UNIVERSIDADE FEDERAL DO AMAPÁ

DEPARTAMENTO DE CIÊNCIAS EXATAS E TECNOLÓGICAS

COORDENAÇÃO DO CURSO DE CIÊNCIA DA COMPUTAÇÃO

Anderson dos Santos Guerra

UMA ABORDAGEM PARA O ENSINO DE ENGENHARIA DE REQUISITOS FOCADA NO ALUNO

Trabalho de Conclusão de Curso

Macapá

2019

Anderson dos Santos Guerra

UMA ABORDAGEM PARA O ENSINO DE ENGENHARIA DE REQUISITOS FOCADA NO ALUNO

Trabalho de Conclusão de Curso submetido à Banca Examinadora do Curso de Ciência da Computação da UNIFAP para a obtenção do Grau de Bacharel em Ciência da Computação.

Orientador: Prof. MSc. Julio Cezar Costa Furtado

Macapá
2019

Anderson dos Santos Guerra

UMA ABORDAGEM PARA O ENSINO DE ENGENHARIA DE REQUISITOS FOCADA NO ALUNO

Trabalho de Conclusão de Curso submetido à Banca Examinadora do Curso de Ciência da Computação da UNIFAP para a obtenção do Grau de Bacharel em Ciência da Computação.

Data da aprovação: ___/___/_____
Nota: _____

BANCA EXAMINADORA

_____

Prof. MSc. Julio Cezar Costa Furtado
Coordenação do Curso de Ciência da Computação - UNIFAP – Orientador

_____

Prof. Esp. Adeildo Telles da Silva
Coordenação do Curso de Ciência da Computação - UNIFAP – Membro Interno

_____

Prof. MSc. Marco Antônio Leal da Silva
Coordenação do Curso de Ciência da Computação - UNIFAP – Membro Interno

AGRADECIMENTOS

Gostaria de agradecer aos meus pais, Euriqui e Maria Aparecida, que independentemente da situação sempre me incentivaram no caminho do estudo e me ajudaram das melhores formas possíveis.

Ao meu orientador, Júlio Cezar, que por algum motivo curioso se demonstrou mais paciente do que eu imaginei que seria, além de sempre dar o devido apoio e dicas na produção deste trabalho.

As amizades que fiz ao longo do curso e que criaram lembranças memoráveis, desde maratonas de estudo pré-prova até conversas estranhas sobre futuros distópicos enquanto acontecia corujão de Age of Empires.

E as surpresas que a vida faz, pois quando menos esperamos ela surge com um motivo pra te fazer sorrir.

RESUMO

A Engenharia de Requisitos é uma área dentro da Engenharia de Software que se preocupa principalmente em garantir que os objetivos para o qual um software é desenvolvido sejam atendidos de maneira satisfatória ao fim da criação do produto. É necessário que o profissional que atue na Engenharia de Requisitos possua capacidades para levantar, alterar e validar requisitos para garantir que no fim do ciclo de desenvolvimento o software atenda às necessidades solicitadas. A indústria de desenvolvimento indica que os profissionais recém ingressados na área não possuem o conhecimento prático que é esperado para poder atuar como um engenheiro de requisitos, um fator que influencia nessa afirmação é o modo como a Engenharia de Requisitos é lecionada em graduações na área de computação, muita das vezes ensinada através de uma abordagem tradicional de ensino. Nesse contexto, este trabalho tem como intuito contribuir com o ensino de Engenharia de Requisitos com uma proposta de abordagem de ensino que faz uso de estratégias focadas no aluno para o desenvolvimento de competências esperadas pela indústria. Para criar a abordagem, foi feita a identificação das competências através do CMMI-DEV. Os resultados obtidos são analisados de maneira que seja possível identificar o que pode ser melhorado na abordagem.

PALAVRAS-CHAVE: Engenharia de Software, Engenharia de Requisitos, Ensino, Abordagem de Ensino, Abordagem de Ensino Tradicional, Abordagem de Ensino Alternativa.

ABSTRACT

Requirements Engineering is an area within Software Engineering that is primarily concerned with ensuring that the objectives for which software is developed are satisfactorily met at the end of product creation. It is necessary that the professional that works in Requirements Engineering has the ability to raise, change and validate requirements to ensure that at the end of the development cycle the software meets the requested requirements. The development industry indicates that newcomers do not have the practical knowledge that is expected to be able to act as a requirement engineer, a factor influencing this assertion is how Requirements Engineering is taught in graduations in the area of computing, often taught through a traditional approach to teaching. In this context, this work aims to contribute with the teaching of Requirements Engineering with a proposal of teaching approach that makes use of student-focused strategies for the development of skills expected by the industry. To create the approach, the competencies were identified through CMMI-DEV. The results obtained are analyzed so that it is possible to identify what can be improved in the approach.

KEYWORDS: Software Engineering, Requirements Engineering, Teaching, Teaching Approach, Traditional Teaching Approach, Alternative Teaching Approach.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

CMMI	*Capability Maturity Model Integration*
CMMI-DEV	CMMI *for Development*
ES	Engenharia de Software
ER	Engenharia de Requisitos
IEEE	*Institute of Electrical and Electronics Engineers*
MPS.BR	Melhoria do Processo de Software Brasileiro
MA-MPS	Método de Avaliação do MPS.BR
OTAN	Organização do Tratado do Atlântico Norte
PMBOK	*Project Management Body of Knowledge*
PMI	*Project Management Institute*
S&SC	Software e Serviços Correlatos
SA-CMM	*Software Acquisiton Capability Maturity Model*
SEI	*Software Engineering Institute*
SOFTEX	Associação para Promoção da Excelência do Software Brasileiro

SUMÁRIO

1 INTRODUÇÃO

Nos dias que correm é impossível não depender de softwares. Diversos tipos de equipamentos, infraestruturas e serviços são controlados por meio do uso de sistemas computacionais, e a grande parte de produtos elétricos possui uma espécie de computador com um software que o controla de alguma forma (SOMMERVILLE, 2011).

Sommerville (2011) afirma que softwares são definidos como abstratos e intangíveis. Não sendo restringidos por leis da física ou processos de manufatura, isso de certa maneira torna simples a engenharia de software uma vez que não existem limites naturais para os potenciais que um software pode atingir. Porém, a falta de restrições físicas, os sistemas de software podem se ter uma complexidade muito grande em pouco tempo, se tornando mais caros para se alterar e difíceis de se entender (SOMMERVILLE, 2011).

Para diminuir as chances de erros no desenvolvimento de sistemas, existe uma área dentro da engenharia de software chamada de Engenharia de Requisitos. Entender um requisito de um cliente é uma das tarefas mais complicadas que um engenheiro de software pode enfrentar, uma vez que o cliente muita das vezes não tem uma noção real do que deseja alcançar com o desenvolvimento do produto, além de não possuir ideia de quais as características fundamentais e funções necessárias (PRESSMAN, 2011).

Tomando como base Menon *et al* (2010), por mais que a engenharia de requisitos seja uma área de grande importância e que auxilie para evitar o fracasso no desenvolvimento de sistemas, o seu ensino em ainda não atinge os desempenhos esperados pela indústria. O que ocorre muitas vezes pelo fato de o ensino tradicional ser utilizado ao invés de ensinos dinâmicos que priorizem as atividades em grupo e o uso de criatividade para solução de problemas (MENON *et al*., 2010).

A pesquisa de Portela (2017) relata sobre a carência existente de profissionais qualificados para atuarem na indústria de desenvolvimento de software, sendo que um dos fatores importantes que influenciam na qualidade dos profissionais atuantes é justamente a educação.

Portela (2017) indica que a falta de profissionais qualificados pode ter relação com as competências necessárias de um engenheiro de requisitos não serem devidamente desenvolvidas durante a graduação, o que torna difícil para a indústria de software conseguir a mão-de-obra qualificada, pois o profissional recém formado só vai efetivamente obter o conhecimento necessário quando for atuar no mercado.

Assim, o principal objetivo deste trabalho é a apresentação de uma abordagem de ensino de engenharia de requisitos para cursos de computação, que seja centrada no aluno, motivadora e que consiga preparar os participantes para que os mesmos possam atuar melhor na engenharia de requisitos em um ambiente real.

Para tal, foi feito o uso de alguns modelos, guias e normas que visam oferecer as boas práticas para o processo de Engenharia de Requisitos, como: CMMI-DEV (SEI, 2010) e MR-MPS-BR (SOFTEX, 2016).

O capítulo 2 dessa pesquisa se trata primariamente da contextualização do trabalho, realizando a explicação de conceitos da engenharia de software e engenharia de requisitos, além da exemplificar alguns problemas que estão relacionados diretamente à engenharia de requisitos. O capítulo 3 é responsável por abordar a motivação para a realização dessa pesquisa, também justificando sua utilidade e de que forma a mesma irá contribuir para a área.

A partir do capítulo 4 é feita a explicação dos objetivos que essa pesquisa tem o intuito de alcançar. O capítulo 5 trata sobre as formas de validação que foram utilizadas na pesquisa. O capítulo 6 contém a metodologia da pesquisa, explicitando suas características e etapas, também mostrando o que já foi realizado e o que falta realizar na pesquisa. O capítulo 7 possui o cronograma do projeto.

1.1. Motivação, Justificativa e Contribuição à área

Com o advento de computadores e da globalização, a demanda para criar produtos e serviços no meio digital aumentou, tornando o desenvolvimento de software um dos pilares da sociedade no século XXI, ao longo do tempo a produção de software sempre foi renovada, por meio do uso de novas práticas e ferramentas para a criação em maior velocidade, menores custos e maior qualidade (SOMMERVILLE, 2011).

Uma vez que a criação de um software é muita das vezes abstrata, como afirma Sommerville (2011), é necessário um conjunto de práticas e indicativos para demonstrarem que a produção está no caminho certo. Muitas das práticas utilizadas no desenvolvimento estão contidas na engenharia de software, especialmente na área da engenharia de requisitos.

O uso da engenharia de requisitos se dá justamente pela capacidade que a mesma tem para a diminuição de erros no desenvolvimento do software, pois a partir dela é feito o levantamento de requisitos, todas as necessidades primárias do cliente, para que o produto possa ser feito de maneira satisfatória e não falhe em testes de validação e verificação (PRESSMAN, 2011). Por esse motivo, a área de engenharia de requisitos é importante e deve ser priorizada no processo de desenvolvimento de software.

Nuseibeh e Easterbrook (2000) indicam que existe um grau de importância na Engenharia de Requisitos dentro do processo de desenvolvimento de software, sendo a parte que se concentra em ancorar o desenvolvimento de atividades para um problema do mundo real, tornando a adequação e o custo-efetivo da solução passível de ser analisada.

A importância da Engenharia de Requisitos acontece justamente pela mesma oferecer uma série de conceitos que formalizam uma adequada elicitação e validação de requisitos, garantindo que um determinado sistema consiga satisfazer de maneira adequada às necessidades do cliente, diminuindo assim a margem para erros e consequentemente o tempo utilizado para o desenvolvimento do produto e custos que poderiam ser desnecessários (SOMMERVILLE, 2011).

Por mais que a Engenharia de Requisitos (ER) e a Engenharia de Software (ES) sejam áreas de grande importância dentro da indústria de desenvolvimento devido a suas características, existe uma dificuldade em encontrar profissionais devidamente qualificados para atuarem na área (PORTELA, 2017).

Conforme Sommerville (2011), as organizações que estão envolvidas com o desenvolvimento de software necessitam possuir processos para nortear a elicitação e validação de requisitos, parte destas organizações não possui um processo estabelecido para elicitação de requisitos, ou se possui, o processo é deficiente, resultando assim, em uma dificuldade em gerir os requisitos ao longo do ciclo de vida do desenvolvimento do software, podendo ocasionar situações onde o produto entregue não atende adequadamente as expectativas cliente.

Seguindo esse contexto, surgiram os modelos, guias e normas que tem como objetivo orientar as organizações com as boas práticas para o processo de engenharia de requisitos, podendo citar os processos constantes no CMMI (SEI, 2010) e modelos específicos que constam com itens em engenharia de requisitos, como o CMMI-DEV (SEI, 2010).

Sendo assim, o fato de adotar as melhores práticas existentes na área de engenharia de requisitos promove a mitigação de grande parte dos problemas mais comuns encontrados, uma vez que a maioria dos problemas encontrados dentro desenvolvimento dos produtos de software são oriundos muitas vezes de um processo de engenharia de requisitos mal feito. Acontecendo casos onde o produto desenvolvido não atende adequadamente os objetivos para os quais foi desenvolvido ou possui custo bem mais elevado do que o previsto (PRESSMAN, 2011).

Um fator a levar em conta para ao analisar os processos de engenharia de requisitos que não são realizados de maneira satisfatória é justamente o fato de professores se aterem muito as abordagens de ensino tradicionais, realizando aulas expositivas e fazendo a aplicação de provas para efetivação do conhecimento. Esse item faz com que os acadêmicos fiquem

desmotivados e torna mais difícil o aprendizado dos mesmos, uma vez que a ER está contida na ES e ambas possuem uma grande base teórica e a efetiva aprendizagem de ambas é feita principalmente através da aplicação seus conteúdos em projetos práticos (WANGENHEIM e SILVA, 2009).

Assim, é possível destacar que uma parte do déficit de profissionais qualificados para atuarem na indústria de desenvolvimento se deve ao fato das aulas de graduações não abordarem de maneira satisfatória os tópicos e competências necessárias da ER (LETHBRIDGE, 2000; PORTELA, 2017). De acordo com Wangenheim e Silva (2009), os tópicos de ES são ensinados de maneira superficial em curso de graduação, além de poucas disciplinas abordarem os itens contidos dentro da ES.

Desta forma, este projeto visa lecionar os itens contidos dentro da área de Engenharia de Requisitos para turmas de computação por meio de atividades centradas no aluno, com o uso de dinâmicas ao invés de aulas tradicionais para que os participantes estejam mais familiarizados em como funciona um ambiente de trabalho com problemas reais, em oposição ao se aterem somente a teoria.

1.2. Objetivos

O projeto tem como principal foco o estudo da eficácia das abordagens de ensinos alternativas para o ensinamento da engenharia de requisitos em turmas de computação, e com base nessa experiência, elaborar um questionário que será respondido pelos participantes para eventualmente ser discutido o nível de eficácia da abordagem utilizada. Para poder atender tais resultados, os objetivos a seguir devem ser contemplados.

1.2.1. Objetivo Geral

Definição de uma abordagem para o ensino de engenharia de requisitos que tome como base as competências e habilidades esperadas por um profissional da área segundo o CMMI-DEV.

1.2.2. Objetivos Específicos

1. Identificar as abordagens utilizadas para o ensino da Engenharia de Software;
2. Elaborar uma abordagem de ensino para engenharia de requisitos;
3. Aplicar a abordagem realizada em uma turma de computação que tenha uma noção em tópicos de engenharia de software;
4. Analisar a eficácia da abordagem alternativa em relação ao método tradicional de ensino.

1.3. Metodologia de Pesquisa

A metodologia foi composta de cinco fases, onde a primeira fase é constituída primariamente de um estudo sobre o que a literatura diz sobre os processos de engenharia de requisitos, o que contemplar e quais dificuldades encontradas na execução de tais processos, assim como quais modelos a seguir. Esta fase apoiou o restante do projeto com todo o referencial teórico necessário para o entendimento do problema.

A segunda fase foi a realização de uma revisão teórica sobre as abordagens de ensino tradicionais e alternativas para o aprendizado de engenharia de software e, mais especificamente, engenharia de requisitos no contexto de cursos de computação. E com base nesta revisão efetuar uma abordagem de ensino de engenharia de requisitos, visando verificar a efetividade do aprendizado.

Na terceira fase foi desenvolvida a abordagem de ensino de engenharia de requisitos, o principal foco da abordagem será o uso de dinâmicas para efetivar o conhecimento teórico. É feita a seleção de materiais que serão utilizados em aula, desde artigos, dinâmicas, jogos e também um projeto prático a ser desenvolvido ao longo da execução da disciplina.

A quarta fase é iniciada com o uso em sala de aula da abordagem definida na fase anterior. Aula esta que, quando concluída, será exposta para a avaliação pelos participantes, onde será dirigida uma solicitação para o preenchimento de um questionário avaliativo formado por questões em escala *likert* e subjetivas, visando a validação do mesmo, bem como, possíveis contribuições técnicas. Nesta fase também, os resultados coletados serão avaliados e priorizados para que os possíveis indicadores de pontos fracos e as oportunidades de melhoria venham a ser incorporados pelo projeto.

Por fim, é iniciada a quinta fase, que trata primariamente da escrita dos resultados obtidos no questionário, bem como a discussão em cima do mesmo, elencando o grau de importância que os participantes deram à área de engenharia de requisitos e o quão efetivo foi o aprendizado por meio do uso de dinâmicas.

Foram recebidas respostas de 22 acadêmicos, todos na fase de graduação em Ciência da Computação na Fundação Universidade Federal do Amapá, os participantes estavam no quarto semestre do curso e já possuíam uma base devido à disciplina de Engenharia de Software que fora cursada anteriormente

Em paralelo à todas as etapas serão executadas a escrita do documento de TCC, bem como possíveis artigos a serem submetidos a eventos de interesse. Conforme demonstrado na seção de cronogramas.

2 CONTEXTUALIZAÇÃO E TERMINOLOGIAS DO TRABALHO

Esta seção apresenta terminologias que serão utilizadas ao longo da pesquisa, bem como uma contextualização da atual situação que se encontra o ensino de engenharia de requisitos, desde o uso de abordagens centradas no professor até o uso de abordagens centradas no aluno, o que a indústria de desenvolvimento de software necessita de profissionais recém formados, além de comentar sobre o modelo CMMI para desenvolvedores.

2.1. A Engenharia de Software

A Engenharia de Software é um termo que foi proposto para ser utilizado no ano de 1969, em uma conferência da Organização do Tratado do Atlântico Norte (OTAN), o termo é utilizado em discussões de problemas que envolvem o desenvolvimento de software, desde grandes softwares que eram entregues com atrasos, possuíam falhas na entrega de funcionalidades desejadas por clientes e custos além do esperado (SOMMERVILLE, 2011).

Ao longo dos anos diversos autores criaram definições para o que seria a Engenharia de Software, neste sentido, Sommerville (2011) diz que a engenharia de software tem como principal objetivo apoiar o desenvolvimento profissional de software, mais do que a programação individual. Ela abrange técnicas que apoiam especificação, projeto e evolução de programas, que normalmente não são relevantes para o desenvolvimento de software pessoal.

Ainda segundo o autor, a engenharia de software é uma disciplina de engenharia que se preocupa com todos os aspectos de produção de software. As principais atividades da engenharia de software são especificação de software, desenvolvimento de software, validação de software e evolução de software.

Naur e Randell (1969) descrevem a engenharia de software de maneira mais direta, sendo ela o estabelecimento e o uso de sólidos princípios de engenharia de uma maneira que seja possível obter software de maneira econômica, que seja confiável e funcione de modo eficiente em máquinas reais.

Tomando como base a definição do IEEE (2013), a engenharia de software pode ser definida como a aplicação de uma abordagem sistemática, disciplinada e quantificável no desenvolvimento, na operação e também na manutenção de software, sendo assim, a aplicação de engenharia ao software. Além da aplicação, a engenharia de software pode ser definida como o estudo das abordagens citadas anteriormente.

