >C.3.3. Aplicar métodos de pesquisa em Engenharia de Software >Entender>Métodos de pesquisa e experimentação em Engenharia de Software >Relato de estudos experimentais deEngenharia de Software	- Competência Geral VII - Competência Derivada C. 3.3 - Relato de estudos experimentais de ES;
	Competência Geral IX (DCN) >C.3.1. Conhecer os direitos e deveres dos criadores, comercializadores, compradores e usuários de software> Registro de Software>Leis acordos e Instruções Normativas Sobre Engenharia de Software.	- Competência Geral IX - Competência Derivada C. 3.1 - Registro de Software; - Leis acórdãos e instruções Normativas sobre ES;
	Competência Geral IX (DCN) >C.3.2. Aplicar métodos de pesquisa em Engenharia de Software >Conhecer>Métodos de pesquisa e experimentação em Engenharia de Software	- Competência Geral IX - Competência Derivada C. 3.2 - Métodos de pesquisa e Experimentação em ES;
	Competência Geral IX (DCN) >C.3.3. Aplicar métodos de pesquisa em Engenharia de Software >Entender>Métodos de pesquisa e experimentação em Engenharia de Software >Relato de estudos experimentais deEngenharia de Software	- Competência Geral IX - Competência Derivada C. 3.3 - Relato de estudos experimentais de ES;
	Competência Geral XII (DCN) >C.3.1. Conhecer os direitos e deveres dos criadores, comercializadores, compradores e usuários de software>Conhecer> Registro de Software>Leis acordos e Instruções Normativas Sobre Engenharia de Software.	- Competência Geral XII - Competência Derivada C. 3.1 - Registro de Software; - Leis acórdãos e instruções Normativas sobre ES;
	Competência Geral XII (DCN) >C.3.2. Aplicar métodos de pesquisa em Engenharia de Software >Aplicar>Métodos de pesquisa e experimentação em Engenharia de Software	- Competência Geral XII - Competência Derivada C. 3.2 - Métodos de pesquisa e Experimentação em ES;
	Competência Geral XII (DCN) >C.3.3. Aplicar métodos de pesquisa em Engenharia de Software >Entender>Métodos de pesquisa e experimentação em Engenharia de Software >Relato de estudos experimentais deEngenharia de Software	- Competência Geral XII - Competência Derivada C. 3.3 - Relato de estudos experimentais de ES;
	Competência Específica IV (DCN) >C.3.1. Conhecer os direitos e deveres dos criadores, comercializadores, compradores e usuários de software>Conhecer> Registro de Software>Noções Básicas de Direito	- Competência Específica IV - Competência Derivada C. 3.1 -Noções Básicas de Direito
	Competência Específica IX (DCN) >C.3.2. Aplicar métodos de pesquisa em Engenharia de Software >Aplicar >Métodos de pesquisa e experimentação em Engenharia de Software >Conhecimento Científico	- Competência Específica IX - Competência Derivada C. 3.2 - Conhecimento Científico;
	Competência Específica IX (DCN) >C.3.3. Aplicar métodos de pesquisa em Engenharia de Software >Entender >Métodos de pesquisa e experimentação em Engenharia de Software >Análise Qualitativa	- Competência Específica IX - Competência Derivada C. 3.3 - Análise Qualitativa;

Eixo de Formação	Unidades de Contexto (Excertos extraídos do Referencial de Formação do RF-ES (Continuação)	Unidades de Registro
Eixo 3	Competência Específica X (DCN) >C.3.4. Aplicar técnicas de comunicação para Engenharia de Software>Aplicar >Técnicas de comunicação	- Competência Específica X - Competência Derivada C.3.4 - Técnicas de Comunicação
	Competência Específica X (DCN) >C.3.5. Entender Técnicas de treinamento em Engenharia de Software>Aplicar >Técnicas de treinamento	- Competência Específica X - Competência Derivada C.3.5 - Técnicas de Treinamento
	Competência Específica X (DCN) >C.3.6. Conhecer métodos de consultoria em Engenharia de Software>Entender >Técnicas de consultoria	- Competência Específica X - Competência Derivada C.3.6 - Técnicas de consultoria
	Competência Específica X (DCN) >C.3.7. Conhecer técnicas de negociação em Engenharia de Software>Conhecer >Técnicas de negociação	- Competência Específica X - Competência Derivada C.3.7 - Técnicas de negociação
Eixo 4	Competência Específica VII >C.4.7. Entender a estrutura dos processos de produção aplicados a software >Entender >Estrutura do processo de bens (manufatura) e serviços (produtos de software)>Competências Competitivas	- Competência Específica VII - Competência Derivada C.4.7 - Competências Competitivas
Eixo 5	Competência Específica I>C.5.1. Conhecer e analisar as características de domínios de aplicação em diversos contextos>Conhecer > Modelagem de processos de negócio>Técnicas de ideação	- Competência Específica I - Competência Derivada C.5.1 - Técnicas de Ideação
	Competência Específica XIV (DCN)>C.5.7. Aplicar métodos e técnicas para design de software >Aplicar >Métodos e técnicas de especificação, modelagem, e análise de arquiteturas de software, Normas, linguagens e ferramentas de arquitetura de Software.>Métodos e técnicas de especificação e modelagem da interação com usuários	- Competência Específica XIV - Competência Derivada C.5.7 - Métodos e técnicas de especificação e modelagem da interação com usuários
Eixo 6	Competência Geral III (DCN) >C.6.1. Aplicar técnicas e procedimentos de desenvolvimento de software >Aplicar >Projeto de Arquitetura de Software, Reutilização de software.	- Competência Geral III - Competência Derivada C.6.1. - Projeto de Arquitetura e Reutilização de Software
	Competência Geral III (DCN)>C.6.2. Aplicar técnicas e procedimentos de validação e verificação (estáticos e dinâmicos)>Aplicar >Técnicas de revisão e análise estática de artefatos de software, Técnicas de revisão e análise dinâmica de artefatos de software	- Competência Geral III - Competência Derivada C.6.2. - Técnicas de revisão e análise de artefatos de Software.
	Competência Geral III (DCN) >C.6.3. Definir o ambiente de construção de software >Aplicar >Ferramentas e frameworks de desenvolvimento de software, Ferramentas e frameworks de gerenciamento de configuração de software.	- Competência Geral III - Competência Derivada C.6.3 - Ferramentas e Frameworks de Desenvolvimento e Gerenciamento de Software
	Competência Geral III (DCN) >C.6.6. Aplicar técnicas de integração de sistemas heterogêneos>Aplicar >Software como serviço	- Competência Geral III - Competência Derivada C.6.6 - Software como serviço
	Competência Geral III (DCN) >C.6.7. Aplicar os princípios, padrões e boas práticas de desenvolvimento de software >Aplicar> Princípios de Engenharia de Software Aplicação de padrões em Engenharia de Software	- Competência Geral III - Competência Derivada C.6.7 - Princípios e Aplicações de Padrões em ES.
	Competência Geral III (DCN) >C.6.9. Aplicar teorias, modelos e técnicas para verificar soluções de software >Aplicar >Técnicas de verificação e análise estática de artefatos de software , Técnicas de análise dinâmica de artefatos de software	- Competência Geral III - Competência Derivada C.6.9. - Técnicas de verificação e análise de artefatos de Software.
	Competência Específica II (DCN) >C.6.1. Aplicar técnicas e procedimentos de desenvolvimento de software>Aplicar>Projeto (design) de arquitetura de software , Reutilização de Software>Projeto (design) de interface comusuários	- Competência Específica II - Competência Derivada C.6.1 - Projeto (design) de interface com usuários
	Competência Específica VI (DCN) >C.6.6. Aplicar técnicas de integração de sistemas heterogêneos >Aplicar >Software como serviço > Sistemas de sistemas.	- Competência Específica VI - Competência Derivada C.6.6 - Sistemas de Sistemas
	Competência Específica XI >C.6.7. Aplicar os princípios, padrões e boas práticas de desenvolvimento de software>Aplicar>Princípios de Engenharia de Software, Aplicação de padrões em Engenharia de Software> Melhoria contínua, Aplicação de gestão de conhecimento	- Competência Específica XI - Competência Derivada C.6.7 - Melhoria Contínua e Gestão do Conhecimento.
Eixo 7	Competência Específica V (DCN)>C.7.2. Aplicar mecanismos de medição da qualidade do produto de software>Aplicar>Métricas de produto de software > Técnicas de avaliação de produto	- Competência Específica V - Competência Derivada C.7.2 - Técnicas de Avaliação de Produto.

Fonte: Elaboração Própria

Apêndice

B

Aqui são demonstradas todas as unidades de contexto e unidades de registro que estruturam a codificação dos Guias Internacionais

ANÁLISE DE CONTEÚDO DOS CURRÍCULOS DA ACM

Tabela B.1 Análise de Conteúdo do CS-2013

Unidades de Contexto	**Unidades de Registro**
IAS. Garantia e segurança da informação (Núcleo-1 3 horas, Núcleo-2 6 horas) >IAS/Engenharia de Software Seguro >[Eletivo]	Engenharia de Software Seguro
SE. Engenharia de Software (Núcleo-1 6 horas, Núcleo-2 21 horas) >SE/Processo de Software >[Núcleo-1 2 horas, Núcleo-2 1 hora]	Processo de Software
SE. Engenharia de Software (Núcleo-1 6 horas, Núcleo-2 21 horas) >SE/Gerenciamento de Projetos de Software>[Núcleo-2 2 horas]	Gerenciamento de Projetos de Software
SE. Engenharia de Software (Núcleo-1 6 horas, Núcleo-2 21 horas) >SE/Ferramentas e Ambientes>[Núcleo-2 2 horas]	Ferramentas e Ambientes em Engenharia de Software
SE. Engenharia de Software (Núcleo-1 6 horas, Núcleo-2 21 horas) >SE/Engenharia de Requisitos>[Núcleo-1 2 horas, Núcleo-2 3 horas]	Engenharia de Requisitos
SE. Engenharia de Software (Núcleo-1 6 horas, Núcleo-2 21 horas) >SE/Desing de Software>[Núcleo-1 3 horas, Núcleo-2 5 horas]	Desing de Software
SE. Engenharia de Software (Núcleo-1 6 horas, Núcleo-2 21 horas) >SE/Construção de Software de Software>[Núcleo-2 2 horas]	Construção de Software
SE. Engenharia de Software (Núcleo-1 6 horas, Núcleo-2 21 horas) >SE/Validação e Verificação de Software>[Núcleo-2 4 horas]	Validação e Verificação de Software
SE. Engenharia de Software (Núcleo-1 6 horas, Núcleo-2 21 horas) >SE/Evolução de Software>[Núcleo-2 2 horas]	Evolução de Software
SE. Engenharia de Software (Núcleo-1 6 horas, Núcleo-2 21 horas) >SE/Confiabilidade de Software>[Núcleo-2 1 hora]	Confiabilidade de Software
SE. Engenharia de Software (Núcleo-1 6 horas, Núcleo-2 21 horas) >SE/Métodos Formais>[Eletivo]	Métodos Formais em Engenharia de Software
SP. Questões Sociais e Prática Profissional. [Núcleo-1 11 horas, Núcleo-2 5 horas] >SP/Contexto Social >[Núcleo-1 1 hora, Núcleo-2 2 horas]	Contexto Social
SP. Questões Sociais e Prática Profissional. [Núcleo-1 11 horas, Núcleo-2 5 horas] >SP/Ferramentas Analíticas >[Núcleo-1 2 horas]	Ferramentas Analíticas em Ética
SP. Questões Sociais e Prática Profissional. [Núcleo-1 11 horas, Núcleo-2 5 horas] >SP/Ética Profissional >[Núcleo-1 2 horas, Núcleo-2 2 horas]	Ética Profissional
SP. Questões Sociais e Prática Profissional. [Núcleo-1 11 horas, Núcleo-2 5 horas] >SP/Propriedade Intelectual>[Núcleo-1 2 horas]	Propriedade Intelectual
SP. Questões Sociais e Prática Profissional. [Núcleo-1 11 horas, Núcleo-2 5 horas] >SP/Privacidade e Liberdades Civis>[Núcleo-1 2 horas]	Privacidade e Liberdades Civis
SP. Questões Sociais e Prática Profissional. [Núcleo-1 11 horas, Núcleo-2 5 horas] >SP/Comunicação Profissional >[Núcleo-1 1 horas]	Comunicação Profissional
SP. Questões Sociais e Prática Profissional. [Núcleo-1 11 horas, Núcleo-2 5 horas] >SP/Sustentabilidade >[Núcleo-1 1 hora, Núcleo-2 1 Hora]	Sustentabilidade
SP. Questões Sociais e Prática Profissional. [Núcleo-1 11 horas, Núcleo-2 5 horas] >SP/História>[Eletivo]	História
SP. Questões Sociais e Prática Profissional. [Núcleo-1 11 horas, Núcleo-2 5 horas] >SP/Economias de Computação>[Eletivo]	Economias de Computação
SP. Questões Sociais e Prática Profissional. [Núcleo-1 11 horas, Núcleo-2 5 horas] >SP/Políticas de Segurança, leis e crimes informáticos>[Eletivo]	Políticas de Segurança, leis e Crimes Informáticos

