PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL
FACULDADE DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

UM MÉTODO DE APRENDIZAGEM BASEADA EM DESAFIOS: UM ESTUDO DE CASO EM AMBIENTES DE DESENVOLVIMENTO DE APLICATIVOS

ALAN RICARDO DOS SANTOS

Tese apresentada como requisito parcial à obtenção do grau de Doutor em Ciência da Computação na Pontifícia Universidade Católica do Rio Grande do Sul.

Orientador: Paulo Henrique Lemelle Fernandes

Porto Alegre
2016

Ficha Catalográfica

S237m Santos, Alan Ricardo dos

Um Método de Aprendizagem Baseada em Desafios : Um Estudo de Caso em Ambientes de Desenvolvimento de Aplicativos / Alan Ricardo dos Santos . – 2016.

179 f.

Tese (Doutorado) – Programa de Pós-Graduação em Ciência da Computação, PUCRS.

Orientador: Prof. Dr. Paulo Henrique Lemelle Fernandes.

1. Abordagem baseada em desafios. 2. Desenvolvimento de software. 3. Dispositivos móveis. 4. Desenvolvimento ágil. I. Fernandes, Paulo Henrique Lemelle. II. Título.

Elaborada pelo Sistema de Geração Automática de Ficha Catalográfica da PUCRS com os dados fornecidos pelo(a) autor(a).

Pontifícia Universidade Católica do Rio Grande do Sul
FACULDADE DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

TERMO DE APRESENTAÇÃO DE TESE DE DOUTORADO

Tese intitulada "Um Método de Aprendizagem Baseada em Desafios: Um Estudo de Caso em Ambientes de Desenvolvimento de Aplicativos" apresentada por Alan Ricardo dos Santos como parte dos requisitos para obtenção do grau de Doutor em Ciência da Computação, aprovada em 8 de agosto de 2016 pela Comissão Examinadora:

Dr. Paulo Henrique Lemelle Fernandes — PPGCC/PUCRS
Orientador

Dr. Rafael Prikladnicki — PPGCC/PUCRS

Dr. Alfredo Goldman vel Lejbman — Universidade de São Paulo - USP

Dr. Alberto Avritzer — Sonatype

Homologada em 06/10/2016, conforme Ata No. 020 pela Comissão Coordenadora.

Prof. Dr. Luiz Gustavo Leão Fernandes
Coordenador.

Campus Central
Av. Ipiranga, 6681 - P. 32 - sala 507 - CEP: 90619-900
Fone: (51) 3320-3611 - Fax (51) 3320-3621
E-mail: ppgcc@pucrs.br
www.pucrs.br/facin/pos

"Not having heard something is not as good as having heard it;
having heard it is not as good as having seen it;
having seen it is not as good as knowing it;
knowing it is not as good as putting it into practice."
Xunzi

AGRADECIMENTOS

Agradeço o orientador Dr. Paulo Fernandes pelo incentivo, apoio, suporte, dedicação e aprendizado durante o doutorado e também em vários momentos difíceis. Sou também grato pelo suporte recebido do Dr. Afonso Sales pela compreensão e apoio durante todas as etapas do meu doutorado e também em vários desafios, seu suporte foi essencial para eu chegar até aqui. Meu sincero obrigado por aceitar o convite para compor a banca de defesa de tese aos professores e pesquisadores Dr. Rafael Prikladnicki, Dr. Alfredo Goldman e Dr. Alberto Avritzer.

Aos funcionários do Programa de Pós Graduação em Ciência da Computação da PUCRS, particularmente aos funcionários Régis Escobal e Diego Cintrão, agradeço o apoio e a paciência durante toda a duração do curso. Agradeço também a toda a Faculdade de informática da PUCRS, pelo incentivo e apoio de diversos funcionários, colegas e professores durante o curso, pela amizade e pelo carinho. Agradeço também a comunidade Tecnopuc a qual me permitiu contato com diferentes profissionais e pesquisadores proporcionando uma experiência fantástica através da troca de conhecimento e aprendizado constante.

Agradeço aos amigos, sou especialmente grato aos amigos Mauricio Cristal, Danilo Santana e Mark Nichols pelo importante suporte para a realização deste trabalho. A minha amiga Dra. Josiane Kroll, por todo o apoio em cada etapa deste trabalho. Agradeço também os amigos Silvio Gomez, Joaquim Assunção, Bernardo Estacio e Dr. Odorico Mendizabal.

Agradeço aos meus pais, Sérgio (em memória) e Maria Amália, obrigado do fundo do coração por sempre acreditar e confiar no meu potencial, eu dedico este trabalho a vocês, pelo empenho constante e por ter me ensinado os valores que tenho hoje. Agradeço também a minha irmã Anne por todo o apoio durante o curso de doutorado. Agradeço a toda a minha familia e a todos os meus amigos que torceram pela minha caminhada durante todo este processo de aprendizado gratificante. Agradeço especialmente a minha esposa Cláudia e meu filho Pedro pela paciência e por tudo que passamos juntos neste período de crescimento, sem a sua ajuda, apoio e compreensão eu não teria conseguido. Obrigado Meu Deus pela tua grandeza, pelo seu amor incondicional. Obrigado pelo carinho, pelo cuidado com minha família, por nunca desistir da minha pessoa e por me amparar em todos os momentos.

UM MÉTODO DE APRENDIZAGEM BASEADA EM DESAFIOS: UM ESTUDO DE CASO EM AMBIENTES DE DESENVOLVIMENTO DE APLICATIVOS

RESUMO

O desenvolvimento de aplicativos para dispositivos móveis é um mercado crescente desde quando surgiram as primeiras plataformas de distribuição de grandes empresas. As plataformas de desenvolvimento mudam constantemente com adições de novos recursos de hardware e software organizados em diferentes *frameworks*, o que motiva o fato do desenvolvimento de aplicativos ser escalável para acompanhar as constantes atualizações de plataforma. Determinar a melhor maneira de preparar desenvolvedores para esta indústria de software emergente é um desafio. Há uma necessidade de se identificar e construir um consenso em torno de práticas de sucesso em ambientes de aprendizagem e desenvolvimento de aplicativos para dispositivos móveis. Esta tese tem como objetivo geral propor um método de apoio ao treinamento de desenvolvimento, através de uma abordagem baseada em desafios e práticas ágeis, em um ambiente de ensino e desenvolvimento de aplicativos. A metodologia de pesquisa foi organizada em três fases de investigação. Na fase um, foram identificadas as melhores práticas para desenvolvimento de aplicativos para dispositivos móveis na literatura. A partir destes resultados, na fase dois foi construído um método preliminar integrando a aprendizagem baseada em desafios e práticas ágeis. Nessa etapa também foi realizado um estudo de campo a fim de compreender o ambiente de desenvolvimento de aplicativos para dispositivos móveis, bem como para mapear vantagens e desvantagens, identificando a viabilidade do uso do método, gerando um conjunto de lições aprendidas. Na fase três um estudo de caso foi realizado em quatro universidades de regiões distintas do Brasil, este estudo permitiu aplicar o método e compreender o seu uso em diferentes regiões, gerando um conjunto de oportunidades de melhoria, melhores práticas e recomendações. A adoção deste método apresentou resultados em termos de percepção de ganho de conhecimento e também resultou em projetos de aplicativos entregues ao mercado.

Palavras-chave: Abordagem baseada em desafios; Desenvolvimento de Software; Dispositivos móveis; Desenvolvimento ágil.

A CHALLENGE BASED LEARNING METHOD: A CASE STUDY ON APPLICATION DEVELOPMENT ENVIRONMENTS

ABSTRACT

The development of mobile applications is a growing market since when the first distribution platforms from large IT companies emerged. The development platforms constantly change with additions of new hardware and software features organized in different frameworks, which motivates the fact of being scalable application development to keep up with the constant platform updates. Determining the best way to prepare developers for this emerging software industry is a challenge. There is a need for identifying and building a consensus around success practices in mobile application learning and development environments. This thesis has as main objective to propose a method to support application development training through challenge based learning and agile practices. This research methodology was organized into three research phases. In phase one, the best practices for mobile application development in the literature have been identified. From these results, in phase two a preliminary method integrating challenge based learning and agile practices was proposed. Also, this step carried out a field study in order to understand the mobile application development environment, as well as to map advantages and drawbacks, identifying the method feasibility, providing a set of lessons learned. In phase three a case study was conducted in four universities in different regions of Brazil, this study allowed to apply the method and to understand its use in different regions and generated a set of improvement opportunities, best practices and recommendations. The adoption of this method presented results in terms of knowledge gain perception and also resulted on mobile application projects delivered to the market.

Keywords: Challenge Based Learning; Software development; Mobile; Agile development.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

API	*Application Programming Interface*
CBL	*Challenge Based Learning*
CMMI	*Capability Maturity Model Integration*
DSDM	*Dynamic Systems Development Method*
GRA	*Guiding Questions Resources and Activities*
IDE	*Integrated Development Environment*
MSD	*Mobile Software Development*
MSL	*Mapeamento Sistemático da Literatura*
MVP	*Minimum Viable Product*
PBL	*Problem Based Learning*
PDCA	*Plan-Do-Check-Act*
PRBL	*Project Based Learning*
SBL	*Studio Based Learning*
SDK	*Software Development Kit*
XP	*Extreme Programming*
TBL	*Task Based Learning*
TDD	*Test Driven Development*
UI	*User Interface*
UX	*User Experience*

SUMÁRIO

1. INTRODUÇÃO

Durante anos, os aplicativos desktop dominaram o mercado, entretanto, com o passar dos anos, a web acabou por conquistar um significativo lugar de destaque como plataforma de desenvolvimento [15]. De acordo com Wasserman [127], essa evolução continua e atualmente está relacionada com o desenvolvimento de aplicativos para dispositivos móveis, que vem se consolidando enquanto processo com foco em pequenos dispositivos. Frente a este contexto, e em função do número de recursos disponíveis no desenvolvimento mobile, observa-se uma crescente existência de diferentes plataformas e tecnologias para o desenvolvimento de aplicativos móveis [21], sendo possível utilizar diferentes *Application Programming Interfaces* (APIs) durante o processo de codificação, o que aponta para a necessidade de um entendimento mais profundo sobre a plataforma utilizada [1].

Para Wasserman [127] *"Usar um dispositivo móvel é diferente de trabalhar com um computador de mesa ou computador portátil. Enquanto gestos, sensores e dados de localização podem ser utilizados em console de jogos e computadores tradicionais, eles tem um papel determinante na maioria dos aplicativos para dispositivos móveis"*. Estes aplicativos, por sua vez, podem ser pré-instalados pelos fabricantes ou obtidos através de plataformas de distribuição [15].

Nesta direção, o mercado de aplicativos conta com basicamente três modelos de desenvolvimento: o modelo nativo que utiliza plataformas como por exemplo Android, iOS e Windows Phone, tendo a habilidade de atingir milhões de clientes através de um simples carregamento do aplicativo na plataforma de distribuição; o modelo web, responsivo, que utiliza tecnologias HTML5, CSS e JavaScript criando um site que tenha características de aplicação, sua vantagem ancora-se por ser um modelo de distribuição rápido e por ter a habilidade de rodar a aplicação imediatamente em múltiplas plataformas utilizando um navegador web [7, 15]; e por fim o modelo híbrido que mistura aspectos do desenvolvimento nativo com design responsivo [41, 45].

Algumas empresas de desenvolvimento de *software* tem um bom entendimento sobre as vantagens e desvantagens de cada plataforma, mas ainda enfrentam dificuldades no momento de escolher qual a que atende melhor as suas necessidades. Isso acontece porque é difícil encontrar uma solução única. Uma decisão equivocada, neste caso, poderia resultar em custos desnecessários, baixo desempenho de aplicativos e uma experiência ruim para os usuários. Para Kareborn e Howcroft [14], o modelo nativo ultrapassa o desenvolvimento baseado em web tanto no número de aplicativos disponíveis [48], como também na quantidade de tempo que os usuários usam o aplicativo no dispositivo [21, 41].

Por outro lado, alguns desenvolvedores veem a facilidade da distribuição web como uma vantagem se comparado com o processo nativo de carregar um aplicativo para uma plataforma de distribuição [111]. Por outra via, os aplicativos com abordagem web sofrem com questões de incompatibilidade de navegadores e uma lenta evolução dos padrões de desenvolvimento web para dispositivos móveis [15].

Neste interim, enquanto existe uma proliferação de aplicativos baseados na web, ainda existe um mercado muito ativo para aplicativos nativos [15, 45]. Um bom entendimento das vantagens e

desvantagens são um fator crítico para a distribuição de aplicativos. Alguns desenvolvedores, por sua vez, também consideram as abordagens híbridas como um caminho natural para desenvolver código multiplataforma, como por exemplo PhoneGap, Titanium, etc. De acordo com alguns estudos [23, 27, 44, 130] a abordagem híbrida é promissora no desenvolvimento de aplicativos para dispositivos móveis.

Paralelamente a este cenário, a relevância do desenvolvimento de *software* para dispositivos móveis atingiu um ponto em que as plataformas de distribuição para dispositivos móveis se tornaram as mais populares [20]. Como muitos usuários tem movido do uso de computadores pessoais para dispositivos móveis, as empresas devem seguir esta tendência, investindo no desenvolvimento de aplicativos para dispositivos móveis. Entretanto, esta mudança representa um desafio por causa de ciclos de inovação muito curtos para *hardware* e sistemas operacionais, como também aplicativos que utilizam serviços em servidores *back-end* [47].

1.1 Justificativa e Relevância

Apesar do acúmulo de conhecimentos gerado nesses últimos anos, o cenário em que se expressa a indústria de desenvolvimento de aplicativos para dispositivos móveis continua em uma crescente ascensão. Nesta direção, o modelo tradicional/cascata não parece um caminho promissor, pois longos ciclos de planejamento e desenvolvimento resultam em aplicativos que podem estar desatualizados [2, 15, 96].

Estas transformações trazem à tona novas questões para a indústria que precisa elaborar formas de aumentar sua agilidade de maneira a responder mais rápido as mudanças de mercado. Logo, essas mudanças geram uma necessidade de reavaliar os processos de desenvolvimento e como os desenvolvedores são treinados. Não obstante, o desafio atual consiste em entender uma melhor forma de preparar os estudantes para atuar nesse mercado emergente [84].

Em um relatório do primeiro Workshop Internacional em Engenharia de Sistemas Móveis, Lewis *et al.* [65] reportaram que os aplicativos móveis tem se tornado uma parte importante para a indústria e sistemas de missão crítica. Um estudo encomendado pela OutSystems [3] revela as principais estatísticas do mercado de desenvolvimento de aplicativos móveis e os desafios que as organizações enfrentam frente a escassez de competências nesta área. Realizado com mais de 200 gerentes e diretores de desenvolvimento de aplicativos de empresas do Reino Unido e Estados Unidos, o estudo concluiu que o mercado tem um desafio significativo na criação de projetos de mobilidade, gerando uma dificuldade em atender as demandas do mercado e impactando receita de empresas. *"O grande desafio, consiste em encontrar desenvolvedores capacitados que estão em falta do mercado"* [3]. A pesquisa comprova essa escassez de mão-de-obra qualificada, revelando que apenas 6% dos entrevistados afirmaram que possuem todas as habilidades necessárias para o desenvolvimento de aplicativos móveis.

A academia, usualmente, ensina conceitos de Engenharia de Software através de disciplinas teóricas, com aulas expositivas, leituras complementares e exercícios teóricos, que abrangem a com-

petência prática em um espaço de tempo limitado [54]. Nesse sentido, quando expostos à indústria, os estudantes encontram um cenário em que técnicas e métodos aprendidos podem ser pouco aplicados [85]. Segundo Santos *et al.* [33], o ensino da engenharia de *software* deve preparar os estudantes para participações efetivas em ambientes colaborativos e interdisciplinares.

Por outra via, Meyer [79] afirma que a academia não deve assumir toda responsabilidade, pois a universidade é diferente de uma empresa, mas ao mesmo tempo precisa preparar os estudantes para os reais desafios. Um professor, consegue desenvolver e implementar planos de aula, avaliar provas e trabalhos, e ensinar os estudantes a analisarem os fatos e serem pensadores críticos, mas o engajamento dos estudantes no processo de aprendizado é um desafio constante para os educadores, pois com frequência os estudantes são atores passivos no aprendizado [8, 74].

De acordo com Endres e Rombach [5] a Tecnologia de Informação (TI) faz parte do campo coberto pela disciplina acadêmica de Ciência da Computação e novas pessoas estão constantemente entrando no campo de TI, mas muitas delas não foram treinadas como cientistas da computação ou engenheiros de software. Pauca e Guy [92] apresentaram um estudo sobre ensino de engenharia de *software* na universidade Wake Forest, e neste estudo eles relatam que a disseminação do uso de programação para dispositivos móveis e desenvolvimento ágil, focando em resolver problemas desafiadores criando benefícios diretos para a sociedade gera um grande impacto no desempenho e motivação dos estudantes durante o curso.

Não obstante, o método tradicional de ensinar continua sendo o mais aplicado ao redor do mundo pelas instituições. Entretanto, quando estudantes são designados como participantes de projetos onde eles exercem um papel ativo, gera um maior engajamento, porque os estudantes constroem um significado baseado no conhecimento que eles possuem e o relacionamento com o material [8, 74]. Teorias sobre aprendizado baseado em experiências tem sido estudadas durante décadas, *"John Dewey em 1910 desenvolveu o início de algumas teorias de aprendizado baseado em experiências que são utilizadas atualmente, ele acreditava que a função principal de um professor era prover acesso a experiências significativas e que isso deveria ser a fundação dos ambientes de ensino modernos"* [102].