2.2. A Engenharia de Requisitos

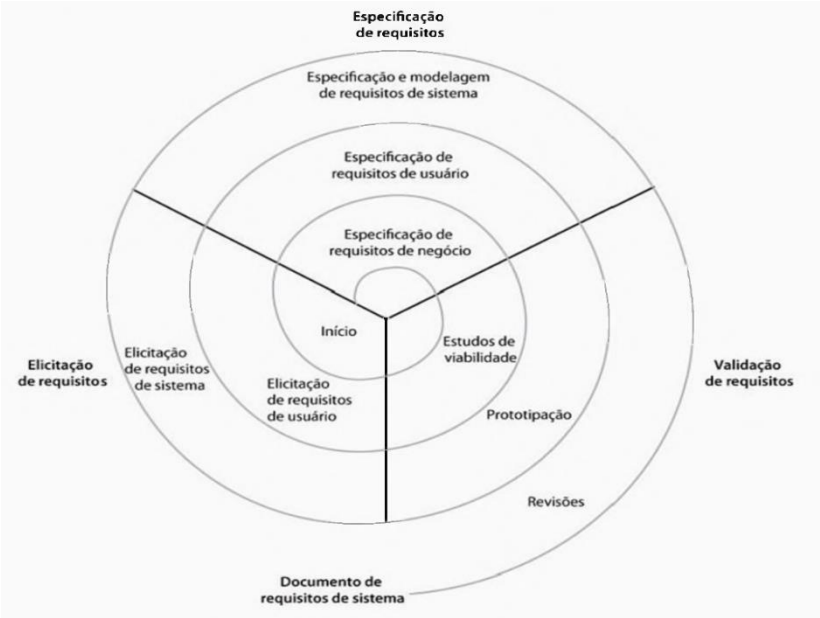
A engenharia de requisitos é responsável por fornecer o mecanismo apropriado para entender aquilo que o cliente deseja, analisando as necessidades, avaliando a viabilidade, negociando uma solução razoável, especificando a solução sem ambiguidades, validando a especificação e gerenciando as necessidades à medida que são transformadas em um sistema operacional (PRESSMAN, 2011)

Segundo Pressman (2011), a engenharia de requisitos abrange sete tarefas distintas: concepção, levantamento, elaboração, negociação, especificação, validação e gestão.

Sommerville (2011) afirma que os requisitos de um sistema são as descrições do que o sistema deve fazer, os serviços oferecem e as restrições a seu funcionamento. Esses requisitos refletem as necessidades dos clientes para um sistema que serve a uma finalidade determinada, como controlar um dispositivo, colocar um pedido ou encontrar informações. O processo de descobrir, analisar, documentar e verificar esses serviços e restrições é chamado engenharia de requisitos.

A partir do que diz Sommerville (2011), requisitos de usuário são declarações em linguagem natural com diagramas de quais serviços o sistema deverá fornecer a seus usuários e as restrições com as quais este deve operar e requisitos de sistema são descrições mais detalhadas das funções, serviços e restrições operacionais do sistema de software. O documento de requisitos do sistema (às vezes chamado especificação funcional) deve definir exatamente o que deve ser implementado. Pode ser parte do contrato entre o comprador do sistema e os desenvolvedores de software.

Figura 1. Espiral do Processo de Engenharia de Requisitos



Fonte: Sommerville (2011)

Sommerville (2011) afirma que os requisitos precisam ser escritos em diversos níveis de detalhamento para que diferentes leitores possam usá-los de diversas maneiras. Com base nisso, o processo de engenharia de requisitos é constituído primariamente de quatro atividades, sendo elas listadas a seguir (SOMMERVILLE, 2011).

2.2.1. Estudo de Viabilidade

No estudo de viabilidade, é realizada uma estimativa para saber se existe a real possibilidade de satisfazer as necessidades dos *stakeholders* utilizando o que há no estado da arte quando se falar de tecnologias de software e hardware.

O estudo avalia se o sistema analisado será rentável do ponto de vista de negócio e se ele pode ser criado no contexto das restrições orçamentárias impostas pelo contratante. O resultado do estudo é capaz de ajudar na decisão de avançar ou não no desenvolvimento do sistema.

2.2.2. Elicitação e Análise de Requisitos

Este item é iniciado após ser realizado o estudo de viabilidade do projeto. Aqui os engenheiros de software trabalham diretamente com os *stakeholders* para poderem realizar a obtenção de informações relativas ao domínio do projeto, bem como quais as características o produto deve possuir, qual será o desempenho esperado, quais as restrições de hardware no qual o sistema estará executando, entre outros.

As atividades incluídas neste processo são:

- Descoberta de Requisitos: é efetivamente a atividade descobrir os requisitos através dos *stakeholders*. Nessa atividade são descobertos os requisitos de domínio.
- Classificação de Requisitos: é feita a organização de requisitos de maneira que fiquem em grupos coerentes.
- Priorização de Requisitos: está ligada a resolução de conflitos por meio de negociação de requisitos, bem como priorização dos mesmos.
- Especificação de Requisitos: os requisitos são documentados. Sendo possível a geração de documentos formais ou informais de requisitos.

2.2.3. Validação de Requisitos

Sommerville (2011) afirma que a validação de requisitos é o processo pelo qual se verifica se os requisitos definem o sistema que o cliente realmente quer. Ela se sobrepõe à análise, pelo fato de estar preocupada em encontrar problemas com os requisitos.

Segundo Sommerville (2011), os cinco tipos de verificação durante a validação são: Verificações de validade, consistência, completude, realismo e verificabilidade.

- Verificações de Validade: neste momento é possível realizar uma análise profunda em que são identificadas funções que podem ser necessárias ou diferentes para o sistema.
- Verificações de Consistência: exige que os requisitos documentados não podem entrar em conflito, não havendo restrições contraditórias.
- Verificações de Completude: o documento de requisitos deve abranger os requisitos que definem as funções e restrições ansiadas pelo usuário do sistema.
- Verificações de Realismo: através da análise de tecnologias existentes, os requisitos devem ser verificados para saber se é possível a sua implementação. Deve ser levado em conta orçamento e cronograma para o desenvolvimento do sistema.
- Verificabilidade: Para evitar conflitos de entendimento entre cliente e contratante, os requisitos do sistema devem passar por testes que demonstrem que o sistema atende os requisitos especificados no documento.

2.2.4. Importância da Engenharia de Requisitos

O processo de Engenharia de Requisitos é uma das principais medidas para obter o grau de sucesso de um sistema de software verificando o nível em que o mesmo consegue atender ao propósito para o qual foi pretendido (NUSEIBEH; EASTERBROOK, 2000)

Ainda segundo Nuseibeh e Easterbrook (2000), existem diversas dificuldades no processo de desenvolvimento de software, como os objetivos dos *stakeholders* serem complicados de articular ou podem não ser explícitos, fazendo com que atingir esses objetivos seja uma tarefa difícil.

Vale ressaltar que o uso do termo Engenharia em Engenharia de Requisitos serve como um lembrete sobre a importância da Engenharia de Requisitos dentro do processo de desenvolvimento de software, sendo a parte que se concentra em ancorar o desenvolvimento de atividades para um problema do mundo real, tornando a adequação e o custo-efetivo da solução passível de ser analisada. (NUSEIBEH; EASTERBROOK, 2000).

Assim, a importância da Engenharia de Requisitos se deve justamente ao fato da mesma oferecer uma série de conceitos que formalizam uma adequada elicitação e validação de requisitos, garantindo que um determinado sistema consiga satisfazer de maneira adequada às necessidades do cliente, diminuindo assim a margem para erros, bem como custos que poderiam ser desnecessários e economizando tempo (SOMMERVILLE, 2011).

Segundo Pressman (2011), a Engenharia de Requisitos é o conjunto de técnicas que constroem uma ponte para o projeto e a construção do mesmo, através dela é possível examinar o contexto do trabalho de software que deverá ser realizado, bem como as necessidades do projeto e a construção do mesmo devem atender, a Engenharia de Requisitos auxilia também na prioridade de ordem na qual o trabalho deve ser executado, formando assim uma área de grande importância dentro da Engenharia de Software.

2.2.5. Problemas relacionados à Engenharia de Requisitos

Os principais problemas do processo de engenharia de requisitos estão relacionados principalmente à coleta de requisitos e verificação de validade, além de muitos sistemas possuírem diversos tipos de clientes, o que pode gerar requisitos muitas vezes contraditórios ou conflitantes entre si. Bem como existe uma dificuldade para os usuários anteciparem de que forma o sistema desenvolvido afetará a forma como seus trabalhos são realizados (SOMMERVILLE, 2011).

Uma vez que a comunicação entre *stakeholder* e desenvolvedores pode ser falha por diversos fatores, como a própria diferença de conhecimento técnico entre as duas partes, outro problema que ocorre é a falha no processo de documentação de requisitos, Menon *et al*. (2010) afirma que esse problema é ocasionado pela falta de preparo dos profissionais no momento de atuação em atividades envolvendo a engenharia de requisitos.

Ainda segundo Menon *et al*. (2010), a maioria dos cursos de computação ensinam os conceitos de engenharia de software por meio de aulas tradicionais, o que não prepara os estudantes para a indústria, existindo uma lacuna entre o que a indústria precisa e o que os acadêmicos aprendem efetivamente em sala de aula.

Zowghi e Paryani (2003), indicam que erros na especificação de requisitos podem gerar um grande impacto no custo do desenvolvimento de software, tornando evidente que é necessária uma detecção prévia dos problemas durante a análise e validação de requisitos para evitar problemas maiores nas fases de teste e manutenção do software.

Um dos problemas encontrados em relação à engenharia de requisitos é a barreira comunicativa entre desenvolvedores e clientes (ZOWGHI; PARYANI, 2003), sendo necessário sair do ensino tradicional e utilizar meios mais didáticos para o ensino de engenharia de requisitos, como o *Role Playing*, que por sua vez melhora o uso da comunicação em projetos.

Com base nisso e a partir dos resultados obtidos por (MENON *et al*., 2010), pesquisa na qual indica que a opinião dos estudantes sobre o nível de aprendizado ganho no ensino é dividida, sendo que metade afirma que as abordagens de ensino alternativas conseguem

habilitar os acadêmicos para usar os conhecimentos obtidos na graduação em um projeto real, é possível afirmar que existe uma necessidade do uso de abordagens alternativas para auxiliar os acadêmicos que estudam tópicos envolvendo a área de engenharia de requisitos.

2.3. Modelo CMMI e o desenvolvimento de produtos

Os modelos do CMMI são coleções que contém as melhores práticas para auxiliar organizações a melhorar diversos processos no âmbito profissional. Esses modelos são desenvolvidos por times que possuem conhecimentos da indústria, governo e do *Software Engineering Institute* (SEI, 2010).

O modelo CMMI-DEV contém um conjunto de guias que abrange conteúdos para o desenvolvimento de produtos e serviços. O CMMI *for Development* contém práticas que abrangem o gerenciamento de processos, gerenciamento de projetos, engenharia de sistemas, engenharia de hardware, engenharia de software e outros processos utilizados para auxiliar no desenvolvimento e manutenção de produtos (SEI, 2010).

O CMMI-DEV não especifica que um projeto deve seguir um processo em específico ou que um determinado número de itens deve ser desenvolvido, nem que uma performance específica deve ser atingida. Para determinar estes itens, o projeto ou organização deve mapear seus processos nas áreas de processos que estão contidas no modelo.

O CMMI-DEV possui uma área de processo voltada para o desenvolvimento de requisitos e nela são descritos três tipos de requisitos: requisitos do cliente, requisitos do produto e requisitos de componente do produto. Requisitos são identificados e refinados ao longo do ciclo de vida do produto, decisões de design, ações corretivas e *feedback* durante cada fase do ciclo de vida são analisadas para percepção do impacto nos requisitos (SEI, 2010).

Todos os projetos de desenvolvimento possuem requisitos. Os requisitos são a base utilizada para a criação do projeto. O desenvolvimento dos requisitos possui as seguintes atividades, com base no que diz SEI (2010):

- Elicitação, análise, validação e comunicação das necessidades do cliente, expectativas, priorização no entendimento dos requisitos que irão satisfazer os *stakeholders*;
- Coletar e coordenar as necessidades dos *stakeholders*;
- Desenvolvimento do ciclo de vida dos requisitos do produto;
- Estabelecimento de atributos funcionais e de qualidade dos clientes;
- Estabelecimento inicial dos requisitos do produto e os componentes do produto de maneira consistente com os requisitos do cliente.

Outra área de processo descrita no CMMI-DEV e que está relacionada com a engenharia de requisitos é o gerenciamento de requisitos, sendo essa área de processo a que possui como principal objetivo o gerenciamento dos requisitos do produto do projeto e de seus componentes, de maneira que garanta o alinhamento entre o que foi planejado e o que foi executado (SEI, 2010).

Com base em SEI (2010), o processo de gerência de requisitos é responsável por realizar o gerenciamento de todos os requisitos que são recebidos ou criados pelo projeto, incluindo tanto os requisitos técnicos quanto os não-técnicos.

SEI (2010), indica que se a área de desenvolvimento de requisitos é implementada no projeto, o seu processo irá gerar requisitos de produtos e de componentes de produtos que irão ser gerenciador pelo processo de gerenciamento de requisitos.

2.4. Ensino da Engenharia de Software

A ACM/IEEE (2013), afirma que a Engenharia de Software é uma disciplina focada na aplicação de teoria, conhecimento e prática para que seja possível o desenvolvimento eficiente de sistemas de software que atendam aos requisitos de usuários. Seguindo a ACM/IEEE (2013), é indicado que cursos de graduação devem possuir tópicos dentro da área de Engenharia de Software que permitam serem desenvolvidas as competências e habilidades que são esperados por profissionais que trabalham na área, uma vez que estes futuros profissionais, necessitam ter a capacidade de compreender o desenvolvimento de software como um processo que visa assegurar prazos, custos e qualidade para o produto (PORTELA *et al*., 2015).

Com o intuito de encontrar a opinião de profissionais em Engenharia de Software sobre o nível de relevância em que a área é abordada em cursos de Ciência da Computação, Wangenheim e Silva (2009), apresentam uma falta de atenção para alguns tópicos de Engenharia de Software durante a graduação, sendo que estes tópicos são ensinados de uma maneira insuficiente, destacam-se: gerência de projetos, garantia de qualidade de software, gerência de requisitos e desenvolvimento de requisitos. Neste sentido, os profissionais da área aprendem mais sobre estes tópicos durante o trabalho do que no período de formação acadêmica (LETHBRIDGE, 2000). Em relação à estratégia de ensino, o currículo na área de Engenharia de Software exige ir além do formato de aula expositiva (ACM/IEEE, 2013). Sendo assim, é importante considerar a variação de técnicas de ensino e aprendizado. As abordagens mais comuns para ensinar ES incluem aulas expositivas, aulas de laboratório, entre outros (PRIKLADNICKI *et al*., 2009).

Utilizando as informações citadas anteriormente, é necessário que a engenharia de requisitos seja ensinada de maneira satisfatória durante a graduação para que um profissional recém formado tenha a capacidade de lidar com situações diversas que podem ocorrer na indústria, um dos meios para atingir esse nível de ensino é realizando o uso de abordagens de ensino que fogem do escopo tradicional de ensino.

Com base nesses indicadores, o próximo tópico irá tratar sobre abordagens tradicionais e alternativas que são utilizadas para o ensino de tópicos de engenharia de software de maneira geral, com foco primário na área de engenharia de requisitos.

2.5. Abordagens de Ensino Tradicionais e Alternativas

A qualidade dos profissionais da área de Engenharia de Software está diretamente ligada à qualidade da educação que os mesmos tiveram, ainda que existam outros fatores que possam contribuir para isto (BECKMAN *et al*., 1997).

Prikladnicki *et al* (2009) afirmam que existem abordagens de ensino centradas no professor e abordagens centradas no aluno, cada uma possuindo suas peculiaridades. Quanto mais expositiva for a aula, maior a sua tendência em ser focada no professor. No entanto, quanto mais dinamismo e praticidade houver na aula, maior o foco no aluno.

Segundo Prikladnicki *et al* (2009), uma aula dada por meio de abordagem expositiva não costuma possuir muita eficiência, uma vez que apenas o sentido da audição é utilizado. Já eventos que simulem problemas reais e atividades vivenciais permitem que o acadêmico assimile situações diferentes na prática. Sendo assim, a aprendizagem prática tem como característica o envolvimento de uma pessoa em uma certa atividade, com resultados usualmente mais positivos que a aula expositiva, uma vez que os acadêmicos se tornam responsáveis pela definição de rumos de uma situação que é proposta.

Em relação às abordagens utilizadas no ensino de Engenharia de Software como um todo, por meio do trabalho de Prikladnicki *et al*. (2009), foram destacados os principais métodos e abordagens de avaliação adotadas nas disciplinas de Engenharia de Software no Brasil. O Quadro 1 indica quais as categorias utilizadas.

Quadro 1. Categorias de Métodos de Ensino

Características	**Focada no Professor**	**Focada no Aluno**
Papel do professor	Principal fornecedor da informação; Especialista; Avaliador do rendimento.	Facilitador; fornece informação para ajudar na compreensão da informação.

Clima de aprendizagem	Individualista.	Coletiva; Foco na coesão de grupo.
Orientação	Baseada na experiência e nos conhecimentos do professor.	Baseada na experiência e conhecimento dos alunos.
Programa de estudos	Definido pelo professor.	Negociado entre professor e alunos.
Objetivo de ensino	Definido pelo professor; resultado padrão.	Definido pelos alunos; Resultados diferentes para cada aluno.
Aquisição de conhecimento	Enfoque na aquisição; Foco na memorização.	Enfoque na utilização e absorção de conhecimento com foco em problemas reais.
Métodos de ensino	Didático; Grande participação do professor.	Métodos que envolvem a participação dos alunos (técnicas dinâmicas).
Foco na educação	Educação individual.	Educação coletiva.
Avaliação	Executada pelo professor; Uso tradicional de provas e notas.	Os alunos também são responsáveis pela avaliação.

Fonte: Prikladnicki *et al.*, 2009

Ainda na pesquisa de Prikladnicki *et al*. (2009), é apresentado um conjunto de experiências obtidas em quatro instituições de ensino superior no que tange o ensino de Engenharia de Software. As experiências foram planejadas e realizadas a partir da identificação de desafios considerados importantes durante o processo de ensino nos últimos anos.

Na PUCRS, a estratégia de ensino se propõe a tratar dos conceitos de gerência de projetos de forma diferenciada. A interação com os alunos é enfatizada através de dinâmicas que exploram assuntos específicos. São características chave da estratégia: (i) diversificação nas técnicas de dinâmicas de grupo: uso de dinâmicas envolvendo raciocínio, mini-fábrica de aviões e jogo de memória (PRIKLADNICKI *et al*., 2009), maiores detalhes podem ser vistos no subtópico 2.5.1.

No curso de Ciência da Computação da UNIVALI, o enfoque principal da estratégia de ensino é a internalização de conhecimento e habilidades pelo "*learning by doing*". Durante a disciplina é simulado um projeto de software, incluindo o planejamento, a execução (cobrindo as fases de análise de requisitos, design, implementação e testes) e a monitoração final (PRIKLADNICKI *et al*., 2009), a abordagem é maior detalhada no subtópico 2.5.2.

Na disciplina de Engenharia de Software ofertada na graduação em Ciência da Computação da UNIFOR são apresentados inicialmente conteúdos relacionados à qualidade de produtos de software para que posteriormente sejam apresentados conteúdos relacionados a processos de software, abrangendo a Normas ISO, processos ágeis e de forma geral o CMMI

(SEI, 2010) e MR-MPS-BR (SOFTEX, 2016). Após uma ampla contextualização, os conteúdos relevantes de cada processo do MR-MPS-BR são apresentados em trabalhos em equipe (PRIKLADNICKI *et al*., 2009).