Fonte: Elaboração Própria

Tabela B.2 Análise de Conteúdo do CE-2016

Unidades de Contexto	**Unidades de Registro**
CE-PPP Preparação para a Prática Profissional [20 Horas Principais]> CE-PPP-1 Histórico e visão geral [1]	Histórico e Visão Geral para prática Profissional.
CE-PPP Preparação para a Prática Profissional [20 Horas Principais]> CE-PPP-2 Ferramentas, padrões e/ou restrições de engenharia relevantes [1]	Ferramentas, padrões e/ou restrições de engenharia relevantes em Prática Profissional
CE-PPP Preparação para a Prática Profissional [20 Horas Principais] >CE-PPP-3 Estratégias de comunicação eficazes [2]	Estratégia de Comunicação Eficazes
CE-PPP Preparação para a Prática Profissional [20 Horas Principais]> CE-PPP-4 Abordagens de equipe interdisciplinar [1]	Abordagens de equipe Interdisciplinar
CE-PPP Preparação para a Prática Profissional [20 Horas Principais] >CE-PPP-5 Enquadramentos filosóficos e questões culturais [2]	Enquadramentos Filosóficos e Questões Culturais
CE-PPP Preparação para a Prática Profissional [20 Horas Principais] >CE-PPP-6 Soluções de engenharia e efeitos sociais [2]	Soluções de engenharia e efeitos sociais
CE-PPP Preparação para a Prática Profissional [20 Horas Principais] >CE-PPP-7 Responsabilidades profissionais e éticas [3]	Responsabilidades profissionais e Éticas
CE-PPP Preparação para a Prática Profissional [20 Horas Principais] >CE-PPP-8 Propriedade intelectual e questões legais [3]	Propriedade Intelectual e questões legais
CE-PPP Preparação para a Prática Profissional [20 Horas Principais] >CE-PPP-9 Questões contemporâneas [2]	Questões Contemporâneas
CE-PPP Preparação para a Prática Profissional [20 Horas Principais] >CE-PPP-10 Questões de negócios e gerenciamento [3]	Questões de Negócios e Gerenciamento
CE-PPP Preparação para a Prática Profissional [20 Horas Principais] CE-PPP-11 Escolhas na prática profissional	Escolhas na Prática profissional
CE-SPE Engenharia de Sistemas e Projetos [35 horas principais] >CE-SPE-6 Processos de Hardware e Software[3]	Processos de Software
CE-SPE Engenharia de Sistemas e Projetos [35 horas principais] >CE-SPE-7 Análise e elicitação de requisitos[3]	Análise e elicitação de requisitos.
CE-SPE Engenharia de Sistemas e Projetos [35 horas principais] >CE-SPE-11 Teste Integração e Validação de sistemas	Testes Integração e validação de Sistemas de Software
CE-SWD Desing de Software [45 horas principais] >CE-SWD-1 Histórico e Visão Gera [2]	Histórico e Visão Geral do Desig de Software
CE-SWD Desing de Software [45 horas principais] >CE-SWD-2 Ferramentas, padrões e/ou restrições de engenharia desoftware relevantes	Ferramentas, padrões e/ou restrições de engenharia de software relevantes
CE-SWD Desing de Software [45 horas principais] >CE-SWD-8 Teste e qualidade de software	Teste e qualidade de software

Fonte: Elaboração Própria

Tabela B.3 Análise de Conteúdo para mapeamento técnico em ES no SI-2010

Unidades de Contexto	Unidades de Registro
Desenvolvimento de Aplicativos > >Curso eletivo>Design do Programa	Design de Software
Desenvolvimento de Aplicativos > >Curso eletivo>Ciclo de vida de desenvolvimento de programa	Ciclo de Vida de Desenvolvimento de software
Desenvolvimento de Aplicativos > >Curso eletivo>Determinantes e análise de requisitos	Análise de Requisitos
Desenvolvimento de Aplicativos > >Curso eletivo>Técnicas para modelar estrutura do programa.	Técnicas para Modelar Estruturas do Software
Introdução à Interação Humano-Computador> >Curso eletivo>Design Centrado no Usuário	Desing de Software Centrado no usuário.
Introdução à Interação Humano-Computador> >Curso eletivo>Teste de Usabilidade	Teste de Software

Fonte: Elaboração Própria

Tabela B.4 Análise de Conteúdo dos conhecimentos Fundamentais do SI-2010

Unidades de Contexto	Unidades de Registro
Conhecimentos e habilidades dos graduados em SI > >Conhecimentos e Habilidades Fundamentais > 1. Liderança e colaboração.	Capacidade de Liderança e Colaboração
Conhecimentos e habilidades dos graduados em SI > >Conhecimentos e Habilidades Fundamentais > 2. Comunicação	Comunicação
Conhecimentos e habilidades dos graduados em SI > >Conhecimentos e Habilidades Fundamentais > 3. Negociação	Negociação
Conhecimentos e habilidades dos graduados em SI > >Conhecimentos e Habilidades Fundamentais > >4.Pensamento analítico e crítico, incluindo criatividade e análise ética	Criatividade e Capacidade de Análise crítica e ética.

Fonte: Elaboração Própria

Tabela B.5 Análise de Conteúdo do TI-2017 da ACM

Unidades de Contexto	Unidades de Registro
Domínios essenciais de TI>ITE-GPP Prática Profissional Global> >ITE-GPP-01 Perspectivas e impacto [L1]	Perspectivas e Impactos da prática profissional
Domínios essenciais de TI> >ITE-GPP Prática Profissional Global> >ITE-GPP-02 Questões e responsabilidades profissionais [L1]	Questões e responsabilidades profissionais
Domínios essenciais de TI>ITE-GPP Prática Profissional Global> ITE-GPP-05 Questões ambientais [L1]	Questões Ambientais
Domínios essenciais de TI>ITE-GPP Prática Profissional Global> ITE-GPP-06 Questões éticas, legais e de privacidade [L1]	Questões éticas, legais e de privacidade
Domínios essenciais de TI>ITE-GPP Prática Profissional Global> ITE-GPP-07 Propriedade intelectual [L1]	Propriedade Intelectual
Domínios essenciais de TI>ITE-GPP Prática Profissional Global> ITE-GPP-09 Comunicações [L1]	Comunicações
Domínios essenciais de TI>ITE-GPP Prática Profissional Global> ITE-GPP-10 Trabalho em equipe e gestão de conflitos [L1]	Trabalho em equipe e gestão de conflitos
Domínios essenciais de TI>ITE-GPP Prática Profissional Global> ITE-GPP-11 Habilidades de empregabilidade e carreiras em TI [L1]	Habilidades de empregabilidade e carreiras em TI
Domínios essenciais de TI>Paradigmas do Sistema ITE-SPA [6%] ITE-SPA-03 Arquitetura do sistema [L1]	Arquitetura de Sistemas
Domínios essenciais de TI>Paradigmas do Sistema ITE-SPA [6%] ITE-SPA-05 Teste e garantia de qualidade [L2]	Teste e garantia de qualidade
Domínios essenciais de TI>ITE-UXD Design de experiência do usuário [3%] ITE-UXD-02 Fatores humanos no design [L2]	Fatores Humanos no Desing
Domínios essenciais de TI>ITE-UXD Design de experiência do usuário [3%] ITE-UXD-08 Defesa do usuário [L1]	Defesa do Usuário
Domínios essenciais de TI>ITE-WMS Sistemas Web e Móveis [3%] >ITE-WMS-04 Conceitos de aplicativos [L2]	Conceitos de Aplicativos
Domínios essenciais de TI>ITE-WMS Sistemas Web e Móveis [3%] >ITE-WMS-05 Frameworks de Desenvolvimento [L2]	Frameworks de Desenvolvimento
Domínios Suplementares >ITS-CEC Desafios emergentes de segurança cibernética [4%] >ITS-CEC-08 Cadeia de suprimentos e garantia de software [L1]	Cadeia de Suprimentos e Garantia de Software
Domínios Suplementares >ITS-SDM Desenvolvimento e Gestão de Software [2%] >ITS-SDM-01 Modelos de processo e atividades [L2]	Modelo de Processos e Atividades de desenvolvimento e Gerenciamento de Software
Domínios Suplementares >ITS-SDM Desenvolvimento e Gestão de Software [2%] >ITS-SDM-02 Desenvolvimento de software baseado em plataforma [L1]	Desenvolvimento de Software Baseado em plataforma
Domínios Suplementares >ITS-SDM Desenvolvimento e Gestão de Software [2%] ITS-SDM-03 Ferramentas e serviços [L2]	Ferramentas e Serviços
Domínios Suplementares >ITS-SDM Desenvolvimento e Gestão de Software [2%] >ITS-SDM-04 Gerenciamento [L2]	Gerenciamento de software e de projetos de software
Domínios Suplementares >ITS-SDM Desenvolvimento e Gestão de Software [2%] ITS-SDM-05 Implantação, operações, manutenção [L2]	Implantação, operações Manutenção de Software
Domínios Suplementares >ITS-SRE Responsabilidade Social [2%]> ITS-SRE-01 Contexto social da computação [L2]	Contexto social da computação
Domínios Suplementares >ITS-SRE Responsabilidade Social [2%]> ITS-SRE-02 Objetivos, planos, tarefas, prazos e riscos [L2]	Objetivos, planos, tarefas, prazos e riscos
Domínios Suplementares >ITS-SRE Responsabilidade Social [2%]> ITS-SRE-03 Papel e regulamentos do governo [L1]	Papel e regulamentos do Governo
Domínios Suplementares >ITS-SRE Responsabilidade Social [2%]> ITS-SRE-04 Desafios e abordagens globais [L1]	Desafios e abordagens Globais
Domínios Suplementares >ITS-SRE Responsabilidade Social [2%]> ITS-SRE-05 Gerenciamento de riscos [L1]	Gerenciamento de Riscos
Domínios Suplementares >ITS-SRE Responsabilidade Social [2%]> ITS-SRE-06 Computação Sustentável [L1]	Computação Sustentável.

Fonte: Elaboração Própria

Tabela B.6 Análise de Conteúdo do Currículo de Engenharia de Software 2014 da ACM

Unidades de Contexto	**Unidades de Registro**
PRF Prática profissional 29> PRF.psy Dinâmica de grupo e psicologia 8 > PRF.psy.1 Dinâmica do trabalho em equipe e grupos a E PRF.psy.2 Cognição individual (por exemplo, limites) k E PRF.psy.3 Complexidade do problema cognitivo k E PRF.psy.4 Interagindo com as partes interessadas c E PRF.psy.5 Lidando com incerteza e ambiguidade k E PRF.psy.6 Lidando com ambientes multiculturais k E > PRF.com Habilidades de comunicação (específicas para SE) 15 PRF.com.1 Leitura, compreensão e resumo da leitura (por exemplo, código-fonte e documentação) a E PRF.com.2 Redação (trabalhos, relatórios, avaliações, justificativas, etc.) a E PRF.com.3 Comunicação de equipe e grupo (oral e escrita, e-mail, etc.) a E PRF.com.4 Técnicas de apresentação a E PRF.pr Profissionalismo 6 PRF.pr.1 Credenciamento, certificação e licenciamento k E PRF.pr.2 Códigos de ética e conduta profissional c E PRF.pr.3 Questões e preocupações sociais, legais, históricas e profissionais c E PRF.pr.4 A natureza e o papel das sociedades profissionais k E PRF.pr.5 A natureza e o papel dos padrões de engenharia de software k E PRF.pr.6 O impacto econômico do software c E PRF.pr.7 Contratos de trabalho k E	- Dinâmica de Grupo e Psicologia - Habilidades de Comunicação - Profissionalismo
Modelagem e análise de software MAA 28 MAA.md Fundamentos de modelagem 8 MAA.md.1 Princípios de modelagem (por exemplo, decomposição, abstração, generalização, projeção/visões e uso de abordagens formais) c E MAA.md.2 Pré-condições, pós-condições, invariantes e design por contrato c E MAA.md.3 Introdução aos modelos matemáticos e notação formal k E MAA.tm Tipos de modelos 12 MAA.tm.1 Modelagem de informações (por exemplo, modelagem entidade-relacionamento e diagramas de classes) a E MAA.tm.2 Modelagem comportamental (por exemplo, diagramas de estado, análise de caso de uso, diagramas de interação , modos de falha e análise de efeitos e análise de árvore de falhas) a E MAA.tm.3 Modelagem de arquitetura (por exemplo, padrões de arquitetura e diagramas de componentes) c E MAA.tm.4 Modelagem de domínio (por exemplo, abordagens de engenharia de domínio) k E MAA.tm.5 Modelagem empresarial (por exemplo, processos de negócios, organizações, metas e fluxo de trabalho) D MAA.tm.6 Modelagem de sistemas embarcados (por exemplo, programação em tempo real análise e protocolos de interface) D MAA.af Fundamentos da análise 8 MAA.af.1 Analisando a forma (por exemplo, integridade, consistência e robustez) c E MAA.af.2 Analisando a exatidão (por exemplo, análise estática, simulação e verificação do modelo) a E MAA.af.3 Analisando a confiabilidade (por exemplo, análise do modo de falha e árvores de falhas) k E MAA.af.4 Análise formal (por exemplo, prova de teorema) k E	- Fundamentos de Modelagem de Software - Timos de Modelos - Fundamentos da Análise