Para Beckman *et al.* [13] uma das formas de aumentar a qualidade do ensino é aperfeiçoar os processos de ensino e aprendizagem, através de didáticas e estratégias inovadoras. Scharff e Verma [105], por sua vez, definiram um modelo de trabalho com métodos ágeis no contexto de ensino na Universidade de Pace em um projeto de desenvolvimento de aplicativo para dispositivos móveis. Concomitantemente, Harleen e Swati [37] realizaram uma revisão e análise da literatura sobre métodos ágeis e reportaram que *agile* é uma abordagem efetiva no desenvolvimento de aplicativos para dispositivos [15].

Por outra vertente, semelhante a outros métodos de pedagogia ativa como *Problem-Based Learning* e *Project-Based Learning*, o *Challenge-Based Learning* (CBL) utiliza reflexões que permitem tempo para pensar em soluções inovadoras sobre diferentes perspectivas, gerando um fluxo contínuo de novos questionamentos e novas reflexões [58]. Outro ponto que diferencia o CBL das demais práticas é que ele é efetivo em ambientes de aprendizado com tecnologia [60], o que permite agilidade

no processo de aprendizado já que suas habilidades são obtidas através do trabalho em problemas do mundo real.

Seguindo esta direção e com o intuito de contribuir para o debate melhorando a preparação de instrutores e estudantes nesse mercado emergente de dispositivos móveis, o presente estudo *"Um método de aprendizagem baseada em desafios: Um estudo de caso em ambientes de desenvolvimento de aplicativos"* buscou propor um método de apoio ao treinamento o qual combina uma abordagem de aprendizado baseada em desafios com práticas ágeis através de um estudo em ambientes de ensino e desenvolvimento de aplicativos para dispositivos móveis. Para tanto, foi desenvolvido neste trabalho um método que utiliza *Challenge-Based Learning* (CBL) [58,60,61] e práticas ágeis [12,19,107,108], avaliando, ainda que de forma incipiente, como essa junção poderá vir a ser adotada em ambientes de aprendizado de desenvolvimento de aplicativos.

Dar visibilidade teórica e empírica no que diz respeito ao uso de CBL e de práticas ágeis aplicados em um ambiente de ensino e desenvolvimento de aplicativos, tornou-se um dos pontos centrais deste estudo na medida em que buscou identificar desafios, oportunidades, práticas e padrões. Logo, acredita-se que a contribuição e singularidade desta pesquisa se dão, no momento em que o mercado de desenvolvimento de aplicativos para dispositivos móveis exige práticas e estratégias de ensino capazes de responder de forma efetiva as complexas demandas do mercado atual.

1.1.1 Problema de pesquisa, Questões Norteadoras e Objetivos

O problema de pesquisa é um processo contínuo de pensar reflexivo, sua caracterização define e identifica o assunto em estudo, devendo ser formulado preferencialmente de forma interrogativa [22]. Atento a esse pressuposto, formulou-se o seguinte problema de pesquisa: **Como treinar estudantes em um ambiente de desenvolvimento de aplicativos para dispositivos móveis?** Como forma de responder ao problema de pesquisa, este estudo se desdobrou em 03 (três) questões norteadoras:

- Questão norteadora 01 - Quais os desafios no desenvolvimento de aplicativos para dispositivos móveis na engenharia de software?
- Questão norteadora 02 - Quais os fatores que tem influência no ensino de desenvolvimento de aplicativos para dispositivos móveis?
- Questão norteadora 03 - Como a abordagem baseada em desafios pode ajudar no ensino de desenvolvimento de aplicativos?

Para explicitar o problema de pesquisa, foram elaborados os respectivos objetivos. O objetivo geral foi o de: ***propor um método de apoio ao treinamento no desenvolvimento de aplicativos para dispositivos móveis, através de uma abordagem de ensino baseada em desafios***. Tendo em vista a sua complementação e o desvendamento das questões norteadoras, foram também formulados três objetivos específicos, apresentados na Tabela 1.1.

Tabela 1.1: Esquema metodológico: questões norteadoras e objetivos específicos

Problema	**Objetivo Geral**
Como treinar estudantes em um ambiente de desenvolvimento de aplicativos para dispositivos móveis?	Propor um método de apoio ao treinamento no desenvolvimento de aplicativos para dispositivos móveis através de uma abordagem de ensino baseada em desafios.
Questões Norteadoras	**Objetivos Específicos**
QN1. Quais os desafios no desenvolvimento de aplicativos para dispositivos móveis na engenharia de software?	1. Explorar a base teórica na área de treinamento e desenvolvimento de aplicativos móveis.
QN2. Quais os fatores que tem influência no ensino de desenvolvimento de aplicativos para dispositivos móveis?	2. Explorar o ambiente de ensino e desenvolvimento de aplicativos e identificar vantagens, desvantagens e lições aprendidas.
QN3. Como a abordagem baseada em desafios pode ajudar no ensino de desenvolvimento de aplicativos?	3. Identificar quais as características e recomendações para o uso de abordagem baseada em desafios em ambientes de ensino e desenvolvimento de aplicativos.

1.2 Estrutura da Tese

O restante desta tese é estruturado da seguinte forma: O capítulo 2 apresenta a base teórica dos principais tópicos discutidos nesta tese. A base teórica descreve o desenvolvimento de aplicativos para dispositivos móveis, os modelos de desenvolvimento de aplicativos, os desafios no desenvolvimento de aplicativos, e apresenta estudos relacionados com ensino e desenvolvimento de aplicativos. O capítulo 3 introduz a abordagem metodológica descrevendo o desenho de pesquisa e descrição de cada uma das três etapas desta pesquisa, as quais englobam o mapeamento sistemático da literatura, a proposta inicial do método, o estudo de campo, o estudo de caso e também a evolução do método.

O capítulo 4 apresenta o mapeamento sistemático da literatura, o protocolo utilizado, bases de dados utilizadas, strings de busca, seleção dos estudos, extração de dados e classificação dos estudos. Além disso, este capítulo descreve os resultados e fatores com influência no ensino e desenvolvimento de aplicativos. O capítulo 5 apresenta a estrutura da abordagem baseada em desafios, a estrutura do *framework* Scrum e a estrutura do método que integra a abordagem baseada em desafios e o desenvolvimento ágil, descrevendo seu processo de definição e as considerações iniciais para seu uso.

O capítulo 6 apresenta o estudo de campo realizado em uma universidade. O estudo de campo descreve as diferenças no desenvolvimento de aplicativos para dispositivos móveis, a opinião dos participantes sobre o uso de desenvolvimento ágil para desenvolver aplicativos, as ferramentas utilizadas para desenvolver os projetos de aplicativos, as práticas de desenvolvimento envolvidas, e a percepção sobre diferenciais do método. Além disso o estudo de campo descreve as características que fazem do método uma solução viável, a percepção das vantagens e desvantagens do método, como também lições aprendidas. O capítulo 7 apresenta o estudo de caso realizado em quatro

universidades em regiões distintas do Brasil, descrevendo como o desenvolvimento ágil influencia o desenvolvimento de aplicativos e as práticas de desenvolvimento utilizadas nos projetos. Apresenta também a influência do uso do método e as percepções sobre o seu uso através das diferentes fases, proporcionando comparações com abordagens tradicionais de ensino e análises, de forma a gerar um conjunto de oportunidades de melhoria, melhores práticas e recomendações.

O capítulo 8 apresenta a construção do método final e as atividades envolvidas em cada uma de suas fases, além de apresentar uma discussão dos efeitos no desenvolvimento de aplicativos relacionados pela literatura. O capítulo 9 discute os objetivos de pesquisa, as principais contribuições desta tese, as limitações de pesquisa e os estudos futuros. Os apêndices desta tese estão organizados da seguinte forma: complemento do protocolo de estudo de campo (Apêndice A), questionário aplicado no estudo de campo (Apêndice B), complemento do protocolo do estudo de caso (Apêndice C) e questionário aplicado no estudo de caso (Apêndice D).

2. BASE TEÓRICA

O referencial teórico representa uma importante etapa de pesquisa, proporcionando uma estrutura para estabelecer a importância do estudo e também estabelecer uma referência para comparar resultados [22]. Na seção 2.1 apresentam-se os conceitos relacionados com o desenvolvimento de aplicativos para dispositivos móveis. A seção 2.2 apresenta os principais conceitos e modelos de uso de pedagogia ativa. A seção 2.3 apresenta os principais desafios e práticas no uso de desenvolvimento ágil.

2.1 Desenvolvimento de Aplicativos para Dispositivos Móveis

Desenvolver aplicativos para dispositivos móveis é um processo com foco em pequenos dispositivos, geralmente chamados de *Smartphones* ou *Tablets*. Estes aplicativos podem ser pré-instalados pelos fabricantes ou obtidos através de plataformas de distribuição [15]. O desenvolvimento de aplicativos para dispositivos móveis é um mercado crescente desde 2008 quando Apple e Google abriram suas lojas de aplicativos para as plataformas iOS e Android. As plataformas de desenvolvimento mudam constantemente com adições de novos recursos de *hardware* e *software* organizados em diferentes *frameworks*, o que motiva o fato do desenvolvimento de aplicativos ser escalável para acompanhar as constantes atualizações de plataforma.

Existem diferentes ambientes de programação para as principais plataformas de desenvolvimento [127]. Para *Windows Phone* existe o ambiente *Microsoft Visual Studio*, para *Android* além do *Android Studio* também existem ferramentas de *plug-in* para o Eclipse e para a plataforma iOS é utilizado o Xcode. Um dos desafios no desenvolvimento de aplicativos, independente da plataforma é manter a qualidade do software. Neste contexto, Haller [47] conduziu em 2013 um estudo sobre o teste de aplicativos para dispositivos móveis e o *feedback* de usuários nas lojas de aplicativos em mais de 1000 avaliações e descobriram que muitos aplicativos não atingem o nível básico de qualidade e que os *feedbacks* dos usuários representam um risco se o aplicativo é grande ou se falha em atingir as demandas dos usuários uma vez que os usuários devem conseguir instalar e iniciar os aplicativos sem problemas de execução.

Desenvolver aplicativos para dispositivos móveis que sejam robustos exige um aprendizado avançado, práticas e ferramentas. Alguns desses são relacionados com métodos de aprendizados e processos, arquitetura de plataformas, *frameworks* e ferramentas. Muppala [83] explorou a plataforma Android como meio de ensinar conceitos avançados em *software* embarcado para estudantes de Ciência da Computação, descrevendo como positivo o fato de que a oportunidade de aprendizado deve ser efetivamente usada para promover o estudo da área para os estudantes.

2.1.1 Modelos de desenvolvimento de aplicativos

As rápidas mudanças e desenvolvimento do mercado de tecnologia da informação tem feito com que tecnologias e plataformas promissoras estejam a disposição [113]. Isso faz com que o mercado de desenvolvimento de aplicativos aumente cada vez mais, utilizando basicamente três modos de desenvolvimento: o modelo nativo que utiliza plataformas como por exemplo Android, iOS e Windows Phone; o modelo web, responsivo, que utiliza tecnologias HTML5, CSS e Javascript criando um *site* que tenha características de aplicação, e por fim o modelo híbrido que mistura aspectos do desenvolvimento nativo com *design* responsivo [41, 45].

No desenvolvimento nativo de plataformas, a vantagem é a possibilidade de utilizar as plataformas de distribuição que atingem uma base muito grande de usuários através de um simples carregamento do aplicativo. O desenvolvimento nativo ultrapassa o desenvolvimento de aplicativos web tanto no número de aplicativos disponíveis [14, 48], como também na quantidade de tempo que os usuários usam o aplicativo no dispositivo [21, 41].

A principal vantagem no desenvolvimento de aplicativos para dispositivos móveis usando o modelo web é um modelo de distribuição rápido e a habilidade de rodar a aplicação imediatamente em múltiplas plataformas utilizando um navegador web [7]. Existem desenvolvedores que encontram facilidade na distribuição web como uma vantagem se comparado com o processo nativo de carregar um aplicativo para uma plataforma de distribuição [111]. Entretanto, os aplicativos com abordagem web sofrem com questões de incompatibilidade de navegadores e uma lenta evolução dos padrões de desenvolvimento web para dispositivos móveis [15].

A abordagem híbrida é diferente da abordagem nativa e também da abordagem web [98]. Não são nativas porque o *layout* é gerado através da renderização do navegador ao invés de usar linguagem e objetos nativos da plataforma e não são puramente web devido a falta de suporte a APIs e outras funcionalidades [41, 45, 98].

As plataformas tecnológicas incluem arquiteturas de *hardware* e *frameworks de software*, permitindo diferentes combinações de *hardware* e software [113]. Algumas são as propostas para a adoção de uma plataforma [113]: Características da Plataforma, Externalidades de Rede, Motivação Pessoal e Características Individuais e Interação Social, as quais são descritas na Tabela 2.1

Tabela 2.1: Classes para adoção de modelo de desenvolvimento móvel

Classe	**Descrição**
Características da Plataforma	Atributos sobre a plataforma que são percebidos como diferenciais pelos utilizadores em potencial.
Externalidades de Rede	Refere-se a valores diretos ou indiretos esperados de uma grande rede de pessoas que adotam uma determinada plataforma.
Características Individuais	Características pessoais que explicam diferentes padrões de adoção de plataforma e motivação em uma população de desenvolvedores .
Interação Social	É um dos fatores mais importantes na adoção de uma plataforma visto que desenvolvedores geralmente buscam novidades de plataformas do mercado. Uma vez que o mercado está sempre em constante mudanças as quais geram impacto nos futuros rendimentos dos desenvolvedores.

Dentre os itens apresentados, os principais fatores motivacionais para a adoção de uma determinada plataforma de desenvolvimento de aplicativos são: disponibilidade de ferramentas, potencial de mercado e negociabilidade [113].

Existem diferentes sistemas operacionais para o desenvolvimento de aplicativos, desde o Symbian utilizado nos dispositivos Nokia e Java ME, até sistemas operacionais mais modernos como Android e iOS. Baseado no estudo de Hammershoj, Sapuppo e Tadayoni [48], a Tabela 2.2 apresenta os três principais sistemas operacionais (SO) utilizados atualmente para o desenvolvimento de aplicativos para dispositivos móveis: iOS, Android e Windows Phone.

Tabela 2.2: Principais sistemas operacionais para dispositivos móveis

SO	**Descrição**
iOS	Este sistema operacional permite o uso de multi tarefas. Os aplicativos são desenvolvidos para dispositivos iPhone, iPad e Apple TV, principalmente utilizando as linguagens Objective-C e Swift através da IDE xCode. O desenvolvimento é controlado e baseado em um guia fornecido pela Apple. O principal canal de distribuição dos aplicativos desenvolvidos nesta plataforma é o Apple Store.
Android	Este sistema operacional é um resultado da Open Handset Alliance (OHA) com o Google. A principal linguagem de desenvolvimento de aplicativos para esta plataforma é o Java. Existem diferentes IDEs para o desenvolvimento, além do Android Studio, também pode ser utilizado o Eclipse com o plug-in Android é utilizado como ferramenta de desenvolvimento [122]. O sistema operacional é *open source*. O principal canal de distribuição dos aplicativos desenvolvidos nesta plataforma é o Google Play.
Windows Phone	É um sistema proprietário e inicialmente baseado no Windows CE kernel. Esta plataforma suporta o desenvolvimento utilizando diferentes linguagens de programação como por exemplo J2ME, Visual Basic, .NET, Visual C++ e outras. O principal canal de distribuição dos aplicativos desenvolvidos nesta plataforma é o Windows Marketplace.

No mercado encontram-se disponíveis diversos sistemas operacionais e ferramentas para o desenvolvimento de aplicativos para dispositivos móveis, muitos destes ambientes são específicos para um determinado SO. Neste sentido, existem soluções para múltiplas plataformas como por exemplo (PhoneGap, Titanium, Cabana, entre outros) os quais permitem a implementação de um aplicativo e sua interface de usuário utilizando tecnologias como HTML e CSS, permitindo dessa forma que um aplicativo pode ser distribuído para diferentes sistemas operacionais (iOS, Android e Windows Phone) [23].

A abordagem multiplataforma possui vantagens e desvantagens quando comparada com o desenvolvimento nativo ou web. Diferentes são os critérios de decisão para adotar como modelo de desenvolvimento [23], dentre eles Qualidade de experiência de usuário, aplicativos, potenciais usuários, custos, segurança, suporte e *time-to-market*, os quais são descritos na Tabela 2.3.

Tabela 2.3: Critérios de decisão para adotar modelos de desenvolvimento de aplicativos

Critério	**Nativo**	**Web**	**Multi Plataforma**
Qualidade UX	Excelente	Muito Boa	Regular
Qualidade Aplicativos	Alta	Média	Regular
Potenciais Usuários	Usuários da plataforma	Máximo (incluindo diferentes dispositivos)	Grande porque atinge usuários de diferentes plataformas
Custo de Desenvolvimento	Alto	Médio	Regular
Segurança do aplicativo	Excelente	Depende do navegador	Ruim
Suportabilidade	Complexa	Simples	Média a Complexa
Time-to-market	Alto	Médio do navegador	Curto

De acordo com os resultados descritos no estudo de critérios de decisão proposto por Dalmasso [23], a maioria dos desenvolvedores gostaria de distribuir aplicativos para as maiores plataformas (iOS, Android), proporcionando uma experiência de usuário (UX) consistente entre as plataformas. Neste sentido, as opções e escolhas são diversas, o que gera diferentes desafios em ambientes de desenvolvimento de aplicativos para dispositivos móveis.