A estratégia está baseada no uso de um instrumento didático que propicia o entendimento dos objetivos, da importância e da dinâmica dos processos de software pelo uso de analogias. O instrumento didático é constituído por um processo de aplicação, instrumentação (quebra-cabeças, um processo definido conforme o propósito, formulários de coleta de dados, *check-lists* e outros) e diretrizes de adaptação (permitem a configuração do instrumento para diferentes contextos) (PRIKLADNICKI *et al*., 2009), como pode ser visto no subtópico 2.5.3.

Uma peculiaridade no artigo de Prikladnicki *et al*. (2009) é que ele se limita a apresentar de forma resumida um conjunto de experiências obtidas com o uso de abordagens alternativas em diversas instituições de ensino, não necessariamente criando uma nova ou analisando abordagens focadas especificamente na área da Engenharia de Requisitos.

2.5.1. Uso de dinâmicas de grupo, educação à distância e atividades práticas

Sabe-se que na Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS) a Engenharia de Software é tratada através de seis disciplinas no curso de Sistemas de Informação. Conforme um estudo realizado na referida instituição, um relato de experiência com a disciplina de Gerência de Projetos de Software, disciplina que têm recebido estimulo de aumento na participação dos alunos desde 2004, constituída de 60 horas-aula e possui envolvimento com o ensino de conceitos básicos utilizados na gerência de projeto de software, interligando o ciclo de desenvolvimento com o ciclo de gerenciamento.

Assim, a estratégia de ensino é composta pela interação entre os acadêmicos através de dinâmicas que tratam assuntos em específico. As principais características da estratégia de ensino são: (i) a diversidade de técnicas para a abordagem de dinâmicas em grupo: raciocínio, mini-fábrica de aviões e jogos de memória. A dinâmica com o uso de aviões foi utilizada para repassar conceitos visto em processos de gerência de projetos. O jogo de memória trabalha com conceitos do PMBOK (PMI, 2008) e o raciocínio é primariamente utilizado em dinâmicas envolvendo gerência de comunicação; (ii) uso de laboratório em aulas práticas: por mais que a gerência de projetos possua um foco maior na teoria, a disciplina foi organizada de maneira que os alunos possam abordar os conceitos de maneira prática, compreendendo criação de estrutura analítica de projeto, definição de escopo e desenvolvimento de cronogramas; (iii) planejamento dos trabalhos definidos pelos acadêmicos: as datas de entrega dos trabalhos das disciplinas são

planejadas pelos acadêmicos. Fazendo com que os alunos possuam uma experiência mais próxima da realidade em relação ao planejamento e monitoramento de projetos, neste caso sendo o trabalho da própria disciplina, a entrega dentro do *deadline* vale nota e alterações devem ser justificadas; (iv) uso de aulas semipresenciais: 20% da disciplina são realizadas fazendo o uso de recursos EAD. Nestas aulas, os alunos são expostos a situações reais de projetos e precisam resolver os problemas estando fora do ambiente de sala de aula. Uma contribuição dessa abordagem é o estímulo do desenvolvimento de trabalho em equipe e comunicação à distância.

A avaliação da disciplina é feita por meio de provas individuais e a entrega de dois trabalhos, todos os itens possuindo pesos iguais. Uma parte das atividades realizadas por meio das aulas EAD compõem a nota do trabalho 1, e parte compõe a nota de uma das provas. Existe também uma atividade feita para avaliar a percepção de evolução do aprendizado, onde os acadêmicos preenchem questionários no primeiro dia de aula e recebem o mesmo questionário no último dia, sendo possível avaliar a evolução do aprendizado e comparar com as respostas que foram dadas do início do semestre.

2.5.2. O uso de *capstone projects* e atividades práticas

Na Universidade do Vale do Itajaí (UNIVALI, campus São José) o curso de graduação em Ciência da Computação possui a área de Engenharia de Software abordada primariamente através de quatro disciplinas. Desse modo, o objetivo das disciplinas iniciais é ensinar conceitos básicos. A disciplina de nome Análise e Projeto de Sistemas 2 faz uso de um *capstone project* de maneira sugerida nas recomendações dadas pela ACM/IEEE (2013), no qual grupos de acadêmicos realizam o planejamento e execução de um projeto de software do início até o fim na duração de um semestre.

A perspectiva principal da disciplina é a internalização das habilidades e conhecimentos por meio do "*learning by doing*". Ao longo da disciplina é feita a simulação de um projeto de software, nisso é incluído as fases de planejamento, execução (abrangendo análise de requisitos, design e implementação) e o monitoramento realizado ao final. Cada equipe de projeto é composta por cerca de seis acadêmicos, onde cada aluno assume uma responsabilidade principal e participa de maneira ativa em todas as fases do projeto. O professor faz o papel do cliente que deseja o produto final, também descrevendo de maneira de alto-nível as necessidades do cliente.

O projeto é inicializado na fase de planejamento, nessa fase é elaborado um projeto. Logo após são iniciadas as atividades técnicas seguindo o plano de projeto e seguindo também

um modelo de processo de desenvolvimento pré-definido nos moldes do modelo de ciclo de vida cascata. Ao longo da execução do projeto, os acadêmicos passam pelas fases de análise de requisitos, *design*, implementação e testes. Paralelamente à execução do projeto, os acadêmicos realizam a coleta de dados referentes aos prazos e esforços gastos nas atividades. Ao fim do projeto estes dados são avaliados e representam um resumo de monitoração de projeto e são comparados com as estimativas iniciais do plano de projeto. Ao fim de cada fase, os acadêmicos realizam a entrega e apresentam os resultados parciais do projeto.

2.5.3. O uso de atividades lúdicas e jogos

A Universidade de Fortaleza possui a disciplina de Engenharia de Software no curso de graduação em Ciências da Computação, a disciplina é ofertada no penúltimo semestre possui como pré-requisitos as disciplinas Análise e Projeto de Sistemas 1 e 2. O ponto principal da disciplina está nos modelos de maturidade de software, com ênfase no MR-MPS-SW (SOFTEX, 2016).

Inicialmente é feita a apresentação de conteúdos relevantes à qualidade de produtos de software e posteriormente são apresentados conteúdos de processos de software, abrangendo desde normas ISO e formas gerais do CMMI (SEI, 2010) e MR-MPS-SW (SOFTEX, 2016). Após isso os conteúdos que possuem relevância dentro do MR MPS são apresentados em equipe, ao fim da disciplina é feita uma atividade utilizando LEGO para consolidar a compreensão dos conhecimentos.

Um dos pontos levados em conta para a utilização de LEGO é a importância de atividades lúdicas para a retenção de conteúdo. Além disso, as peças desse brinquedo possibilitam projetar, tomando como base requisitos definidos, um produto a ser construído, havendo um certo nível de paridade com a construção de software. Outro ponto a ser observado para esta abordagem é o fato do professor ser capaz de realizar mais intervenções junto aos acadêmicos e observar de perto as dificuldades enfrentadas ao longo da execução dos processos, podendo assim saber quais conteúdos foram menos compreendidos. O planejamento para o uso de LEGO é feito para ser executado em três aulas, seguindo um roteiro pré-definido. Os acadêmicos devem planejar e construir um projeto real, fazendo também o monitoramento e controle de projeto a partir dos processos que foram definidos.

O próximo tópico abordará trabalhos que possuem relação com a criação ou o uso de abordagens de ensino que auxiliem na capacitação de acadêmicos de computação dentro da área de engenharia de software

2.6. Trabalhos relacionados

Furtado e Oliveira (2018), realizam uma abordagem de ensino com foco no indivíduo, com um estudo voltado ao acadêmico, visando identificar se o aprendizado voltado ao acadêmico é preferível pelos participantes e qual o grau de adoção utilizado pelos professores de Engenharia de Software. O resultado do estudo apresenta um indicador de que a abordagem alternativa consegue obter melhores resultados do que as abordagens tradicionais de ensino. Contudo, essa abordagem é utilizada no contexto da Engenharia de Software e não exclusivamente na Engenharia de Requisitos.

Outro trabalho a destacar é o de Santos *et al*. (2014), que realiza uma pesquisa com o intuito de identificar, analisar e discutir ferramentas e métodos que apoiem o ensino de engenharia de software. Por meio de uma revisão sistemática da literatura, foi feita a análise de 26 pesquisas. A partir da pesquisa foi possível identificar três perspectivas de aspectos conceituais: o tipo de proposta utilizado na pesquisa, qual o tópico dentro da área de Engenharia de Software que foi abrangido pela proposta e qual o público alvo.

No que tange aos tópicos abordados nas pesquisas de Santos *et al* (2014), 28% dos estudos não possuem um foco específico, enquanto 72% das pesquisaram estavam relacionadas principalmente aos tópicos de Gerência de Projeto de Software e Testes de Software. Das propostas, 48% estavam voltadas para a criação de ferramentas computacionais que dessem suporte ao ensino de Engenharia de Software, enquanto 32% das propostas tinham ligação com atividades didáticas sem necessariamente utilizar algum software educativo e os últimos 20% das propostas não possuíam uma nova abordagem para o ensino de Engenharia de Software, mas contaram o relato de experiências de professores ao lecionarem a disciplina em cursos de graduação ou especialização.

Semelhantemente ao trabalho de Prikladnicki *et al*. (2009), o artigo de Santos *et al*. (2014) trata de analisar diversas propostas sobre a maneira como o ensino de Engenharia de Software é realizada, sua pesquisa é realizada por meio de uma revisão sistemática da literatura, não focando na criação de uma abordagem, mas sim em encontrar e comentar as já existentes.

Destaca-se também a pesquisa de Zowghi e Paryani (2003) faz o uso de uma atividade dinâmica para o ensino de engenharia de requisitos, por meio de *role playing* para incentivar e melhorar a comunicação entre desenvolvedores e clientes. A dinâmica foi feita em uma turma do segundo ano da graduação em Tecnologia da Informação, na Universidade de Tecnologia, em Sydney, no ano de 2002, a ideia da disciplina era ensinar aos acadêmicos três habilidades consideradas fundamentais para engenheiros de requisitos:

- Habilidades de entrevistas e *groupware* para a elicitação e validação de requisitos;

- Habilidades de análise e modelagem para a resolução de problemas;
- Habilidades de escrita para a especificação de requisitos.

A abordagem utilizada por Zowghi (2003) divide os acadêmicos nos papéis de usuários (clientes) e engenheiros de requisitos (analistas). Existe o revezamento de papéis ao longo da disciplina, desta forma os participantes podem experimentar as dificuldades encontradas por cada grupo e aprender a como revisar e refinar os modelos de requisitos que os mesmos estão criando iterativamente.

Após o término da disciplina ministrada em Zowgui (2003), foi feita uma análise dos dados coletados ao longo do tempo da disciplina, os dados coletados se baseiam principalmente em feedback pós aula e comentários dos participantes. O resultado final afirma que os acadêmicos gostaram de participar das dinâmicas e se sentiram motivados pelos desafios que uma abordagem de ensino ativa proporciona. Como um todo, os estudantes disseram que aprenderam um gama de detalhes tanto técnico quanto não-técnico na área de Engenharia de Requisitos.

A pesquisa de Zowgui (2003) foca no ensino de Engenharia de Requisitos por meio de uma abordagem alternativa, no entanto sua abordagem se limita ao uso de atividade de *role playing*, descartando outras possibilidades de atividades práticas que podem ser utilizadas para a efetivação do conhecimento.

Por meio dos trabalhos apresentados neste tópico é possível observar que o uso de abordagens alternativas para o ensino de tópicos de engenharia de software consegue obter um grau de aprendizado maior se comparado com o uso de abordagens tradicionais e expositivas

3 METODOLOGIA DE ENSINO

Segundo Nunes (2015), os gestores acadêmicos devem estabelecer quais as competências necessárias na Engenharia de Software e a partir disso encontrar os conteúdos que permitam os acadêmicos conseguirem essas competências, sendo necessário determinar como as disciplinas serão abordadas: à distância, por meio de seminários, presencial, em grupos ou por outra abordagem.

Uma vez que a identificação, validação e todo o processo de Engenharia de Requisitos possui um grau de complexidade, foi necessário identificar quais as competências são esperadas por um profissional atuante na área, para que a abordagem de ensino possa focar na aquisição e melhoramento dessas habilidades por parte dos acadêmicos participantes, como dito por Nunes (2015).

O modelo CMMI-DEV foi utilizado como referência, uma vez que seus processos são utilizados como base em diversas áreas (SEI, 2010). Foram identificadas um total de 10 competências necessárias para um profissional poder atuar na área de ER de maneira satisfatória para a indústria.

No CMMI-DEV, a Engenharia de Requisitos é vista na área de processo *Requirements Development* (RD), no nível 3 do modelo, e na área *Requirements Management* (REQM), no nível 2 do modelo, no Quadro 1 é possível observar as competências e habilidades, com os indicadores de *Specific Goals* (SG) e *Specific Practices* (SP).

Quadro 2. Competências e Habilidades esperadas de um profissional da Engenharia de Requisitos, com base no CMMI-DEV (SEI, 2010)

Competências	**Habilidades**	**CMMI-DEV**
Elicitar as necessidades e expectativas dos stakeholders para todas as fases do ciclo de vida do produto	● Utilizar métodos para elicitar requisitos e expectativas dos stakeholders;	RD SG 1 SP 1.1
Transformar as necessidades e expectativas dos stakeholders em requisitos do cliente	● Traduzir as necessidades e expectativas dos stakeholders em requisitos do cliente devidamente documentados; ● Estabelecer e manter uma ordem de prioridade dos requisitos funcionais e não funcionais; ● Definir condições de validação e verificação	RD SG 1 SP 1.2

Competências	Habilidades	CMMI-DEV
Estabelecer e manter requisitos de produto, que são baseados nos requisitos do cliente	● Desenvolver requisitos em termos técnicos para poderem ser utilizados no design do produto; ● Derivar requisitos que são resultados de decisões de design; ● Desenvolver requisitos arquiteturais, obtendo a atributos de qualidade críticos e as medidas necessárias para estabelecer a arquitetura do produto e o design;	RD SG 2 SP 2.1
Identificar requisitos de interface	● Identificar interfaces tanto externa ao produto quanto interna; ● Desenvolver os requisitos para as interfaces identificadas	RD SG 2 SP 2.3
Estabelecer e manter uma definição de funcionalidades e atributos de qualidade necessários	● Identificar funcionalidades e atributos de qualidade desejados; ● Analisar e quantificar as funcionalidades solicitadas pelo usuário final; ● Analisar os requisitos para identificar divisões lógicas ou funcionais; ● Dividir requisitos em subgrupos, seguindo um critério estabelecido, para facilitar a análise de requisitos;	RD SG 3 SP 3.2
Analisar requisitos para garantir que os mesmos são necessários e suficientes	● Analisar os requisitos para determinar se satisfazem os objetivos de requisitos de nível superior; ● Analisar requisitos para garantir que são completos, exequíveis e verificáveis; ● Identificar requisitos chave que possuem grande influência no custo, calendário, performance ou risco; ● Identificar medidas de performance técnicas que serão rastreadas durante o desenvolvimento;	RD SG 3 SP 3.3
Validar requisitos para garantir que o produto resultante terá a performance desejada no ambiente do usuário final	● Analisar os requerimentos para determinar o risco de o produto não ter a performance desejada no ambiente de uso; ● Explorar a adequação dos requerimentos através de representações do produto (protótipos, modelos) e obtendo o feedback dos stakeholders; ● Avaliar o design enquanto o mesmo se torna complexo no contexto do ambiente de	RD SG 3 SP 3.5

Competências	Habilidades	CMMI-DEV
	validação de requisitos para identificar erros e expor necessidades e requisitos de cliente;	
Desenvolver o entendimento dos requisitos através de quem provê o requisito	● Estabelecer critérios que distingam adequadamente os provedores de requisitos; ● Criar critério para avaliar a aceitação dos requisitos; ● Analisar requisitos para garantir que os critérios estabelecidos estão sendo atingidos; ● Atingir o entendimento dos requisitos com os provedores de requisitos	REQM SP 1.1
Obter comprometimento aos requisitos a partir dos participantes do projeto	● Avaliar o impacto dos requisitos nos compromissos já existentes;	REQM SP 1.2
Gerenciar as mudanças nos requisitos ao longo do projeto	● Documentar todos os requisitos e as devidas mudanças que foram originadas do projeto; ● Manter o histórico de mudanças de requisitos; ● Avaliar o impacto das mudanças de requisitos no ponto de vista de stakeholders;	REQM SP 1.3
Manter o rastreamento bidirecional nos requisitos	● Manter o rastreamento dos requisitos para garantir que a origem de requisitos de baixo nível está documentada; ● Manter o rastreamento dos requisitos de um requisito para seu requisito derivado; ● Gerar uma matriz de rastreabilidade de requisitos	REQM SP 1.4
Garantir que os planos de projeto se mantenham alinhados com os requisitos	● Revisar os planos de projetos para garantir consistência com os requisitos e suas mudanças; ● Identificar a origem de inconsistências (se existirem); ● Identificar necessidade de mudanças no plano de projeto devido às mudanças de requisitos; ● Iniciar as medidas necessárias para eventuais correções	REQM SP 1.5

Fonte: Autoria própria (2018)

A seleção de técnicas, métodos e recursos de ensino adotados nesta metodologia foi baseada no trabalho de Portela (2017), que tem por objetivo potencializar a adoção conjunta destes itens, através de um ciclo iterativo a fim de atender os diferentes perfis de aprendizagem.

O modelo de ensino de Portela (2017) é fundamentado no ciclo de aprendizagem de Kolb (1984) e na metodologia iterativa de ensino proposta por Gary *et al*. (2013).

3.1. Base para a criação da abordagem

A pesquisa de Portela (2017) não realiza a criação de uma abordagem nova, mas possui a intenção de integrar as abordagens alternativas que foram identificadas na literatura em um único modelo iterativo, de maneira que seja possível realizar um ciclo de aprendizagem e abordar os diferentes tipos de aprendizagem dos acadêmicos.

Portela (2017) faz uso de diversas abordagens em sua pesquisa, todas voltadas para um lado mais centrado no aluno para que seja possível efetivar o conhecimento. O principal intuito de utilizar diversas abordagens é realizar a integração das mesmas em um modelo iterativo, podendo analisar diferentes tipos de aprendizagem nos participantes (PORTELA, 2017).