Unidades de Contexto (Continuação)	**Unidades de Registro**
REQ Análise e especificação de requisitos 30 REQ.rfd Fundamentos dos requisitos 6 REQ.rfd.1 Definição de requisitos (por exemplo, produto, projeto, restrições, c E REQ.rfd.2 Processo de requisitos c E REQ.rfd.3 Camadas/níveis de requisitos (por exemplo, necessidades, objetivos, usuário requisitos, requisitos de sistema e software requisitos) c E REQ.rfd.4 Características dos requisitos (por exemplo, testáveis, inequívocos, consistentes , corretos, rastreáveis e prioritários) c E REQ.rfd.5 Analisar requisitos de qualidade (não funcionais) (por exemplo, segurança, proteção, usabilidade e desempenho) a E REQ.rfd.6 Requisitos de software no contexto da engenharia de sistemas k E REQ.rfd.7 Evolução dos requisitos c E REQ.rfd.8 Rastreabilidade c E REQ.rfd.9 Priorização, análise de trade-off, análise de risco e análise de impacto c E REQ.rfd.10 Gerenciamento de requisitos (por exemplo, gerenciamento de consistência, planejamento de liberação e reutilização) k E REQ.rfd.11 Interação entre requisitos e arquitetura k E REQ.er Elicitando requisitos 10 REQ.er.1 Fontes de elicitação (por exemplo, partes interessadas, especialistas de domínio e ambientes operacionais e organizacionais) c E REQ.er.2 Técnicas de elicitação (por exemplo, entrevistas, questionários/pesquisas, protótipos, casos de uso, observação e técnicas participativas) a E REQ.rsd Especificação e documentação de requisitos 10 REQ.rsd.1 Noções básicas de documentação de requisitos (por exemplo, tipos, público, estrutura, qualidade, atributos e padrões) k E REQ.rsd.2 Técnicas de especificação de requisitos de software (por exemplo, documentação de requisitos planejados, tabelas de decisão, histórias de usuários e especificações comportamentais) a E REQ.rv Validação de requisitos 4 REQ.rv.1 Revisões e inspeções a E REQ.rv.2 Prototipagem para validar requisitos k E REQ.rv.3 Projeto de teste de aceitação c E REQ.rv.4 Validação dos atributos de qualidade do produto c E REQ.rv.5 Análise de interação de requisitos (por exemplo, interação de recurso) k E REQ.rv.6 Análise formal dos requisitos D	- Fundamentos de Requisitos - Elicitação de Requisitos - Validação de Requisitos
DES Desing de Software 48 DES.con Conceitos de design 3 DES.con.1 Definição de design c E DES.con.2 Questões fundamentais de design (por exemplo, dados persistentes, gerenciamento de armazenamento e exceções) c E DES.con.3 Contexto do design em vários ciclos de vida de desenvolvimento de software k E DES.con.4 Princípios de design (ocultação de informações, coesão e acoplamento) a E DES.con.5 Interações entre design e requisitos c E DES.con.6 Design para atributos de qualidade (por exemplo, confiabilidade, usabilidade, manutenibilidade, desempenho, testabilidade, segurança e tolerância a falhas).k E DES.con.7 Compromissos do projeto k E DES.str Estratégias de design 6 DES.str.1 Design orientado a funções c E DES.str.2 Projeto orientado a objetos a E DES.str.3 Projeto centrado na estrutura de dados D DES.str.4 Projeto orientado a aspectos D DES.ar Design Arquitetural 12 DES.ar.1 Estilos arquitetônicos, padrões e estruturas a E DES.ar.2 Compensações arquitetônicas entre vários atributos a E DES.ar.3 Hardware e problemas de engenharia de sistemas em software arquitetura k E DES.ar.4 Rastreabilidade de requisitos em arquitetura k E DES.ar.5 Arquiteturas orientadas a serviços k E DES.ar.6 Arquiteturas para sistemas de rede, móveis e embarcados k E DES.ar.7 Relação entre a arquitetura do produto e a estrutura da organização de desenvolvimento e mercado k E DES.hci Design de interação humano-computador 10 DES.hci.1 Princípios gerais de projeto HCI a E DES.hci.2 Uso de modos e navegação a E DES.hci.3 Técnicas de codificação e design visual (por exemplo, cores, ícones e fontes) c E DES.hci.4 Tempo de resposta e feedback a E DES.hci.5 Modalidades de design (por exemplo, manipulação direta, seleção de menu, formulários, pergunta-resposta e comandos) a E DES.hci.6 Localização e internacionalização c E DES.hci.7 Métodos de projeto HCI c E DES.hci.8 Modalidades de interface (por exemplo, fala e linguagem natural, áudio/vídeo e tátil) D DES.hci.9 Metáforas e modelos conceituais D DES.hci.10 Psicologia da IHC D DES.dd Design detalhado 14 DES.dd.1 Padrões de design a E DES.dd.2 Design de banco de dados a E DES.dd.3 Design de sistemas móveis e em rede a E DES.dd.4 Notações de projeto (por exemplo, diagramas de classe e objeto, UML, diagramas de estado e especificação formal) c E Avaliação do projeto DES.ev 3 DES.ev.1 Atributos de design (por exemplo, acoplamento, coesão, ocultação de informações e separação de preocupações) k E DES.ev.2 Métricas de projeto a E DES.ev.3 Análise de projeto formal D	-Conceitos de Desig de Software -Estratégias de Design de Software - Design Arquitetural - Design de Interação Humano-Computador - Design Detalhado

Unidades de Contexto (Continuação)	**Unidades de Registro**
Verificação e validação do Software VAV 37 VAV.fnd V&V terminologia e fundamentos 5 VAV.fnd.1 V&V objetivos e restrições k E VAV.fnd.2 Planejando o esforço de V&V k E VAV.fnd.3 Documentando a estratégia de V&V, incluindo testes e outros artefatos a E VAV.fnd.4 Métricas e medições (por exemplo, confiabilidade, usabilidade e desempenho) k E VAV.fnd.5 Envolvimento de V&V em diferentes pontos do ciclo de vida k E VAV.rev Comentários e análise estática 9 VAV.rev.1 Revisões pessoais (design, código, etc.) a E VAV.rev.2 Revisões por pares (inspeções, orientações, etc.) a E VAV.rev.3 Análise estática (detecção de defeitos comuns, verificação contra especificações formais, etc.) a E Teste VAV.tst 18 VAV.tst.1 Teste de unidade e desenvolvimento orientado a testes a E VAV.tst.2 Tratamento de exceção (teste de casos extremos e limite condições) a E VAV.tst.3 Análise de cobertura e teste baseado em estrutura a E VAV.tst.4 Técnicas de teste funcional de caixa preta a E VAV.tst.5 Teste de integração c E VAV.tst.6 Desenvolvimento de casos de teste com base em casos de uso e/ou usuário histórias a E VAV.tst.7 Testes baseados em perfis operacionais (por exemplo, os mais usados operações primeiro) k E VAV.tst.8 Sistema e teste de aceitação a E VAV.tst.9 Testar atributos de qualidade (por exemplo, usabilidade, segurança, compatibilidade e acessibilidade) a E VAV.tst.10 Teste de regressão c E VAV.tst.11 Ferramentas de teste e automação a E VAV.tst.12 Teste de interface do usuário k E VAV.tst.13 Teste de usabilidade a E VAV.tst.14 Teste de desempenho k E VAV.par Análise de problemas e relatórios 5 VAV.par.1 Analisando relatórios de falha c E VAV.par.2 Técnicas de depuração e isolamento de falhas a E VAV.par.3 Análise de defeitos (por exemplo, identificação do produto ou raiz do processo causa para injeção de defeito crítico ou detecção tardia) k E VAV.par.4 Rastreamento de problemas c E	- V&V terminologia e fundamentos -Comentários e análise estática teste - Análise de problemas e relatórios
Qualidade de Software QUA 10 Conceitos e cultura de qualidade de software QUA.cc 2 QUA.cc.1 Definições de qualidade k E QUA.cc.2 Preocupação da sociedade com a qualidade k E QUA.cc.3 Os custos e impactos da má qualidade k E QUA.cc.4 Um modelo de custo da qualidade c E QUA.cc.5 Atributos de qualidade para software (por exemplo, confiabilidade, usabilidade e segurança) k E QUA.cc.6 Papéis de pessoas, processos, métodos, ferramentas e tecnologia k E QUA.pca Garantia de processo 4 QUA.pca.1 A natureza da garantia de processo k E QUA.pca.2 Planejamento da qualidade k E QUA.pca.3 Técnicas de garantia de processo k E QUA.pda Garantia do produto 4 QUA.pda.1 A natureza da garantia do produto k E QUA.pda.2 Distinções entre garantia e V&V k E QUA.pda.3 Modelos de produtos de qualidade k E QUA.pda.4 Análise de causa raiz e prevenção de defeitos c E QUA.pda.5 Métricas e medição de produtos de qualidade c E QUA.pda.6 Avaliação dos atributos de qualidade do produto (por exemplo, usabilidade, confiabilidade e disponibilidade) c E	- Conceitos e cultura de qualidade de software - Garantia de processo - Garantia do produto

Unidades de Contexto (Continuação)	**Unidades de Registro**
Processo de Software PRO 33 PRO.con Conceitos de processo 3 PRO.con.1 Temas e terminologia k E PRO.con.2 Infraestrutura de processo de engenharia de software (por exemplo, pessoal, ferramentas e treinamento) k E PRO.con.3 Modelagem e especificação de processos de software c E PRO.con.4 Medição e análise de processos de software c E PRO.con.5 Melhoria do processo de engenharia de software (individual, equipe e organização) c E PRO.con.6 Análise e controle de qualidade (por exemplo, prevenção de defeitos, processos de revisão, métricas de qualidade e análise de causa raiz de problemas críticos defeitos para melhorar processos e práticas) c E PRO.con.7 Modelos de ciclo de vida de engenharia de sistemas D PRO.imp Implementação do processo 8 PRO.imp.1 Níveis de definição do processo (por exemplo, organização, projeto, equipe e indivíduo) k E PRO.imp.2 Características do modelo de ciclo de vida (por exemplo, baseado em plano, incremental, iterativo e ágil) c E PRO.imp.3 Processo de software individual (modelo, definição, medição, análise e melhoria) a E PRO.imp.4 Processo de equipe (modelo, definição, organização, medição, análise e melhoria) a E PRO.imp.5 Implementação de processos de software no contexto da engenharia de sistemas k E PRO.imp.6 Adaptação do processo k E PRO.imp.7 Efeito de fatores externos (por exemplo, contrato e requisitos, padrões e práticas de aquisição) em processo de software k E PRO.pp Planejamento e acompanhamento de projetos 8 PRO.pp.1 Gerenciamento de requisitos (por exemplo, lista de pendências do produto, prioridades, dependências e mudanças) a E PRO.pp.2 Estimativa de esforço (por exemplo, uso de dados históricos e técnicas de estimativa baseadas em consenso) a E PRO.pp.3 Divisão do trabalho e agendamento de tarefas a E PRO.pp.4 Alocação de recursos c E PRO.pp.5 Gestão de riscos (por exemplo, identificação, mitigação, remediação e rastreamento de status) a E PRO.pp.6 Métricas e técnicas de rastreamento de projetos (por exemplo, valor agregado, velocidade, gráficos de burndown, rastreamento de defeitos e gerenciamento de dívida técnica) a E PRO.pp.7 Autogestão da equipe (por exemplo, acompanhamento de progresso, dinâmica alocação de carga de trabalho e resposta a questões emergentes) a E Gerenciamento de configuração de software PRO.cm 6 PRO.cm.1 Controle de revisão a E PRO.cm.2 Gerenciamento de liberação c E PRO.cm.3 Ferramentas de gerenciamento de configuração c E PRO.cm.4 Construir processos e ferramentas, incluindo testes automatizados e integração contínua a E PRO.cm.5 Processos de gerenciamento de configuração de software k E PRO.cm.6 Problemas de manutenção k E PRO.cm.7 Distribuição e backup D PRO.evo Evolution processos e atividades 8 PRO.evo.1 Conceitos básicos de evolução e manutenção k E PRO.evo.2 Trabalhando com sistemas legados k E PRO.evo.3 Refatoração c E	- Conceitos de processo - Implementação do processo - Planejamento e acompanhamento de projetos - Gestão de configuração de software - Evolution processos e atividades
Segurança da SEC SEC.dev Desenvolvendo software seguro 8 SEC.dev.1 Construindo segurança no ciclo de vida de desenvolvimento de software c E SEC.dev.2 Segurança na análise e especificação de requisitos a E SEC.dev.3 Princípios e padrões de design seguros a E SEC.dev.4 Técnicas de construção segura de software a E SEC.dev.5 Verificação e validação relacionadas à segurança a E	Desenvolvimento de software seguro

Fonte: Elaboração Própria

Apêndice C

Aqui são demonstradas todas as unidades de contexto e unidades de registro que estruturam a codificação dos Estudos Selecionados que versam sobre Uso de FLOSS no Ensino de Engenharia de Software