2.1.2 Desafios no desenvolvimento de aplicativos

Os ambientes de desenvolvimento de aplicativos para dispositivos móveis apresentam algumas particularidades quando comparados com outros ambientes de desenvolvimento [26], e isso se deve primeiramente ao fato de existir diferentes plataformas de desenvolvimento (iOS, Android, Windows Phone, entre outros) e também diferentes fabricantes de *hardware* (HTC, Samsung, Apple, etc) o que exige uma maneira de repensar um mesmo aplicativo em diferentes contextos [127], e em segundo lugar porque as interfaces de usuário tratam de um novo paradigma de interação homem-computador [11, 91].

As novidades de plataformas de desenvolvimento de aplicativos proporcionam oportunidades, desafios e constantes mudanças [127]. Roman, Picco e Murphy relatam que a mobilidade representa um colapso total de todos os pressupostos de estabilidade [99]. A instabilidade de requisitos é um problema crítico em ambientes de desenvolvimento de aplicativos para dispositivos móveis [127], devido a constante introdução e atualização de tecnologias, e este problema é endereçado através de iterações curtas e prototipagem, o que faz o desenvolvimento ágil ser muito eficiente para tratar novas tecnologias [52]. Neste contexto, Dehlinger e Jeremy [26] descreveram quatro desafios da engenharia de *software* para desenvolvimento de aplicativos: Projetar interfaces de usuário que sejam universais, o desenvolvimento de linhas de produtos de aplicativos móveis, prover suporte a aplicações sensíveis ao contexto e equilibrar agilidade com incerteza na especificação de requisitos.

Através da evolução dos *software* e tecnologias, a dinâmica de requisitos emergentes tem se tornado ainda mais desafiadora de ser solucionada, fazendo com que flexibilidade e personalização de *software* torne-se necessária, a fim de modificar um sistema para melhor atender as necessidades dos usuários [125]. Uma outra área de pesquisa também envolve questões de manutenção de aplicativos neste mundo de mudanças rápidas nas plataformas móveis, pois enquanto usuários comuns frequentemente atualizam os seus dispositivos e seus aplicativos, muitos usuários corporativos são menos favoráveis a este tipo de comportamento [127]. Neste sentido, Gordon [42] apresentou um estudo que reune alguns dos desafios no desenvolvimento de aplicativos para dispositivos móveis, os quais são descritos na Tabela 2.4.

Tabela 2.4: Desafios no desenvolvimento de aplicativos

Desafio	**Descrição**
Interface de usuário e Usabilidade	Este desafio inclui diferentes formas de pensar as entradas e saídas produzidas, lidando com uma tela relativamente pequena, com foco na capacidade de resposta e fornecendo interações sensíveis ao contexto.
Cooperação de dispositivos	Os aplicativos em dispositivos móveis são muitas vezes uma parte de uma solução multi-dispositivo, utilizando também outras camadas de *software* e servidores.
Hardware	Lidar com características de múltiplos sensores, gasto de bateria, memória limitada, entre outros.
Manipulação de dados	Armazenamento de dados em dispositivos móveis nem sempre segue o tradicional modelo de sistema de arquivos. Os dados podem ser armazenados em arquivos específicos de aplicativos, arquivos de preferência, em bancos de dados específicos ou remotamente em um ou mais servidores.
Interação de aplicativos	Aplicativos em dispositivos móveis são escritos para interagir uns com os outros sempre que necessário .
Questões de programação	Questões de programação incluem o ciclo de vida do aplicativo, projeto para múltiplas plataformas, segurança e privacidade, e teste e depuração.

Os desafios para desenvolver aplicativos também são relatados por diferentes estudos, citando que os aplicativos precisam ser desenvolvidos para uma variedade de dispositivos os quais operam em diferentes plataformas de *hardware* e *software* que vivem em constante atualização [122, 123, 125, 127], apresentando uma necessidade do gerenciamento da evolução constante das plataformas e requisitos, como também de um *time-to-market* adequado. Com o aumento da disponibilidade de orientação dos fabricantes e comunidades, os desenvolvedores podem programar seus aplicativos com algumas facilidades. Dessa forma, conhecimentos de engenharia podem ser utilizados criar arquiteturas e SDKs que ajudam desenvolvedores a interagir com os recursos existentes. No entanto, esses aspectos técnicos não abordam as grandes questões de desenvolvimento de aplicativos em escala com diferentes níveis de suporte [62]. As plataformas Android e iOS tem feito um bom

trabalho para ajudar a lidar com diferenças relacionadas a interface de usuário, experiência de usuário, desempenho e sensores, entretanto não prestar atenção a estes problemas pode fazer uma grande aplicação em um dispositivo aparecer desajeitada em outro [122].

Em um curso de engenharia de software, os estudantes são expostos a conceitos de processos e qualidade de software [116]. Alguns desses conceitos precisam de uma atenção especial no desenvolvimento de aplicativos para dispositivos móveis, o que inclui segurança, integridade, eficiência, portabilidade e usabilidade. Existem algumas outras questões importantes a serem consideradas, no que diz respeito ao desenvolvimento de tais aplicações em cursos baseados em projetos, dentre elas o papel ativo dos estudantes e o engajamento em resolver problemas reais [75, 116].

2.1.3 Ensino e aprendizado de desenvolvimento de aplicativos

Os cursos de desenvolvimento de aplicativos apresentam uma série de características, e neste sentido Burd *et al.* [15] publicaram um levantamento de centenas de cursos em computação móvel, apresentando as principais características de mobilidade que podem ser utilizadas para moldar esses cursos, as quais são descritas na Tabela 2.5.

Tabela 2.5: Principais características para moldar cursos de desenvolvimento de aplicativos

Característica	**Descrição**
Mobilidade e Difusão	Dispositivos tendem a ser propriedade de indivíduos e o *software* é integrado com a dinâmica dos proprietários seguindo mudanças de localização e atividades.
Interface de usuário e programação orientada a eventos	Aplicativos móveis utilizam largamente recursos de interfaces gráficas de usuário e a maioria delas são orientadas a eventos.
Interrupções	Lidar com características de múltiplos sensores, gasto de bateria, memória limitada, entre outros.
Entradas baseadas em sensores	A computação em desktop se baseia principalmente em um teclado e mouse para a entrada, enquanto dispositivos móveis tipicamente integram diversos tipos de *hardware* de entrada: toque, multitoque, acelerômetros, GPS, giroscópio, magnetômetro, sensores de proximidade, câmeras, sensores infravermelhos, microfones, etc.
Recursos finitos	Os benefícios do consumo de recursos de forma conservadora, utilizando algoritmos de baixa complexidade, mantendo serviços leves, e limitando o uso intensivo de sensores e gráficos são muito mais aparentes em dispositivos móveis do que em desenvolvimento tradicional.
Proximidade dos usuários	Os desenvolvedores têm acesso direto a um enorme e diversificada base de usuários diversificada por meio de canais de distribuição de aplicativos de terceiros.

A evolução na indústria de dispositivos móveis também oferece novas experiências e recursos, onde o desafio para os educadores é entender e explorar a melhor maneira de utilizar estes recursos para suporte no aprendizado [84]. A importância do uso de dispositivos móveis no processo de aprendizagem e suas vantagens foram estudadas por Fetaji [36]. Dessa forma, diversos métodos

de ensino de desenvolvimento aplicativos para dispositivos móveis foram apresentados em diferentes estudos [15,36,56,72,84,116]. A literatura apresenta também autores [4,36,39,75,104] que aplicam diferentes métodos de pedagogia ativa para o ensino e aprendizado de desenvolvimento de aplicativos para dispositivos móveis.

2.2 Pedagogia Ativa

Diferentes formas de ensino experimental tem sido estudadas nas últimas décadas e muitos destes estudos são classificados pelo uso de Pedagogia Ativa [31, 56, 72]. Técnicas de pedagogia ativa engajam os estudantes e melhoram seu desempenho positivamente [31, 75, 121]. Este tipo de abordagem move os estudantes para além de um papel expectador em aulas tradicionais para atividades que engajam os estudantes em problemas reais [57, 58, 60, 61].

> [...] mudanças sociais, econômicas e tecnológicas estão transformando a vida profissional e aumentam o nível de requisitos exigidos dos profissionais, não influenciando somente os programas de treinamento mas também as expectativas formuladas para educação superior e especialistas conhecidos na área de ensino e aprendizado tem compartilhado de maneira geral que adquirir capacidade critica e conhecimentos em solução de problemas é um objetivo primário na educação [...] Dochy *et al.* [31].

Ambientes de aprendizado poderosos tem como objetivo o desenvolver uma configuração educacional na qual o aprendizado dos estudantes é o principal foco e que o ensino é baseado na concepção construtivista de aprendizado e suas aplicações pedagógicas [31].

Este tipo de abordagem gera oportunidades para a criatividade dos estudantes, mesmo que o professor talvez planeje um problema com certa dificuldade, sempre existem diferentes maneiras de se resolver um dado problema e isto significa também que talvez o professor não saiba mais tudo sobre o problema [56]. Marcangelo e Gibbon [72] citam que o PBL ficou popular nos anos 60 como resultado de uma pesquisa realizada por Barrows e Tamblyn nas capacidades de raciocínio de estudantes, onde argumentaram que o PBL *(Problem Based Learning)* é baseado em duas premissas: A primeira é que o aprendizado através de resolução de problemas é muito mais efetivo que o aprendizado baseado na memória para criar um corpo de conhecimento utilizável; A segunda, era que as habilidades, as quais são muito importante no tratamento de pacientes, são também resolução de problemas, e não habilidades de memória, significando que PBL reflete a forma como pessoas aprendem na vida real. Mais recentemente, o modelo do PBL tem sido extendido para outras disciplinas como matemática, ciências e estudos sociais [121].

O PBL aborda inicialmente problemas ou situações ao invés da exposição direta do conhecimento, e o currículo oferece experiências de pedagogia ativa integrando disciplinas [72]. Neste contexto, o PBL é um método baseado numa abordagem construtivista onde os estudantes trabalham em grupos colaborativos para identificar o que eles precisam aprender [59] e estes ambientes são guiados por

um *framework* no qual os estudantes sistematicamente exploram e analisam problemas a serem resolvidos [31].

Problem-Based Learning é baseado em uma filosofia construtivista a qual sugere que o que é aprendido não pode ser separado de como o aprendizado ocorre [75], é uma das mais conhecidas formas de pedagogia ativa. As principais características do PBL envolvem a apresentação de um problema ou definição e formulação do problema (declaração do problema), seguida da geração de um inventário de conhecimento através de uma lista do que precisa se saber sobre o problema para gerar soluções possíveis proporcionando um compartilhamento de resultados e soluções.

Uma abordagem similar ao PBL é chamada de *Project-Based Learning* (PRBL), sendo definida como uma abordagem de instrução centrada no aluno, na qual os estudantes trabalham em equipes para completar um projeto que não tem limites específicos [35]. PRBL é um modelo que organiza o aprendizado através de projetos. De acordo com definições encontradas em manuais para professores, projetos são atividades complexas, baseados em questões ou problemas, que envolvem os estudantes em planejamento, resolução de problema, tomada de decisão e atividades de investigação [121]. *"De modo semelhante, a limitação do âmbito da revisão para pesquisar artigos nos quais os autores descrevem seu trabalho como PRBL parece deixar de fora a pesquisa prévia em focada no projeto, a educação experiencial ou pedagogia ativa. De qualquer forma, a ideia de colocar os estudantes a trabalhar em projetos não é nova"* [121]. Existe uma tradição em escolas de atividades práticas e realização de projetos para desenvolver temas interdisciplinares [121]. PRBL considera o aprendizado através da realização de projetos reais e trabalho em equipe [38, 104], utilizada em diversos estudos. Algumas das características desta abordagem são:

- Pode ser explorada para organizar a aprendizagem em torno de projetos.
- Considera uma abordagem baseada na colaboração e cooperação.
- Os aprendizes participam da construção do conhecimento.
- Busca resolver problemas reais.

De acordo com uma pesquisa publicada em 2009 por Ikonen *et al.* [56] existem alguns desafios na implementação de PRBL ou outros métodos de pedagogia ativa i) a mudança no papel do professor-instrutor tradicional em tutor-facilitador exigindo que os próprios professores devem estar dispostos a adaptar novas ideias e métodos pedagógicos; ii) o papel dos estudantes também muda de uma postura passiva que recebe as informações para um membro ativo de um grupo de projeto o que nem sempre é fácil e o professor tem que estar presente para ativar e guiar o processo de aprendizado; iii) manter os grupos de projeto juntos talvez seja difícil.

O *Task-Based Learning* é uma estratégia de instrução eficiente para o aprendizado de atividades através de tarefas. Esta abordagem oferece ação e reflexão, um contraste com o aprendizado mecânico que oferece pouca ação e reflexão [36]. Assim, Fetaji [36] apresenta esta abordagem adaptada para o ensino e desenvolvimento de aplicativos para dispositivos móveis, dividida em quatro estágios, conforme apresentado na Tabela 2.6.

Tabela 2.6: Estágios da abordagem *Task-Based Learning*

Estágio	**Descrição**
Pré-tarefa	Introdução ao tópico e tarefa
Planejamento de tarefa	Aprendizes preparam-se para se familiarizar com a tarefa
Realização de tarefa	Aprendizado e prática
Após Tarefa	Avaliação

Neste contexto, o *Studio-Based Learning* é um modelo pedagógico usado extensivamente em artes e arquitetura, neste tipo de modelo o instrutor geralmente atribui na aula uma especificação que os estudantes devem implementar. Durante o projeto os instrutores tem revisões periódicas com cada estudante [4, 17]. Este modelo funciona melhor com um espaço dedicado como um estúdio, mas isto não é um componente mandatório, ele pode ser adaptado para uso em outros ambientes [4]. Dessa forma, Carter [17] identifica as seguintes características na Tabela 2.7.

Tabela 2.7: Características da abordagem *Studio-Based Learning*

Característica	**Descrição**
Definição	As atribuições de tarefas devem ser primariamente baseadas em projetos
Avaliação	O trabalho dos estudantes é periodicamente avaliado formalmente e informalmente através das revisões de projetos
Engajamento	Os estudantes são engajados nas revisões de projetos para criticar e contribuir o trabalho dos outros estudantes
Artefatos	As críticas do projeto devem girar em torno dos artefatos geralmente criados pela disciplina

Existem diferentes abordagens de pedagogia ativa como PBL, PRBL e também o (CBL)[1], também conhecido como aprendizado baseado em desafios, o qual é uma abordagem que simula um ambiente de trabalho [89].

2.2.1 Aprendizagem baseada em desafios

Os primeiros esforços para construção desta abordagem baseada em desafios foram publicados em 2008 [57], através da iniciativa chamada *Apple Classrooms of Tomorrow* (ACOT2). Esta iniciativa foi um esforço colaborativo com a comunidade de educação para identificar os princípios básicos para as escolas do século 21, e o objetivo desta iniciativa é ajudar as escolas a se aproximarem da criação de um tipo de ambiente de aprendizado que esta geração de estudantes precisa, de forma a aumentar o engajamento com as escolas [57]. Os princípios desta iniciativa são descritos na Tabela 2.8.

[1] Detalhes e exemplos sobre o uso de CBL estão disponíveis em http://www.challengebasedlearning.org

Tabela 2.8: Princípios básicos para escolas do século 21

Princípio	**Descrição**
Entender as habilidades do século 21	Estabelece uma linha base onde educadores, estudantes e comunidade devem ser ensinados sobre as habilidades do século 21 que os estudantes precisam adquirir para ter sucesso.
Currículo aplicado	Oferece uma visão inovadora sobre o tipo de ambiente de aprendizado que deve ser proporcionado para esta geração de estudantes. Onde estudantes precisam ser engajados em contextos relevantes de abordagem baseada em problemas e abordagem baseadas em projetos.
Avaliação informativa	Identifica novos tipos de sistemas para avaliação devido a um papel de independência que os estudantes tem que tomar e em como monitorar e ajustar o aprendizado.
Cultura de inovação	Criar uma cultura que suporta e reforça o uso de inovação para o aprendizado dos estudantes.
Conexão social e emocional	Prover um reconhecimento apropriado de forma pessoal e profissional, onde cada estudante deve ter uma conexão clara com propósito com o ambiente da escola.
Acesso a tecnologia	Prover acesso 24 por 7 aos ambientes de aprendizado, informação, recursos e tecnologias necessárias para o engajamento e apropriação da pesquisa e resultados a serem obtidos e publicados.

Os elementos acima descritos na Tabela 2.8 são a base da aprendizagem baseada em desafios. Este tipo de abordagem visa capacitar os alunos para realizar pesquisas, integrando práticas com a teoria e aplicação de conhecimentos e habilidades, tais como colaboração, resolução de problemas e flexibilidade, a fim de tomar medidas no contexto da comunidade [58], [89], [61], [60]. Em 2009 foi publicado o primeiro relatório técnico [58] definindo a abordagem baseada em desafios como um processo que inicia com uma Idéia, seguido pelas questões essenciais, o desafio, questões guia, atividades e recursos, de forma a determinar uma solução.

Uma das características do CBL é a relação com a capacidade que os alunos possuem de expandir e compartilhar estes impactos com a comunidade através do uso de tecnologia educacional e mídia social [73]. A aprendizagem baseada desafio (CBL) foi construída sobre a prática do PBL, em que os estudantes trabalham em problemas do mundo real em equipes colaborativas, mas com diferenças. No centro da aprendizagem baseada desafio é uma chamada à ação que inerentemente exige que os alunos façam algo acontecer [61].