O modelo desenvolvido por Portela (2017) se demonstra bastante flexível em sua aplicação, sendo capaz de ser utilizado tanto para o ensino de uma unidade de conhecimento específica quanto para uma disciplina da área de ES completa, além de possuir um sistema *web*, que tem como intuito realizar, de maneira sistemática, a aplicação do modelo em sala de aula, permitindo que possa ser seguido um fluxo iterativo do modelo (como é possível observar na Figura 2), registrar dados da execução e até mesmo acessar materiais de apoio. Para execução em uma disciplina específica, o modelo de Portela (2017) possui uma preparação a ser realizada:

Figura 2. Modelo Iterativo para o Ensino de Engenharia de Software

Fonte: Portela (2017)

1. Preparação da disciplina: com o professor realizando a preparação da disciplina a ser abordada, é possível definir a(s) unidade(s) de conhecimento que serão abordadas na aplicação do modelo e também selecionar quais competências devem ser desenvolvidas nos acadêmicos. A preparação inclui a escolha de material instrucional, como dinâmicas, videoaulas, jogos e planejamento de um projeto prático que será desenvolvido. Ao fim da etapa de preparação, o professor responsável pela execução do modelo pode criar uma conta no sistema do modelo e realizar o cadastro da disciplina que será alvo da aplicação do modelo;
2. Aplicação do modelo: na segunda fase é feita a aplicação do modelo em sala, dada a adequação do modelo para o uso em aula, o mesmo pode ser aplicado fazendo uso do cronograma de aulas da disciplina, onde a cada horário (usualmente 50 minutos), é necessário ser realizada uma etapa do modelo, a exceção da etapa de contextualização que deve ter sua duração adaptada para o nível de complexidade e tempo necessários para realizar as atividades práticas determinadas do projeto. Ao longo da execução do modelo, o professor possui acesso à base de dados do modelo, podendo acessar artigos, jogos, videoaulas, propostas de dinâmicas, bem como atividades do projeto;
3. Atualização da base de dados: ao fim da aplicação do modelo iterativo, o professor responsável pode realizar a colaboração na base de dados do modelo fazendo a submissão do material utilizado ao longo da execução do modelo, com o intuito de manter a base de dados crescendo de maneira colaborativa e sempre atualizada.

Na pesquisa de Portela (2017), um dos objetivos é incorporar o modelo iterativo as práticas de capacitação utilizadas amplamente na indústria do desenvolvimento de software. Essas práticas fazem com que os profissionais da aplicação estejam aptos tanto para adquirir novas competências quanto para aprimorar as já existentes. Desse modo, é esperado que o acadêmico possa adquirir e aprimorar técnicas que sejam cobradas por um profissional que seja iniciante na área de computação.

Com base no modelo de Portela (2017), a abordagem de ensino utilizada na presente pesquisa é centrada na leitura de artigos e relatos de experiência, discussão de casos práticos, uso de jogos e simuladores, além da realização de projetos práticos e reflexão, por parte do aluno, sobre o conteúdo aprendido e as atividades realizadas. Os parágrafos seguintes detalham como foi realizado o planejamento da disciplina desta pesquisa.

A disciplina foi planejada para ser executada como um tópico optativo no curso de Ciência da Computação, para alunos que já tivessem cursado a disciplina base de Engenharia de Software. Assim, a disciplina tinha 60h de carga-horária, dividida em 4 horas de aula na

semana. Neste contexto, no primeiro encontro com a turma, foi fornecido o material de apoio necessário para que os alunos fossem capazes de realizarem a etapa de preparação para o curso.

O Quadro 3 abaixo lista planejamento realizado para a disciplina, onde são apresentados: o conteúdo programático, tendo em vista as competências planejadas para a disciplina; a estratégia de ensino, definida conforme o nível de aprendizagem pretendido para o tópico e seus resultados esperados; os resultados esperados, o que o aluno deve ser capaz de realizar após o estudo da unidade; e o nível de aprendizagem, utilizando-se uma terminologia baseada na taxonomia de Bloom (1956), que consiste em conhecimento, entendimento e aplicação, onde: Conhecer, refere-se a lembrar do material previamente ensinado; Compreender, diz respeito a entender a informação e o significado do material ensinado; e Aplicar, refere-se a usar o material aprendido em situações novas e concretas. É importante ressaltar que aplicar engloba compreender que engloba conhecer (NUNES; YAMAGUTI; NUNES, 2016).

Quadro 3. Agenda da Disciplina

<table>
<tr><th>Conteúdo</th><th>Estratégia de Ensino</th><th>Resultados Esperados</th><th>Tempo</th><th>Nível de Aprendizagem</th></tr>
<tr><td colspan="5">1. Introdução à Engenharia de Requisitos</td></tr>
<tr><td>1.1. Apresentação da disciplina</td><td>Dinâmica das cartas para criação de produto.</td><td>O aluno deve ser ter uma noção sobre os anseios, estresse e dificuldade na execução de projeto quando os requisitos são coletados de maneira errônea</td><td>45min</td><td>Conhecer</td></tr>
<tr><td rowspan="2">1.2. A importância da Engenharia de Requisitos</td><td>Leitura e Discussão em sala do material de apoio.</td><td rowspan="2">O aluno deve ser capaz de conhecer a importância da Engenheira de Requisitos para a qualidade do produto de software</td><td>45min</td><td rowspan="2">Conhecer</td></tr>
<tr><td>Jogo: Software Quantum</td><td>45min</td></tr>
<tr><td rowspan="2">1.3. O processo de Engenharia de Requisitos</td><td>Leitura e Discussão em sala do material de apoio.</td><td rowspan="2">O aluno deve conhecer as etapas, papéis e atividades envolvidas no processo de engenharia de requisitos</td><td>45min</td><td rowspan="2">Conhecer</td></tr>
<tr><td>Jogo: A Ilha dos Requisitos</td><td>30min</td></tr>
<tr><td colspan="5">2. Descoberta de Requisitos</td></tr>
<tr><td>2.1. Atividades envolvidas</td><td>Leitura e Discussão em sala do material de apoio.</td><td>O aluno deve ser capaz de compreender as relações entre as</td><td>45min</td><td>Compreender</td></tr>
</table>

Conteúdo	Estratégia de Ensino	Resultados Esperados	Tempo	Nível de Aprendizagem
	Dinâmica da Descoberta de Requisitos	atividades desenvolvidas na descoberta de requisitos	45min	
2.3. Principais dificuldades	Leitura e Discussão em sala do material de apoio.	O aluno deve compreender as principais dificuldades da descoberta de requisitos e ser capaz de desenvolver formas de mitigar estes problemas	45min	Compreender
	Seminário sobre as dificuldades encontradas na dinâmica anterior		45min	
2.2. Técnicas	Leitura e Discussão em sala do material de apoio.	O aluno deve ser capaz de compreender as diversas técnicas de descoberta de requisitos	45min	Compreender
	Dinâmica do GameMaker		50min	
3. Especificação e Documentação				
3.1. Tipos de Requisitos	Leitura e Discussão em sala do material de apoio.	O aluno deve conhecer e saber identificar os tipos de requisitos	45min	Conhecer
3.2. Formas de documentar os requisitos	Leitura e Discussão em sala do material de apoio.	O aluno deve ser capaz de compreender as diversas técnicas de documentação de requisitos	45min	Compreender
	Continuação da Dinâmica do GameMaker		50min	
4. Projeto Final				
4.1. Projeto prático	Dinâmica da entrevista com o cliente	O aluno deve ser capaz de aplicar os conhecimentos obtidos ao longo da disciplina e realizar todo o processo de engenharia de requisitos (com a supervisão do professor)	4 semanas	Aplicar

Fonte: Autoria própria (2018)

3.2. Os Jogos e Projetos Práticos Utilizados

Para cada tópico com o nível de aprendizagem de Conhecer ou Aplicar foi definida uma atividade lúdica e/ou um projeto prático para que os alunos possam internalizar os conceitos aprendidos e aplicá-los para resolver problemas reais no contexto da Engenharia de Requisitos. A duração de cada atividade foi planejada levando em consideração o tempo disponível para cada encontro semanal (cerca de 100 minutos), se existiam mais de uma atividade planejada

para o mesmo encontro, a dificuldade dos problemas propostos e também norteada através de uma pré-execução realizada pelo professor durante o desenvolvimento das atividades.

3.2.1. Dinâmica das Cartas

A dinâmica das cartas para a criação de produto se dá da seguinte forma: a turma é dividida em grupos, cada grupo deve criar o mesmo produto com base nos requisitos solicitados pelo cliente (o professor). Ao longo da aula o cliente adiciona ou remove um requisito, obrigando os grupos a realizarem os ajustes necessários no protótipo. Dessa forma, é possível fornecer para os participantes uma visão inicial da volatilidade dos requisitos, das dificuldades ocasionadas pela má coleta destes requisitos e as consequências disso ao longo de um projeto.

A ideia de utilizar a dinâmica das cartas se deu justamente pelo rápida compreensão e execução da mesma, permitindo uma maior quantidade de tempo para o aluno realizar a prática e estabelecer os conceitos iniciais da Engenharia de Requisitos.

3.2.2. Jogo: Software Quantum

Software Quantum (KNAUSS *et al.*, 2008) é um jogo criado para ser executado na *web* e que simula o processo de engenharia de requisitos. O mesmo foi criado com o intuito de oferecer um modelo que seja simples e fácil de aprender em alguns minutos, mas ao mesmo tempo tenha a capacidade de transmitir as principais ideias contidas na Engenharia de Requisitos. É possível observar alguns detalhes do jogo na Figura 3.

Figura 3. Jogo Software Quantum

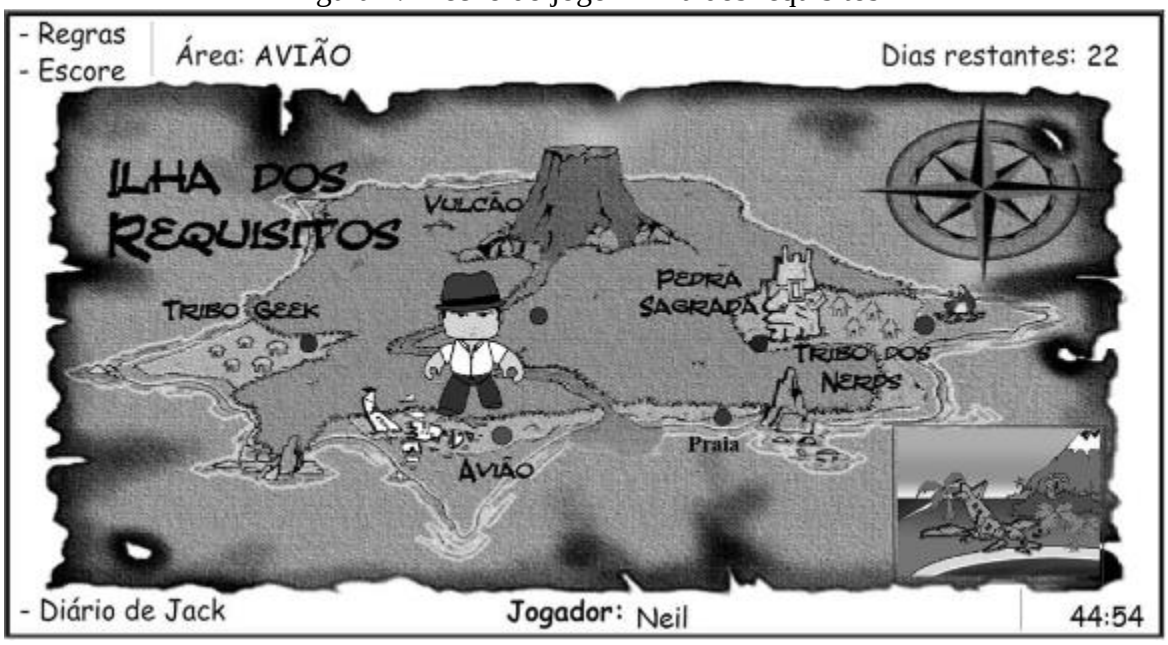
Fonte: Knaus *et al.* (2008)

O jogo não cobre necessariamente todos os aspectos dinâmicos de projetos de software, mas a ideia é que o exista um foco na tensão entre desenvolver um sistema corretamente e desenvolver o sistema correto para o cliente.

3.2.3. Jogo: Ilha dos Requisitos

O jogo ilha dos requisitos (THIRY *et al*., 2010) possui uma estória na qual o protagonista sofre um acidente em uma viagem e vai parar em uma ilha desconhecida. O principal objetivo do jogo é ajudar o protagonista a escapar da ilha junto com os membros da tribo dos Nerds, antes que a ilha seja destruída por um vulcão. Ao longo do jogo são apresentados sete desafios que auxiliam o jogador a entender melhor os conceitos e o processo de engenharia de requisitos, na Figura 4 é possível identificar o mapa do jogo.

Figura 4. Trecho do jogo A ilha dos requisitos



Fonte: Thiry *et al.* (2010)

Ao longo dos desafios o jogador consegue receber um feedback imediato de suas ações, como dicas para relembrar conceitos que estão relacionados a engenharia de requisitos e resultados que foram obtidos ao final de cada desafio. Uma característica do jogo é que ele procura envolver o participante em uma situação análoga as situações que poderiam ser resolvidas por meio de práticas relacionadas a Engenharia de Requisitos. Os desafios são:

- Processo de Engenharia de Requisitos: o jogador deve ordenar as fases do processo de engenharia de requisitos;
- Validação de requisitos: é necessário levar os requisitos ao cliente para que seja feita a validação deles antes do início execução efetiva do projeto;
- Papel do analista de requisitos: o jogador deve identificar corretamente quais as habilidades de um analista de requisitos;
- Análise do problema: é necessário realizar a distinção entre os problemas e suas possíveis soluções;

- Especificação dos Requisitos: o jogador deve realizar a classificação dos requisitos entre funcionais ou não funcionais;
- Classificação dos requisitos: é necessário fazer a classificação dos requisitos apresentados como funcionais ou não funcionais;
- Gerência dos requisitos: o jogador deve ordenar o processo de mudança dos requisitos listados e identificar as atividades que fazem parte da gerência de requisitos.

Ao término do jogo o acadêmico deverá estar apto a identificar como se dá o processo de coleta de requisitos.

3.2.4. Dinâmica do *GameMaker*®

GameMaker® é uma ferramenta proprietária utilizada para o desenvolvimento de jogos. O uso do *GameMaker*® se baseia no ensino de Engenharia de Software por meio do *Game Design* (KAJAL; MARK, 2005), que possui um fator de diversão que pode engajar os acadêmicos a participarem mais ativamente do ensino. A ferramenta foi utilizada de forma que os participantes deveriam desenvolver um protótipo de jogo baseado no que o cliente solicitava.

Para realizar a atividade, os acadêmicos foram instruídos de forma que o professor em sala seria o cliente e os acadêmicos seriam parte de um time de desenvolvimento, o cliente solicitava requisitos para serem implementados no produto (o jogo que estava sendo criado na ferramenta *GameMaker*®), ao longo da atividade os participantes puderam experimentar com funciona uma coleta de requisitos e a sua implementação no produto em desenvolvimento, bem como experimentar as dificuldades ocasionadas por falhas de comunicação e consequentemente na coleta de requisitos.

3.2.5. Projeto Prático

No Projeto Prático, os participantes criaram “empresas” que deveriam entrevistar um cliente externo para a criação de um produto, com reuniões semanais e apresentações de protótipos para verificar se o projeto estava de acordo com as necessidades do cliente, para isso os acadêmicos fizeram uso do processo de engenharia de requisitos e todas as boas práticas que foram aprendidas ao longo da disciplina.

O cliente era a secretaria da coordenação do curso de Ciência da Computação, onde o principal problema que os grupos precisavam resolver era a sistematização do processo seletivo de um projeto de extensão.

Durante as reuniões semanais as equipes entrevistavam o cliente com o intuito de obter os requisitos necessários para o desenvolvimento do produto, quando as equipes possuíam protótipos estes eram apresentados para o cliente com o objetivo de validar os requisitos que foram coletados anteriormente e verificar quais mudanças poderiam ser realizadas na experiência de usuário.

Ao final do projeto os grupos apresentaram os softwares finais para o cliente, desenvolvidos a partir dos requisitos levantados, que fez a escolha do que melhor atendeu suas necessidades e foi posto em uso pelo projeto de extensão.

Este capítulo descreveu o processo de concepção, elaboração e construção da abordagem de ensino. A abordagem tem como intuito apoiar o ensino da Engenharia de Requisitos em cursos de Computação de forma que estimule e motive os alunos e que esteja alinhada com as abordagens de ensino humanísticas, onde o processo de aprendizagem e ensino é centrado no aluno. A metodologia é apoiada em Problem Based Learning (PBL) e na Teoria de Aprendizado de Kolb (1984), através da aplicação de uma adaptação do Ciclo de Aprendizagem de Kolb proposto por Portela (2017). A metodologia é composta por leitura de relatos de experiência, com PBL, discussão de casos práticos, uso de jogos lúdicos e dinâmicas, realização de projetos práticos e reflexão sobre o aprendido.

4 *SURVEY*

Este capítulo abordará os detalhes dos resultados obtidos com o *survey* que foi aplicado aos alunos que participaram da proposta de disciplina de Engenharia de Requisitos, com duração de um semestre.

O principal objetivo para a elaboração do *survey* é a coleta de dados para a verificação do percentual de melhora no aprendizado de tópicos dentro de Engenharia de Requisitos, uma vez que o ensino tradicional por vezes se mostra incapaz de preparar os acadêmicos para as situações que podem ocorrer no mercado de trabalho.

A população-alvo deste *survey* é caracterizada por acadêmicos de cursos de computação que possuam disciplinas relacionadas à Engenharia de Software, a turma em específico cursa o quarto semestre de Ciência da Computação em uma instituição pública de ensino. No que tange ao design de coleta dos dados, ele pode ser considerado de corte, transversal, uma vez que os participantes informam dados relacionados às suas experiências passadas.

O *survey* utilizado coleta dados quantitativo sobre os acadêmicos que participaram do experimento, relativo a suas informações e preferências individuais. Deste modo, para a coleta de dados, foi utilizado um questionário formado por questões objetivas e subjetivas.

Visando reduzir as chances de questionários respondidos de forma incompleta, errônea no que tange à inserção de informações e para avaliar o tempo gasto a tarefa, foi feito um pré-teste de aplicação do *survey*.

4.1. As Questões Do *Survey*

Para realizar a aplicação do *survey*, foi feito o uso de um questionário formado de questões objetivas e subjetivas, como supracitado. Para efetuar a definição das perguntas do questionário, foram pesquisados referenciais sobre Engenharia de Requisitos que fossem utilizados amplamente pela comunidade científica e indústria de software.

Tomando por base os referenciais citados na seção anterior, foi realizada a definição das questões do *survey*. O Quadro 4 exibe questões com o intuito de elencar o background e motivações do aluno para o aprendizado. Tendo em vista que um aluno motivado possui mais chances de aprender do que um aluno desmotivado.

Quadro 4. Questões com respostas em escala *likert*

Questão	**Opções de Resposta**

A Engenharia de Software uma é área importante do desenvolvimento de software	Resposta realizada a através de escala *likert*, sendo 1. Discordo totalmente; 2. Discordo parcialmente; 3. Neutro; 4. Concordo parcialmente; 5. Concordo totalmente.
A Engenharia de Requisitos é importante para a entrega de produtos que atendam às necessidades do cliente	
Estou motivado a aprender mais sobre a engenharia de requisitos	
O conteúdo ensinado na disciplina foi relevante	

Fonte: Autoria própria (2018)

O Quadro 5 mostra as questões criadas para os participantes do experimento com o objetivo de avaliar os tópicos de Engenharia de Requisitos contemplados em aulas.

Quadro 5. Questões com respostas em escala *likert*

O conteúdo abordado pela disciplina foi suficiente para entender como a Engenharia de Requisitos funciona em uma organização	Resposta realizada a através de escala *likert*, sendo 1. Discordo totalmente; 2. Discordo parcialmente; 3. Neutro; 4. Concordo parcialmente; 5. Concordo totalmente.
A abordagem escolhida para a disciplina teve uma boa integração da teoria com a prática	
As dinâmicas/práticas foram realizadas em tempo adequado:	
As dinâmicas/práticas tinham um nível de complexidade adequado	
As dinâmicas/práticas desenvolvidas não restringiam a criatividade dos alunos para pensarem em suas próprias soluções	
As dinâmicas/práticas tornaram o processo de aprendizagem divertido e desafiador	

Ao longo da disciplina, a abordagem de ensino me manteve motivado a aprender	

Fonte: Autoria própria (2018)

A seguir, no Quadro 6, é possível observar as questões que fizeram uso de respostas com múltipla escolha e subjetivas.