ANÁLISE DE CONTEÚDO DE TRABALHOS SOBRE USO DE FLOSS NA EES

Tabela C.1 Exploração do Trabalho ID-89

A. Villarrubia ; H. Kim			
Título	Building a community system to teach collaborative software development	ID	89
Unidades de Contexto		Unidades de Registro	
Versão reduzida de Complexidade e Infraestrutura do processo de desenvolvimnto de Software do Mundo Real		Complexidade e Infraestrutura de Desenvolvimento	
Participação dos estudantes em diversas atividades de aprimoramento técnico e sociotécnico nos projetos independente do seu nível de formação no curso.		- Participação dos Estudantes nas atividades - Nível de Formação no curso	
Produtos desenvolvidos nos Projetos podem motivar os discentes e atrair para a comunidade de desenvolvimento		Motivação dos discentes	
Complexidade de Projetos OSS podem ser um Obstáculo na Atratividade dos alunos para as atividades		- Complexidade do Projeto - Atratividade para as atividades	
Pode haver Resistência à mudanças no aspecto curricular		Mudanças Curriculares	
Alunos podem necessitar de tempo desprendido para aprender o uso de ferramentas de controle.		-Tempo para aprender -Ferramentas de Controle	
Necessidade de apresentação prévia dos alunos às ferramentas envolvidas nos princípios de comunidade de desenvolvimento OSS		- Apresentação prévia dos alunos - Ferramentas de princípios de comunidade Floss	
Necessidade de desenvolver um Método de Avaliação e acompanhamento do Aluno em Trabalho Colaborativo por meio de OSS		- Método de avaliação e acompanhamento	

Fonte: (VILLARRUBIA; KIM, 2015)

Tabela C.2 Tabela de Exploração do Trabalho ID-93

Smrithi Rekha V ; V. Adinarayanan			
Título	An Open Source approach to enhance industry preparedness of students	ID	93
Unidades de Contexto		Unidades de Registro	
Projetos permite indicação de parâmetros de avaliação para as atividades desenvolvidas		Parâmetros de avaliação das atividades	
Necessidade de Programação conhecida pelos estudantes		Programação conhecida	
Tamanho do Projeto com complexidade Administrável		Tamanho do Projeto	
Necessidade prévia de Permissões Técnicas ou Acadêmicas para a execução das atividades		Permissões técnicas ou acadêmicas prévias	

(ADINARAYANAN et al., 2014)

Tabela C.3 Tabela de Exploração do Trabalho ID-106

M. Dorodchi ; N. Dehbozorgi			
Título	Utilizing open source software in teaching practice-based software engineering courses	ID	106
Unidades de Contexto		Unidades de Registro	
Alunos tem noção de Aprimoramento nas habilidades com computação		Noção de aprimoramento	
Alunos sentem conforto em trabalhar com outros profissionais.		Conforto em trabalho em equipe	
Iniciar o desenvolvimento da comunicação em comunidade Open-Source		Comunicação em Comunidade em open-source.	
Familiaridade com Ferramentas Gratuitas		Familiaridade com Ferramentas	
Contato com Projetos Complexos e intervenção nestes		ação em Projetos complexos	
Perda de confiança em lidar com Projetos Muito Complexos e Grandes do Mundo real		-Perda de Confiança -Projetos Grandes e Complexos	
Cultura OSS pode ser difícil tanto para o professor como para o aluno no processo de ensino-aprendizagem		Cultura OSS	

Fonte: (DORODCHI; DEHBOZORGI, 2016)

Tabela C.4 Tabela de Exploração do Trabalho ID-115

V. Krishnamoorthy ; B. Appasamy ; C. Scaffidi			
Título	Using Intelligent Tutors to Teach Students How APIs Are Used for Software Engineering in Practice	ID	115
Unidades de Contexto		Unidades de Registro	
Participar do prjeto auxilia a percepção de funções e noções teóricas de engenharia de software aprimoradas e vistas na prática		Percepções de funções e teorias na prática	
Possibilidade de Consulta a repositórios para obter assistência e conteúdos adicionais para alunos com dificuldade de aprendizagem baseada em projetos		Consulta a Repositórios	

(KRISHNAMOORTHY; APPASAMY; SCAFFIDI, 2013)

Tabela C.5 Tabela de Exploração do Trabalho ID-135

S. Gokhale ; T. Smith ; R. McCartney			
Título	Teaching software maintenance with open source software: Experiences and lessons	ID	135
Unidades de Contexto		Unidades de Registro	
Os projetos OSS abrangem uma variedade de domínios, como ferramentas para desenvolvimento de software, análise financeira, segurança e rede, manipulação e visualização de dados.		Ferramentas de Vários Domínios	
Estrutura Social do OSS permite escalabilidade dos projetos com qualidade.		Estrutura social do OSS	
Os alunos aprenderam uma ampla Gama de habilidades e aprenderam a compensar deficiências de habilidades e de conhecimento em curto espaço de tempo.		Percepção de habilidades e deficiências compensadas	
OSS tem vocação para atrair participantes comprometidos com as reais aplicações dos princípios de ES		Vocação para atrair participantes comprometidos com a ES	
Projetos incompletos ou em andamento fornece flexibilidade para novas intervenções.		Projeto em estado inacabado ou em andamento	
Projetos menores podem não fornecer dificuldade na compreensão e aprimoramento, limitando os desafios de aprendizagem.		Projetos menores não dificulta ou desafia a aprendizagem	
Projetos muito grandes podem estar alem das habilidades dos estudantes.		Projetos grandes	
		Habilidades dos Estudantes	
Não são todos os projetos que podem desenvolver os alunos sobre os aspectos de aprendizagem em manutenção de software		Restrição de Projeto a aprender Manutenção de Software	
Frustração inicial dos alunos substituídas por um senso saudável de orgulho ao conseguir desenvolver atividades nos projetos.		Frustração e posterior senso de orgulho.	
Projetos com Arquitetura Fragil inibe profundamente o envolvimento dos estudantes nas atividades de manutenção de software.		Arquitetura Frágil inviabiliza participação na atividade	
Projetos incompletos ou em andamento pode parecer assustador e inibir a confiança		Projeto em estado inacabado ou em andamento	
		Assustar ou inibir a confiança dos alunos	
Necessidade de volume de código, de complexidade e qualidades razoáveis para aprendizagem em Projetos de ES Baseados em Manutenção		- Volume de código, complexidade - Qualidade razoável	
Os estudantes participantes do estudo tinham experiência limitada em programação Java.		Experiência em programação Java	
A escolha do projetos OSS para ensinar foi baseado na proporcionalidade da preparação dos alunos		Decisões baseadas em Preparação	
Projetos escolhidos foram modulares e com levemente documentado.		Projetos modulares e documentados	
As atividades de manutenção desenvolvidas foram desafiadoras, com intensivo uso de recursos e com consumo de tempo, significativamente apoiado por dodumentação e comentários e facilitado por boas arquiteturas.		Intensivo uso de tempo	
		Projeto documentado e com boa arquitetura	
Frustração inicial dos alunos ao se deparar com os projetos		Frustração inicial	
Estado do Projeto, projetos devem fornecer necessidades de melhorias com dificuldades variadas para os alunos desenvolverem atividades. Que não sejam excessivamente difíceis e que seja progressiva durante o semestre.		Estado do projeto com necessidades de melhorias	
Projetos totalmente polidos que também não sugerem melhorias óbvias de funcionalidade podem ser menos satisfatórios aos alunos.		Projetos polidos são poucos satisfatórios	
O Assunto dos projetos escolhidos, como música, jogos, produtividade geral e questões humanísticas influenciam drasticamente no envolvimento dos estudantes		Assunto do projeto influencia no envolvimento	
Experiência com programação foi usada pelos docentes para avaliar a estimativa de impacto do código do projeto e seu aprimoramento pelos alunos.		Experiência com programação	
Projetos para participação do estudante devem considerar maturidade, estabilidade, crescimento potencial e apoio aos objetivos de aprendizagem.		Considerar maturidade, estabilidade e crescimento do projeto	

Fonte: (GOKHALE; SMITH; MCCARTNEY, 2013b)

Tabela C.6 Tabela de Exploração do Trabalho ID-1088

K. Buffardi			
Título	Localized open source collaboration in software engineering education	ID	1088
Unidades de Contexto		Unidades de Registro	
Envolvimento do estudante em open source pode estimular aprendizagem através de prática construtivista		Envolvimento pela prática construtisvista	
Envolve os Estudantes em causas altruístas e de interesse coletivo		Causas Altruístas e de Interesse coletivo	
Projetos Floss Inativos ou considerados "projetos de brinquedo"podem não associar a prática acadêmica com a indústria		Projetos inativos ou de brinquedo	
Recomenda-se que o envolvimento do estudante deve ser voluntario.		Necessidade de envolvimento voluntário	
Metodologia de escolha do projeto deve ser pensado sobre escolha voluntaria ou indicado pelo professor		Escolha voluntaria ou indicada pelo professor.	

Fonte: (BUFFARDI, 2015)

Tabela C.7 Tabela de Exploração do Trabalho ID-1097

G. W. Hislop ; H. J. C. Ellis			
Título	Humanitarian Open Source Software in Computing Education	ID	1097
Unidades de Contexto		Unidades de Registro	
Estimula experiência com complexidade de projetos de porte real e deixam o professor sobre controle das ações		- Complexidade de projetos de porte real - Professor controla as ações	
Estimular compreensão de responsabilidade social e percepção de valor em contribuir com a comunidade		- Responsabilidade Social -Contribuição com a comunidade	
Possibilita o professor usar o projeto em escalas de atividades maiores ou menores de acordo com sua experiência de ensino usando FLOSS		- Escolha de Projetos pelo professor -Sua Experiência de Ensino usando Floss	
Baixo impacto dos FLOSS na escolha do plano de carreira dos alunos (Provavelmente por que estudos abordam estudantes já decididos finalizando cursos)		Percepção do impacto de FLOSS nas decisões de carreira	
A abordagem do estudo usando FLOSS pelo professor interfere no seu controle sobre as atividades e os limites de uso do recursos do Projeto		Abordagem docente interfere no seu controle sobre atividades	

Fonte: (HISLOP; ELLIS, 2017)

Tabela C.8 Tabela de Exploração do Trabalho ID-1192

G. C. Diniz ; M. A. G. Silva ; M. A. Gerosa ; I. Steinmacher			
Título	Using Gamification to Orient and Motivate Students to Contribute to OSS Projects	ID	1192
Unidades de Contexto		Unidades de Registro	
Floss auxilia no Engajamento de estudantes em atividades e processos próximos à realidade de desenvolvimento da indústria de software.		Engajamento do estudante em atividades e processos	
Alunos Recém-Chegados ao mundo Open Source podem encontrar problemas de orientação que os desmotivem a fazer a sua primeira contribuição		- Alunos Recém-Chegados -Problemas de Orientação -Desmotivação em contribuir	
A organização de tarefas sequenciais e objetivas possibilita menos rejeição e impacto menor a estudantes novatos no uso de OSS.		- Organização de tarefas sequenciais e objetivas -Rejeição e impacto - Estudantes novatos em OSS	

Fonte: (HISLOP; ELLIS, 2017)

Tabela C.9 Tabela de Exploração do Trabalho ID-1193

H. J. C. Ellis ; M. Chua ; G. W. Hislop ; M. Purcell ; S. Dziallas			
Título	Towards a model of faculty development for FOSS in education	ID	1193
Unidades de Contexto		Unidades de Registro	
Diminue barreiras de acesso ao código e permite que novos estudantes possam entrar no ambiente OSS através da observação		-Barreiras de Acesso ao código - Entrar no ambiente OSS pela observação	
Ambientes Floss estimulam a participação de mulheres na comunidade		-Participação de Mulheres	
HFOSS estimula os estudantes a estudar e desenvolver as habilidades devido a motivação altruísta		- Motivação Altruísta	
Tantos Alunos como professores precisam desenvolver uma curva de aprendizado de elementos que envolve o uso de FLOSS para se tornarem produtivos na comunidade como ferramentas de comunicação e controle no projeto		-Curva de Aprendizado -Ferramentas de Comunicação e controle	
Colaboradores precisam aprender como a aplicação de ferramentas montam a estrutura de projeto funcional para a comunidade.		-Aplicação de Ferramentas	
Possibilidade alta de Incompatibilidade de cronograma, processos e objetivos entre FOSS e aulas acadêmicas		-Incompatibilidade de Cronograma processo e objetivo com aulas.	
Atenção a possibilidade de Falta de Experiência de Instrutores com desenvolvimento de sofwtare de grande escala.		Falta de Experiência de Instrutores	
Possibilidade de Falta de Experiência de participantes da comunidade FOSS com cursos de computação.		Falta de experiencia da comunidade com cursos	
Instrutores podem ter dificuldade em desenvolver conhecimento local suficiente de um projeto FOSS.		Instrutores podem ter dificuldade de conhecimento local de um projeto,	
Projetos FOSS podem não fornecer documentação e orientação para novos participantes de forma suficiente. (14/4)		-Falta de Documentação e Orientação - Novos participantes	
Participar de Projeto Real pode gerar eventos imprevisíveis e gerar surpresas e perda de controle para o instrutor.		Surpresas e perda de controle para o instrutor.	
Resistência de outros instrutores à adoção da participação em HFOSS para o ensino		Restrição de outros Instrutores a adoção de HFOSS	
Contribuintes precisam conhecer a cultura básica do projeto escolho.		Cultura básica do projeto	
Frequencia de trabalho acadêmico usando o mesmo projeto e necessidade de mudança de materiais e atribuiçoes a cada aula.		- Frequência de Uso do floss pelo professor	
Barreiras técnicas para a participação dos alunos em projetos HFOSS		Barreiras técnicas para participação dos alunos	
Necessidade de Suporte do instrutor para a participação do aluno no HFOSS		Suporte do Instrutor para participação	
Necessidade de Diretrizes para membros do corpo docente ao envolver os alunos em projeto HFOSS		-Diretrizes ao corpo docente na inserção do aluno em projetos.	
Necessidade de Abordagem para seleção de Projetos HFOSS para uso em uma classe		Abordagem de seleção para projetos	
Limitações de tempo para adequação do Projeto HFOSS ao ensino		Limitações de tempo	
Limitações de suporte do Setor de TI para participação em HFOSS como atividade acadêmica.		Suporte do Setor de TI	
Capacidade de modificar currículos para adoção da participação dos alunos em HFOSS.		Modificar Currículos	