Em 2011, Johnson e Adams [60] realizaram um estudo sobre a abordagem baseada em desafios e como resultados eles observaram um aumento no engajamento dos estudantes, um aumento no tempo investido para trabalhar nos desafios, aplicação criativa de tecnologia e uma maior satisfação dos estudantes o que melhorou também o aprendizado. *"O acesso a tecnologia (uma parte que integra o CBL), provê significado para os estudantes explorarem opções assim que eles começam a criar e também fornece ferramentas para eles comunicarem o seu trabalho"* [61]. Dessa forma, o CBL ajuda a melhorar diversas áreas de conhecimento [60], 90% dos professores relataram melhora significante em áreas como liderança, colaboração, flexibilidade, criatividade, resolução de problemas

e inovação. Além disso 75% dos professores citam o aumento no engajamento dos estudantes. Mais detalhes sobre a opinião de professores no uso de CBL, como também resultados encontrados em diferentes níveis de instrução estão disponíveis no estudo realizado por Johnson e Adams em 2011 [60].

Os desafios da atualidade geram mudanças, uma escola não pode ser mais um lugar onde o currículo seja a parte da realidade das oportunidades, os estudantes estão buscando ser desafiados de maneira autêntica [58]. O CBL é uma abordagem multidisciplinar que encoraja os estudantes a utilizarem tecnologias em suas vidas diárias para solucionar problemas do mundo real, criando um espaço onde os estudantes podem direcionar suas próprias pesquisas utilizando pensamento crítico sobre como aplicar o que eles aprendem [61]. O CBL possui algumas características principais [58, 60, 61], dentre elas:

- Como parte da metodologia CBL o professor é também um colaborador, não havendo hierarquia.

- O uso de tecnologia é a diferença chave entre as abordagens, o CBL já provou ser efetivo em ambiente rico em tecnologia.

- Reflexão e publicação, perspectivas claras e concisas sobre o que se aprendeu sobre um tópico, conteúdo ou processo.

- Engajamento e co-autoria do aprendizado.

Neste contexto, Timothy *et al.* [89] apresentou um comparativo entre aprendizagem tradicional e aprendizagem baseada em desafios numa configuração de ambiente de trabalho. As comparações entre estes métodos apresentaram que a interação entre os participantes do grupo da abordagem baseada em desafios foi significantemente melhor em termos de itens pós-teste que exigem integração, destacando os benefícios em explorar a aprendizagem em contexto de trabalho. Luis e Marrero [68] reportaram que abordagem baseada em desafios gera um aumento no aprendizado colaborativo porque permite o uso do conhecimento em um objeto real e que isso produz melhores resultados ao fornecer uma experiência com mais engajamento para estudantes.

2.2.2 Comparativo entre abordagens

Cada uma das abordagens possui características que podem ser utilizadas em diferentes contextos e cenários de aprendizado. Neste sentido, buscou-se identificar características em comum entre estas abordagens, como também entender onde a abordagem baseada em desafios se diferencia das demais, a Tabela 2.9 apresenta um comparativo entre as abordagens analisadas neste estudo.

Tabela 2.9: Quadro comparativo entre abordagens de pedagogia ativa

Característica	**CBL**	**PBL**	**PRBL**	**SBL**	**TBL**
Aplicação de conteúdo e habilidades	X	X	X	X	X
Ênfase na independência do estudante	X	X	X		
Trabalho em equipe	X	X	X	X	X
Reflexão sobre os conhecimentos adquiridos	X	X	X	X	X
O professor deixa de ser o ponto central	X				
Simula o ambiente de trabalho	X				
Uso de tecnologia	X				

Um ponto importante de ressaltar é que mesmo que o CBL tenha sido concebido com foco no uso de tecnologia, as demais abordagens também tem sido adaptadas para aplicação em ambientes ricos em tecnologia, por exemplo [17, 36, 38, 75, 104] (embora elas não tenham sido concebidas com este foco). As abordagens de pedagogia ativa avaliadas possuem muitas similaridades e algumas diferenças, algumas dessas podem ser tratadas como questão de escopo e estilo durante a aplicação da abordagem.

Existem diferentes estudos sobre processos e formas de ensino de desenvolvimento de aplicativos para dispositivos móveis [17, 36, 38, 75, 104]. Neste sentido, um estudo sobre o uso de abordagem baseada em desafios e Scrum relacionado a esta tese foi recentemente publicado em uma conferência da ACM [102].

2.3 Desenvolvimento Ágil

O desenvolvimento de *software* adaptativo e incremental que é a base do desenvolvimento ágil vem sendo estudado desde os anos 70. O uso de métodos ágeis no desenvolvimento de *software* flexibiliza o processo de forma a utilizar melhoria continua, entrega incremental, como também se adapta rapidamente a mudanças de requisitos e tecnologias. A metodologia ágil foca mais em aspectos humanos de engenharia de *software* do que processos, interações humanas antes de ferramentas e processos [37]. Neste sentido, um mal entendimento comum é que os projetos ágeis em geral não tem equipes que criam e mantém documentação de projeto [97].

O desenvolvimento ágil é considerado atualmente uma das principais abordagens de desenvolvimento de *software* [103]. Além disso, os métodos ágeis são considerados adequados para diversos tipos de projetos, e as empresas tem aumentado o reconhecimento sobre a necessidade de agilidade. Entretanto, os métodos ágeis também possuem desafios em alguns casos como falta de planejamento de arquitetura, uma ênfase muito grande em resultados antecipados e um baixo nível de cobertura de teste [16].

Existem diferentes métodos ágeis de desenvolvimento de software, em 2010 Hasnain [49] realizou uma revisão da literatura que ajuda pesquisadores a determinar o estado da arte na pesquisa sobre desenvolvimento ágil, reportando que nas conferências da IEEE tem aumentado a cada ano o

número de estudos sobre os métodos Scrum e *Extreme Programming* (XP). Da mesma forma, anteriormente Tore e Torgeir [34] também realizaram uma revisão da literatura sobre estudos empíricos de desenvolvimento ágil, fornecendo uma base dos estudos correntes e referências sobre métodos ágeis de desenvolvimento, tais como: *Feature Driven Development, Crystal Clear, Lean Software Development, Scrum e Extreme Programming*. Sendo assim, existem diferentes métodos ágeis de desenvolvimento e diferentes razões para a adoção do desenvolvimento ágil.

2.3.1 Razões para adotar desenvolvimento ágil

Baseado em um estudo sobre a evolução do desenvolvimento ágil de *software* no Brasil publicado por Melo *et al.* [88], foi possível obter um panorama das principais razões na adoção de métodos ágeis, como também a percepção dos benefícios:

- Acelerar o *time-to-market*.
- Aumento de produtividade.
- Melhorar a disciplina técnica.
- Melhorar a habilidade de mudanças.
- Melhorar a manutenção do software.
- Melhorar a qualidade do software.
- Melhorar o alinhamento entre TI e área de negócios.
- Melhorar o moral da equipe.
- Reduzir custos.
- Reduzir riscos.
- Simplificar o processo de desenvolvimento.
- Visibilidade do projeto.

De acordo com o estudo de Melo *et al.* [88], a habilidade de gerenciar mudanças de prioridades pontua com 67,94% das respostas, processo simplificado de desenvolvimento 60,93%, e um ponto importante encontrado na pesquisa é que 67,1% dos respondentes indicaram que projetos ágeis contribuem para acelerar a entrega do produto, permitindo finalizar os projetos mais rapidamente.

Traçando um comparativo entre os percentuais das razões encontradas por Melo *et al.* [88] e o relatório publicado pela Version One em 2015 [90], é possível mencionar que existe uma semelhança nos percentuais de algumas das motivações de adoção de desenvolvimento ágil. De acordo com o estudo mais recente [90] as três principais razões para adoção de desenvolvimento ágil são: acelerar

a entrega do produto 59% , gerenciar mudanças de prioridades 56% e melhorar a produtividade 53%. Dessa forma, a habilidade de gerenciar mudanças de prioridade, aceleração de entrega do produto, processo simplificado de desenvolvimento e melhoria na produtividade são fatores diferenciais no uso de desenvolvimento ágil, o qual se baseia em diferentes métodos e práticas ágeis de desenvolvimento.

2.3.2 Práticas ágeis

O desenvolvimento ágil tem sido visto com uma boa maneira de conduzir projetos de desenvolvimento de aplicativos [96]. As diferentes metodologias são vistas como uma forma de melhorar a flexibilidade e produtividade do desenvolvimento de software, através da disponibilidade de meios para se adaptar às mudanças nas exigências do ambiente de desenvolvimento, e também para aprender com as experiências de desenvolvimento [96]. Estes métodos envolvem diferentes práticas de desenvolvimento. De acordo com a pesquisa publicada recentemente em 2015 pela Version One [90] as práticas ágeis comumente utilizadas são apresentadas na Tabela 2.10.

Tabela 2.10: Práticas Ágeis

Percentual	Prática	Descrição
80%	Reunião diária	Reunião diária entre todos os integrantes da equipe
79%	Iterações curtas	Curtos ciclos de desenvolvimento
79%	*Backlogs* priorizados	Lista de requisitos (*user stories*) priorizada
71%	Planejamento iterativo	Planejamento iterativo em diferentes ciclos de desenvolvimento
69%	Retrospectivas	Retrospectivas sobre o que se aprendeu em cada iteração de desenvolvimento
65%	Planejamento de *release*	Planejamento de *releases* do produto para os usuários
65%	Teste unitário	Testes nas assinaturas de entrada e saída de um aplicativo
56%	Estimativa	Estimativa baseada na opinião da equipe
53%	Revisão de iteração	Revisões dos produtos funcionais
53%	Quadro de tarefas	Quadro de tarefas com os respectivos status
50%	Integração contínua	Integração contínua de código
48%	PO dedicado	Dedicação de um *"dono do produto"*.
46%	Time único	Equipes de desenvolvimento e teste como um única equipe
43%	Padrões de código	Padrões de desenvolvimento de código
38%	Área de trabalho	Área de trabalho em espaços abertos para a equipe
36%	Refatoração	Processo de reestruturar código existente
34%	*Test-Driven Development*	Processo que persiste na repetição de um curto ciclo de desenvolvimento a partir da descrição do caso de teste
31%	Kanban	Sistema de controle logístico de um ponto de vista de produção

Neste contexto, as cinco práticas mais utilizadas da lista são Reuniões Diárias 80% das respostas, seguida de Iterações Curtas 79%, *Backlogs* Priorizados 71%, Planejamento Iterativo 69% e Retrospectivas 69%. Existem outras práticas ágeis que também foram apresentadas nesta tabela, as quais tem um percentual menor de utilização.

Os processos de desenvolvimento ágil buscam aumentar a transparência na condução dos problemas de desenvolvimento. A agilidade busca dar poder aos desenvolvedores para resolver problemas técnicos importantes, mesmo que eles não sejam diretamente visíveis para o cliente [9]. Alguns dos principais métodos ágeis são *Scrum, Kanban, Lean, Feature Driven Development e Extreme Programming (XP)*. Em 2010, Hasnain [49] apresentou uma pesquisa sobre estudos de desenvolvimento ágil, na qual ele cita que os métodos Scrum e XP incluem a maioria das características e práticas do desenvolvimento ágil.

2.3.3 *Extreme programming*

Extreme Programming (XP), juntamente com outros métodos ágeis, surgiu nos últimos anos como uma abordagem para o desenvolvimento de software [19]. XP é uma disciplina de desenvolvimento de *software* que enfatiza produtividade, flexibilidade, informalidade, trabalho em equipe e o uso limitado de tecnologia fora da programação, trabalhando em ciclos curtos e cada ciclo se iniciam com a escolha de requisitos de um *backlog* [69].

Nos anos 2000, Beck [12] apresentou o XP como uma metodologia leve para pequenas e médias equipes de desenvolvimento de *software* que lidam com requisitos vagos ou que mudam rapidamente, propondo as práticas listadas na Tabela 2.11:

Tabela 2.11: Descrição das práticas do método XP

Nome	Descrição
O jogo de planejamento	Determinar rapidamente o escopo das *releases* combinando prioridades de negócio e estimativas técnicas.
Pequenas *releases*	Colocar um sistema simples rapidamente em produção, e então liberar novas versões brevemente.
Metáfora	Compartilhar com todo o desenvolvimento a história de como o sistema deve funcionar.
Design simples	O sistema deve ser definido da forma mais simples possível e qualquer complexidade extra deve ser removida assim que descoberta.
Testes	Os programadores escrevem testes de unidade e os clientes escrevem testes para funcionalidades finalizadas.
Refatoração	Os programadores reestruturam o código sem mudar o comportamento do mesmo.
Programação em par	Todo o código produzido é escrito com dois programadores em cada máquina.
Propriedade coletiva	Qualquer um pode mudar qualquer código em qualquer lugar e em qualquer tempo.
Integração contínua	Integrar e fazer build do código diversas vezes ao dia.
Semana de 40 horas	Seguir a regra de não trabalhar mais que 40 horas por semana.
Cliente no local	Incluir um usuário real como parte da equipe.
Padrões de código	Os programadores escrevem o código de acordo com as regras enfatizando a comunicação através do código.

Algumas situações onde não é recomendado o uso de XP também foram recomendadas por Beck [12] : i) Quando o cliente ou gerente insistir em completar uma especificação, *design* ou

análise antes de iniciar uma pequena parte da programação; ii) Quando a equipe necessitar trabalhar muitas horas extras; iii) Quando os equipes forem muito grandes; iv) Quando existirem barreiras tecnológicas para manter o código simples e limpo ou quando em um ambiente onde é necessário muito tempo para obter *feedback*; v) Quando o ambiente não promover *collocation* permitindo que todos trabalhem de forma fisicamente próxima.

Existem diferentes métodos ágeis, e de acordo com a pesquisa realizada recentemente pela Version One realizada com 3925 respondentes na América do norte e Europa [90] os métodos e práticas Scrum são utilizados pela maioria da indústria. Mariz *et al.* [25] apresentou um estudo que investiga a relação entre práticas ágeis e o sucesso de projetos utilizando Scrum, através de um levantamento aplicado em 62 engenheiros de *software* associados a 11 projetos de nove empresas diferentes, os resultados mostram que oito de cada 25 atributos associados com as melhores práticas ágeis tem uma correlação significante com sucesso de projetos, sugerindo que é importante se considerar o desenvolvimento ágil como uma maneira de melhorar a eficácia de projetos na indústria de *software*.

2.3.4 Scrum

Scrum é uma abordagem ágil de desenvolvimento de *software* iterativa e incremental, o qual foi apresentado em 1995 por Ken Schwaber em um artigo que descreve a metodologia Scrum [107]. Em 2004, Schwaber [108] relatou que o Scrum é também utilizado em trabalhos complexos, nos quais é impossível prever como tudo vai acontecer. Além disso, o Scrum oferece um *framework* e um conjunto de práticas que mantém tudo visível, permitindo a equipe saber exatamente o que esta acontecendo e fazer os ajustes necessários para manter o projeto indo de acordo com os objetivos desejados. As práticas Scrum são parte de um processo iterativo e incremental e a saída de cada iteração é um incremento no produto e as iterações se repetem até que o projeto não tenha mais orçamento. O Scrum utiliza três artefatos principais: *Product Backlog, Sprint Backlog* e Incremento de um produto potencialmente funcional [108]. O *Product backlog* é uma lista de requisitos, também conhecidos como *user stories* ordenados em prioridade. O *Sprint backlog* é organizado durante o planejamento de cada sprint, onde as user stories são identificadas e trabalhadas por ordem prioridade. As atividades são estimadas em horas pelas equipes [97], entretanto a abordagem do uso de estimativas em horas não é a única opção, pois existem outras como por exemplo pontos de complexidade. Cada iteração do Scrum se chama *sprint* que tem duração de uma a quatro semanas. A equipe tem como base de trabalho o *product backlog*, que é uma lista de requisitos e prioridades.

Em cada Sprint ocorrem reuniões diárias onde cada membro da equipe responde o que foi realizado no último dia, o que será realizado no dia corrente e se tem algum impedimento para mover adiante nas atividades de desenvolvimento. No final de cada Sprint ocorre uma demonstração do produto chamado *Sprint Review* e após isso são tratadas as lições aprendidas na *Sprint Retrospective* [105]. O *framework* Scrum [107] é apresentado na Figura 2.1

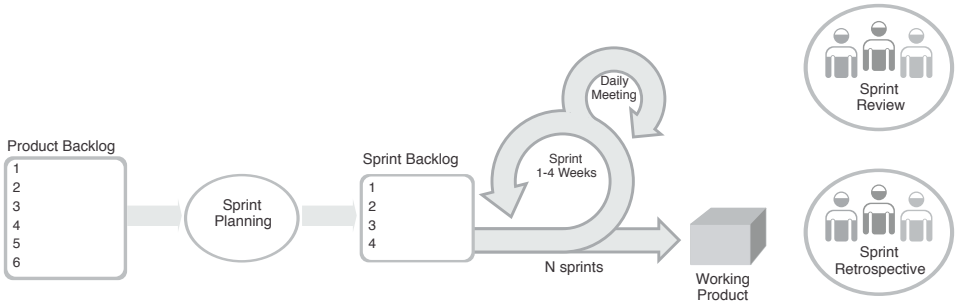

Figura 2.1: Scrum *framework*

Neste contexto, em 2013 Zorzo *et al.* [32] realizaram um estudo de caso descrevendo uma abordagem sobre o uso de Scrum no ensino de engenharia de software, onde o curso inteiro foi planejado para seguir os princípios Scrum, de forma que os estudantes puderam aplicar o desenvolvimento ágil junto com o aprendizado. Após dois anos usando esta abordagem, os autores observaram alguns pontos positivos, como a prática do aprendizado natural por exemplo, e também uma melhor preparação dos estudantes no contexto de desenvolvimento ágil. Como ponto negativo os autores relataram que ocorreram casos de impossibilidade de entrega do produto nas primeiras Sprints, e alguns desafios de interação e colaboração entre os estudantes. O Scrum apresenta as seguintes características:

- Requer iterações curtas (Sprints de 1 a 4 semanas).
- Mudanças são bem-vindas porque o escopo é flexível e pode ser ajustado.
- Rápido *time-to-market* entregando um produto funcional e incremental ao final de cada Sprint.
- Processo leve, passos curtos e *feedback* contínuo onde mal-entendidos são endereçados rapidamente.
- Melhor qualidade porque os defeitos são descobertos antecipadamente como parte de cada Sprint.
- A equipe inteiro é envolvido no processo de tomada de decisão, promovendo visibilidade e entendimento sobre o trabalho a ser realizado.
- Melhora no valor agregado com foco em fazer o que é imediatamente necessário reduzindo perda de tempo em funcionalidades de pouco valor.