Quadro 6. Questões com respostas de múltipla escolha e subjetivas

Questões	**Opções de Resposta**
Quais das técnicas estudadas foram utilizadas durante a descoberta de requisitos?	Alternativas Brainstorming, Entrevistas, Questionários, Fluxogramas, Prototipagem
Quais das técnicas aplicadas você considerou mais fácil para descobrir e garantir o entendimento das necessidades do cliente?	Alternativas Brainstorming, Entrevistas, Questionários, Fluxogramas, Prototipagem
Quais das técnicas aplicadas para a descoberta de requisitos você usaria novamente?	Alternativas Brainstorming, Entrevistas, Questionários, Fluxogramas, Prototipagem, Etnografia
Quais das técnicas não aplicadas para a descoberta de requisitos você utilizaria em uma próxima oportunidade?	Alternativas Brainstorming, Entrevistas, Questionários, Fluxogramas, Prototipagem, Etnografia
Quais das técnicas aplicadas para a descoberta de requisitos foi mais fácil de aprender?	Alternativas Brainstorming, Entrevistas, Questionários, Fluxogramas, Prototipagem, Etnografia
Quanto a documentação dos requisitos, qual o documento você considera que atingiu os melhores resultados para a comunicação com o cliente?	Alternativas: Casos de Uso, Documento de Requisitos, Discovery Canvas
Quanto a documentação dos requisitos, qual o documento você considera que atingiu os melhores resultados para a comunicação com o time de desenvolvimento?	Alternativas: Casos de Uso, Documento de Requisitos, Discovery Canvas
Quanto a documentação dos requisitos, qual a técnica você considera mais fácil de aprender?	Alternativas: Casos de Uso, Documento de Requisitos, Discovery Canvas
Quais os pontos fortes da disciplina?	Resposta subjetiva

E quais os pontos fracos, suas observações e sugestões de melhoria?	Resposta subjetiva

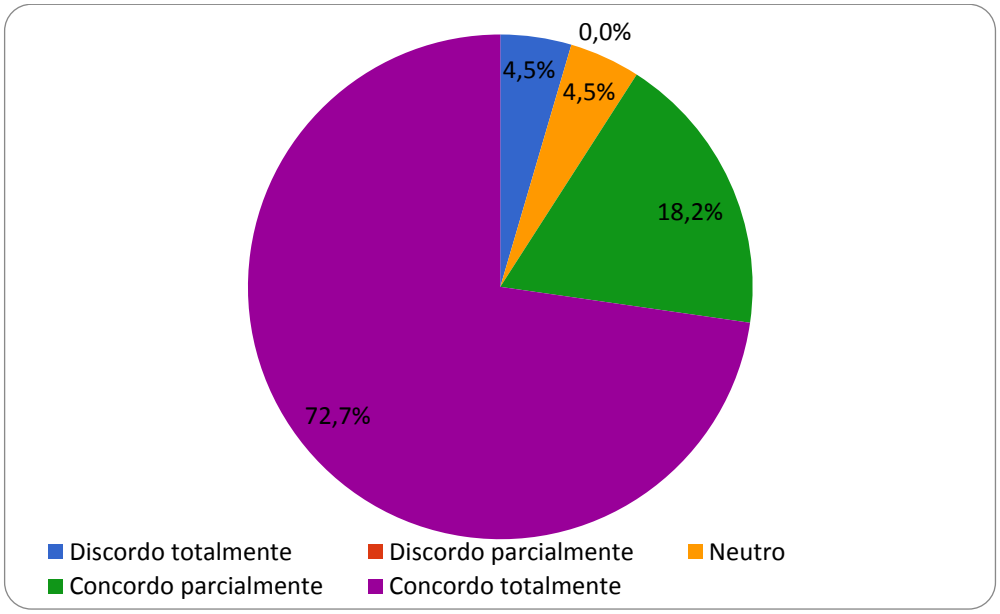
Fonte: Autoria própria (2018)

4.2. Quanto aos resultados obtidos pelo *Survey*

Nesta subseção serão apresentados os dados obtidos no *survey* que foi respondido pelos acadêmicos participantes da proposta de disciplina, os comentários sobre as respostas serão discutidos na seção subsequente.

Em relação à importância da área de Engenharia de Software para o desenvolvimento de software, 72.7% dos acadêmicos concordam totalmente e 18.2% concordam parcialmente, os resultados podem ser visualizados na Figura 5.

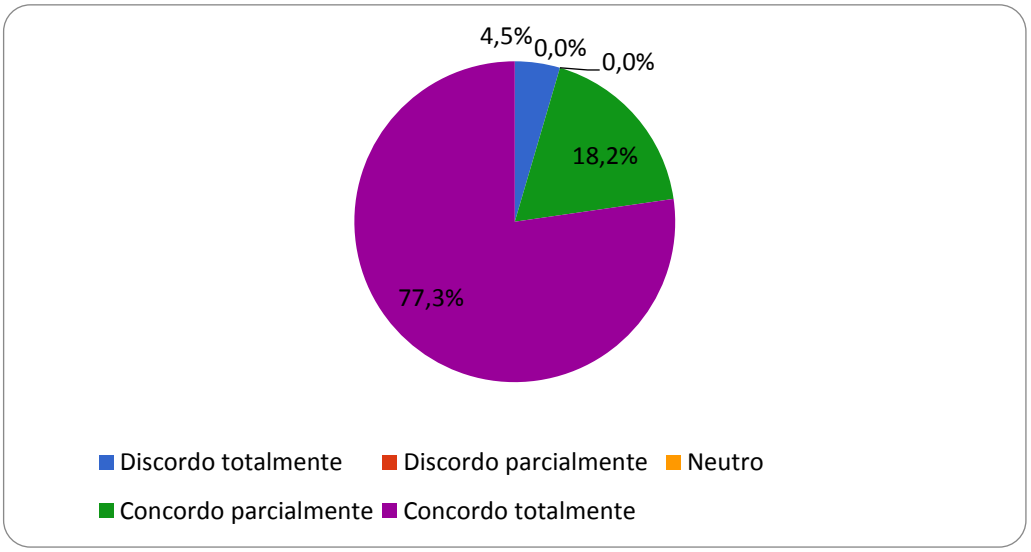
Figura 5. A Engenharia de Software é uma área importante do desenvolvimento de software



Fonte: Autoria própria (2018)

No que diz respeito à importância da Engenharia de Requisitos para atender as necessidades do cliente, um total de 77.3% dos participantes concordam totalmente com a afirmação, em seguida 18.2% concordam parcialmente e 4.5% discordam totalmente, os dados de resposta para essa pergunta do questionário pode ser visualizado na Figura 6.

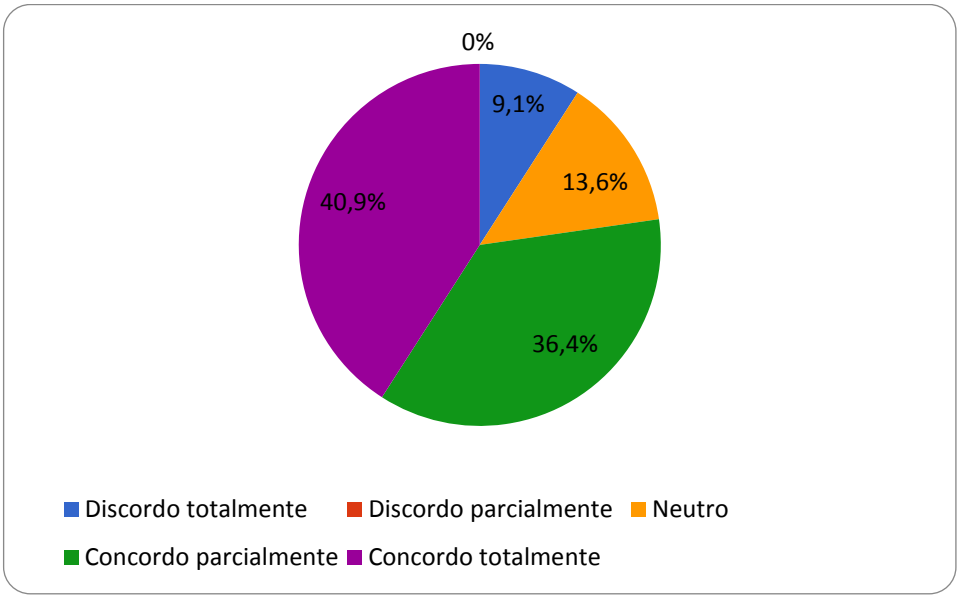
Figura 6. A Engenharia de Requisitos é importante para a entrega de produtos que atendam as necessidades do cliente



Fonte: Autoria própria (2018)

Em relação à motivação para aprender mais sobre Engenharia de Requisitos, 40.9% concordam totalmente com a afirmação, 36.4% concordam parcialmente, seguidos de acadêmicos que são neutros (13.6%), os dados podem ser vistos na Figura 7.

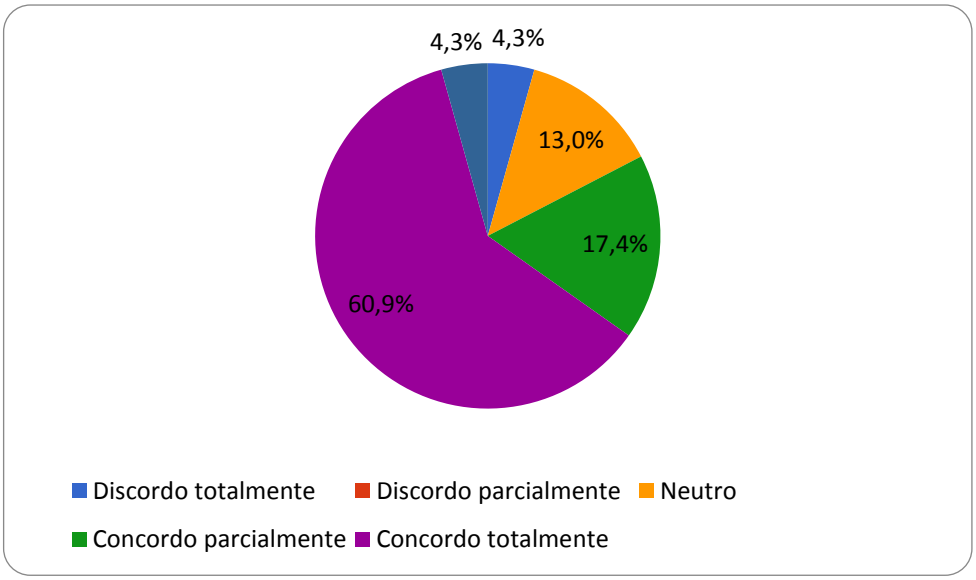
Figura 7. Estou motivado a aprender mais sobre a engenharia de requisitos



Fonte: Autoria própria (2018)

Quanto à relevância dos conteúdos ministrados na disciplina, 60.9% concordam totalmente que os conteúdos foram significativos, seguidos por 17.4% concordando parcialmente, 13% neutros e 4.3% não acham que os conteúdos foram expressivos, como visto na Figura 8.

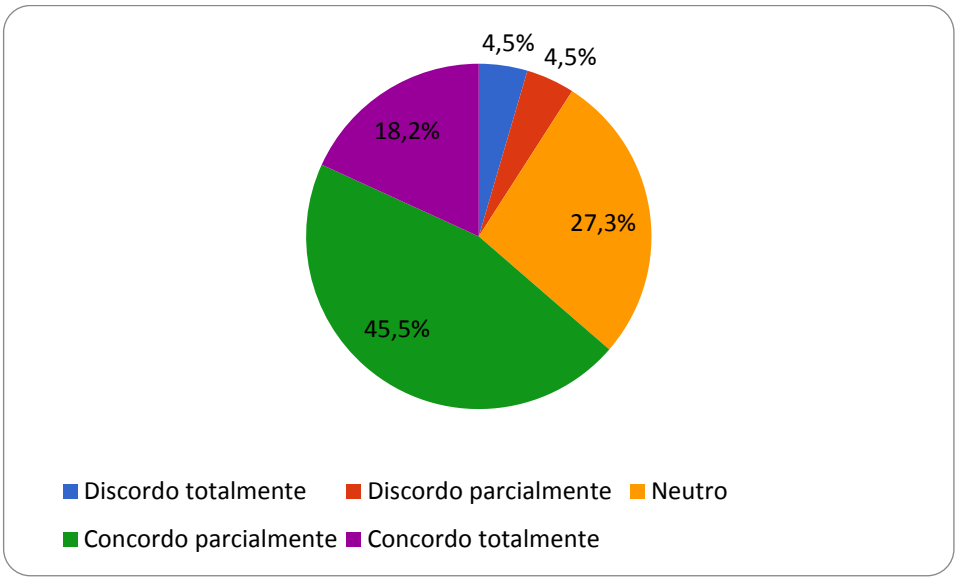
Figura 8. O conteúdo ensinado na disciplina foi relevante



Fonte: Autoria própria (2018)

No tocante à visão dos alunos em relação à eficiência dos conteúdos da disciplina para entender como a engenharia de requisitos funciona em uma organização, 18.2% concordam totalmente e 45.5% concordam parcialmente, é possível observar os resultados na Figura 9.

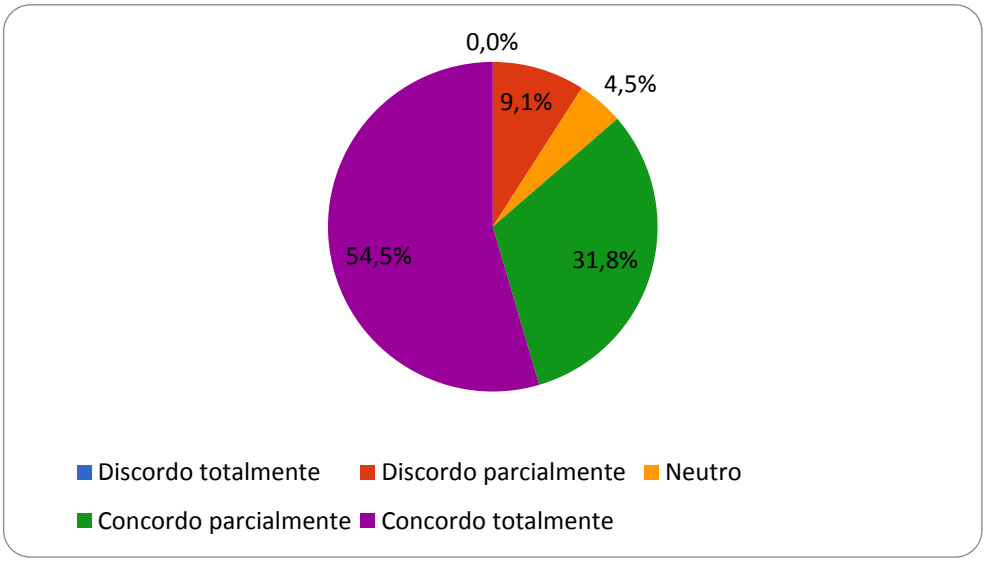
Figura 9. O Conteúdo abordado pela disciplina foi suficiente para entender como a Engenharia de Requisitos funciona em uma organização



Fonte: Autoria própria (2018)

Quanto à verificação se a abordagem utilizada na disciplina teve uma boa integração da teoria com a prática, é possível observar através da Figura 10 que 54.5% dos acadêmicos que participaram concordam totalmente com a afirmação, seguidos pelos acadêmicos que concordam parcialmente (31.8%), neutros (4.5%) e discordam parcialmente (9.1%)

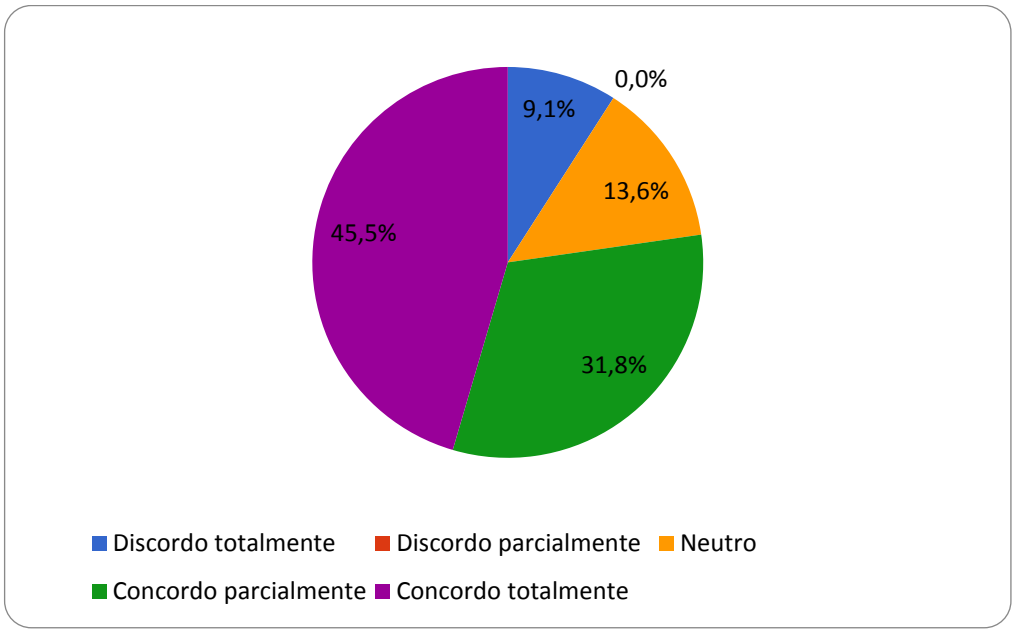
Figura 10. A abordagem escolhida para a disciplina teve uma boa integração da teoria com a prática



Fonte: Autoria própria (2018)

No que tange ao tempo utilizado para realizar as atividades práticas da disciplina, é possível observar na Figura 11 que 45.5% concordam totalmente que o tempo foi suficiente, logo em seguida vem os acadêmicos que concordam parcialmente (31.8%).

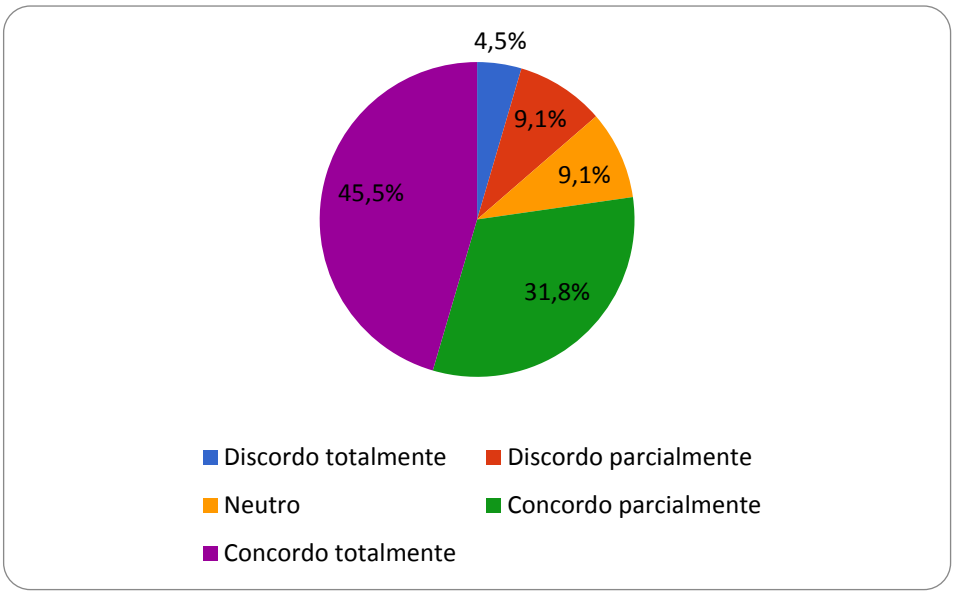
Figura 11. As dinâmicas/práticas foram realizadas em tempo adequado



Fonte: Autoria própria (2018)

Tendo em consideração o nível de complexidade das atividades práticas nas, 45.5% dos participantes concordam totalmente que o nível de complexidade foi adequado, em seguida 31.8% concordam parcialmente com a afirmação, como consta na Figura 12.