Fonte:(ELLIS et al., 2013)

Tabela C.10 Tabela de Exploração do Trabalho ID-4503

MacKellar, B.K. ; Sabin, M. ; Tucker, A.B.			
Título	Bridging the academia-industry gap in software engineering: A client-oriented open source software projects course	ID	4503
Unidades de Contexto		Unidades de Registro	
Projetos de Código Aberto agregam ferramentas e práticas padrão que tendem a dar certo com estudantes, como ferramentas tambem de código aberto e acessíveis ao ambiente acadêmico.		Agregam Ferramentas de código aberto e acessíveis	
O uso de Projetos de Código Aberto Obrigam os Alunos a conhecerem técnicas e ferramentas de desenvolvimento distribuído, como repositórios de projetos, controle de versão e rastreamento de problemas on-line.		Alunos necessitam conhecem ferramentas de desenvolvimento distribuído	
O uso de Código Aberto estimula o uso de ferramentas para comunicação na prática de desenvolvimento, ferramentas que por sua vez podem auxiliar o docente no rastreamento das interações dos estudantes.		- Uso de ferramentas para a comunicação - Auxiliar docente no rastreamento das interações	
Algumas atividades podem não ser triviais pois os alunos podem não ter experiência prévia com técnicas ou métodos comuns nas atividades envolvendo FOSS como compartilhamento de código.		- Atividades podem não ser triviais -Alunos sem experiência prévia com métodos	
Projetos de Código Aberto tem reputação de serem mal documentados, o que pode ser indesejável numa abordagem para o Ensino de Engenharia de Software e exigirá muito do instrutor para estabelecer padrões de documentação.		-Projetos podem ser mal documentados -Projeto pode ser indesejável pra ensinar ES -Exigir esforço do instrutor para padronizar documentação	
Instrutores tem problemas em avaliar contribuições dos alunos em projetos de equipe, principalmente contribuições individuais em atividades feitas em grupo.		Problemas em avaliar contribuições individuais em atividades em grupo	
Alunos podem ter várias restrições a participação constante nas contribuições do projeto, restrições impostas por trabalho e questões pessoais.		Restrições de participação constante no projeto por questões pessoais e de trabalho.	
O docente deve administrar a conclusão do Projeto ou das atividades desenvolvidas neste dentro do prazo acadêmico de acordo com o objetivo do seu uso.		Docente deve administrar prazo da conclusão do projeto ou atividade	
Docentes devem auxiliar os alunos com materiais suficientes para o aprendizado em organização de sua estrutura de ferramentas e ambientes onde as atividades serão desenvolvidas		- Docentes devem auxiliar alunos com materiais para aprendizado. - Estrutura de ferramentas e ambientes onde atividades serão desenvolvidas.	
A abordagem de uso de FOSS exige redimensinamento dos projetos usados ao tamanho e complexidade compatível com o perfil dos alunos executantes das atividades.		-Redimensionamento de tamanho e complexidade - Compatibilidade com o perfil dos alunos	
É necessário adaptar os objetivos do uso de projetos ao currículo de acordo com o perfil dos estudantes, como o tamanho das equipes, experiência com tamanho e complexidade de códigos e experiência com colaboração em projeto de software.		-Adaptar objetivos do uso do projeto no ensino - Perfil dos estudantes - experiência com tamanho e complexidade - experiência com colaboração	

Fonte: (MACKELLAR; SABIN; TUCKER, 2015)

Tabela C.11 Tabela de Exploração do Trabalho ID-4663

Buffardi, K.			
Título	Localized open source software projects: Exploring realism and motivation	ID	4663
Unidades de Contexto		Unidades de Registro	
Fornecer liberdade de escolha dos estudantes a cerca dos Projetos de Software para o curso pode motivar o aluno a trabalhar naquilo(domínio) que gostam.		- Fornecer liberdade de escolha - Domínio que gostam - Motivar a participação	
Trabanho com HFOSS pode possibilitar melhor percepção do impacto social da computação e desenvolvimento do senso altruísta		- Percepção do impacto social da computação - Desenvolver senso altruísta	
Trabalhar com mentores Profissionais de Software nos Projetos Auxiliou os estudantes a adquirirem hábitos, habilidades e comunicação profissionais.		- Mentoria profissional auxilia estudantes	
O uso de HFOSS pode se mostrar desafiador a instrutores e estudantes devido a imprevisibilidade da comunicação e colaboração devido a distribuição global dos colaboradores.		- Comunicação e colaboração imprevisível desafia docente e aluno	
Uso de Projetos para curso de Engenharia de Software em períodos semestrais mudam o foco em diretrizes rígidas para foco em processos no ciclo de vida do desenvolvimento.		- Mudança do foco em diretrizes para foco em processos em cursos semestrais	

Fonte: (BUFFARDI, 2016)

Tabela C.12 Tabela de Exploração do Trabalho ID-4811

Bowring, J. ; Burke, Q.			
Título	Shaping software engineering curricula using open source communities	ID	4811
Unidades de Contexto		Unidades de Registro	
Importante se atentar a falta de experiência em requisitos prévios para o domínio do ambiente de desenvolvimento do mundo real, como controle de versão, teste, script, sistemas de construção e lançamento e gerenciamento de projetos.		- Se atentar a falta de experiência - Ambiente e Ferramentas	
Interessante introduzir os conceitos de código aberto e familiaridade com o ambiente H/FOSS para auxiliar envolvimento dos estudantes.		- Introduzir conceitos de Open Source - Familiarizar com Ambiente H/FOSS	
Uso de recursos didáticos de apoio de aprendizagem como blogs e livros pode auxiliar no uso de projetos.		- Uso de recursos de apoio a aprendizagem auxilia no uso de projetos.	

Fonte: (BOWRING; BURKE, 2016)

Tabela C.13 Tabela de Exploração do Trabalho ID-4815

Fernandes, S. ; Barbosa, L.S.		
Título	Collaborative environments in software engineering teaching: A FLOSS approach	ID 4815
Unidades de Contexto		Unidades de Registro
Atividades em FLOSS permite definir perfil do estudante e os graus de conhecimento nas técnicas de desenvolvimento de acordo com os papéis desenvolvidos		Percepção do perfil do aluno de acordo com os papéis
Atividades desenvolvidas em projetos iguais a outros já estudados ou compatíveis com experiências anteriores facilita a interações e dinâmicas.		Interação facilitada por uso de projetos próximos a experiências anteriores
Grupos foram ativos usando as ferramentas de interação para troca de ideias, dúvidas e conquistas.		Ferramentas de interação facilita grupos ativos
Estudantes se sentem motivados a expor suas experiências e resultados em eventos acadêmicos abertos a sua comunidade.		Motivação em expor resultados
Possibilita observar o desenvolvimento que o "Aprender Fazendo"causa no processo de aprendizagem dos estudantes		Percepção da aprendizagem observando práticas
Alunos quando pressionados nas tarefas mostram esforço para ultrapassar seus próprios limites		Alunos percebem e buscam ultrapassar seus limites
Alunos ficaram orgulhosos do trabalho que realizaram e viram como interessante os elogios e motivações de pessoas do mundo todo		Senso de orgulho e motivação vinda de pessoas do mundo todo
Quase todos os alunos vêem seu trabalho válido e apreciado		Aluno percebe valor de seu trabalho
Ingressantes nas atividades com Floss apresentam Formação modesta comparado com habilidades de programação de membros da comunidade FLOSS		Habilidades de programação diferentes entre iniciantes e experientes em FLOSS
Dificuldade dos estudantes em estabelecer conexão com uma comunidade Floss escolhida.		Dificuldade de comunicação com comunidade
Incompatibilidade de compreensão entre o interesses dos estudantes com a comunidade FLOSS		Choque de compreensão de interesses entre alunos e comunidade FLOSS
Incompatibilidade entre a interconexão dos estudantes com a comunidade FLOSS no que tange modelo de respostas e compreensão mutuas		Compreensão e interconexão incompatíveis entre alunos e comunidade
Gestores dos Projetos da Comunidade podem ter resistência com a contribuição de curto prazo dos participantes.		Gestores dos projetos da comunidade resistir a contribuição dos alunos
Alunos com mais experiência profissional e mais velhos tendem a ser mais conservadores e evitam "pensar fora da caixa"		Alunos com experiência profissional podem evitar novas experiências
Contribuintes dos Projetos FLOSS não estão dispostos a ensinar, mas discutir ideias e problemas e mostrar que sabem resolver os problemas		Contribuintes resistem ensinar e focam em discutir e resolver problemas
A relação da comunidade com os grupos podem variar, com demora nas resposas, boa recepção, atenção somente após os primeiros commits, solicitação de mais tarefas após perceber seriedade nas tarefas já desenvolvidas		Relação da comunidade com os alunos pode variar
Alunos precisam ter uma boa base de desenvolvimento de software e programação e comunidades devem ser consideradas apenas como apoiadoras;		Base de programação e desenvolvimento
Habilidades em programação tem influência no engajamento das funções exercidas nas atividades		- Habilidades de programação - Engajamento das funções nas atividades
É importante garantir interação relevante com a comunidade FLOSS		Interação com a comunidade Floss
Compatibilidade entre Perfil da Comunidade FLOSS com a proposta das atividades com os estudantes		Atividades dos alunos devem atender perfil da comunidade.
É importante inserir nos estudantes a consciência de participação, análise e estudo		Consciência de participação, análise e estudo.
Pequenos grupos são mais capazes de especializar os membros uma tarefa particular a ser desenvolvida		Formação dos grupos define perfil dos membros
O uso das atividades a fim de avaliação acadêmica estimulou a proatividade mesmo quando enfrentaram difuculdades de interação e receptividade da comunidade escolhida.		-Proatividade devido a avaliação acadêmica -Interação e receptividade da comunidade
Alunos com formação mais interdisciplinar tendem escolher tarefas como analistas em vez de desenvolvedor ou analista de código		Perfil de formação do aluno influencia na sua participação
Formação acadêmica e profissional limitou a escolha de Tarefas		Formação do aluno limitou sua escolhas
Alunos com dificuldade em Engenharia de Software foram capazes de identificar seus pontos fracos, mas também onde e como podiam contribuir melhor		Percepção do aluno com seus pontos fracos

Fonte: (FERNANDES; BARBOSA, 2016)