Um outro fator também relacionado com projetos Scrum é uma melhora na satisfação das partes interessadas no projeto, porque as partes interessadas podem ser mais envolvidas, influenciando o desenvolvimento do produto e também gerando uma maior afinidade com o projeto [108]. Em 2012, McNely *et al.* [77] apresentaram um estudo etnográfico de 15 semanas, descrevendo como o Scrum mediou as ações diárias de 13 participantes estudados. De acordo com os resultados, eles reportaram

que o Scrum facilita a prática de desenvolvimento coletivo através da intermediação de diferentes dinâmicas de colaboração e aprendizado contínuo.

2.3.5 Método Lean e método *Dynamic Systems Development*

Dentre os métodos ágeis, o método Lean tem um grande destaque, e de acordo com Middleton e Joyce [80] *Lean* surgiu em 1990 como um termo para descrever o processo de produção automotiva desenvolvido desde os anos 50 pela Toyota, o desenvolvimento de *software* Lean é a aplicação dos princípios Toyota de desenvolvimento de produto no desenvolvimento de software. O desenvolvimento de *software* lean provê uma abordagem contemporânea para a engenharia de *software* em conjunto com ferramentas práticas para melhorar eficiência de processo [80] e alguns dos princípios lean são: eliminar o desperdício, expandir a aprendizagem, decidir o mais tarde possível, entregar o mais rápido possível e capacitação da equipe.

Uma outra abordagem ágil se chama DSDM *Dynamic Systems Development Method*. O DSDM oferece um *framework* de controle e melhores práticas para desenvolvimento rápido de aplicações, ele foi criado por um grupo de organizações e teve sua primeira publicação em Janeiro de 1995, e provou ser provou ser extremamente eficaz no fornecimento de sistemas sustentáveis [115]. O DSDM é baseado na premissa que projetos de *software* falham por problemas nas pessoas ao invés de problemas na tecnologia [78].

3. METODOLOGIA DE PESQUISA

Ao tomar a produção do conhecimento no âmbito da Ciência da Computação, percebe-se a complexa tarefa do pesquisador em buscar caminhos para abordar o assunto em estudo. Visto que, o mercado de Tecnologia da Informação (TI) continua em crescente ascensão, o que exige a todo momento, responder rapidamente as mudanças de mercado.

Dentro de um esforço, de ampliar o debate teórico/prático sobre: um método de suporte ao aprendizado no desenvolvimento de aplicativos para dispositivos móveis, através de uma abordagem de ensino baseada em desafios, é que se apresenta o percurso metodológico trilhado no decorrer do processo deste estudo.

3.1 Abordagem Metodológica

Para o desenvolvimento deste estudo, foi realizada uma pesquisa de caráter exploratório com abordagem qualitativa, tendo como objeto de estudo: *"Um Método de Aprendizagem Baseada em Desafios: Um Estudo de Caso em Ambientes de Desenvolvimento de Aplicativos"*. Conforme Sampieri *et al.* [51] os estudos exploratórios têm o objetivo de examinar um tema ou um problema de investigação pouco estudado ou que não tenha sido abordado anteriormente por outros estudos. Para Yin [132] a pesquisa exploratória permite ainda escolher as mais variadas técnicas de coleta de dados adequadas para a pesquisa, decidindo sobre as questões que mais necessitam de atenção e investigação detalhada, podendo ainda, alertar o pesquisador quanto às dificuldades em potencial, sensibilidades e áreas de resistência. No que tange a pesquisa qualitativa, esta utiliza informações enquanto fenômenos que não se restrigem às percepções sensíveis e aparentes, já que estão sujeitas a mudanças e transformações. Logo, os pesquisadores estão particularmente interessados em entender como as coisas acontecem, ao invés de generalizá-las [22].

Neste ensejo, conforme reforça Creswell [22] a pesquisa qualitativa dá conta de uma realidade que não pode ser quantificada, mas aprendida em seu movimento. Logo, caracteriza-se como um processo investigativo, no qual o pesquisador pouco a pouco extrai sentido de um fenômeno contrastando, comparando, replicando, catalogando e/ou classificando o objeto do estudo [22].

Dadas as estratégias de pesquisa e buscando alcançar os objetivos propostos na seção 1.1.1 foi definido um desenho de pesquisa composto por 3 (três) etapas: Etapa 1 (levantamento da base teórica); Etapa 2 (proposta do método inicial e estudo de campo); e Etapa 3 (estudo de caso e método final), conforme ilustra a Figura 3.1.

3.1.1 Etapa 1

Com a finalidade de explorar o referencial teórico existente na área de treinamento e desenvolvimento de aplicativos móveis, optou-se pela realização de um Mapeamento Sistemático da Literatura (MSL), que é apresentado no Capítulo 4.

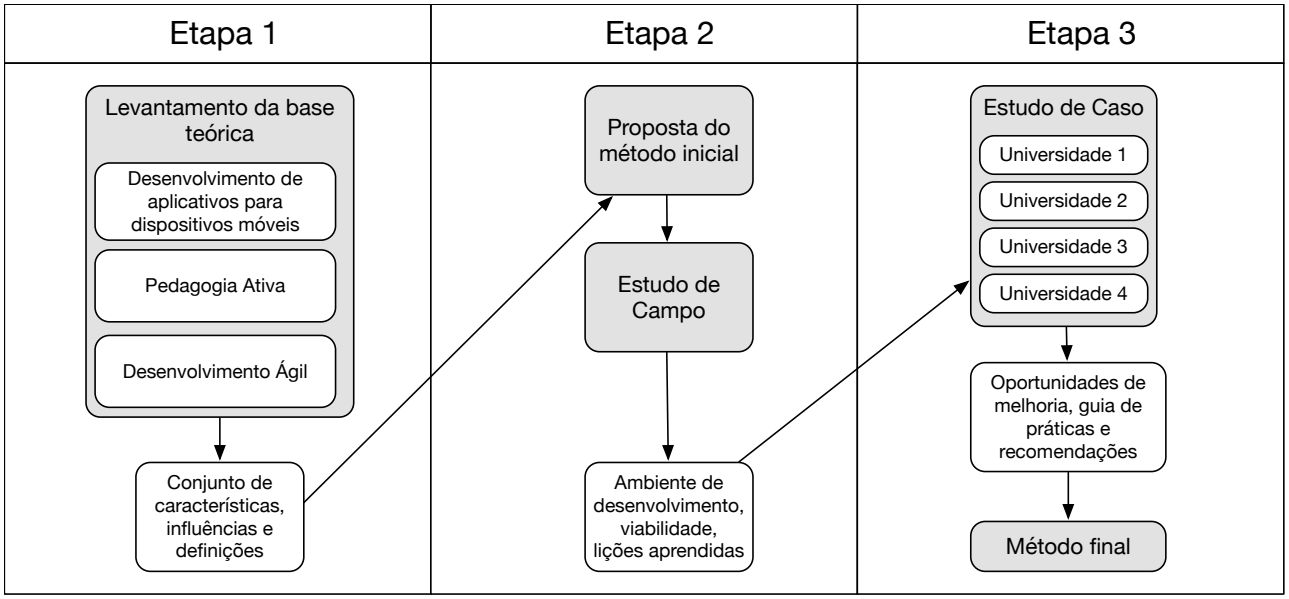

Figura 3.1: Desenho de pesquisa

Segundo Petersen et *et al.* [95], o MSL é um tipo de estudo secundário que visa a realização de um estudo em abrangência dos tópicos de interesse. Logo, o planejamento deste mapeamento é representado pela definição de um protocolo de pesquisa que ajuda a delimitar o escopo da pesquisa. O principal objetivo de um MSL é fornecer uma visão geral de uma área de pesquisa, identificar a quantidade e tipo de pesquisa, bem como os resultados disponíveis dentro destas pesquisas.

Concomitantemente, Kitchenham [64] reforça que a presença deste protocolo ao definir de forma explícita as questões de pesquisa, os critérios de seleção de fontes bibliográficas, os critérios de inclusão e exclusão de artigos, e os critérios de qualidade a serem observados nos estudos, ajuda a amadurecer o problema que vem sendo investigado. Logo, este método permite identificar, obter e consultar a bibliografia e outros materiais úteis para o objetivo do estudo, do qual poderá ser extraído informações relevantes para o problema de pesquisa [22].

Em justa posição, e entendendo que estudos de mapeamento sistemático da literatura em engenharia de software tem sido recomendados especialmente para áreas de pesquisa emergentes e/ou com falta de estudos primários com alta qualidade [95], foi seguido para o desenvolvimento deste estudo as recomendações propostas por Petersen [95], no que tange ao MSL. Tão logo, os construtos teóricos gerados por este método, permitiram uma análise dos conceitos essenciais que contribuíram posteriormente para a elaboração da proposta do método inicial.

3.1.2 Etapa 2

Tomando como base a definição das perspectivas e dados resultantes do MSL proposto na etapa 1, foi definida a proposta do método inicial. Esta proposta, por sua vez, agrupa os principais conceitos da abordagem baseada em desafios e do *framework* scrum, detalhada e apresentada no Capítulo 5.

Definida a proposta do método inicial, optou-se pela realização de um estudo de campo objetivando realizar uma análise sobre a viabilidade do método proposto em ambientes de aprendizado e desenvolvimento de aplicativos móveis.

Os estudos de campo, usualmente não fazem generalizações de resultados, mas permitem que os pesquisadores busquem ilustrar um fenômeno em particular dentro do seu contexto [46]. Além disso, um estudo de campo pode ser utilizado em conjunto com outros métodos de pesquisa [46].

Sob esta perspectiva, o estudo de campo proposto para este estudo, e detalhado no Capítulo 6, centrou-se em analisar o uso do método, de forma a detectar características da amostra, mapear o ambiente de desenvolvimento, ferramentas utilizadas e práticas de desenvolvimento, buscando compreender os fatores que fazem do método inicial uma solução viável e também identificar vantagens e desvantagens, criando assim um corpo de conhecimento sobre o método proposto. Buscando melhorar a confiabilidade e validade deste estudo, um protocolo de pesquisa foi previamente planejado e seguido tanto na aplicação do teste piloto, quanto nos questionários objetivando uniformizar e sistematizar a coleta e análise dos dados [10, 46].

3.1.3 Etapa 3

Com base nas lições aprendidas na Etapa 2, e seguindo as recomendações de Yin [132] foi realizado durante esta etapa um estudo de caso, conforme detalhado no Capítulo 7, com o objetivo de identificar as oportunidades de melhorias, melhores práticas e recomendações de forma a propor o método final.

A opção por utilizar o estudo de caso como referência nesta etapa, se dá em função deste tipo de estudo fazer questionamentos do tipo *"como"* e *"por que"* [132], no que diz respeito a ambientes de aprendizado e desenvolvimento de aplicativos para dispositivos móveis, e por analisar um fenômeno contemporâneo dentro do seu contexto de vida real, especialmente quando as fronteiras entre o fenômeno e o contexto não são claramente evidentes [132].

Dessa forma, os objetivos propostos para a realização do estudo de caso foram: a) aplicar o método inicial proposto em diferentes locais de pesquisa; b) analisar o uso do método inicial proposto; e c) identificar as melhores práticas, oportunidades de melhoria e recomendações sobre o uso do método proposto de forma a desenvolver o método final.

Buscando melhorar a confiabilidade e validade deste estudo, um protocolo de pesquisa foi previamente planejado e seguido tanto na aplicação do teste piloto, quanto nas entrevistas semi-estruturadas objetivando uniformizar e sistematizar a coleta e análise dos dados [22, 132].

Concluído o processo de coleta dos dados da Etapa 2 e da Etapa 3, a exploração do material de análise foi desenvolvida tomando como referência as recomendações de análise de dados apresentadas por Creswell [22] através de seis passos, a saber:

Passo 01 Organizar e preparar os dados para análise, envolve transcrever as entrevistas e digitar anotações de campo.

Passo 02 Ler todos os dados para obter uma percepção geral das informações e refletir sobre o seu significado.

Passo 03 Realizar uma análise detalhada com um processo de codificação, segmentando sentenças ou parágrafos em categorias e rotulando essas categorias com um termo.

Passo 04 Utilizar este processo para gerar uma descrição do local, das pessoas ou temas para análise.

Passo 05 Utilizar passagens narrativas para comunicar os resultados da análise.

Passo 06 Realizar uma interpretação ou extrair um significado dos dados.

Para Creswell [22], estes passos envolvem o pesquisador em um processo sistemático de análise de dados textuais, permitindo assim um entendimento sobre os dados produzidos pela pesquisa, bem como lições aprendidas através da interpretação do pesquisador. Seguindo estas recomendações, os resultados obtidos neste estudo, serão apresentados no Capítulo 7, que compõe este volume.

4. MAPEAMENTO SISTEMÁTICO DA LITERATURA

O Mapeamento Sistemático da Literatura (MSL), ajuda a construir um esquema de classificação, além de estruturar um campo de interesse na Engenharia de Software [95]. Nesse sentido, buscando explorar a base teórica existente na área de treinamento e desenvolvimento de aplicativos móveis, apresenta-se neste capítulo os resultados obtidos com a realização do MSL, seguindo as recomendações definidas por Petersen *et al.* [95]. Na seção 4.1 é apresentado o protocolo utilizado para a realização do MSL, na seção 4.2 apresentam-se os resultados específicos do mapeamento relacionado ao ensino e aprendizado de desenvolvimento de aplicativos, e por fim, na seção 4.3 apresentam-se as considerações tecidas sobre o MSL.

4.1 Protocolo

Com o objetivo de estudar o estado atual do conhecimento sobre treinamento e desenvolvimento de aplicativos móveis, fornecendo assim uma estrutura como a categorização dos tipos de relatórios de pesquisa e resultados que têm sido publicados, esta seção apresenta as principais decisões tomadas durante o MSL, embasados na metodologia supracitada para este estudo.

4.1.1 Questões de pesquisa

Segundo Petersen *et al.* [95], um dos passos iniciais para a realização de um MSL diz respeito a definição das questões de pesquisa para determinar o foco do estudo. Nesse sentido, foram definidas as seguintes questões de pesquisa:

(**RQ1**) *O que a literatura reporta sobre ensino e aprendizado de desenvolvimento de aplicativos?*

(**RQ2**) *Quais fatores tem influência no processo de ensino e desenvolvimento de aplicativos?*

(**RQ3**) *Como o desenvolvimento ágil ajuda o ensino e desenvolvimento de aplicativos?*

Com a RQ1, buscou-se identificar estudos que discutiam o tema: ensino e aprendizado de desenvolvimento de aplicativos para dispositivos móveis, sendo que posteriormente foi elaborado um catálogo com os estudos relacionados ao campo de pesquisa. Na RQ2 buscou-se por fatores que influenciavam positivamente ou negativamente o ensino e desenvolvimento de aplicativos para dispositivos móveis. Para este estudo definiu-se influência positiva quando um fator em particular ajuda a atingir um objetivo de ensino ou aprendizado. Por outro lado, a influência negativa foi definida quando um fator causa um problema nos objetivos de ensino ou aprendizado. Adicionalmente, com a RQ3 buscou-se investigar o desenvolvimento ágil no contexto de ensino e desenvolvimento de aplicativos para dispositivos móveis.

4.1.2 Bases de dados

Buscando responder as questões deste mapeamento, *strings* de busca foram executadas em três bases digitais: *IEEEXplore (http://ieeexplore.ieee.org), ACM Digital Library (http://dl.acm.org/), e Scopus (https://www.scopus.com)*. A duplicação de artigos e o número de publicações não relevantes oriundas de outras bases, contribuíram para a escolha das três bases na condução deste estudo

4.1.3 *Strings* de busca

Neste mapeamento, o idioma padrão utilizado na busca foi a língua inglesa, devido a maior disponibilidade de artigos atualizados na área. Inicialmente as *strings* de busca foram mais restritivas utilizando o filtro (*teaching* OR *learning*) AND (*"mobile software development"* OR *"mobile application development"* OR *"mobile software engineering"*), entretanto este filtro resultou em menos de 100 estudos (incluindo aqui os duplicados), filtro que deixava de incluir artigos de controle. Neste sentido, optou-se por manter as *strings* na forma ampla dos termos removendo (teaching OR learning) de maneira a obter um panorama geral de estudos na área. Dessa forma, as *strings* de busca que guiaram a pesquisa para responder as questões deste estudo foram: *"mobile application development"*, *"mobile software development"* e *"mobile software engineering"*. As *strings* de busca foram estabelecidas combinando a lista de palavras chave com o conector lógico *"OR"*.