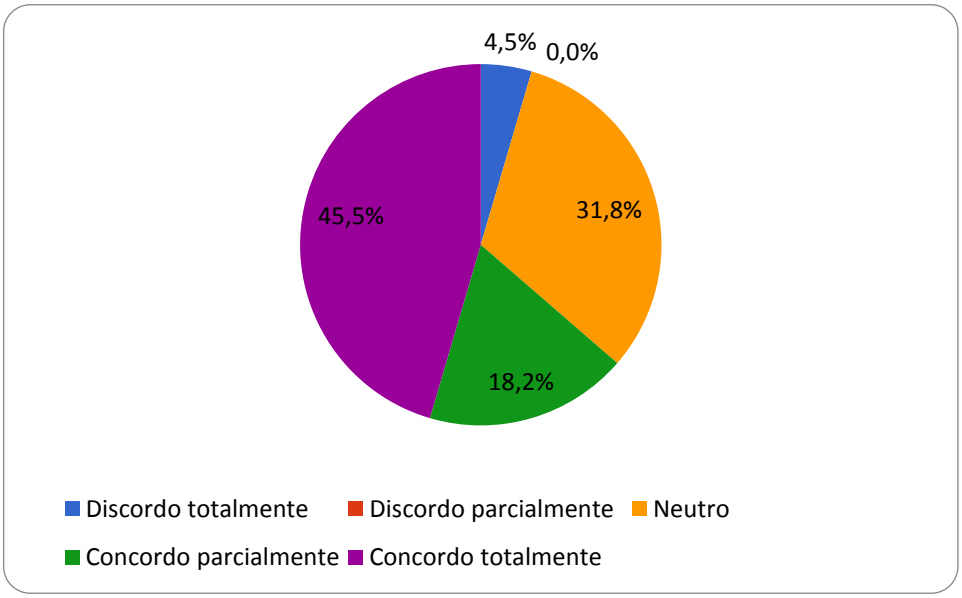
Figura 12. As dinâmicas/práticas tinham um nível de complexidade adequado



Fonte: Autoria própria (2018)

Quanto ao fato de as atividades práticas realizadas na disciplina não restringirem a criatividade dos participantes para elaborarem suas próprias soluções, 45.5% dos acadêmicos concordaram totalmente com a afirmação supracitada, seguido de 18.2% que concordaram parcialmente, segue a Figura 13 com os dados.

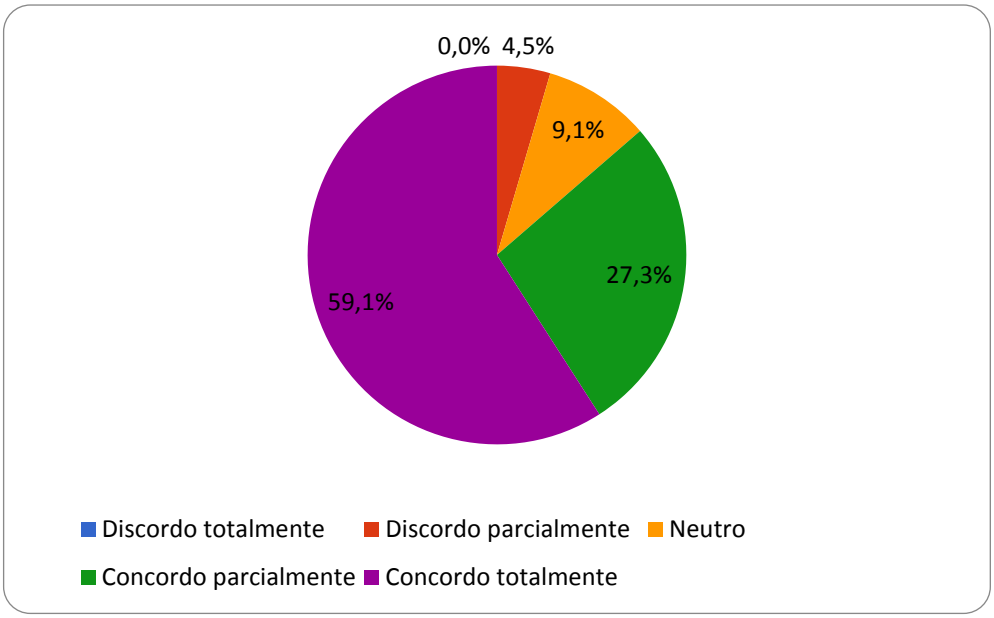
Figura 13. As dinâmicas/práticas desenvolvidas não restringiam a criatividade dos alunos para pensarem em suas próprias soluções



Fonte: Autoria própria (2018)

Na Figura 14 é possível observar os dados tendo em consideração o fato de as atividades práticas realizadas na disciplina tornarem o processo de aprendizado divertido e desafiador, onde 59.1% concordam totalmente com a afirmação e 27.3% concordaram parcialmente.

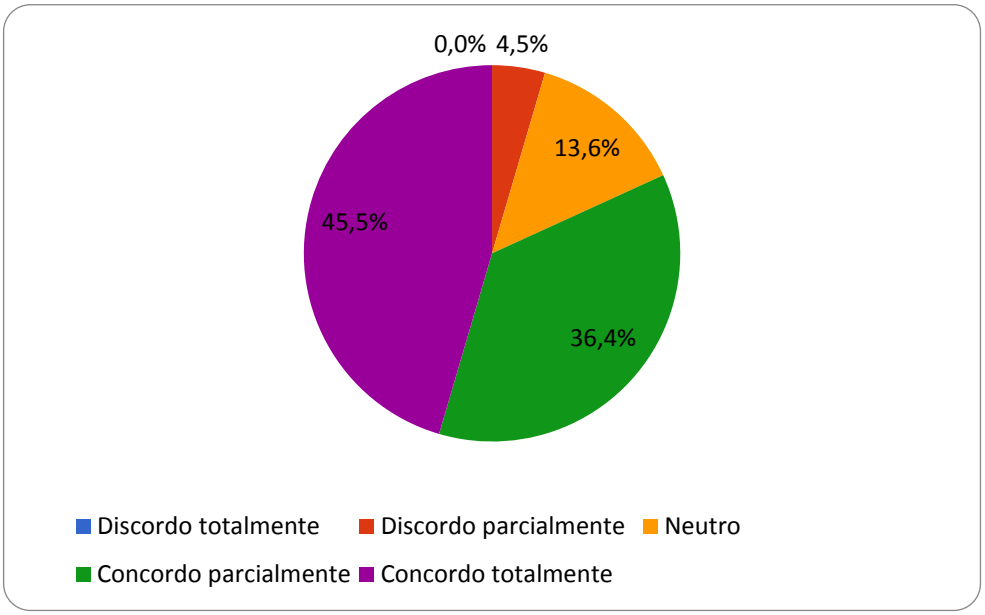
Figura 14. As dinâmicas/práticas tornaram o processo de aprendizagem divertido e desafiador



Fonte: Autoria própria (2018)

Quanto à abordagem de ensino ser um motivo para que o acadêmico se motivasse a aprender, 45.5% dos participantes concordam totalmente e 36.4% concordam parcialmente, sendo possível visualizar os resultados na Figura 15.

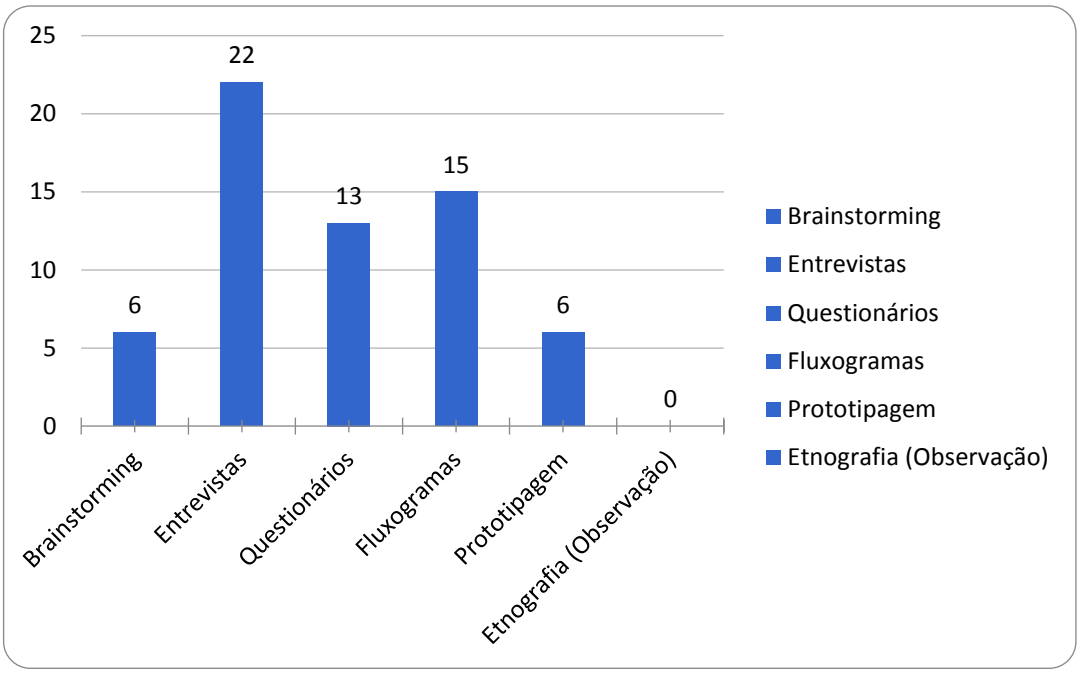
Figura 15. Ao longo da disciplina, a abordagem de ensino me manteve motivado a aprender



Fonte: Autoria própria (2018)

Quanto às técnicas utilizadas para realizar a descoberta de requisitos, a totalidade dos acadêmicos utilizaram a entrevista (22), seguido do uso de fluxogramas (15). A Figura 16 mostra as técnicas mais usadas.

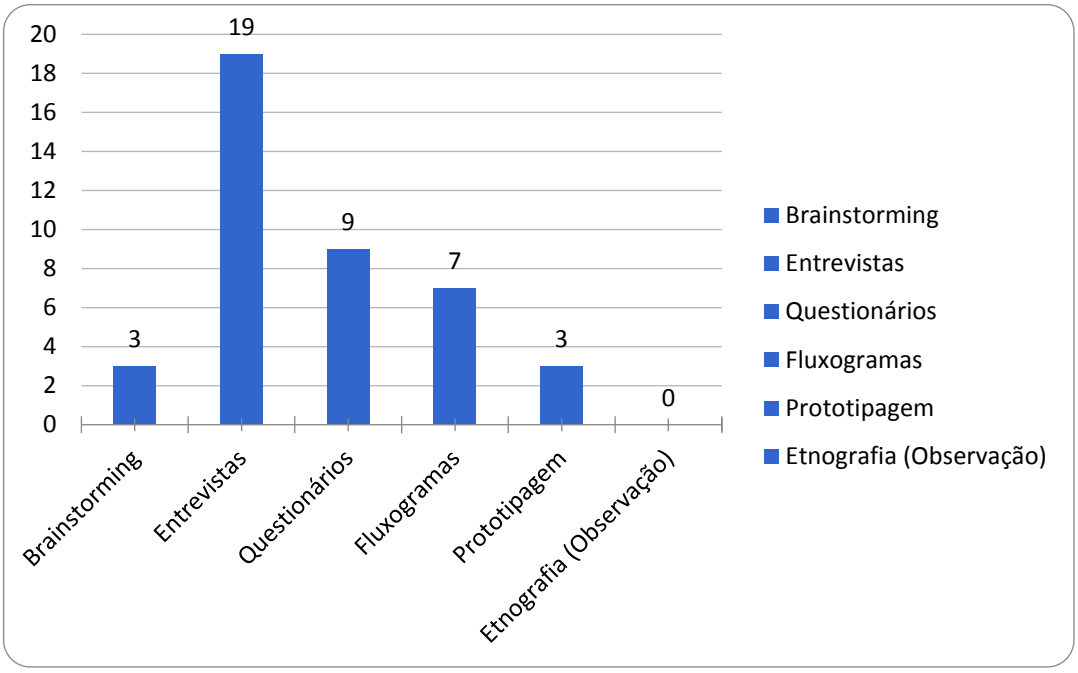
Figura 16. Quais das técnicas estudadas foram utilizadas durante a descoberta de requisitos



Fonte: Autoria própria (2018)

Levando em conta qual a técnica considerada a mais fácil para descobrir as necessidades do cliente, 19 participantes escolheram as entrevistas, 9 escolheram questionários, logo em seguida vem fluxogramas (7), e *brainstorming* e prototipagem com a mesma quantidade (3), é possível visualizar os dados na Figura 17.

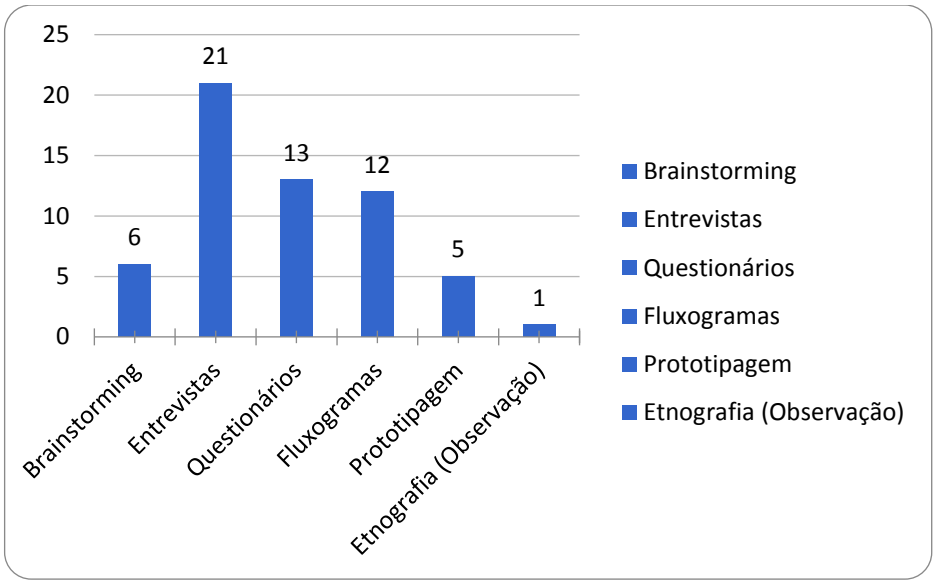
Figura 17. Quais das técnicas aplicadas você considerou mais fácil para descobrir e garantir o entendimento das necessidades do cliente



Fonte: Autoria própria (2018)

Em relação às técnicas que os participantes usariam novamente, a entrevista foi escolhida pela maioria (21), em segundo lugar vem o uso de questionários (13). A Figura 18 demonstra os dados.

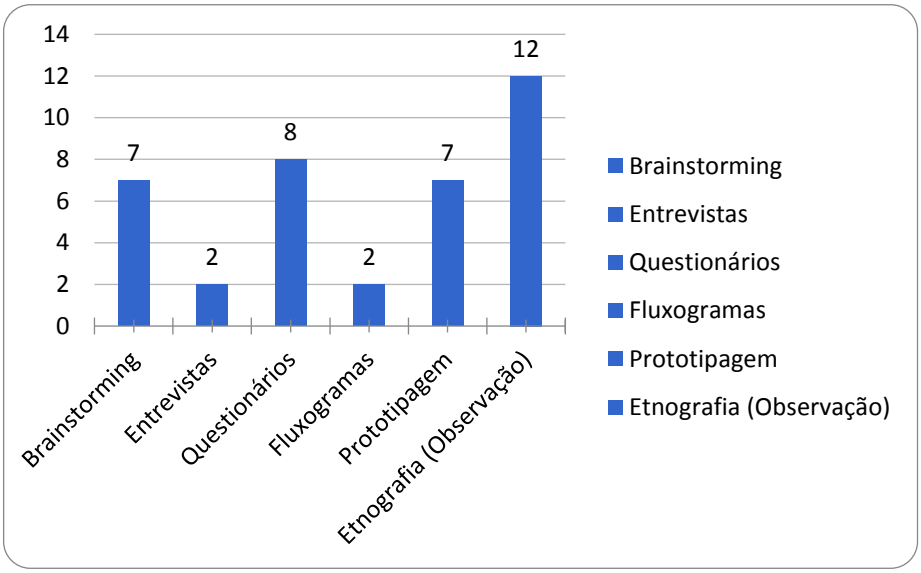
Figura 18. Quais das técnicas aplicadas para a descoberta de requisitos você usaria novamente



Fonte: Autoria própria (2018)

Quanto às técnicas não utilizadas, mas que os participantes gostariam de usar em uma próxima oportunidade, etnografia ficou em primeiro lugar com 12 votos, logo após vem o uso de questionário (8 votos), brainstorming e prototipagem (ambos com 7 votos), entrevistas e fluxogramas (ambos com 2 votos), a Figura 19 possui os dados de resposta.

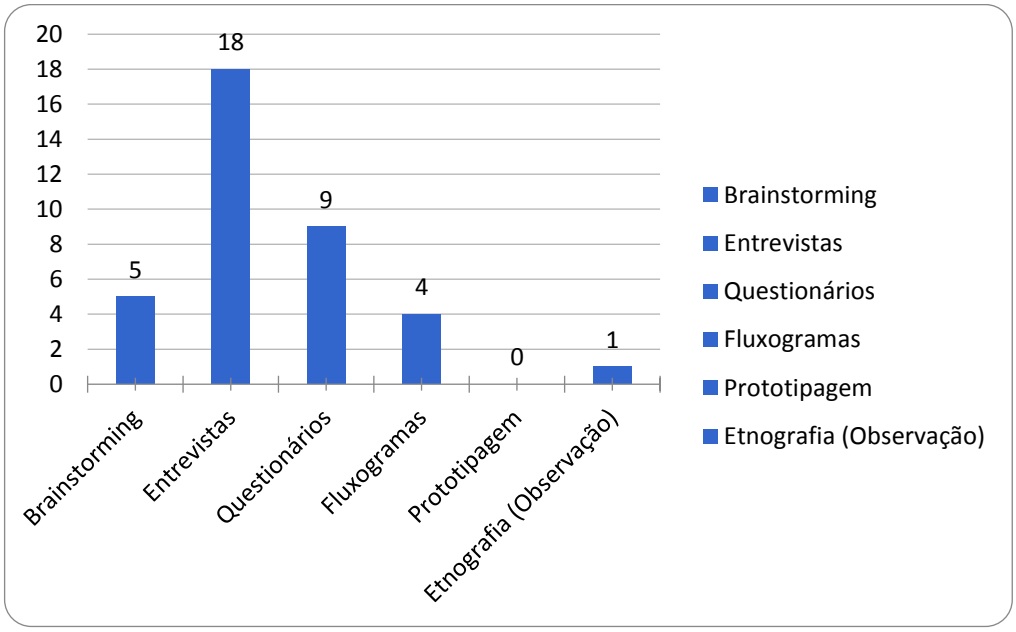
Figura 19. Quais das técnicas não aplicadas para a descoberta de requisitos você utilizaria em uma próxima oportunidade



Fonte: Autoria própria (2018)

Sobre quais técnicas utilizadas para a elicitação de requisitos foi mais fácil aprender, entrevista levou 20 votos, é possível verificar os dados na Figura 18.