Tabela C.14 Tabela de Exploração do Trabalho ID-4966

Ellis, H.J.C. ; Hislop, G.W. ; Jackson, S. ; Postner, L.			
Título	Team project experiences in humanitarian free and open source software (HFOSS)	ID	4966
Unidades de Contexto		Unidades de Registro	
Projetos com natureza Altruísta parece fazer com que a comunidade seja mais acolhedora		-Assuntos altruístas -Comunidade Acolhedora	
Projetos com natureza Altruísta parecem ser mais úteis para alunos participantes		-Assuntos altruístas	
A proposta humanitária e de mundo real dos HFOSS são importantes para atrair o interesse de estudantes, principalmente mulheres		- Proposta Humanitária - Atrair interesse do aluno - Atrair participação feminina	
Alunos podem adquirir habilidades técnicas não comumente ensinadas em sala de aula, como manutenção de código, ferramentas de desenvolvimento e controle de versão.		Aprender Ferramentas incomuns	
O desenvolvimento distribuído permite aos alunos oportunidade única de interação com profissionais de várias localidades e culturas para desenvolver software.		Interação com profissionais de vários locais	
Os projetos HFLOSS e seus variados tamanhos e complexidades permitem os alunos compreender como a complexidade e o tamanho afeta na opção de desenvolvimento.		Tamanho e complexidade na opção de desenvolvimento	
HFLOSS são projetos em andamento e com ciclo de vida longo, o que permite ao estudante entender melhor o processo de manutenção e tomadas de decisão ao longo da vida útil.		Projetos em andamento em com ciclo de vida longo	
Projetos HFLOSS normalmente usam processo ágeis de software que os alunos devem aprender para participar, além de muitas abordagens de desenvolvimento		Necessidade de aprendizagem prévia em processos ágeis	
Aprendizagem no HFLOSS estimula a conscientização e responsabilidade social, além de permitir que os alunos observem suas habilidades de computação para ajudar os outros.		Observar suas habilidades de computação	
Uma base real de clientes e o propósito Humanitário torna os HFLOSS eficazes em promover Motivação aos estudantes		-Motivação dos estudantes - Propósito Humanitário do HFLOSS	
HFLOSS também beneficia os instrutores, pois auxilia os mesmos a manterem-se atualizados sobre tendencias e inovações.		Ajuda instrutor a manter-se atualizado	
O auxilio dos grupos aos estudantes no que tange o uso de ferramentas e abordagens mais modernas nas atividades desenvolvidas auxilia a aprendizagem e reduz a pressão sobre o professor		-Auxílio da comunidade nas atividades - Uso de Ferramentas e abordagens -Reduzir pressão sobre o docente	
O esforço inicial na seleção de um projeto pode beneficiar as atividades de aprendizagem em uma série de cursos e não somente em um curso		Esforço na escolha do projeto pelo professor pode beneficiar atividades	
Alunos podem se sentir motivados a trabalhar em comunidades diversificadas e ter aumento da confiança sobre sua habilidade de computação		-Motivação em comunidades diversificadas -Confiança em do aluno em sua habilidade	
A expectativa de estudantes em um aprendizado estruturado e previsível pode ir de encontro com a realidade da aprendizagem aberta e menos estruturada comuns em Projetos HFLOSS		Choque de expectativas de aprendizado em ambiente menos estruturado	
Alunos podem confundir os papeis e a autoridade entre o instrutor e a comunidade nas atividades.		Confusão dos papéis e autoridade entre docentes e comunidade.	
Alunos Enfrentam várias curvas de aprendizado ao entrar em um Projeto FLOSS, estas são influenciadas pela complexidade do projeto, pela compreensão da cultura de FOSS e necessidade de aprendizado de domínio e ferramentas.		-Curva de Aprendizagem -Familiarizar com ferramentas -Complexidade do projeto -Compreensão da cultura FOSS -Aprendizagem de domínio	
Alunos podem não estar familiarizado com ferramentas, abordagens ou linguagens usadas no projeto e pode precisar aprender esses elementos simultaneamente.		Necessidade de aprender Ferramentas e abordagens	
Professores também podem enfrentar curvas de aprendizagem para ferramentas, abordagens e cultura FOSS, bem como conhecimento de domínio.		-Curva de aprendizagem do professor -Ferramentas e Abordagens -Cultura FOSS -Conhecimento de domínio	
Sincronização de cronogramas entre o calendário acadêmico e o ciclo de lançamentos de HFOSS podem enfrentar problemas se os estudantes contribuírem com o código.		Problemas entre sincronização de calendário acadêmico com ciclo de lançamentos do HFOSS	
A imprevisibilidade pode ser um desafio se ocorrer mudanças drásticas no projeto no meio do prazo, como mudanças nos tipos de contribuições aceitas pela comunidade.		Imprevisibilidade das ações da comunidade	
Em alguns casos, um projeto HFOSS pode fornecer um artefato de estudo com mínima ou nenhuma interação com a comunidade HFOSS		Possibilidade de artefatos de estudo sem interação com a comunidade	
Em alguns casos,os alunos podem se tornar membros da comunidade HFOSS e contribuir para ele como parte de uma aula ou outra atividade acadêmica.		Alunos podem ser membros da comunidade como parte de aula ou atividade	
As pessoas tendem a pensar nas primeiras contribuições dos alunos como contribuições de código, mas é essencial lembrar que Projetos OSS executam toda gama de atividades de engenharia de software de forma transparente.		Contribuição dos alunos podem ser de várias atividades de ES, não apenas código.	
Projetos OSS permite interação com profissionais de desenvolvimento e trabalhar com aspectos não codificados (Especificação, Planejamento, Suporte, Teste)		Projetos permite interação com profissionais	
Projetos OSS permite que estudantes aprendam aspectos não técnicos como analisar produtos, investigar uso de uma ferramenta dentro do projeto, explorar normas sociais de comunicação em comunidades etc.		-Possibilidade de Investigar ferramentas -Explorar normas de interação	
A aprendizagem do aluno em uma comunidade Floss é uma forma de partipação periférica legítima, onde são apresentados à comunidade em etapas, inicialmente assistindo e observando e progredindo até se tornar membro.		Participação dos alunos na comunidade em etapas	
Instrutores devem promover a participação dos alunos, sendo necessário aprender como criar atribuições, resultados e rúbricas de avaliações para eles.		Docentes devem promover a participação do aluno e criar critérios de avaliação nos projetos	
Professores devem negociar comunicação e suporte com a comunidade HFOSS, indicando onde a comunidade irá contribuir com a aprendizagem e onde os alunos irão contribuir com a comunidade		- Professores devem negociar comunicação com a comunidade HFOSS	
Os instrutores também precisam estar atentos aos potenciais Direitos Educacionais da Família e à Lei de Privacidade (FERPA) e às questões de propriedade intelectual.		Atentar a Direitos Educacionais da Família Lei de Privacidade e Propriedade Intelectual	
A seleção do projeto é crítica ao usar HFLOSS em um curso, o projeto precisa se ajustar aos objetivos de aprendizagem, estejam abertos a contribuições, seja receptivos aos alunos e tenham bons mecanismos de comunicação.		-Projetos devem ser compatíveis com objetivos de aprendizagem -Receptividade aos Alunos -Bons mecanismos de comunicação	
Normalmente professores encontram dificuldades em relação ao desafio da seleção do projeto quando estes adotam o HFLOSS no ensino pela primeira ou segunda vez		-Dificuldade do professor selecionar projetos quando tem baixa experiência com HFOSS	
Professores precisam planejar como encerrar um semestre com elegância, para que a comunidade entenda o que é realizado e o que não é realizado quando os alunos terminam um semestre.		Compatibilidade entre ciclos de atividades na academia e na comunidade	
As instituições precisam apoiar os esforços de HFOSS fornecendo acesso a aplicativos de código aberto e infraestrutura em laboratórios.		Instituições devem apoiar participação em HFOSS	
A forma como o FOSS é utilizado em sala de aula determina o grau de domínio sobre a seleção controle, acompanhamento e aproximação com a rotina da comunidade.		Forma de uso do FOSS na sala de aula determina a aproximação com a rotina da comunidade	
A complexidade do projeto e abordagem do uso do projeto no ensino pode interferir na motivação e na percepção dos alunos sobre os avanços em suas habilidades		- Complexidade do projeto - Interferência na Motivação - Percepção do Aluno sobre o avanço em suas habilidades	

Fonte:(ELLIS et al., 2015a)

Tabela C.15 Tabela de Exploração do Trabalho ID-5147

Ellis, H.J.C. ; Hislop, G.W. ; Pulimood, S.M. ; Morgan, B. ; Coleman, B.			
Título	Software engineering learning in HFOSS: A multi-institutional study	ID	5147
Unidades de Contexto		Unidades de Registro	
Alunos podem aprender novas ferramentas e linguagens de programação, adquirir experiência profissional e criar uma rede profissional.		Aprender Ferramentas e Linguagens de Programação	
Expõe o aluno a projetos com complexidade significativa.		Exposição a Complexidade significativa	
Alunos que se envolvem em projetos FLOSS tem impacto positivo sobre aprendizagem em Engenharia de Software, ganhando conhecimento sobre ferramentas, processos e visão de complexidade do Projeto.		-Aprendizagem sobre Ferramentas -Visão de Complexidade	
Os membros do corpo docente podem enfrentar desafios ao negociar a comunicação e o suporte com a comunidade HFOSS para selecionar um projeto associado.		-Desafios docentes ao negociar Comunicação com a Comunidade	
Os membros do corpo docente podem enfrentar desafios para identificar as contribuições dos alunos e ajustar os cronogramas do curso aos cronogramas de lançamento do projeto.		- Desafio docente para identificar contribuições - Ajustar cronograma do curso ao cronograma da comunidade	
Lei de Direitos Educacionais e Privacidade da Família (FERPA), regras de política intelectual e requisitos institucionais podem colocar restrições adicionais sobre a participação dos alunos em projetos HFOSS.		-Restrições na participação dos alunos devido a Leis, regras de política intelectual e requisitos institucionais	
Ferramentas, abordagens, conhecimento de domínio, cultura FOSS e interações profissionais exercem influência na curva de aprendizagem de alunos e professores.		- Curva de Aprendizagem - Ferramentas e Abordagens - Conhecimento de Domínio - Cultura FOSS - Interação com a comunidade	
Alunos podem ter dificuldade em se ajustar a uma abordagem mais flexível de aprendizado a um abiente menos estruturado comparado a sala de aula.		- Dificuldade do aluno em se ajustar a abordagem de aprendizado no FLOSS	
Não existe diferença significativa entre a percepção da capacidade de colaboração, das vantagens e desvantagens do HFOSS, do uso de ferramentas, e tecnicas e compreensão de interações profissionais entre alunos com alta habilidade e baixa habilidade de programação		- Habilidades de Programação - Percepção da capacidade de colaboração - Percepção do uso de Ferramentas - Compreensão da interação profissional	
A percepção sobre a capacidade de conhecer e aplicar os processos de desenvolvimento de software podem ser diferentes para quem trabalhou com Floss de que não trabalhou com Floss.		- Experiência de quem trabalho e não trabalhou com Floss	
A percepção sobre o impacto da complexidade e do tamanho do projeto no desenvolvimento varia para alunos que tiveram ou não exeperiência com FLOSS.		- Impacto da complexidade e do tamanho do Projeto - Experiência com FLOSS	
Mulheres podem se mostrar mais confiantes sobre sua capacidade de participar do planejamento e desenvolvimento de um projeto de software do mundo real		- Confiança sobre capacidade de participar	
Alunos com Habilidades de programação mais altas podem se mostrar mais confiantes para participar de uma comunidade FOSS para desenvolvimento e para manter um projeto HFOSS		- Habilidades de programação - Confiança para participar de Comunidade	

Fonte:(ELLIS et al., 2015b)

Tabela C.16 Tabela de Exploração do Trabalho ID-5328

Mishra, D. ; Hacaloglu, T. ; Mishra, A.			
Título	Teaching software verification and validation course: A case study	ID	5328
Unidades de Contexto		Unidades de Registro	
Alunos se mostraram mais motivados e produtivos ao desenvolver as atividades em equipe e discutir as soluções entre si		- Motivação dos Alunos - Atividades e discussões em Equipe	
Alunos podem ter o desempenho otimizado ao fazer as atividades em problemas de domínio conhecidos		- Melhor desempenho ao conhecer o dominio	
Alunos tiveram dificuldade de compreender a real função do programa na inspeção do código e solicitavam uma explicação prévia da funcionalidade do código.		- Complexidade do código - Necessidade de explicação prévia	
Falta de conhecimento e experiência de requisitos de diferentes domínios de aplicação podem dificultar a compreensão das funções do código.		- Falta de experiência em domínios	
Alunos tiveram dificuldade de compreender especificações de requisitos em Inglês,		Dificuldade com a língua inglesa	
Restrições de tempo podem interferir na compreensão de ferramentas que sejam utilizadas para executar as atividades no software.		- Restrições de Tempo - Compreensão de Ferramentas	
Professores devem primeiramente avaliar a exatidão do programa, pois, devido a limitações de tempo, pode não ser viáveis para testes.		- Docentes devem testar projeto antes de indicar atividades - Limitações de tempo podem inviabilizar atividades	
Trabalhar em pequenos fragmentos de código melhorou o desempenho dos alunos em atividades de teste.		- Atividades em fragmentos de código podem interferir na aprendizagem	
Embora uma ferramenta usada para apoiar as atividades possam ser de fácil uso, ela pode exigir bom conhecimento de programação.		-Uso de Ferramentas para apoiar atividades -Necessidade de bom conhecimento de programação	
Escolher o domínio de problemas baseados em tópicos que os alunos sejam familiarizados podem auxiliar na interferência da baixa habilidade analítica e a baixa experiência com várias áreas de aplicação.		- Decisões do professor - Tópicos familiares aos alunos	
A dificuldade dos alunos em usar uma ferramenta para a execução das atividades pode ser resolvido com introdução do uso da ferramenta antes das atividades.		- Professor pode fazer introdução prévia - Dificuldade em usar ferramentas	

Fonte:(MISHRA; HACALOGLU; MISHRA, 2014)

Tabela C.17 Tabela de Exploração do Trabalho ID-5329

Morgan, B. ; Jensen, C.			
Título	Lessons learned from teaching open source software development	ID	5329
Unidades de Contexto		Unidades de Registro	
Uso do FOSS possibilitou engajamento mútuo, motivação e compatibilidade com as práticas do aprender fazendo.		- Engajamento e motivação devido ao aprender fazendo	
Alunos que interagem com os mentores da comunidade podem achar mais fácil contribuir para um projeto.		- Interação com mentores da comunidade	
Alunos que participam de projetos menores, que ja tiveram alguma experiência ou com muitos mentores relataram menos sobre estarem sobrecarregados.		- Projetos menores - Experiência prévia em projetos - Presença de mentores	
Os recém-chegados ao FOSS geralmente têm dificuldade em encontrar um ponto de entrada em um projeto.		- Inexperientes em FOSS	
Documentação incompleta e / ou desatualizada, nenhuma resposta dos membros da comunidade quando são feitas perguntas e a necessidade de aprender um novo conjunto de ferramentas para participar são barreiras de entrada aos iniciantes em FLOSS.		-Documentação incompleta/desatualizada -Falta de respostas da comunidade - Aprender novo Conjunto de Ferramentas - Barreira na entrada de alunos - Alunos Iniciantes em Floss	
Frustrações com a escolha do projeto devido a falta de documentação ou respostas da comunidade pode interferir no interesse dos alunos pelo FOSS		-Frustração com a escolha do projeto - Falta de documentação -Falta de respostas da comunidade -Interferência no interesse pelo FLOSS	
Projetos grandes e complexos com grande documentação pode deixar alunos sobrecarregados e com dificuldade de encontrar um ponto de partida na contribuição		- Projetos grandes e complexos - Grande documentação - Sobrecarga dos alunos - Inserção dos alunos no FLOSS	
Dar liberdade aos alunos em escolha de projetos pode limitar o apoio da Classe e do Instrutor devido a incompatibilidade de ferramentas, costumes e disponibilidade de documentação.		- Liberdade de escolha aos alunos limita intervenção docente - Incompatibilidade de Ferramentas - Diferenças da cultura de Floss -Disponibilidade de Documentação	
Devido a curta duração do período escolar e um processo de avaliação longo quando envolve vários projetos, a avaliação se baseou na quantidade de esforço e qualidade do processo envolvido ao invés da aceitação das contribuições.		- Curta duração do período escolar - Avaliação baseada na quantidade de esforço não na aceitação das contribuições	
Liberdade de participação em projetos com tamanho extenso pode ser um obstáculo para a participação dos alunos.		- Liberdade de participação dos alunos nos projetos - Projetos extensos -Obstáculo para a participação	
Usar projetos menores e com linguagem na qual os alunos tenham experiência proporciona curva de aprendizado mais moderada.		- Projetos menores - Linguagem que alunos tenham experiência - Curva de Aprendizagem moderada	