4.1.4 Seleção de estudos

Para a inclusão de trabalhos nesta pesquisa foram definidos previamente alguns critérios de seleção: estar disponível de forma online, ser escrito em inglês, e descrever sobre desenvolvimento de software. A busca incluiu artigos publicados em revistas, jornais, conferências e *workshops* publicados entre 2004 e Setembro de 2014, de forma a cobrir os últimos 10 anos de publicação na área. Aplicados os critérios de inclusão, foram removidos os estudos duplicados e aqueles que não atendiam aos critérios de seleção. Quanto a base *ACM Digital Library* esta foi utilizada como fonte primária por contemplar muitos dos estudos disponíveis nas outras bases. Foram retornados das fontes de busca supracitadas 662 (seiscentos e sessenta e dois) trabalhos, sendo extraídos para análise: título, palavras-chave e resumo. Após a primeira iteração foram selecionados 251 (duzentos e cinquenta e um) estudos para leitura completa. Após a leitura completa, 47 (quarenta e sete) estudos relacionados com ensino e desenvolvimento de aplicativos permaneceram na pesquisa sendo selecionados para uma análise completa onde buscou-se identificar as especificidades de cada estudo (descritas na subseção 4.1.6). A Tabela 4.1 apresenta de forma geral a evolução do processo de seleção dos estudos. Na primeira coluna, estão listadas as ferramentas de busca; na segunda coluna, é mostrado o número de estudos científicos retornados durante a primeira etapa; na terceira coluna, é exposto a quantidade de estudos que foram excluídos; e na última coluna da tabela apresenta-se o número de estudos selecionados, segundo os critérios de inclusão e exclusão.

Tabela 4.1: Seleção de estudos

Base de dados	Total resultados encontrados	Não selecionados	Seleção final
ACM Digital Library	291	261	30
IEEEXplore	89	82	7
Scopus	282	272	10
Total	**662**	**615**	**47**

4.1.5 Processo de extração de dados

Para a extração de dados, foi adotado inicialmente o uso de ferramentas como: MS Excel [1] e o catálogo Mendeley [2] que corroboraram no mapeamento de informações gerais sobre o tema apresentado. Metadados como: autor, título, ano, veículo de publicação, nome da fonte e métodos de pesquisa foram coletados e organizados em campos descritivos como: área, classificação de qualidade, e características de cada estudo. Metadados foram coletados e organizados em campos descritivos que englobam: autor, título, ano, veículo de publicação, nome da fonte, método de pesquisa, classificação de qualidade e características de cada estudo.

4.1.6 Processo de classificação de qualidade

Buscando apoiar a extração de dados, conforme apresentado na Tabela 4.2, este estudo adotou o uso de um *checklist* que ajudou a avaliar a qualidade dos estudos baseado em quatro critérios. Este checklist por sua vez, decorre de uma adaptação efetuada a partir de um estudo apresentado por Salleh *et al.* [100]. Logo, foram definidas três escalas para avaliar os critério de qualidade: SIM (vale 1 ponto), PARCIALMENTE (vale 0,5 pontos); e INCONCLUSIVO (vale 0 pontos). A escala SIM, significa que o estudo atinge o critério de qualidade. A escala PARCIALMENTE, significa que o estudo atinge parte do critério de qualidade. A escala INCONCLUSIVO, significa que o artigo não atinge o critério de qualidade ou o critério não fica claro no artigo.

Tabela 4.2: Checklist de avaliação de qualidade de artigos

Critério	Escala
#1: O trabalho é bem/adequadamente referenciado (apresenta trabalhos relacionados/semelhantes e baseia-se em modelos e teorias da literatura)?	( ) Sim ( ) Parcialmente ( ) Inconclusivo
#2: O objetivo da pesquisa é claro?	( ) Sim ( ) Parcialmente ( ) Inconclusivo
#3: O método de pesquisa foi apropriado para alcançar os objetivos da pesquisa?	( ) Sim ( ) Parcialmente ( ) Inconclusivo
#4: Existe uma clara descrição do contexto no qual a pesquisa foi realizada?	( ) Sim ( ) Parcialmente ( ) Inconclusivo

[1] microsoft.office.com/excel
[2] https://www.mendeley.com

No que diz respeito ao processo de classificação da qualidade adotado para este estudo, este contribui por sua vez, para identificar a base conceitual e a relevância dos estudos para a área, como também para a análise dos fatores com influência positiva ou negativa em ambientes e aprendizado de desenvolvimento de aplicativos para dispositivos móveis.

4.1.7 Validade do processo

As principais ameaças de validade do processo são a seleção dos estudos, imprecisão na extração de dados, classificação incorreta dos estudos, métodos de pesquisa e tipos, e um potencial viés do autor. De forma a garantir o processo de seleção, classificação correta e não viés, este mapeamento seguiu as recomendações de Petersen *et al.* [95].

Sobre as classificações dos estudos e resultados, pelo menos dois pesquisadores discutiram cada estudo. Em caso de discordância, o problema era discutido até encontrar um consenso. Entretanto, existe a possibilidade do processo de extração talvez resultar em algum dado incorreto.

4.2 Resultados do Mapeamento

Esta seção apresenta os resultados de cada pergunta relacionada ao MSL. A primeira subseção apresenta resultados relacionados ao ensino e aprendizado de desenvolvimento de aplicativos, a segunda apresenta os fatores que tem influência em ambientes de ensino e aprendizado de desenvolvimento de aplicativos, seguida da subseção que apresenta as contribuições deste mapeamento sistemático e por fim as considerações finais deste MSL.

4.2.1 Ensino e aprendizado de desenvolvimento de aplicativos

Baseado no processo de extração de dados, foram encontrados 47 estudos relacionados com ensino e aprendizado de desenvolvimento de aplicativos para dispositivos móveis, os quais foram classificados inicialmente de acordo com a Tabela 4.3. A maioria dos estudos foram publicados em *Conferências* 59,57%, seguidos de *Revistas* 21,28%, *Simpósios* 10,64%, e *Workshops* 8,51%. Estes estudos discutem os temas de: *Ensino e Aprendizado* 48,94%, seguido de *Ferramentas/Práticas* 27,66% e *Modelos/Frameworks/Metodologias* 17,02%. Também se observou que 6,38% se classificam nas categorias *Ensino e Aprendizado* e *Ferramentas/Práticas*.

Dentre os métodos de pesquisa mais utilizados pelos estudos selecionados *Estudo de caso* representa 40,42%, seguido por uma mistura de estratégias de pesquisa 29,79%, os quais foram classificados como *Outros* (incluindo position papers). Neste sentido, *Revisão da literatura* corresponde a 10,64%, seguido de *Experimentos Controlados* 8,51% e *Survey* 8,51%. Além disso, foi encontrada uma combinação de *Quasi-experiment* e *Survey* em um estudo. Os dados disponíveis na Tabela 4.3 são organizados seguindo a seguinte estrutura:

- **Fonte**: Conferência (C), Revista/Periódico (J), Simpósio (S), e Workshop (W);

- **Categoria**: Modelo/Framework/Metodologia (MF), Ensino e Aprendizado (TL), e Ferramentas/Práticas (T/P);
- **Método de Pesquisa**: Estudo de Caso (CAS), *Survey* (SRV), Experimento Controlado (CTR), Revisão da Literatura (LIT), e Quasi-Experimento (QUE);
- **Classificação**: *Evaluation Research* (ER), *Experience Paper* (EP), *Opinion Paper* (OP), *Philosophical Paper* (PP), *Proposal of Solution* (PS), e *Validation Research* (VR).

O esquema de classificação detalhado no trabalho de Wieringa's [128] foi aplicado para classificar os estudos baseados no tipo de pesquisa (Coluna 5 na Tabela 4.3). Os resultados mostram que a maioria dos estudos são relatórios de experiência, nos quais praticantes relatam suas próprias experiências em determinadas ocasiões.

Tabela 4.3: Classificação dos estudos

Estudo	Fonte	Cat.	Método	Classif.	Critério				Pontos
					#1	#2	#3	#4	
S1 [43]	C	T/P	CAS	ER	1	1	1	1	4
S2 [130]	C	T/P	CAS	EP	0,5	1	0,5	0,5	2,5
S3 [71]	C	TL	Outro	OP	0,5	0,5	0,5	1	2,5
S4 [106]	C	TL	CAS	EP	1	1	0,5	1	3,5
S5 [109]	C	TL	SRV	OP	0,5	0,5	0,5	0,5	2
S6 [70]	C	TL	Outro	PP	0,5	1	0,5	1	3
S7 [87]	W	T/P	CTR	ER	0,5	1	1	1	3,5
S8 [36]	C	TL	Outro	PP	1	1	0,5	1	3,5
S9 [44]	C	T/P	CAS	OP	0,5	1	0,5	1	3
S10 [129]	S	T/P	CAS	ER	0,5	1	0,5	1	3
S11 [39]	J	TL, T/P	CAS	PS	1	1	0,5	1	3,5
S12 [76]	J	TL	CAS	PS	0,5	1	0,5	1	3
S13 [27]	S	TL	CAS	EP	1	1	0,5	1	3,5
S14 [131]	J	TL	Outro	OP	0,5	0,5	0,5	1	2,5
S15 [42]	C	TL	LIT	OP	0,5	0,5	0,5	1	2,5
S16 [75]	J	TL	CAS	ER	1	1	0,5	1	3,5
S17 [96]	C	MF	LIT	PP	1	1	0,5	1	3,5
S18 [82]	C	MF	Outro	PS	0,5	0,5	0,5	0,5	2
S19 [15]	W	TL	SRV	VR	1	1	0,5	1	3,5
S20 [55]	C	T/P	SRV	OP	0,5	1	0	1	2,5
S21 [86]	W	MF	CTR	EP	0,5	1	0,5	1	3
S22 [114]	C	TL	Outro	OP	0,5	0,5	0,5	1	2,5
S23 [24]	C	T/P	Outro	PS	0,5	1	0,5	0,5	2,5
S24 [50]	C	MF	Outro	EP	0,5	1	0,5	1	3
S25 [104]	C	MF	Outro	EP	0,5	1	0,5	0,5	2,5
S26 [117]	C	TL	CAS	EP	1	1	0,5	1	3,5
S27 [120]	C	TL	CAS	EP	1	1	0,5	1	3,5
S28 [124]	C	TL	CAS	EP	0,5	0,5	0,5	1	2,5
S29 [94]	C	MF	CAS	EP	1	1	0,5	0,5	3
S30 [2]	C	MF	CAS	EP	0,5	1	0,5	1	3
S31 [118]	J	MF	SRV, QUE	EP	1	1	1	1	4
S32 [1]	C	T/P	CTR	PS	0,5	1	1	1	3,5
S33 [29]	S	T/P	CTR	PS	0,5	1	0,5	1	3
S34 [14]	J	T/P	CAS	PS	1	1	1	1	4
S35 [62]	C	T/P	LIT	OP	0,5	1	0,5	0,5	2,5
S36 [105]	W	TL	CAS	ER	1	1	1	0,5	3,5
S37 [116]	J	TL	CAS	OP	1	0,5	0,5	0,5	2,5
S38 [127]	C	T/P	LIT	PP	1	1	0,5	0,5	3
S39 [4]	S	TL	CAS	EP	1	1	0,5	1	3,5
S40 [93]	J	TL	Outro	EP	0,5	1	0,5	1	3
S41 [111]	C	T/P	CAS	EP	1	1	0,5	0,5	3
S42 [28]	J	TL, T/P	Outro	EP	0,5	1	0,5	1	3
S43 [7]	C	TL	Outro	PS	0,5	1	0,5	1	3
S44 [112]	J	TL	LIT	OP	0,5	1	0,5	0,5	2,5
S45 [67]	C	TL	Outro	OP	0,5	0,5	0,5	1	2,5
S46 [126]	S	TL, T/P	Outro	EP	1	1	0,5	1	3,5
S47 [6]	C	TL	SRV	EP	0	0,5	0,5	1	2

Na avaliação de qualidade, 17 estudos 36,17% obtiveram uma pontuação maior que 3; estes estudos foram classificados como *qualidade alta*. Vinte e sete estudos 57,45% obtiveram uma pontuação maior que 2 e menor ou igual a 3, estes estudos foram classificados como *qualidade média*. Três estudos 6,38% obtiveram uma pontuação menor que dois e estes estudos foram classificados como *inconclusivos*. É importante ressaltar que esta avaliação não possui viés de classificação quantitativa, uma vez que independente do contexto do artigo ser classificado como Ensino e Aprendizado, Ferramentas/Práticas ou Modelo/Framework/Metodologia a pontuação média continua em torno de 3 para todos os contextos, conforme apresentado na Tabela 4.3.

4.2.2 Fatores com influência no ensino e aprendizado de desenvolvimento

Diferentes fatores com influência no ensino e aprendizado de desenvolvimento de aplicativos foram avaliados. Neste sentido, 20 fatores de sucesso e suas influências foram encontrados e categorizados, conforme Tabela 4.4. As colunas disponíveis na Tabela 4.4 são organizadas utilizando a seguinte estrutura: Fator de Sucesso (SF) e Categoria, onde TL é Ensino e Aprendizado, T/P é Ferramentas/Práticas e MF é Modelo/Framework/Metodologia.

Tabela 4.4: Fatores e Categorias

No.	SF	Efeito			Categoria		
		Positivo	Negat.	Inconcl.	TL	T/P	MF
01	Pedagogia ativa	[4, 36, 39, 75, 104]	-	-	✓	✓	✓
02	Métodos ágeis	[2, 50, 96, 105, 106, 124, 127]	-	-	✓	✓	✓
03	Atitude	[4]	-	-	✓	-	-
04	Geração de código	[27, 28, 82, 109, 126, 130]	-	[29]	✓	✓	✓
05	Colaboração	[76]	-	-	✓	-	-
06	Confiança	[6, 7]	-	-	✓	-	-
07	Paradigma de desenvolvimento	-	-	[43, 55]	-	✓	-
08	Plataforma de desenvolvimento	[42, 44, 86]	-	[14, 87, 111]	✓	✓	-
09	Facilidade de aprender	[1]	-	-	-	✓	-
10	Motivação	[4, 15, 39, 76, 106, 118, 129]	-	-	✓	✓	✓
11	Desempenho	[4, 70, 71]	-	-	✓	-	-
12	Sessões práticas	[7, 93, 112, 116, 120, 131]	-	-	✓	-	-
13	Protótipos	-	-	[24]	-	✓	-
14	Gerenciar riscos	-	-	[62]	-	✓	-
15	*Design* simples	[94]	-	-	-	-	✓
16	Pequenas releases	[94]	-	-	-	-	✓
17	TDD	[94]	-	-	-	-	✓
18	Utilidade	[67, 118]	-	-	✓	-	✓
19	*Whole team*	[94]	-	-	-	-	✓
20	Carga de trabalho	-	[117]	-	✓	-	-

SF01: Pedagogia ativa

De acordo com Dochy *et al.* [31] as mudanças tecnológicas transformam a vida profissional e aumentam os requisitos para os profissionais, não só influenciando programas de treinamentos de funcionários mas também as expectativas sobre a educação superior. Especialistas na área de ensino e aprendizado tem compartilhado que é de senso comum que a aquisição de habilidades de resolução de problemas é um objetivo da educação [31].

Neste contexto, diferentes estudos tem reportado active learning como um fator com influência positiva no ensino e aprendizado de desenvolvimento de aplicativos móveis. Gestwicki e Ahmad [4,39] apresentaram um curso introdutório experimental de Ciência da Computação (CS) focando em dois fatores pedagógicos: O uso de App Inventor para Android, o qual é uma linguagem de programação visual que permite os usuários escreverem apps utilizando uma interface de blocos através arrastando os objetos; e a adoção de *Studio-Based Learning* (uma abordagem de pedagogia ativa) como principal metodologia de ensino. O estudo revelou uma resposta positiva dos estudantes aos fatores pedagógicos e atitudes positivas relacionadas com Ciência da Computação, e os estudantes também demostraram desempenho acadêmico significativo.

Neste sentido, um ambiente pedagógico também foi proposto por Massey *et al.* [75] para influenciar experiências de aprendizado de estudantes como desenvolvedores e tomadores de decisão. O sucesso dos esforços descritos pelos autores deve encorajar outras instituições a usar *Problem-Based learning* (PBL) como um meio de influenciar experiências de aprendizado de estudantes interessados em áreas emergentes de tecnologia como o desenvolvimento de aplicativos para dispositivos móveis. Um estudo sobre PBL foi apresentado por Sarif e Shiratuddin [104] como um método de ensino de prática de desenvolvimento de aplicativos para dispositivos móveis no contexto de um curso de graduação em um módulo de programação para dispositivos móveis. Eles utilizaram uma ferramenta chamada MD-Matrix para ajudar os estudantes a escolherem a metodologia apropriada para o desenvolvimento de seus projetos facilitando o processo de aprendizado.

Em 2009, Fetaji [36] analisou modelos de aprendizado e soluções desenvolvidas e revisões de modelos de aprendizado, focando em *Task-Based learning* TBL, e apresentando indicativos que uma abordagem de pedagogia ativa aplicada através de TBL e inovação tem influência positiva no processo de desenvolvimento de aplicativos móveis.

SF02: Métodos ágeis

Um modelo de ensino de desenvolvimento de aplicativos móveis foi apresentado por Scharff *et al.* [106], enfatizando práticas de engenharia de software, compartilhando como lições aprendidas que estudantes trabalhando em projetos e utilizando desenvolvimento ágil apresentam uma melhora no engajamento e motivação. Além disso, Scharff e Verma [105] relatam que o uso de Scrum é uma das razões de sucesso de projetos quando tempo é considerado uma limitação e o desenvolvimento de aplicativos móveis é aprendido *just-in-time*.