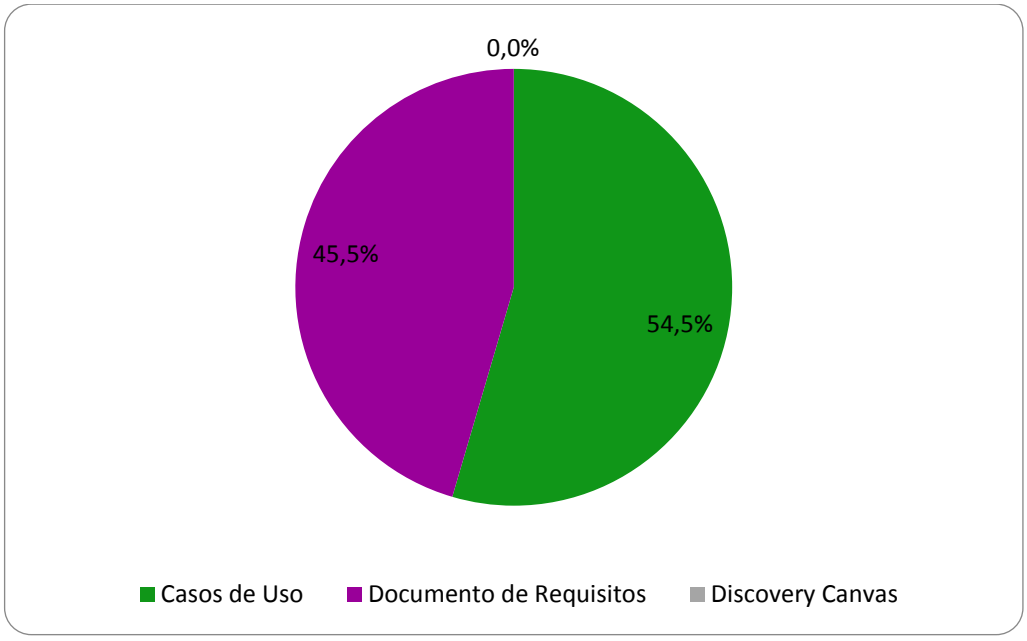
Figura 20. Quais das técnicas aplicadas para a descoberta de requisitos foi mais fácil de aprender



Fonte: Autoria própria (2018)

Quanto à qual documento atingiu os melhores resultados para a comunicação com o cliente, 54.5% dos acadêmicos escolheram o documento de casos de uso, 45.5% escolheram o documento de requisitos, como visto na Figura 21.

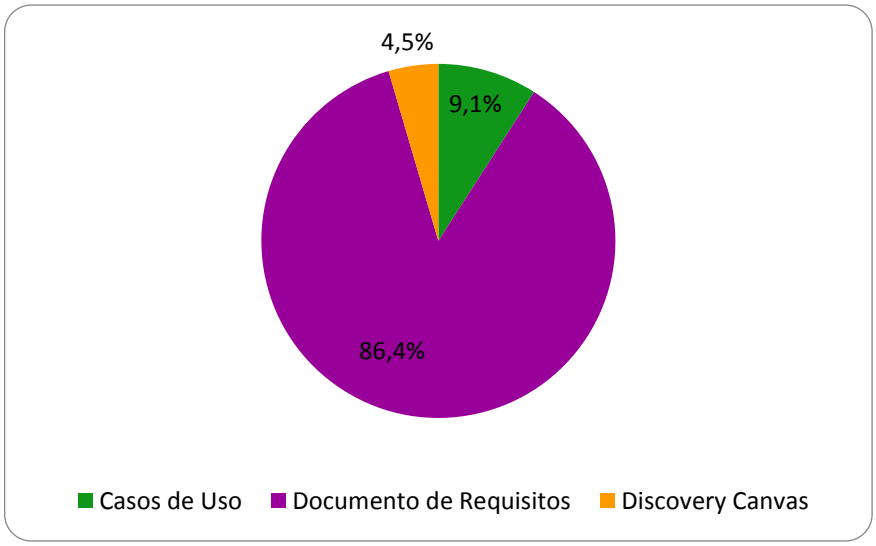
Figura 21. Quanto a documentação dos requisitos, qual o documento você considera que atingiu os melhores resultados para a comunicação com o cliente



Fonte: Autoria própria (2018)

Em relação ao documento que obteve melhores resultados da comunicação com o time de desenvolvedores, 86.4% escolheram o documento de requisitos, 9.1% escolheram o documento de casos de uso e 4.5% escolheram o *discovery canvas*, sendo possível visualizar na Figura 22.

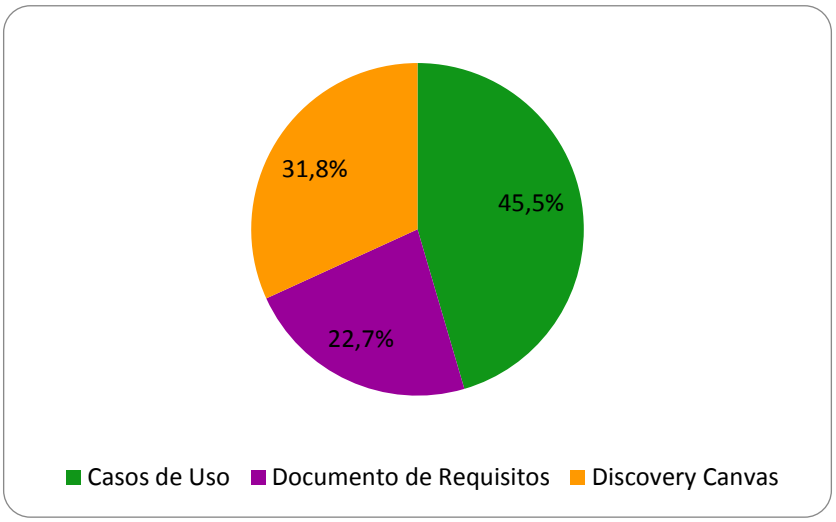
Figura 22. Quanto a documentação dos requisitos, qual o documento você considera que atingiu os melhores resultados para a comunicação com o time de desenvolvimento



Fonte: Autoria própria (2018)

Quanto à documentação dos requisitos, os participantes consideraram os casos de uso a documentação mais fácil de se aprender, com 45.5% dos votos, em segundo lugar veio o *discovery canvas*, com 31.8% e em último veio o documento de requisitos, com 22.7%, a Figura 23 ilustra os dados.

Figura 23. Quanto a documentação dos requisitos, qual a técnica você considera mais fácil de aprender?



Fonte: Autoria própria (2018)

5 DISCUSSÕES SOBRE O *SURVEY*

Neste capítulo será feita a discussão dos resultados do *survey* que foi aplicado no final da execução da disciplina, de maneira que possa ser analisado para identificar pontos negativos e positivos da abordagem de ensino utilizada.

5.1. Sobre a importância da Engenharia de Software

No que diz respeito à importância da engenharia de software como uma área do desenvolvimento de software, 90.9% dos participantes do *survey* o consideram de suma importância, concordam completamente ou parcialmente com tal afirmativa.

Já em relação a importância da Engenharia de Requisitos para atender as necessidades do cliente, 95.5% dos participantes concordaram completamente ou parcialmente com a afirmação, tal percentual mostra, na visão dos alunos, que a Engenharia de Requisitos é parte fundamental dentro da Engenharia de Software.

É evidente através dos resultados obtidos que os participantes consideram de extrema importância a área de Engenharia de Software e Engenharia de Requisitos, isso se deve a diversos fatores, dentre os principais estão as disciplinas da área, que abordam sobre a relevância para o bom desenvolvimento de software, bem como as oportunidades de estágios que os acadêmicos tiveram, onde houve a necessidade de encarar problemas reais e criar desenvolver soluções adequadas para os mesmos.

5.2. Sobre as Abordagens de Ensino

No ponto que toca ao ensino, observando a Figura 5, é possível notar que 77.3% dos participantes responderam que se sentem motivados a aprender mais sobre o conteúdo de Engenharia de Software. Já em relação à abordagem de ensino (Figura 13), 81.9% dos participantes afirmam de maneira total ou parcial que se sentiram motivados a aprender mais sobre a engenharia de requisitos devido às dinâmicas utilizadas em aula.

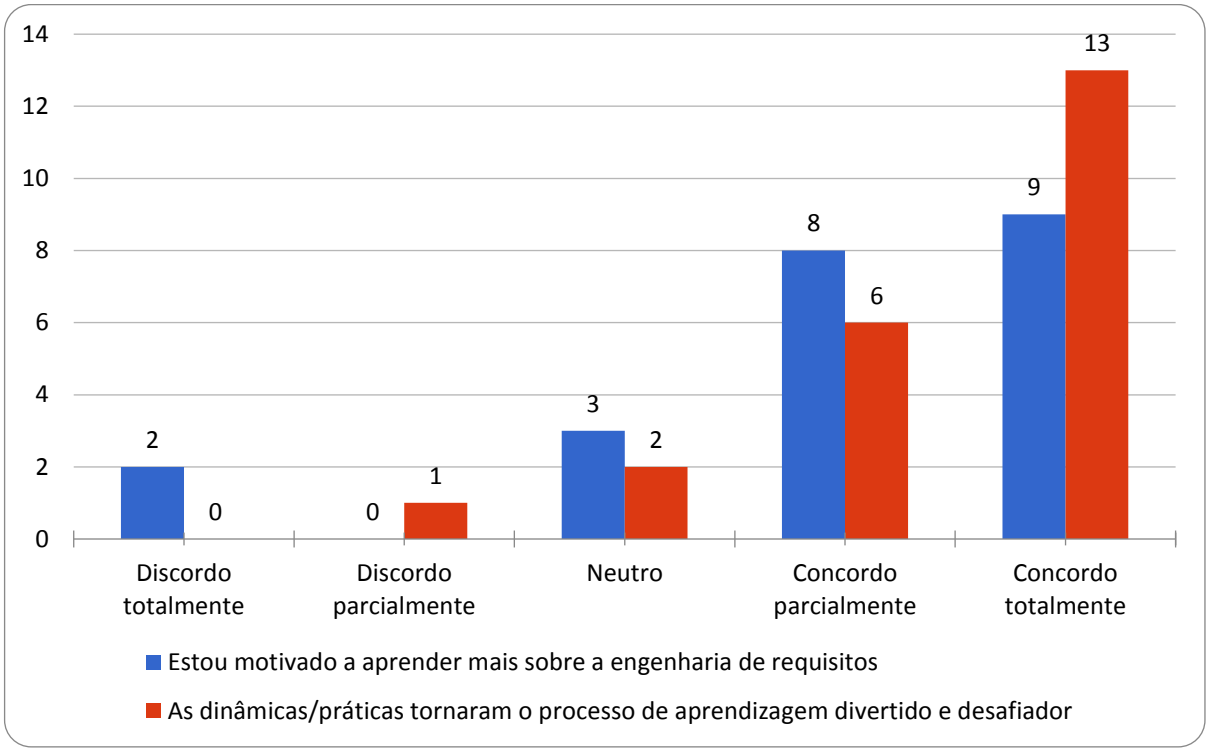
Isso evidencia que as dinâmicas adotadas ao longo do minicurso motivam os acadêmicos a aprender mais sobre o conteúdo. Juntamente com isso, outro dado importante é que 86.4% (sendo possível observar através da Figura 12) dos acadêmicos concordam totalmente ou parcialmente que as dinâmicas/práticas tornaram o processo de aprendizagem divertido e desafiador.

A partir desses resultados, foi possível evidenciar que os alunos preferem abordagens de ensino mais práticas, que os motivem a praticar o conteúdo de Engenharia de software, além do incentivo por meio de práticas, também é notado que os acadêmicos ficam com uma

permanência de um conjunto de conceitos através dessas abordagens, além de fomentar o aprendizado e participação, como visto nos resultados das estratégias de ensino destacadas por Prikladnicki *et al.* (2009). As dinâmicas, juntamente com as atividades práticas refletem no interesse dos alunos que se sentem mais motivados a aprender mais sobre a disciplina. Isso revela que a disciplina de Engenharia de Requisitos é muito mais entendida partindo de situações práticas, do que do modo de ensino convencional somente com aulas teóricas.

Ainda analisando os dados que dizem respeito a abordagem de ensino escolhida para o minicurso, 86.3% dos participantes concordaram total ou de forma parcial que a abordagem escolhida para a disciplina teve uma boa integração de teoria com a prática. Isso corrobora ainda mais sobre como abordagens mais práticas são mais eficientes no processo de aprendizagem dos alunos na disciplina de engenharia de software.

Figura 24. Relação entre motivação para o aprendizado e o processo de aprendizagem ser divertido



Fonte: Autoria própria (2018)

Voltando a análise dos dados obtidos através do *survey*, outro ponto importante é que 81.8% dos participantes do minicurso concordam totalmente ou parcialmente que o conteúdo ensinado na disciplina foi relevante. Isso é explicado pelo fato dos acadêmicos, através de dinâmicas e práticas relacionadas a Engenharia de requisitos, terem realmente entendido a real relevância do conteúdo. Pois os mesmos puseram as "mãos-na-massa" e com isso conseguiram perceber como essa fase é crucial e acarreta em consequências em todo o desenvolvimento de

um software, com os acadêmicos compreendendo melhor os conceitos de engenharia de requisitos em um número de aulas práticas significativamente menor que aulas teóricas, resultado semelhante ao obtido por Prikladnicki *et al.* (2009), na estratégia de ensino que envolve o uso de atividades lúdicas e jogos no geral.

A partir desse entendimento mais aprofundado do conteúdo, os participantes tinham mais noção de como a engenharia de requisitos é importante em um cenário organizacional. Isso ficou evidente quando os participantes foram questionados se o conteúdo abordado pela disciplina foi suficiente para entender como a Engenharia de requisitos funciona em uma organização, e a resposta foi que 63.7% dos participantes concordaram totalmente ou parcialmente com tal afirmação enquanto apenas uma totalidade de 9% discordam da afirmação de alguma forma, o percentual restante de 27.3% vai para os que foram neutros em relação a afirmação, evidenciando que por mais que o experimento tenha utilizado práticas para se assemelhar ao funcionamento de engenharia de software em uma organização, uma quantidade considerável de participantes não puderam confirmar essa semelhança, uma vez que não tiveram a experiência de trabalho na área. Apesar de não ser uma porcentagem tão alta, ainda sim é uma porcentagem satisfatória haja vista que entender como funciona um ambiente organizacional não é tarefa simples e leva algum tempo.

A principal constatação com à relação aos pontos negativos não foi sobre a abordagem em si, mas sobre o curto período que foi dado para a realização das atividades práticas e da teoria, haja vista que o experimento teve a duração de um semestre. Quando questionados sobre se as dinâmicas/práticas foram realizadas em tempo adequado, 77.3% concordaram totalmente ou parcialmente com tal constatação. Mesmo com o período do minicurso sendo curto a porcentagem de aprovação ainda é bastante satisfatória.

Outro pontos analisados pelo *survey* que dizem respeito a abordagem de ensino, nos mostram que 77.3% dos participantes concordaram totalmente ou parcialmente que as dinâmicas/práticas tinham um nível de complexidade adequado, isso é um fator importante, uma vez que a maioria dos acadêmicos já haviam tido algum contato com o assunto, principalmente por meio da disciplina de engenharia de software. E que 63.7% dos participantes do referido experimento concordaram totalmente ou parcialmente que as dinâmicas/práticas desenvolvidas não restringiam a criatividade dos alunos para pensarem em suas próprias soluções, que foi outro ponto muito positivo da abordagem adotada, pois assim os participantes puderam fazer uso de suas próprias ideias para a resolução de alguns problemas dados.

Em relação a quais técnicas foram utilizadas durante o projeto final, quais foram as mais fáceis e quais seriam usadas novamente, existe uma predominância no uso de entrevistas e

questionários (como pode ser visto nas Figuras 16, 17 e 18), isso indica que as técnicas que possuem uma relação direta com o cliente possuem uma maior preferências nos participantes, uma vez que o contato direto muitas vezes gera uma quantidade maior de requisitos que podem ser analisados, além disso também são os tipos de técnicas mais fáceis de aprender, como consta na Figura 20. No que tange a que tipo de técnica os participantes não utilizaram, mas gostariam de tentar utilizar em uma próxima oportunidade (Figura 19), os participantes demonstraram um interesse no uso da etnografia, indicando que os acadêmicos gostariam de realizar visitas nos locais do cliente para que seja possível observar de maneira melhor o funcionamento das regras de negócios que devem ser implementadas.

Os participantes indicaram que o documento de casos de uso foi o que obteve uma melhor comunicação com o cliente (Figura 21), evidenciando que o mesmo possui uma maior facilidade de ser entendido pelo cliente se comparado com o documento de requisitos. Já em relação ao time de desenvolvedores, os mesmos acharam que o documento de requisitos foi o que melhor atende suas necessidades, uma vez que esse documento possui um maior nível de detalhamento dos requisitos que devem ser desenvolvidos para o produto final. Em relação a que tipo de documentação foi mais fácil de aprender (Figura 23), os participantes indicaram que o documento de caso de uso foi o que teve uma facilidade maior de aprendizado, demostrando que seu grau de complexidade foi usualmente menor que o documento de requisitos.

5.3. Validade

Um dos principais itens no que tange a realização de um experimento é saber o quão válido são os seus resultados. A ordem de prioridade dos tipos de validação é realizada através dos objetivos que o experimento possui. Travassos (2002) afirma que para experimentos aplicados, a ordem de importância das validades é: interna, externa, construção e conclusão.

5.3.1. Validade Interna

A validade interna é utilizada para definir se a relação entre o tratamento em si e o resultado obtido é causal e não resultado de um fator que não foi medido ou que não pode ser controlado. Neste experimento, algumas ameaças à validade interna podem ser descritas como o efeito da maturação ou mesmo da instrumentação

No que tange a maturação, a mesma pode ter ocorrido por meio de estudos e atividades de aprendizagem que não foram planejadas no escopo do experimento, por meio de iniciativa própria do participante em buscar por mais conteúdo. Para amenizar esse fator, os alunos tiveram a instrução de não buscar por questões que não estivessem dentro das atividades

planejadas para o experimento. No entanto não existe uma maneira de confirmar que as instruções foram devidamente seguidas.

5.3.2. Validade Externa

Por meio da avaliação externa é definida as condições que limitam a capacidade de generalizar os resultados obtidos para outras populações em outros contextos. O contexto do experimento foi acadêmico, sendo assim os resultados generalizados devem ser limitados pelo contexto acadêmico.

Outro fator limitante nesse quesito é que o experimento foi realizado com uma amostra pequena de participantes e, até a presente data, não foi replicado em outras populações, grupos ou universidades.