Fonte:(MORGAN; JENSEN, 2014)

Tabela C.18 Tabela de Exploração do Trabalho ID-5335

Ellis, H.J.C. ; Jackson, S. ; Hislop, G.W. ; Diggs, J. ; Burdge, D. ; Postner, L.			
Título	Learning within a professional environment: Shared ownership of an hfoss project	ID	5335
Unidades de Contexto		Unidades de Registro	
Trabalhar no FOSS beneficia na percepção do aluno sobre o potencial da computação para o benefício da sociedade		Percepção do aluno sobre potencial da computação.	
Comunidades que apoiam, não julgam e ajudam o estudante auxilia na motivação para participação em FOSS		- Comunidades que apoiam não julgam e ajudam - Motivação para participação em Floss	
Dependência da programação imprevisível de projetos FOSS podem ser uma Desvantagem para uso no ensino.		- Incompatibilidade da rotina do FLOSS para o ensino	
Curva de Aprendizagem com a cultura e Ferramentas do FOSS podem ser um fator limitante para uso do FOSS no ensino		- Curva de Aprendizagem - Cultura do FLOSS - Ferramentas do FLOSS - Limitação do Floss ao ensino	
A primeira exposição do aluno a projetos complexos e considerável podem afetar na confiança da capacidade em computação		- Projetos Complexos -Primeira Experiência do Aluno -Confiança da capacidade em computação afetada	
Atualizar ou Retomar projetos FOSS parados pode gerar curva de aprendizado devido a necessidade de entender as novas versões de cada tecnologia e suas interdependências.		- Continuar projetos parados - Curva de Aprendizagem - Atualizar-se a novas versões de tecnologia	

Fonte:(ELLIS et al., 2014)

Tabela C.19 Tabela de Exploração do Trabalho ID-5343

Krutz, D.E. ; Malachowsky, S.A. ; Reichlmayr, T.			
Título	Using a real world project in a software testing course	ID	5343
Unidades de Contexto		Unidades de Registro	
O Entusiasmo e a Motivação dos alunos ao trabalhar nos projetos estimulam que novos alunos se interessem pela aprendizagem por meio de Projetos FLOSS		- Motivação e entusiasmo dos alunos - Novos alunos se interessem	
O entusiasmo dos professores da turma em relação ao novo projeto e sua natureza variativa inerente pode ser um fator de satisfação dos alunos em relação a aprendizagem nos projetos.		- Entusiasmo dos professores pode gerar satisfação nos alunos	
A liberdade dos estudantes nas decisões sobre suas atividades e autogerenciamento pode gerar problemas como necessidade mais orientação e dificuldade de entregas no prazo estipulado		- Liberdade dos alunos nas decisões - Necessidade de mais orientação - Dificuldade de Entregas no Prazo	
Os projetos escolhido deve ser grande e compléxo o suficiente para desenvolver as atividades de teste de tamanho adequado, mas não grante ao ponto de sobrecarrecar as equipes de alunos		- Projetos devem ser grandes e complexos suficiente para as atividades	
Projetos com alto nível de documentação são preferidos pelos alunos para ter uma uma base de aceitação avaliada e tbm permitir avaliar a documentação.		- Projetos com alto nível de documentação são preferência dos alunos	

Fonte:(KRUTZ; MALACHOWSKY; REICHLMAYR, 2014)

Tabela C.20 Tabela de Exploração do Trabalho ID-5353

Chen, Z. ; Memon, A. ; Luo, B.			
Título	Combining research and education of software testing: A preliminary study	ID	5353
Unidades de Contexto		Unidades de Registro	
O uso de projetos com tamanho relevante e complexo o suficiente permite ao aluno compreender direfentes tipos de testes para diferentes tipos de aplicações.		- Projetos com Tamanho e complexidade relevante	
É um desafio avaliar muitas tarefas de teste dentro de um FOSS em um tempo limitado.		-Limitações de tempo para avaliar muitas tarefas em um FOSS	
Atividades de teste nos projetos permitem aos alunos uma percepção da importância do teste como demanda de mão de obra e estimulará a busca por ferramentas de automação e redução de custos nas atividades.		- Percepção da importância do teste - Estímulo a busca por ferramentas	

Fonte:(CHEN; MEMON; LUO, 2014)

Tabela C.21 Tabela de Exploração do Trabalho ID-5357

Smith, T. ; Gokhale, S. ; McCartney, R.			
Título	Understanding students'preferences of software engineering projects	ID	5357
Unidades de Contexto		Unidades de Registro	
Alunos se preocupam com os benefícios de seu trabalho para si mesmos, para usuários e para outros profissionais		- Percepção da importância de seu trabalho	
Questões humanitárias e nível de escolha de carreira, poder de escolha, autenticidade e relação ao mundo real são mencionadas como motivadoras e atrativas os alunos.		- Questões humanitárias - Relação com o mundo real - Motivação aos alunos - Atratividade dos alunos	
Demora e necessidade de suporte adicional são riscos envolvidos na liberdade de escolha dos projetos pelos alunos.		- Demora e necessidade de suporte nos projetos - Liberdade de Escolha dos Alunos	
Alunos tem mais familiaridade com projetos de acordo com conceitos familiares e com o tamanho da comunidade de usuários, transmitindo a sensação de benefícios do trabalho		- Alunos confortáveis com projetos de assuntos familiares -Alunos confortáveis com comunidades de tamanho familiares -Percepção da sensação de benefício de suas ações	
Alunos que viram importância em produzir valor nas suas atividades se preocuparam com o esforço que poderia estar envolvido para a conclusão da atividade.		- Percepção da importância de produzir valor nas atividades	
Projetos podem ser escolhidos pelos professores de acordo com muitos fatores principalmente a adequação aos objetivos do curso		- Escolha dos projetos pelo professor - Noção de Adequação aos objetivos do curso	
Complexidade apropriada e satisfação do aluno são critérios importantes na seleção de projetos		- Complexidade apropriada - Satisfação do aluno	

Fonte:(SMITH; GOKHALE; MCCARTNEY, 2014)

Tabela C.22 Tabela de Exploração do Trabalho ID-5546

Fernandes, S. ; Martinho, M.H. ; Cerone, A. ; Barbosa, L.S.			
Título	Integrating formal and informal learning through a FLOSS-based innovative approach	ID	5546
Unidades de Contexto		Unidades de Registro	
A atitude do aluno é pró-ativa para lidar com as dificuldades de estabelecer uma conexão com a comunidade Floss Escolhida		- Atitude pró ativa do aluno - Conexão com a comunidade	
Conhecimento prévio entre os membros pode auxiar a interação e a dinâmica dos grupos de alunos.		- Conhecimento prévio entre os membros auxilia dinâmica dos grupos	
Alunos tendem a organizar eventos como Workshops para divulgar os resultados de suas atividades no Projeto		- Alunos tendem a divulgar seus resultados	
A curva de aprendizado foi excepcional, muito mais íngreme do que aquela ao observar em aulas formais para o aprendizado em programação		- Curva de Aprendizado	
Alunos enfrentaram uma linguagem totalmente nova e a dominaram em um curto período, demonstrando uma curva de aprendizagem íngrime.		-Alunos aprenderam linguagem nova - Curva de aprendizagem íngrime	
Alunos adquiriram experiência valiosa em novas ferramentas e tecnicas de desenvolvimento por meio das atividades no projeto.		Experiência em Novas Ferramentas	
Alunos citam o impacto de suas atividades para a comunidade tanto como "cliente" como "fornecedor"de serviços em um ambiente distribuído.		Alunos citam o impacto de suas atividades	
Projetos orientados a FLOSS fornecem um ambiente muito interessante para exercitar o "aprender fazendo"e, em geral,abordagens autônomas e pró-ativas de aprendizagem		Floss fornece ambiente para exercitar atividades pró-ativas	
Demasiada demora da comunidade em responder aos alunos ou a dificuldade de entender o que o grupo se propunha a fazer são um desafio inicial aos estudantes.		Demasiada demora de respostas da comunidade	
Alunos ficaram inicialmente um pouco decepcionados com a lacuna entre a experiência e domínio de computação da comunidade e a sua própria		Decepção do aluno com lacuna de experiência da computação	
Alguns alunos, temendo suas próprias habilidades de programação fracas, optaram por funções mais “observacionais”, ou seja, analista de requisitos.		Habilidades de programação	
Pequenos Grupos tendem a especializar funções dos membros nas atividades envolvendo o Projeto.		Tamanho de Grupo influencia nas funções dos membros na atividade	
A interação com a comunidade é entendida pelos alunos como oprincipal força motriz do seu progresso de aprendizagem.		- Interação com a comunidade - Percepção do seu progresso de aprendizagem	

Fonte:(FERNANDES et al., 2013)

Tabela C.23 Tabela de Exploração do Trabalho ID-5676

Papadopoulos, P.M. ; Stamelos, I.G. ; Meiszner, A.			
Título	Enhancing software engineering education through open source projects: Four years of students’ perspectives	ID	5676
Unidades de Contexto		Unidades de Registro	
Alunos se sentem motivados a continuar contribuindo na comunidade mesmo após a conclusão do curso.		Motivação a contribuir com a comunidade	
Alunos informam que o impacto das atividades no Projeto FLOSS foi importante para uma atitude positiva sobre o prosseguimento da carreira.		Percepção do impacto das atividades na visão de carreira	
Alunos gostaram de ter acesso a um repositório onde pudesse depositar e recuperar seu trabalho.		Alunos gostaram de ter acesso a um repositório	
Alunos que não tem experiência com programação se sentem menos confiantes de exercer funções de desenvolvimento nas atividades do Projeto.		-Baixa experiência de programação - Menor confiança em sua participação no projeto	
Alunos podem ter subestimado suas habilidades e por medo de comentários negativos sobre suas contribuiçoes escolhiam funções e projetos aparentemente mais viáveis para eles.		Alunos podem subestimar suas habilidades	
Alunos sentiram preocupação sobre organização dos dados de contribuição no ambiente de aprendizagem do curso, solicitando orientação para busca de dados em relatórios anteriores.		Alunos solicitam feedback das suas contribuições	
Baixo nível de interação entre os alunos da própria classe no ambiente de colaboração demanda busca por métodos que aproximem a interação entre esses estudantes.		Baixa interação entre alunos demanda ação docente para aproximá-los	
Interessados em aplicar a abordagem de aprendizagem por meio de projetos FLOSS devem se atentar a sobrecarga dos Instrutores e a demanda por ambientes de aprendizagem com caixa de ferramentas básicas para as atividades.		-Se atentar a sobrecarga dos instrutores -Ambientes com caixa de ferramentas básica	
Os Alunos veem importância nas funções que se adaptam com suas características na participação das atividades no projeto, sendo elas a capacidade de programar, de testar ou de ter uma visão mais analítica.		Alunos percebem a importância de suas funções	
Alunos sentiram dificuldade em escolher projetos compatíveis com as atividades que exerceriam, tendo que mudar de projeto algumas vezes por falta de comunicação e receptividade da comunidade.		-Alunos tiveram dificuldade em escolher projetos - Falta de comunicação e receptividade da comunidade	
Alunos sentiram necessidade de orientação para escolha dos projetos.		Alunos necessitaram orientação docente	
O fator de sobrecarga dos instrutores está baseado principalmente no monitoramento do progresso dos alunos nas diferentes etapas, principalmente na aprovação dos projetos escolhidos.		- Sobrecarga dos instrutores - Monitoramento do progresso dos alunos	

Fonte:(PAPADOPOULOS; STAMELOS; MEISZNER, 2013)