Os desafios no desenvolvimento e aplicativos móveis foram examinados por Rahimian e Ramsin [96], onde os autores indicam que o uso de desenvolvimento ágil como apropriado para o desen-

volvimento de aplicativos para dispositivos móveis e eles apresentaram um modelo Híbrido. Neste sentido, Wasserman [127] publicou uma visão geral das questões de pesquisa de engenharia de *software* importantes relacionadas com o desenvolvimento de aplicativos que são executados em dispositivos móveis, citando que os desenvolvedores de aplicativos para dispositivos seguem processos de desenvolvimento ágil.

Uma abordagem de desenvolvimento ágil voltada para o desenvolvimento de aplicativos para o mercado foi apresentada por Heredia *et al.* [50], onde o processo foi seguido durante dois anos e o seu uso levou o desenvolvimento com êxito de diversos aplicativos para o mercado. Uskov [124] apresentou um estudo colaborativo *"estudante-universidade"*, através do *design* e desenvolvimento de um projeto onde estudantes pesquisaram exemplos do mundo real e suas melhores práticas para diferentes métodos ágeis, pesquisando metodologias como por exemplo (*e.g.*, Scrum [107]), a qual foi apresentada como uma das opções para suporte ao desenvolvimento de aplicativos.

Existem diferentes estudos sobre o uso de desenvolvimento ágil e desenvolvimento de aplicativos para dispositivos móveis. Em 2013, Harleen e Swati [37] realizaram uma revisão e análise de desenvolvimento de aplicativo para dispositivos móveis utilizando ágil, e existem estudos que recomendam desenvolvimento ágil como uma boa opção para abordar as diferentes fases do ciclo de desenvolvimento de aplicativos para dispositivos móveis, os seguintes processos de desenvolvimento foram avaliados: Mobile D, RaPiD 7, Hybrid Methodology Design, MASAM e SLeSS [37].

A metodologia tem um papel muito importante em ambientes de desenvolvimento de aplicativos para dispositivos móveis, porque os aplicativos mudam e evoluem constantemente baseados em necessidades de usuários imediatas [63]. Dessa forma, Kaleel e Ssowjanya [63] estudaram práticas ágeis e Scrum que melhor atendem requisitos de desenvolvimento de *software* android e aplicaram estas práticas em uma metodologia de desenvolvimento de software, na qual eles conseguiram desenvolver com sucesso um aplicativo de segurança utilizando importante práticas de desenvolvimento ágil e Scrum, como por exemplo adaptabilidade para requisitos em constante evolução, equipes de desenvolvimento com forte base técnica e comunicação efetiva através de reuniões diárias.

SF03: Atitude

Ahmad e Gestwicki [4] relataram atitude como um fator positivamente influenciado pelo aprendizado de desenvolvimento de aplicativos para dispositivos móveis. O estudo apresenta um curso introdutório experimental de Ciência da Computação, utilizando *App Inventor*. Os resultados mostraram atitudes muito positivas referentes ao aprendizado no campo de CS.

SF04: Geração de código

Seis estudos [27, 28, 82, 109, 126, 130] relacionaram a geração de código através de abordagens multi-plataforma como uma influência positiva no desenvolvimento de aplicativos. Dickson [27] introduziu um estudo sobre o Cabana, uma abordagem multi-plataforma que habilita um rápido *design* de interface e desenvolvimento de aplicativos sem a curva de aprendizado mais tradicional de construção de aplicativos. O uso desta abordagem foi descrito em duas disciplinas diferentes para desenvolver aplicativos que podem rodar em múltiplas plataformas como um fator positivo. Neste

sentido, Xanthopoulos e Xinogalos [130] apresentaram uma pesquisa com foco nas tendências atuais de desenvolvimento de aplicativos, descrevendo que a implementação de aplicativos interpretados é uma solução promissora.

Um relatório de experiência também utilizando o *App Inventor* foi apresentado por Wagner *et al.* [126] através de observações em um curso de curta duração de ensino técnico. De acordo com os autores, o uso do App Inventor ajudou como vantagem no aumento do interesse que os alunos possuem por smartphones. Sewell e Ringenberg [109] também compartilharam experiências sobre o uso da ferramenta para ensinar princípios clássicos de Ciência da Computação, utilizando levantamentos de diferentes experiências os autores relataram que o uso de desenvolvimento de aplicativos móveis para o ensino de conceitos introdutórios de Ciência da Computação resultou em *feedback* positivo dos estudantes.

Uma IDE (*Integrated Development Environment*) com abordagem *model-driven development*, que pode apoiar a criação de interfaces de usuários para aplicativos destinados a diferentes plataformas de dispositivos móveis também foi apresentado como um caminho que ajuda a salvar tempo de desenvolvimento devido ao fato de criar um modelo abstrato do domínio de negócio para facilitar as atividades de desenvolvimento [29].

Neste contexto, Dickson [28] apresentou um estudo sobre um curso de desenvolvimento de aplicativos para dispositivos móveis utilizando o Cabana. De acordo com o autor, o curso foi um sucesso para estudantes com variedade de níveis de experiência em programação. Além disso, um gerador independente de aplicativos para dispositivos móveis foi proposto por Miravet *et al.* [82] com o intuito de gerar aplicativos para múltiplas plataformas. Os resultados mostraram que o *framework* proposto é viável como uma solução para fragmentação de dispositivos na área de desenvolvimento de aplicativos móveis.

SF05: Colaboração

Colaboração também foi citada como um fator que influencia positivamente a motivação de estudantes, pois a colaboração entre as equipes oferece oportunidades para auto-crítica e auto-aperfeiçoamento, bem como de avaliação pelos pares [76].

SF06: Confiança

Dois estudos [6,7] relatam confiança como um fator positivamente influenciado pelo aprendizado de desenvolvimento de aplicativos para dispositivos móveis. Alonso *et al.* [6] criou um curso de uma semana para universitários desenvolver aplicações para a plataforma Android do Google. Dez estudantes com muito pouca ou nenhuma experiência em programação, desenvolveram e testaram aplicativos na plataforma Android. Após o desenvolvimento os estudantes responderam uma Survey na qual eles relataram um aumento no nível de confiança, 100% dos estudantes relataram de uma mudança pequena a uma grande mudança no aumento do nível de confiança que eles podem se sobressair no desenvolvimento de aplicativos.

Além disso, Alston [7] compartilhou experiências na entrega de um módulo de desenvolvimento de aplicativos para dispositivos móveis com estudantes de graduação e também os desafios envolvidos

em desenvolver um currículo alternativo. As avaliações do módulo indicam que o currículo alternativo aplicado nos estudantes foi bem recebido que os hands-on práticos e os tutoriais utilizados na entrega do módulo deram aos alunos um sentimento de capacitação e a confiança que precisam para ter sucesso.

SF07: Paradigma de desenvolvimento

Em 2012, Huy e Do vanThanh [55] realizaram uma pesquisa para oferecer *guidelines* sobre quais paradigmas de desenvolvimento são mais aplicáveis para desenvolvimento de aplicativos para dispositivos móveis. Após análise e avaliação, os autores concluíram que aplicativos nativos e aplicativos utilizando HTML5 figuravam as primeiras opções como paradigma de desenvolvimento. Os resultados indicam que a escolha final de um paradigma de desenvolvimento de aplicativos móveis vai depender das preferências de contexto de aplicativos e desenvolvedores.

Experimentos utilizando abordagens de desenvolvimento baseado na Web foram descritos por Granlund *et al.* [43], estudo no qual reportaram que este tipo de abordagem talvez tenha mais benefícios que a utilização de desenvolvimento nativo porque atinge um maior número de usuários com uma única implementação. Entretanto mesmo que com este tipo de abordagem não seja necessário lidar com as diferentes plataformas de desenvolvimento, este tipo de tecnologia ainda não está madura ao mesmo tempo que ainda falta um suporte de todos os browsers disponíveis no mercado.

SF08: Plataforma de desenvolvimento

Quatro estudos relataram que as plataformas de desenvolvimento ajudam a produzir uma influência positiva no desenvolvimento de aplicativos para dispositivos móveis. Morten *et al.* [44] realizou uma pesquisa para comparar plataformas de desenvolvimento em diferentes categorias. Utilizando a implementação de uma versão aplicativo de um jogo *tic-tac-toe* em quatro diferentes plataformas, s autores reportaram diferenças quando avaliaram ambientes de comunidade, habilidades de *hardware* e maturidade de plataforma. Neste sentido, eles reportaram que a geração de código tem uma influência positiva quando é necessário desenvolver aplicativos que precisam rodar em diferentes plataformas.

Gordon [42] realizou uma pesquisa onde estudantes desenvolveram para a plataforma android usando App Inventor e Java. De acordo com o autor, programas de CS devem ensinar disciplinas que cobrem desenvolvimento de aplicativos para dispositivos móveis e estas disciplinas devem enfatizar os conceitos agrupados em seis categorias: Interface de Usuário, Relação entre dispositivos, Desafios de hardware, Gerenciamento de Dados, Interação entre aplicativos e Desafios de programação. Um experimento com quatro desenvolvedores foi conduzido por Nebeling *et al.* [86], neste estudo os autores analisaram diariamente os fatores comuns e as diferenças em desenvolver aplicativos para dispositivos móveis utilizando abordagens nativa e abordagens web, incluindo diferentes tipos de atividades. De acordo com os autores, a produtividade aumenta quando ao utilizar abordagens web pela possibilidade de atingir diferentes plataformas com o mesmo aplicativo.

Outro tipo de plataforma de desenvolvimento foi apresentado por Nguyen *et al.* [87] através de um experimento em sujeitos independentes para compara a produtividade de um programador utilizando TouchDevelop (uma plataforma que fornece a habilidade de desenvolver o código no próprio dispositivo móvel como plataforma) comparando com uma abordagem mais tradicional de desenvolvimento de aplicativos para dispositivos móveis. Os autores reportaram que para tarefas pequenas um programador que não tem experiência anterior em Java ou android é mais produtivo escrevendo código utilizando o TouchDevelop do que utilizando Android em um PC. Neste contexto, Simon e Cornforth [111] apresentaram um estudo sobre o uso de TouchDevelop para ensinar desenvolvimento de aplicativos para dispositivos móveis, e de acordo com as percepções dos estudantes participantes no estudo o curso foi claramente envolvente, e os alunos apreciaram a oportunidade de desenvolver aplicativos para um dispositivo móvel.

Além disso, um estudo sobre o desenvolvimento e distribuição de aplicativos para dispositivos móveis foi descrito por Bergvall-Kareborn e Howcroft [14], os autores compartilharam que as plataformas de desenvolvimento móvel existentes tem proporcionado benefícios consideráveis para alguns desenvolvedores, entretanto os autores relataram que plataformas de crowdsourcing oferecem uma change para o empreendedorismo. Entretanto os autores não disponibilizaram uma direção clara em qual plataforma deve ser utilizada, classificando este estudo como inconclusivo.

SF09: Facilidade de aprendizado

Abadi *et al.* [1] estudaram a funcionalidade e a usabilidade do NitroGen com estudantes de Ciência da Computação (CS) no terceiro ano de graduação, relatando que os participantes que conhecem ferramentas e técnicas para desenvolvimento de aplicações web ainda não têm experiência com o desenvolvimento de aplicações móveis. As percepções dos estudantes participantes no estudo foi que eles gostaram de utilizar a ferramenta NitroGen e também acharam que isto facilitou o aprendizado melhorando o trabalho em ambientes de desenvolvimento de aplicativos para dispositivos móveis.

SF10: Motivação

Seis estudos [4, 15, 76, 106, 118, 129] relatam que a motivação dos estudantes é positivamente impactada pelo aprendizado de desenvolvimento de projetos de aplicativos para dispositivos móveis. Em 2010, Matos e Grasser [76] compartilharam uma experiência utilizando a plataforma Android. A experiência foi descrita e compartilhada através de um guia para organizar um curso de desenvolvimento para dispositivos Android. Os autores reportaram que a motivação no curso foi gerada através da novidade e oportunidade de inovação e que a produção de aplicativos para o mercado ajuda a melhorar a motivação dos estudantes. Uma survey foi realizada em 2012 por Burd *et al.* [15] em aproximadamente 200 cursos que cobriam algum aspecto de computação móvel, este estudo discutiu a influência positiva que o desenvolvimento para dispositivos móveis gera na motivação dos estudantes, além disso este estudo fornece um guia de escolha de plataformas para os educadores. Um outro estudo [4] apresentou indicativos que o desenvolvimento móvel aumenta a motivação dos estudantes. Neste sentido, Wolber [129] descreve o uso do App Inventor com os estudantes criando aplicativos simples mas reais, motivando-os a resolver problemas de lógica como um segundo passo.

Em 2014, Sykes [118] através de seu estudo fornece uma visão geral do ambiente de desenvolvimento iOS e uma avaliação de um curso, incluindo pesquisas qualitativas, observações informais e uma análise quantitativa envolvendo resultados de pontuação de desempenho dos alunos, descrevendo o aproveitamento e a utilidade de acordo com a opinião dos estudantes. Quando os estudantes trabalham em projetos que tem impacto na sociedade a motivação deles aumenta [106].

SF11: Desempenho

Três estudos [4, 70, 71] apresentam resultados sobre a influência positiva do aprendizado de desenvolvimento de aplicativos para dispositivos móveis no desempenho dos estudantes.

Qusay *et al.* [70] propôs um kit escolar para ajudar as universidades na integração de dispositivos móveis no currículo de Ciência da Computação. O kit aplica uma abordagem de utilizar o desenvolvimento móvel para ensinar conteúdo de Ciência da Computação com foco em desenvolvimento de aplicativos Java ME e BlackBerry. De acordo com os autores esta abordagem ajudou os estudantes a atingirem bons resultados no seu desempenho e satisfação. Em outro estudo, Ahmad e Gestwicki [4] também apresentaram resultados que indicam que o uso de desenvolvimento de aplicativos para dispositivos móveis ajuda a melhorar o desempenho dos estudantes.

Adicionalmente, Qusay e Popowicz [71] apresentaram uma abordagem de introdução ao desenvolvimento de aplicativos para dispositivos móveis nos estágios iniciais do currículo de Ciência da Computação, mostrando que esta abordagem pode melhorar o desempenho de algumas habilidades e a satisfação dos estudantes de Ciência da Computação.

SF12: Sessões práticas de desenvolvimento

Cinco estudos [7,93,112,116,120] apresentam sessões práticas como influência positiva no ensino e aprendizado de desenvolvimento de aplicativos para dispositivos móveis. Teng e Helps [120] realizaram um projeto para um curso nível júnior de sistemas operacionais e solicitaram aos estudantes para desenvolver um aplicativo utilizando alguma das principais plataformas: Apple iOS, Microsoft Windows Phone e Google Android. Os estudantes poderiam escolher uma das plataformas para definir os seus aplicativos, os estudantes reportaram que as sessões praticas tiveram uma influência positiva na sua experiência de aprendizado. Neste sentido, Payne [93] realizou um estudo onde um único curso de desenvolvimento de aplicativos é viável e onde o acesso para ambas plataformas PC e MAC era disponível e suportado, reportando que um curso de desenvolvimento nativo iOS ou Android pode dar aos estudantes amplitude de experiência em duas ou mais plataformas, permitindo simultaneamente uma profundidade razoável em uma única plataforma

Problemas e desafios no desenvolvimento de aplicativos para dispositivos móveis foram descritos por Stringfellow e Mule [116] utilizando aplicativos para smartphone como projetos em um curso de engenharia de software. De acordo com os relatórios de experiência dos estudantes, o projeto ensinou-lhes uma grande quantidade de conteúdo técnico e não técnico, os quais são tão importantes na sala de aula como também no mundo real. Xu [131] apresentou um conceito chamado Classroom Flip para o ensino de desenvolvimento de aplicativos para dispositivos móveis como uma forma de diversificar as formas de aprendizado. O conceito se baseia em um curso prático, e de acordo com os

resultados do estudo este tipo de abordagem gera um resultado positivo além de que os estudantes melhoram suas habilidades no desenvolvimento de aplicativos para dispositivos móveis.

Neste contexto, experiências na entrega de módulo de desenvolvimento de aplicativos para dispositivos móveis com estudantes de graduação foi apresentada por Alston [7] mostrando também os desafios envolvidos em desenvolver um currículo alternativo. As avaliações do módulo indicam que os estudantes receberam bem as sessões práticas e tutoriais utilizados durante o módulo, melhorando a sensação de empoderamento e confiança que os estudantes precisam para ter sucesso. Além disso, Skelton *et al.* [112] descreve os esforços para ensinar conceitos avançados de engenharia de *software* na Jackson State University através de um laboratório de desenvolvimento de aplicativos para dispositivos móveis, estudo o qual descreve o uso de sessões práticas como uma influência positiva no aprendizado.

SF13: Protótipo

Sa e Carriço [24] apresentaram um estudo sobre os conceitos de usuário com o foco em metodologias de *design* aplicadas ao desenvolvimento móvel, e de acordo com os resultados do estudo o engajamento da equipe de *design* com protótipos nos estágios iniciais deve ser considerado como um fator que contribui para o sucesso no processo de desenvolvimento de aplicativos para dispositivos móveis.

SF14: Gerenciar riscos

Kakkar *et al.* [62] estudaram os desafios no desenvolvimento tradicional de aplicativos e apresentaram diferentes aspectos sobre gerenciamento de risco em ambientes de desenvolvimento de aplicativos para dispositivos móveis, onde o gerenciamento de risco é apontado como um fator importante para gerenciar as constantes mudanças e desafios em projetos de desenvolvimento de aplicativos.