A abordagem de ensino utilizada tentou inserir o aluno na aplicação prática dos conceitos aprendidos, no entanto as atividades práticas eram baseadas em cenários reais simplificados, sendo assim não existem garantias que as habilidades obtidas serão refletidas em um ambiente real de desenvolvimento.

5.3.3. Validade de Construção

A validade de construção considera a relação entre a teoria e observação, ou seja, verifica se o tratamento reflete a causa de maneira satisfatória e o resultado reflete o efeito de maneira satisfatória.

Durante a avaliação da validade de construção os fatores humanos e os aspectos relevantes ao projeto do experimento devem ser relevados. Desta forma, não é possível afirmar que os participantes aprenderam a utilizar estes novos conhecimentos em um contexto diferente do utilizado no experimento.

5.3.4. Validade de Conclusão

A validade de conclusão é a capacidade de chegar a uma conclusão correta a respeito da relação entre o tratamento e o resultado do experimento. Durante a avaliação desta qualidade é preciso levar em conta conceitos como: a confiabilidade das medidas; a confiabilidade da implementação dos tratamentos e a escolha do teste estatístico.

Mesmo realizando o agrupamento de todos os dados, a amostra permanece pequena, impossibilitando a demonstração de qualquer relação estatística válida.

6 CONSIDERAÇÕES FINAIS

Esta pesquisa teve como objetivo analisar se esse tipo de abordagem criada era benéfico no processo de ensino/aprendizagem do conteúdo de Engenharia de Requisitos, que é parte fundamental da Engenharia de Software. É fato que a área de engenharia de software é muitas vezes deixada de lado pelos alunos que cursam Ciência da Computação. Isso ocorre por ser uma disciplina muito teórica e muitas vezes isso desmotiva os acadêmicos.

O principal problema levantado ao longo da pesquisa é a falta de profissionais qualificados para atuar na indústria do desenvolvimento de software, mais especificamente como engenheiros de requisitos. Essa falta de qualificação que por vezes está atrelada a forma de como os tópicos de engenharia de requisitos são ensinados ao longo da graduação. Com base nisso, a abordagem criada na pesquisa teve o intuito de diminuir essa lacuna entre a academia e a indústria, fazendo uso de leitura de artigos, jogos e simuladores, discussão de casos práticos e outras atividades que centradas no aluno.

Quanto aos resultados do estudo, forneceram um indicador de que a abordagem alcança um efeito melhor no aluno sobre a adequação do curso se comparado às aulas de expositivas tradicionais como o principal método instrucional.

Através dos dados obtidos pelo *survey* foi mostrado que a abordagem adotada no minicurso tende a ser efetiva. Foi abordado, além da teoria, aspectos práticos da disciplina possibilitando aos participantes um ótimo entendimento e um convívio maior com a área de Engenharia de Requisitos.

Embora os resultados da abordagem sejam significativos, mais pesquisas são necessárias para avaliar o ganho real de aprendizagem que foi alcançado pela proposta em comparação com outros modelos que podem ser utilizados para realizar o ensino de Engenharia de Requisitos.

REFERÊNCIAS BIBLIOGRÁFICAS

ACM/IEEE. **Computer Science Curricula 2013:** Curriculum Guidelines for Undergraduate Degree Programs in Computer Science. New York: ACM, 2013.

BECKMAN, K.; COULTER, N.; KHAJENOURI, S.; MEAD, N. **Collaborations:** Closing the industry–academia gap. IEEE Software 14 (6), pp. 49–57, 1997.

FURTADO, J. C. C.; OLIVEIRA, S. R. B. "**Evaluating Students' Perception of Their Learning in a Student-centered Software Engineering Course -** A Experimental Study." ICSOFT, 2018.

LETHBRIDGE, T. **What Knowledge Is Important to a Software Professional?**. IEEE Computer Society, Ottawa, May 2000. 44-50.

MALIK, Bushra; ZAFAR, Saad. **A Systematic Mapping Study on Software Engineering Education**. 2013.

MEMON, R.N.; AHMAD, R.B.; SALIM, S.S. **Problems in requirements engineering education:** a *survey*. 2010. In *Proceedings of the 8th International Conference on Frontiers of Information Technology* (FIT '10). ACM, New York, NY, USA, Article 5, 6 pages.

NAUR, Peter; RANDEL, Brian. **Software Engineering:** Report of a Conference Sponsored by the NATO Science Committee, Garmisch, Germany, 7-11 Oct. 1968, Brussels, Scientific Affairs Division, NATO. 1969.

NUNES, Daltro. **Competências do Engenheiro de Software**. Revista da SBC: Engenharia de Software - Qual é o impacto da ES no mercado de Computação e na sociedade como um todo? 1ª. ed. Porto Alegre: Sociedade Brasileira de Computação, Cap. 2, p. 16-20, 2015.

NUNES, D. J., YAMAGUTI, M. H. E NUNES, I. **Refinement of student competences of the software engineering course**. IX Forum on Education in Software Engineering, p. 143-146, 2016.

NUSEIBEH, Bashar; EASTERBROOK, Steve. **Requirements engineering:** a roadmap. 2000. In *Proceedings of the Conference on The Future of Software Engineering* (ICSE '00). ACM, New York, NY, USA, 35-46.

PMI – Project Management Institute. **A guide to the project management body of knowledge**. 4. Ed., 2008.

PORTELA, C. S., VASCONCELOS, A. M. L. E OLIVEIRA, S. R. B. **How to Develop Competencies and Abilities Professional in Software Engineering in Undergraduate Students?**, in Int'l Conf. Frontiers in Education: CS and CE - FECS'15, pp. 91-94, 2015.

PORTELA, Carlos. **Um modelo iterativo para o ensino de engenharia de software baseado em abordagens focadas no aluno e práticas de capacitação da indústria**. 2017. Tese (Doutorado em Ciência da Computação) — Centro de Informática, Universidade Federal de Pernambuco, Recife.

PRESSMAN, Roger. **Engenharia de Software:** Uma abordagem profissional. 7. ed. Brasil. AMGH Editora Ltda, 2011.

PRIKLADNICKI, Rafael; ALBUQUERQUE, Adriano; WANGENHEIM, C. G.; CABRAL, Reinaldo. **Ensino de Engenharia de Software:** Desafios, Estratégias de Ensino e Lições Aprendidas. 2009.

PUCRS. **Bacharelado em Sistemas de Informação, Grade curricular**, disponível online em http://www.pucrs.br/politecnica/curso/sistemas-de-informacao/#informacoes-academicas, acesso em julho/18.

SANTOS, Ronnie; MAGALHAES, Cleyton V. C.; CORREIA-NETO, Jorge; SOUZA, Polliana; ELLEN; VILAR, Guilherme. **Ferramentas, métodos e experiências no ensino de Engenharia de Software:** Um mapeamento sistemático. 2014. 10.5753/cbie.sbie.2014.544.

SEI. **CMMI® for Development**. Version 1.3. CMU/SEI-2010-TR-033 ESC-TR-2010-033. Software Engineering Institute-SEI, Carnegie Mellon University: 561, 2010.

SOFTEX. **MR-MPS-BR -** Modelo de Referência MPS para Sofware – Guia Geral. 2016. Disponível em: www.softex.br

SOMMERVILLE, Ian. **Engenharia de Software**. 9. ed. Brasil. Pearson Education do Brasil, 2011.

TRAVASSOS, G. H. et al. **Introdução à Engenharia de Software Experimental**, Relatório Técnico RT-ES-590/02 do Programa de Engenharia de Sistemas e Computação, COPPE/UFRJ, Rio de Janeiro, Brasil, 2002.

ZOWGHI, Didar; CLELAND-HUANG, Jane. **Transforming the Requirements Engineering Classroom Experience**. 2008. *16th IEEE International Requirements Engineering Conference*, Catalunya, 2008, pp. 297-297.

ZOWGHI, Didar; PARYANI, Suresh. **Teaching requirements engineering through role playing:** lessons learnt. *Proceedings. 11th IEEE International Requirements Engineering Conference, 2003.*, Monterey Bay, CA, USA, 2003, pp. 233-241.

WANGENHEIM, C. G.; SILVA, D. A. **Qual conhecimento de engenharia de software é importante para um profissional de software?** In: FÓRUM DE EDUCAÇÃO EM ENGENHARIA DE SOFTWARE, 2009, Fortaleza. Anais. Fortaleza: UFC.

APÊNDICE I – *SURVEY* (QUESTÕES OBJETIVAS)

Responda as afirmações abaixo:

Q1. A Engenharia de Software uma é área importante do desenvolvimento de software:

- Discordo fortemente
- Discordo
- Neutro
- Concordo
- Concordo fortemente

Q2. A Engenharia de Requisitos é importante para a a entrega de produtos que atendam as necessidades do cliente:

- Discordo fortemente
- Discordo
- Neutro
- Concordo
- Concordo fortemente

Q3. Estou motivado a aprender mais sobre a Engenharia de Requisitos:

- Discordo fortemente
- Discordo
- Neutro
- Concordo
- Concordo fortemente

Q4. O conteúdo ensinado na disciplina foi relevante:

- Discordo fortemente
- Discordo
- Neutro
- Concordo
- Concordo fortemente

Q5. O conteúdo abordado pela disciplina foi suficiente para entender como a Engenharia de Requisitos funciona em uma organização:

- Discordo fortemente
- Discordo
- Neutro
- Concordo
- Concordo fortemente

Q6. A abordagem escolhida para a disciplina teve uma boa integração da teoria com a prática:

- Discordo fortemente

- Discordo
- Neutro
- Concordo
- Concordo fortemente

Q7. As dinâmicas/práticas foram realizadas em tempo adequado:

- Discordo fortemente
- Discordo
- Neutro
- Concordo
- Concordo fortemente

Q8. As dinâmicas/práticas tinham um nível de complexidade adequado:

- Discordo fortemente
- Discordo
- Neutro
- Concordo
- Concordo fortemente

Q9. As dinâmicas/práticas desenvolvidas não restringiam a criatividade dos alunos para pensarem em suas próprias soluções:

- Discordo fortemente
- Discordo
- Neutro
- Concordo
- Concordo fortemente

Q10. As dinâmicas/práticas tornaram o processo de aprendizagem divertido e desafiador:

- Discordo fortemente
- Discordo
- Neutro
- Concordo
- Concordo fortemente

Q11. Ao longo da disciplina, a abordagem de ensino me manteve motivado a aprender:

- Discordo fortemente
- Discordo
- Neutro
- Concordo
- Concordo fortemente

APÊNDICE II – *SURVEY* (QUESTÕES DE MÚLTIPLA ESCOLHA)

Q12. Quais das técnicas estudadas foram utilizadas durante a descoberta de requisitos?

- ☐ Brainstorming
- ☐ Entrevistas
- ☐ Questionários
- ☐ Fluxogramas
- ☐ Prototipagem
- ☐ Etnografia (Observação)

Q13. Quais das técnicas aplicadas você considerou mais fácil para descobrir e garantir o entendimento das necessidades do cliente?

- ☐ Brainstorming
- ☐ Entrevistas
- ☐ Questionários
- ☐ Fluxogramas
- ☐ Prototipagem
- ☐ Etnografia (Observação)

Q14. Quais das técnicas aplicadas para a descoberta de requisitos você usaria novamente?

- ☐ Brainstorming
- ☐ Entrevistas
- ☐ Questionários
- ☐ Fluxogramas
- ☐ Prototipagem
- ☐ Etnografia (Observação)

Q15. Quais das técnicas não aplicadas para a descoberta de requisitos você utilizaria em uma próxima oportunidade?

- ☐ Brainstorming
- ☐ Entrevistas
- ☐ Questionários
- ☐ Fluxogramas
- ☐ Prototipagem
- ☐ Etnografia (Observação)

Q16. Quais das técnicas aplicadas para a descoberta de requisitos foi mais fácil de aprender?

- ☐ Brainstorming
- ☐ Entrevistas
- ☐ Questionários
- ☐ Fluxogramas
- ☐ Prototipagem
- ☐ Etnografia (Observação)

Q17. Quanto a documentação dos requisitos, qual o documento você considera que atingiu os melhores resultados para a comunicação com o cliente?

- Casos de uso
- Documento de requisitos
- Discovery canvas

Q18. Quanto a documentação dos requisitos, qual o documento você considera que atingiu os melhores resultados para a comunicação com o time de desenvolvimento?

- Casos de uso
- Documento de requisitos
- Discovery canvas

Q19. Quanto a documentação dos requisitos, qual a técnica você considera mais fácil de aprender?

- Casos de uso
- Documento de requisitos
- Discovery canvas

APÊNDICE III – CONJUNTO DE DADOS DA ABORDAGEM DE ENSINO

Para as questões Q1 à Q11, de acordo com a escala *likert*:

1. Discordo fortemente
2. Discordo
3. Neutro
4. Concordo
5. Concordo Fortemente

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11
5	5	4	4	3	5	5	5	5	5	4
5	5	4	5	4	5	5	5	4	5	4
5	4	4	5	3	4	4	4	5	4	3
4	5	3	3	2	2	1	2	3	3	3
4	5	5	5	4	5	5	5	4	5	5
5	5	5	5	4	5	5	5	5	5	5
5	5	5	5	4	5	4	4	5	5	4
4	4	4	5	4	3	4	5	5	5	4
5	5	5	4	4	5	5	4	5	5	5
3	4	3	3	3	4	3	2	3	3	4
5	5	5	5	5	5	5	5	5	5	5
5	5	4	4	3	4	3	4	3	5	5
1	1	1	1	1	2	1	1	1	2	2
5	5	5	5	4	5	5	4	4	4	4
5	5	5	5	4	4	5	4	3	4	4
5	5	5	5	5	5	4	5	5	5	5
4	4	1	3	3	4	3	3	4	4	5
5	5	4	5	4	5	5	5	5	5	5
5	5	5	5	5	5	5	5	5	5	5
5	5	4	5	5	5	4	5	3	5	5
5	5	4	4	4	4	4	4	3	4	4
5	5	3	5	3	4	4	3	3	4	3

Q12	Q13	Q14	Q15	Q16
Brainstorming, Entrevistas, Questionários, Fluxogramas	Entrevistas, Questionários, Fluxogramas	Brainstorming, Prototipagem, Entrevistas, Questionários, Fluxogramas	Etnografia (Observação)	Brainstorming, Entrevistas, Questionários, Fluxogramas
Entrevistas	Entrevistas	Entrevistas	Etnografia (Observação)	Entrevistas, Questionários
Brainstorming, Entrevistas, Fluxogramas	Brainstorming	Brainstorming, Fluxogramas	Prototipagem, Questionários	Brainstorming, Entrevistas, Fluxogramas
Prototipagem, Entrevistas, Questionários	Entrevistas	Entrevistas, Questionários	Brainstorming, Etnografia (Observação)	Entrevistas
Prototipagem, Entrevistas, Fluxogramas	Entrevistas, Fluxogramas	Entrevistas, Fluxogramas	Questionários, Etnografia (Observação)	Entrevistas

Entrevistas, Fluxogramas	Entrevistas	Entrevistas, Fluxogramas	Brainstorming, Prototipagem, Questionários, Etnografia (Observação)	Entrevistas
Brainstorming, Entrevistas, Questionários, Fluxogramas	Brainstorming, Entrevistas, Questionários, Fluxogramas	Brainstorming, Entrevistas, Questionários, Fluxogramas	Prototipagem, Etnografia (Observação)	Brainstorming
Entrevistas, Questionários, Fluxogramas	Entrevistas, Questionários	Entrevistas, Questionários, Fluxogramas	Entrevistas, Questionários	Entrevistas
Brainstorming, Entrevistas, Questionários, Fluxogramas	Fluxogramas	Brainstorming, Prototipagem, Entrevistas, Questionários, Fluxogramas, Etnografia (Observação)	Prototipagem, Etnografia (Observação)	Entrevistas, Fluxogramas
Prototipagem, Entrevistas, Questionários	Prototipagem, Entrevistas	Prototipagem, Entrevistas, Questionários	Brainstorming, Prototipagem, Entrevistas, Questionários	Entrevistas, Questionários
Entrevistas, Questionários, Fluxogramas	Entrevistas, Questionários	Entrevistas, Questionários, Fluxogramas	Fluxogramas	Questionários
Entrevistas, Questionários, Fluxogramas	Entrevistas, Questionários	Brainstorming, Entrevistas, Questionários, Fluxogramas	Etnografia (Observação)	Brainstorming, Entrevistas, Questionários
Entrevistas, Questionários, Fluxogramas	Entrevistas, Questionários	Entrevistas, Questionários	Brainstorming	Entrevistas, Questionários
Entrevistas, Fluxogramas	Entrevistas, Fluxogramas	Entrevistas, Fluxogramas	Questionários, Etnografia (Observação)	Entrevistas, Etnografia (Observação)
Brainstorming, Entrevistas, Fluxogramas	Brainstorming, Fluxogramas	Brainstorming, Entrevistas, Fluxogramas	Questionários	Brainstorming, Entrevistas, Questionários
Entrevistas, Questionários, Fluxogramas	Entrevistas, Questionários	Entrevistas, Questionários	Prototipagem	Entrevistas
Prototipagem, Entrevistas, Questionários	Prototipagem, Entrevistas, Questionários	Prototipagem, Entrevistas, Questionários	Brainstorming, Etnografia (Observação)	Entrevistas, Questionários
Brainstorming, Entrevistas, Questionários	Entrevistas, Questionários	Entrevistas, Questionários	Prototipagem, Etnografia (Observação)	Questionários
Entrevistas	Entrevistas	Entrevistas	Questionários	Entrevistas
Entrevistas, Questionários, Fluxogramas	Entrevistas, Fluxogramas	Entrevistas, Fluxogramas	Etnografia (Observação)	Fluxogramas
Prototipagem, Entrevistas, Fluxogramas	Prototipagem, Entrevistas	Entrevistas, Questionários	Brainstorming, Fluxogramas	Entrevistas
Prototipagem, Entrevistas	Entrevistas	Prototipagem, Entrevistas	Brainstorming	Entrevistas

Q17	Q18	Q19
Casos de Uso	Documento de Requisitos	Casos de Uso
Documento de Requisitos	Documento de Requisitos	Casos de Uso
Documento de Requisitos	Documento de Requisitos	Documento de Requisitos

Documento de Requisitos	Documento de Requisitos	Casos de Uso
Documento de Requisitos	Documento de Requisitos	Discovery Canvas
Casos de Uso	Documento de Requisitos	Discovery Canvas
Casos de Uso	Documento de Requisitos	Casos de Uso
Casos de Uso	Documento de Requisitos	Casos de Uso
Casos de Uso	Documento de Requisitos	Casos de Uso
Documento de Requisitos	Documento de Requisitos	Casos de Uso
Casos de Uso	Documento de Requisitos	Documento de Requisitos
Documento de Requisitos	Documento de Requisitos	Documento de Requisitos
Documento de Requisitos	Documento de Requisitos	Documento de Requisitos
Casos de Uso	Documento de Requisitos	Discovery Canvas
Documento de Requisitos	Documento de Requisitos	Documento de Requisitos
Casos de Uso	Documento de Requisitos	Discovery Canvas
Casos de Uso	Documento de Requisitos	Casos de Uso
Casos de Uso	Documento de Requisitos	Casos de Uso
Documento de Requisitos	Casos de Uso	Discovery Canvas
Casos de Uso	Discovery Canvas	Discovery Canvas
Casos de Uso	Documento de Requisitos	Discovery Canvas
Documento de Requisitos	Casos de Uso	Casos de Uso