Tabela C.24 Tabela de Exploração do Trabalho ID-17796

Smith, Therese Mary and McCartney, Robert and Gokhale, Swapna S. and Kaczmarczyk, Lisa C.			
Título	Selecting Open Source Software Projects to Teach Software Engineering	ID	17796
Unidades de Contexto		Unidades de Registro	
Ao concluir as atividades os alunos substituiam a frustração original por um senso de orgulho e realização.		Percepção de sua evolução e senso de orgulho	
Devido a grande variação de complexidade, a seleção de projetos de forma espontânea pelos alunos pode inviabilizar os objetivos de aprendizagem e demandar muito tempo.		Variação de complexidade	
Projetos de tamanho reduzido e com equipe de suporte não são encontrados para seleção em atividades		- Projetos de tamanho Reduzido - Projetos com equipe de suporte	
Alunos se sentiam incialmente frustrados com a dificuldade de compreender códigos não escritos por eles.		Frustração por perceber dificuldade com código de terceiros	
O fardo de Selecionar e Preparar os Projetos OSS devem ser mitigados para estimular o uso do OSS no Ensino de ES		Fardo de Selecionar e preparar projetos devem ser mitigados	
Grande Variedade de Tamanho complexidade e qualidade torna difícil a escolha do projeto ideal para uma atividade.		-Variedade de Complexidade e tamanho do projeto	
Critérios Preliminares como a linguagem de programação conhecida, tamanho do projeto e equipe de suporte são relevantes para a escolha dos projetos		-Linguagem de programação escolhida - Tamanho do Projeto -Equipe de Suporte	
Características dos Repositórios de Projetos Floss podem exercer influência direta na seleção dos projetos por alunos e professores (Coleção, Termos de Pesquisa Adequados e Distinção entre executáveis e códigos fonte).		- Características do Repositório	
Critérios de seleção de projetos para o objetivo do curso :Tamanho do Código, A Linguagem de Programação, Domínio da Aplicação, Força de Acoplamento, Atividades Recentes, Qualidade de Documentação, Capacidade de Construção		-Tamanho do código -Linguagem de programação -Domínio da aplicação -Estado do projeto	

Fonte:(SMITH et al., 2014)

Tabela C.25 Tabela de Exploração do Trabalho ID-17800

Gokhale, Swapna and Smith, Therese and McCartney, Robert			
Título	Teaching Software Engineering from a Maintenance-centric View Using Open-source Software	ID	17800
Unidades de Contexto		Unidades de Registro	
Encontrar Projetos Apropriados, Projetar Tarefas e Encontrar Tarefas Apropriadas podem gerar sobrecarga no Instrutor.		Sobrecarga Docente no planejamento da atividade	

Fonte:(GOKHALE; SMITH; MCCARTNEY, 2013a)

Tabela C.26 Tabela de Exploração do Trabalho ID-17805

Castelluccia, Daniela and Visaggio, Giuseppe			
Título	Teaching Evidence-based Software Engineering: Learning by a Collaborative Mapping Study of Open Source Software	ID	17805
Unidades de Contexto		Unidades de Registro	
Problemas provenientes da colaboração Aberta como coordenação de equipes distribuídas, falta de documentação, má utilização do código,evolução não planejada de funções são fatores de risco para avaliação de componentes OSS		-Estado do projeto - Coordenação de equipes distribuídas.	

Fonte:(CASTELLUCCIA; VISAGGIO, 2013)

Tabela C.27 Tabela de Exploração do Trabalho ID-17830

Hislop, Gregory W. and Ellis, Heidi J.C. and Pulimood, S. Monisha and Morgan, Becka and Mello-Stark, Suzanne and Coleman, Ben and Macdonell, Cam			
Título	A Multi-Institutional Study of Learning via Student Involvement in Humanitarian Free and Open Source Software Projects	ID	17830
Unidades de Contexto		Unidades de Registro	
Transparência do Código aberto permite que os alunos entendam quem contribuiu com várias partes do projeto e permite contribuição destes		Transparência do código aberto na participação do aluno	
Ambiente FOSS pode auxiliar no ganho de confiança do aluno em suas habilidades e em Crescer Profissionalmente		Aluno tem ganho de confiança em sua habilidade	
Alunos podem ver os benefícios de usar habilidades de computação para ajudar os outros e ajuda a preparar os alunos para se tornarem membros contribuintes da comunidade.		Alunos percebem benefícios do uso de sua habilidade	
Foss Humanitários são atraentes aos alunos, principalmente mulheres para a área de computação		Mulheres se atraem por FOSS Humanitários	
A participação no HFOSS melhora a motivação e a perspectiva de carreira.		Motivação e perspectiva de carreira	
Interação com as comunidades HFOSS aumenta confiança do aluno		-Interação com a comunidade -Aumento da confiança	
Alunos e Instrutores devem se adequar a um ambiente de aprendizagem menos estruturado onde o aluno deve aprender com uma variedade de recursos, incluindo documentação onde o professor não é mais a única autoridade para o material.		- Adequação do aluno e docente a ambiente de aprendizagem não estruturado	
Os alunos podem não estar familiarizados com as ferramentas, abordagens ou idiomas usados no projeto, e pode precisar aprender todos esses simultaneamente.		-Falta de familiaridade com ferramentas -Familiaridade com o idioma	
Os instrutores podem enfrentar curvas de aprendizagem semelhantes para ferramentas, abordagens, Cultura FOSS e conhecimento de domínio		- Instrutores pode enfrentar curva de aprendizagem - Ferramentas e abordagens -Cultura FOSS -Conhecimento de domínio	
Alunos podem ter expectativas não realizadas sobre como interagir com a comunidade HFOSS		Frustração da expectativa ao interagir com comunidade HFOSS	
Os alunos devem compreender a Cultura FOSS, incluindo formas e modos de comunicação, e deve frequentemente obter conhecimento de domínio significativo		-Alunos devem compreender - Obter conhecimento de domínio	
A complexidade do próprio projeto apresenta outra curva de aprendizado, pois os alunos devem aprender como abordar e entender um projeto existente.		- Complexidade do próprio projeto - Curva de aprendizagem	
Além disso, instrutores deve promover a participação dos alunos, aprender a criar tarefas e resultados do curso, e criar rubricas de avaliação.		Professor deve promover a participação criar tarefas e rubricas de avaliação	
Instrutores podem ter que negociar comunicação e suporte com a comunidade HFOSS, por exemplo, identificando o que apoiador da comunidade fornecerá (como um mentor para o instrutor ou respondendo perguntas dos alunos), e o que os alunos irão fornecer, que pode incluir documentação ou correções de código		Professor pode negociar com a comunidade as atividades feitas pelos alunos	
Os instrutores também precisam estar atento às questões potenciais do FERPA e de propriedade intelectual.		Atenção a questões legais e propriedade intelectual	
O projeto deve se adequar aos objetivos de aprendizagem do curso,esteja aberto a contribuições e boas-vindas aos alunos, e tenha bons mecanismos de comunicação		-Projeto deve se adequar aos Objetivos de aprendizagem - Receptividade aos alunos -bons mecanismos de comunicação	
Sincronização dos horários entre o calendário acadêmico e o Ciclo de lançamento de HFOSS e imprevisibilidade onde o projeto pode fazer uma grande mudança em sua direção no médio prazo.		-Horário acadêmico e ciclo de lançamento HFOSS compatíveis - Imprevisibilidade do projeto	
Alunos podem sentir que avançaram na capacidade de programação, mas podem ficar menos confiantes por estar mais longe do status de especialistas do que imaginavam.		- Percepção de avanço na capacidade de programação -Falta de confiança	

Fonte: (HISLOP et al., 2015)

Tabela C.28 Tabela de Exploração do Trabalho ID-17882

Van Deursen, Arie and Aniche, Maurício and Aué, Joop and Slag, Rogier and De Jong, Michael and Nederlof, Alex and Bouwers, Eric			
Título	A Collaborative Approach to Teaching Software Architecture	ID	17882
Unidades de Contexto		Unidades de Registro	
Alunos indicaram capacidade de se avaliar nas atividades		Capacidade do aluno se avaliar	
Alunos indicaram Possibilidade de Avaliar os colegas		Percepção de aprendizagem do colega	
Alunos indicaram oportunidade de Interação com comunidade Profissional envolvida no projeto		Interação com a comunidade	
Tempo Maior utilizado para fazer o curso: Alunos relataram que gastaram mais tempo do que em demais cursos		Tempo maior para fazer o curso	
Interferência do aspecto emocional e interpessoal na avaliação da atividade: Alunos podem ter dificuldades com relações com colegas e com a participação destes.		Questão emocional e interpessoal	
Largura e Profundidade entre o Projeto e o Assunto Abordado podem fazer o aluno ter dificuldade de aplicar muitos dos conceitos teóricos na prática por meio do projeto.		Compatibilidade entre o projeto e o conteúdo abordado	
Experiência dos participantes em programação		Experiência em Programação	
Experiência com Ambientes de Controle de Versão (GIT)		Experiência em ferramenta específica	
Largura e Profundidade entre o Projeto e o Assunto Abordado auxiliam alunos mergulhar no Projeto em que analisam e aprender com apresentações das análises de outros colegas		Largura e profundidade entre projeto e assunto abordado	
Interferência da Aceitação do Projeto na Aprendizagem: Alunos podem ter resistência ou podem ficar satisfeitos em trabalhar no projeto escolhido		Interferência na aceitação do projeto na aprenidizagem	

Fonte:(DEURSEN et al., 2017)

Tabela C.29 Tabela de Exploração do Trabalho ID-18359

Kussmaul, Clifton			
Título	Experience Report: Guiding Faculty Students to Participate in Humanitarian FOSS Communities	ID	18359
Unidades de Contexto		Unidades de Registro	
Alunos pode ter aumento na motivação e perspectiva de carreira		Motivação e perspectiva de carreira	
Alunos indicam que podem descrever impacto da complexidade e tamanho do projeto.		Impacto da Complexidade e tamanho do projeto	
Alunos indicam que podem usar ferramentas e tecnicas no projeto		Uso de Ferramentas e técnicas	
Alunos indicam que sua capacidade de programação aumentou.		Percepção da capacidade de programação	
Alunos podem ter queda na confiança sobre suas habilidades em computação		Queda na confiança sobre suas habilidades	
Alunos podem perder a confiança em trabalhar com diferentes equipes e culturas.		Interferência de diferentes equipes e culturas na confiança	
Entender como novas pessoas se integram a uma comunidade FOSS, Identificar projetos FOSS que dão as boas-vindas, orientam e apóiam novas pessoas, Decidir o que os alunos farão e como irão interagir, Apresentar a comunidade e apoiar o aluno e documentar o progresso são diretrizes que podem ajudar a envolver alunos em FOSS		-Integração a uma comunidade em FOSS -Projetos que apoiam novos membros -Decisões sobre interações dos alunos -Acompanhar o progresso do aluno	

Fonte: (KUSSMAUL, 2016)

Tabela C.30 Tabela de Exploração do Trabalho ID-18433

Kussmaul, Clifton L. and Ellis, Heidi and Hislop, Gregory W. and Postner, Lori and Burdge, Darci			
Título	Helping faculty students to participate in Humanitarian Free Open Source software: The OpenFE OpenPath projects	ID	18433
Unidades de Contexto		Unidades de Registro	
Há evidências de que Hfoss exerce impacto positivo na Motivação do Estudantes para Estudar Computação		Impacto na motivação dos estudantes	
Há Evidências de Impactos positivos na aprendizagem percebida em Engenharia de Software		Percepção na aprendizagem em ES	
Há Evidências de Impactos positivos em planos de carreira		Impactos no plano de carreira	
O tempo limitado de preparação dos instrutores, muito material para cobrir e pouca capacidade de mudar currículo podem influenciar na participação em HFOSS		- Tempo Limitado - Preparação dos instrutores - Capacidade de mudança de currículo	

Fonte: (KUSSMAUL et al., 2017)

Apêndice

D

Aqui se apresenta a formação acadêmica e a trajetória profissional do pesquisador

TRAJETÓRIA DO PESQUISADOR

ERINALDO SANTOS OLIVEIRA

Professor do Ensino Básico, Técnico e Tecnológico (EBTT) do Instituto Federal Baiano - Ifbaiano.
Especialista em Engenharia de Sistemas pela Escola Superior Aberta do Brasil - ESAB. Trabalho de Conclusão: Modelagem de Sistema de Informação para Auxílio da Administração de Polo EAD do Instituto Federal Baiano.
Bacharel em Sistemas de Informação pela Universidade Salvador - UNIFACS. Trabalho de Conclusão: Modelagem de um Sistema de Informação para auxílio do Projeto Acadêmico Pedagógico.
Atua na educação profissional em computação desde 2009, sendo quatro anos na Educação Profissional Estadual da Bahia e dez anos como Professor EBTT pelo Ifbaiano, lecionando disciplinas nas áreas de Lógica e Técnicas de Programação, Banco de Dados, Análise e Projeto de Sistemas, Sistemas de Informação e Projetos de TI, Sistemas Operacionais, Arquitetura de Computadores, Operação de Computadores, Fundamentos de Informática, Segurança da Informação, Gestão de TI, etc.
Tem experiência com Coordenação de Polo Presencial de Educação à Distância pela Rede E- Tec Brasil e como Professor Formador no curso de Secretaria Escolar EAD do Instituto Federal Baiano. Trabalhou na reformulação de três PPCs (Projeto Político Pedagógico) de Cursos de Nível Médio da Educação Profissional no Ifbaiano, sendo um deles o Curso Profissional de Educação de Jovens e Adultos - Proeja . Tem experiência como Conselheiro de Curso Técnico em Manutenção e Suporte em Informática, Curso Técnico em Informática - Proeja, Curso Técnico em Informática, e Curso Técnico em Agroindústria. Atuou como Membro de Núcleo de Assessoramento Pedagógico nos Cursos Citados e Atuou como Coordenador do Curso Técnico em Manutenção e Suporte em Informática.