SF15: *Design* simples

Um estudo sobre uma visão de processo no desenvolvimento de aplicativos para dispositivos móveis, explorando as vantagens de limitações do uso de XP foi apresentado por Pedersen *et al.* [94]. Seguindo uma abordagem experimental de projeto durante quatro meses eles descrevem que simple *design* tem uma influencia e impacto positivo em projetos de desenvolvimento de aplicativos para dispositivos móveis.

SF16: Pequenas releases

Pequenas *releases* ajudam a equipe de desenvolvimento a entregar com frequência versões iterativas e incrementais de software. Isto ajuda a manter o cliente e os desenvolvedores trabalhando juntos de forma a compartilhar e resolver problemas encontrados durante o desenvolvimento [94].

SF17: TDD - Test Driven Development

TDD é uma abordagem de desenvolvimento de software, na qual unidades de *software* são desenvolvidas em pequenas partes. Uma pequena parte do código contribui para verificar os requisitos

de funcionalidades implementada, gerando um impacto positivo em projetos de desenvolvimento de aplicativos [94].

SF18: Utilidade

Liu [67] apresentou um curso e suas reflexões sobre o uso de uma abordagem orientada ao desenvolvimento no ensino de desenvolvimento de aplicativos para dispositvos móveis. O autor reportou comparações, vantagens e desvantagens entre o uso de ensino tradicional e uma abordagem orientada ao desenvolvimento, relatando que a utilidade da prática dos estudantes na resolução de problemas ainda é a forma preferida pelos alunos de cursos de programação. Neste sentido, Sykes [118] também relatou utilidade como um fator de impacto positivo para o engajamento dos estudantes em ambientes de ensino e aprendizado de desenvolvimento de aplicativos para dispositivos móveis.

SF19: *Whole team*

Whole team é uma estratégia de projeto na qual a equipe compartilha entre si os objetivos de projeto e a responsabilidade para atingi-los. Pedersen *et al.* [94] descreve esta estratégia como promissora em ambientes de desenvolvimento de aplicativos para dispositivos móveis.

SF20: Carga de trabalho

Um estudo relatou a influência negativa da carga de trabalho durante o aprendizado de desenvolvimento de aplicativos. De acordo com Sung e Samuel [117] durante a pesquisa de um novo curso de desenvolvimento de aplicativos para dispositivos móveis baseado nas necessidades dos estudantes e o estado atual da tecnologia, eles relatam que uma carga de trabalho muito pesada dificulta que os estudantes tenham tempo para entender e apreciar detalhes importantes no desenvolvimento de aplicativos.

4.2.3 Influência do desenvolvimento ágil no ensino e aprendizado de desenvolvimento de aplicativos

Alguns estudos discutem o uso de práticas ágeis no ensino e desenvolvimento de aplicativos para dispositivos móveis [30, 44, 106, 109]. Estes estudos relatam que as práticas ágeis ajudam no ensino de desenvolvimento de aplicativos através de diferentes fases do ciclo de vida de desenvolvimento. Além de que outros estudos também foram conduzidos para avaliar a eficácia do uso de práticas ágeis em sala de aula [105, 124].

Existem estudos que discutem a adoção de métodos ágeis em ambientes de ensino de desenvolvimento de aplicativos para dispositivos móveis. Uskov [124] realizou uma pesquisa colaborativa para modelar um currículo de engenharia de *software* para desenvolvimento de aplicativos. Neste estudo os estudantes realizaram uma pesquisa de exemplos reais e melhores práticas de desenvolvimento ágil como opções para dar suporte ao desenvolvimento de aplicativos. No mesmo sentido, Scharff e Verma [105] apresentaram uma pesquisa que avalia o uso de Scrum com sucesso em ambientes de desenvolvimento de aplicativos para dispositivos móveis.

No contexto de ferramentas e práticas para ambientes de desenvolvimento de aplicativos utilizando métodos ágeis, Pedersen *et al.* [94] relatam que o uso de pequenas *releases*, design simples e TDD tem impacto positivo nos projetos de desenvolvimento de aplicativos.

Existem outros estudos mencionados anteriormente [2,50,96] que discutem desenvolvimento ágil em ambientes ambientes de desenvolvimento de aplicativos para dispositivos móveis. Heredia *et al.* [50] apresentou um processo de desenvolvimento ágil voltado para o desenvolvimento de aplicativos, tentando medir a agilidade do processo através das atividades que eram realmente implementadas pelas equipes de desenvolvimento. Além do mais, Rahimian e Ramsin [96] examinaram os desafios em ambientes de desenvolvimento de aplicativos e relataram que métodos ágeis são a melhor abordagem para ambientes de desenvolvimento de aplicativos, relatando que este tipo de abordagem pode facilitar a aplicação de conceitos de engenharia de *software* para produzir aplicativos para dispositivos móveis. Neste sentido, o desenvolvimento ágil pode ser utilizado em conjunto com métodos de pedagogia ativa onde estudantes desenvolvem projetos do inicio ao fim. Esta abordagem pode melhorar a experiência dos estudantes além das sessões práticas.

4.2.4 Contribuições do estudo de mapeamento sistemático da literatura

Uma análise quantitativa dos estudos selecionados foi conduzida em diferentes contextos exemplo:, *Ensino e Aprendizado*, *Ferramentas/Práticas*, *Modelo/Framework/Metodologia*, e foi encontrado que 48,94% dos estudos 23 de 47) estavam relacionados com Ensino e Aprendizado, seguido de Ferramentas/Práticas 27,66%, 13 de 47) e Modelo/ Framework/Methodologia 17,02%, 8 de 47). Três estudos apresentaram uma mistura de Ensino e Aprendizado e Ferramentas/Práticas 6,38%.

De forma a observar as correlações entre os atributos dos estudos e sua avaliação de qualidade, foi aplicado um algoritmo de associação de data mining [66] para detectar as regras de associação mais relevantes. Este algoritmo processa relevância estatística entre todas as informações descritas dos estudos analisados (Tabela 4.3). Dentre as saídas do processamento do algoritmo duas correlações fortes foram encontradas e são descritas a seguir:

Foi encontrado uma correlação de 91% 10 de 11 entre a qualidade média (*i.e.*, qualidade média da pontuação final na Tabela 4.3) e *Opinion Papers* [42, 44, 55, 62, 67, 71, 109, 112, 114, 116, 131]. De fato, destes 11 estudos, somente Sewell e Ringenberg [109] tem uma classificação inconclusiva, uma vez que para este estudo os quatro critérios foram parcialmente satisfeitos (Tabela 4.2).

A segunda correlação relevante foi encontrada entre estudos de *Ensino e Aprendizado* descrevendo Estudos de caso [4,27,75,76,105,106,116,117,120,124] que tendem a ter mais contribuições de qualidade. Sete destes dez estudos 70% foram classificados como alta qualidade, e os três estudos restantes [76,116,124] continuam entregando uma qualidade média de acordo com a avaliação realizada neste estudo.

Desenhando um paralelo entre estes dois subconjuntos de estudos mencionados, (a) *Opinion Papers* com qualidade média e (b) Ensino e Aprendizado descrevendo Estudos de Caso com alta qualidade, foram observados diferentes comportamentos de acordo com as respostas dos quatro critérios mencionados na Tabela 4.2. Tabela 4.5 apresenta resultados médios de cada critério de

cada subconjunto. Observando a Tabela 4.5 existe um comportamento similar entre os critérios #3 e #4, mas um comportamento muito distinto para os critérios #1 e #2. Portanto, a diferença de qualidade encontrada para os estudos nestes dois subconjuntos é basicamente resultado de falta de definições base nas referências dos artigos (critério #1) e definição de objetivo (critério #2).

Tabela 4.5: Resposta média para cada critério

Subconjunto	**#1**	**#2**	**#3**	**#4**
(a)	0,55	0,68	0,45	0,82
(b)	0,90	0,90	0,55	0,90

Ambientes de aprendizado poderosos tem como objetivo desenvolver uma configuração educacional onde o aprendizado os estudantes seja o objetivo principal e a instrução é definida com base na concepção construtivista da aprendizagem e suas aplicações pedagógicas [31]. Motivação foi encontrada como fator positivo em diversos estudos, seguido de desenvolvimento ágil, pedagogia ativa e sessões práticas. Além disso estudos relatam que uso de desenvolvimento ágil, incluindo Scrum tem afetado positivamente ambientes de desenvolvimento de aplicativos para dispositivos móveis [2, 50, 94, 96, 105, 124]. Neste contexto, Rahimian e Ramsin [96] identificaram alguns dos requisitos de desenvolvimento de aplicativos com abordagens híbridas. Outros estudos apresentaram propostas *e.g.*, Abrahamsson *et al.* [2] apresentou um estudo de caso da abordagem chamada Mobile-D. Pedersen *et al.* [94] reportou uma experiência com foco em XP que pode ser considerada uma adição com outros estudos relacionados tais como Scharff [105].

Realizando uma análise dos 20 fatores de sucesso, motivação 14,89% é relatada como uma influência positiva, seguida por Desenvolvimento ágil 4,89%, Pedagogia ativa 12,76%, Geração de código 14,89% e Sessões práticas 12,76%. Os demais fatores tem uma frequência de até 5%. Além disso seis fatores principais possuem impacto positivo, tais como: Atitude, Colaboração, Confiança, Facilidade de aprendizado, Motivação e Desempenho. No total os estudos indicam 19 fatores como influência positiva no aprendizado e desenvolvimento de aplicativos, especialmente Pedagogia Ativa, Desenvolvimento Ágil e Sessões Práticas. Estes resultados nos guiaram a propor um modelo conceitual para ambientes de aprendizado e desenvolvimento de aplicativos conforme apresentado na Figura 4.1.

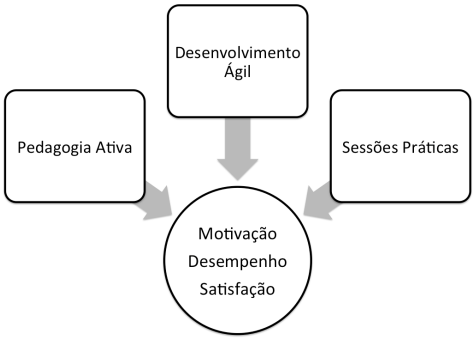

Figura 4.1: Modelo conceitual da integração de pedagogia ativa e práticas ágeis

Este modelo conceitual será utilizado como guia na construção da proposta inicial do método de integração. As ideias fundamentais da aprendizagem baseada em problemas reais têm gerado uma grande variedade de estruturas de aprendizagem, incluindo *Problem-Based Learning* [31], *Project-Based Learning* [119], *Studio-Based Learning* [4], *Task-Based Learning* [36], e *Challenge-Based Learning* [58], entre outros. Este mapeamento sistemático da literatura revisou estes métodos, os quais colocam o estudante como o centro do processo de aprendizado e foca nos estudantes trabalhando juntos para descobrir o conhecimento, trabalhando suas habilidades, ao invés de adquirir conhecimento através de métodos tradicionais e diretos de um instrutor/professor. Estas abordagens de ensino recentemente têm sido defendidas como uma maneira melhor de preparar os alunos para um ambiente de trabalho moderno fluido e dinâmico. Estudos de abordagens experimentais de aprendizado tem demonstrado que os estudantes adquirem habilidades de trabalho mais sólidas quando comparados ao modelo de ensino tradicional [49].

4.3 Considerações Finais do Mapeamento Sistemático

Existem estudos sobre pedagogia ativa através das diferentes abordagens previamente citadas, PBL, PRBL, SBL, CBL, TBL entre outros [4, 31, 36, 58, 119]. Além disso, estudos mostram que o desenvolvimento ágil influencia positivamente ambientes de desenvolvimento de aplicativos para dispositivos móveis [30, 44, 105, 106, 109, 124].

Existem indicativos que motivação é influenciada positivamente em ambientes de desenvolvimento de aplicativos. De acordo com os estudos selecionados, Sessões Práticas, Pedagogia Ativa e Desenvolvimento ágil são contribuintes na influência positiva do fator motivação. Sobre o ponto de vista de ambientes de desenvolvimento, alguns estudos [93, 111, 130] relataram diferentes formas de desenvolver aplicativos, tais como ambientes nativos, web e híbridos. Neste contexto, foram verificados estudos que usam geração de código através de abordagens multi-plataforma no desenvolvimento de aplicativos [27, 29, 130]. Na avaliação de qualidade dos estudos selecionados, existem relações entre atributos dos estudos e sua qualidade, encontramos uma correlação de 91% entre a qualidade média dos estudos e *Opinion papers*.

Baseados nos dados e discussões apresentados neste MSL, encontramos uma oportunidade de trabalho futuro para analisar empiricamente a expansão do modelo proposto através de um método aplicado em ambientes de ensino e desenvolvimento de aplicativos para dispositivos móveis.

5. PROPOSTA INICIAL DO MÉTODO

Neste capítulo é apresentada a proposta inicial do método, conforme definido na subseção 3.1.2 da metodologia de pesquisa. A proposta inicial do método foi construída baseada nas principais características e fatores identificados na Etapa 1 da pesquisa. Na seção 5.1 é apresentada a estrutura da abordagem baseada em desafios, na seção 5.2 é apresentada a estrutura do *framework* scrum, na seção 5.3 é apresentada a estrutura da proposta inicial do método, e na seção 5.4 (pág. 78) é apresentado um guia base para uso do método.

5.1 Estrutura da abordagem baseada em desafios

As principais características que diferenciam o uso de abordagem baseada em desafios comparado com outras abordagens de pedagogia ativa são a facilidade de uso em ambientes ricos em tecnologia e a simulação de um ambiente de trabalho. O *framework* CBL [58] foi apresentado em 2009 conforme descrito na Figura 5.1.

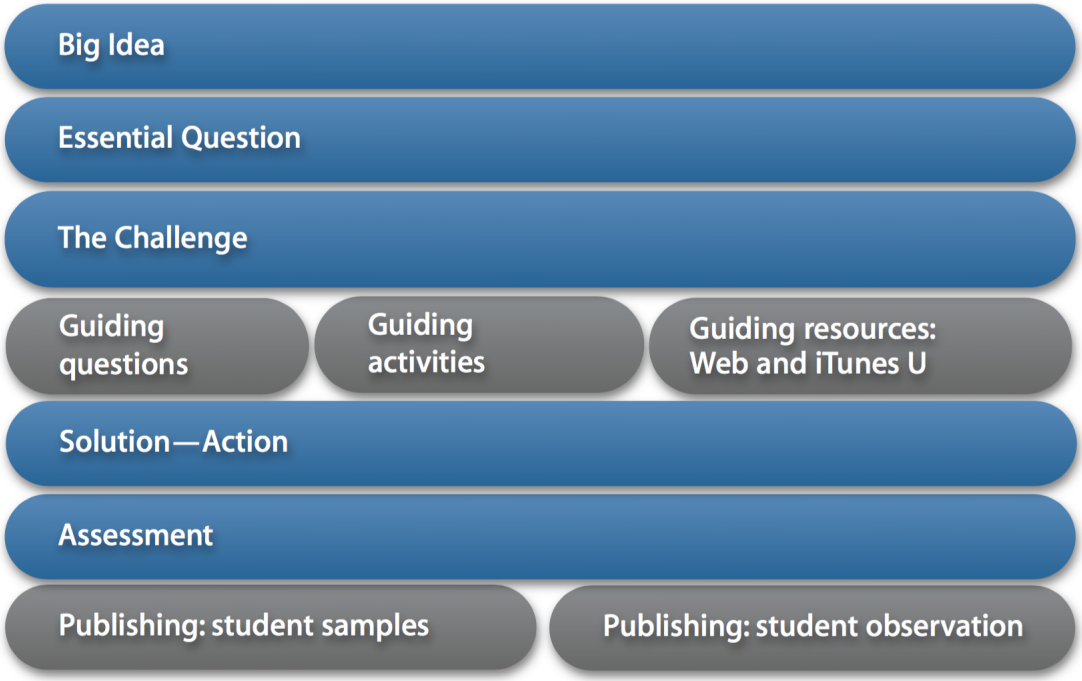

Figura 5.1: Conceito da abordagem baseada em desafios

A abordagem baseada em desafios basicamente é composta pelos estágios que incluem *Big Idea*, *Essential Question*, *Challenge*, *Guiding Questions*, *Guiding Activities*, *Guiding Resources*, *Solution*, *Assessment*, *Publishing*. Estes foram apresentados no relatório técnico divulgado em 2009 [58] e são descritos na Tabela 5.1.

Tabela 5.1: Principais características da abordagem baseada em desafios

Nome	**Descrição**
Big Idea	Um conceito amplo que pode ser explorado de diversas formas, é envolvente, e tem importância para os alunos e a sociedade em geral.
Essential Question	Deve identificar o que é importante a ser descoberto da *Big Idea* e refinar a ideia.
Challenge	Os estudantes devem articular um desafio para criar uma resposta específica ou solução que pode resultar em ação concreta.
Guiding Questions	Representa o conhecimento que os estudantes precisam descobrir para atingir o desafio.
Guiding Activities	Qualquer atividade que possa ajudar os estudantes a responderem as *guiding questions*.
Guiding Resources	Recursos diversos que podem ser *podcasts, websites*, vídeos, banco de dados, artigos ou especialistas.
Solution	Cada desafio é indicado de forma suficientemente ampla para permitir uma variedade de soluções que devem ser concretas, atingíveis e claramente articuladas.
Assessment	A solução pode ser avaliada pela sua conexão com o desafio, precisão do conteúdo, clareza de comunicação, aplicabilidade para implementação, etc.
Publishing	O desafio proporciona múltiplas oportunidades de documentar a experiência e publicação da mesma. Os estudantes são encorajados a publicar resultados e obter feedback.

Esta abordagem busca proporcionar espaço e liberdade para os estudantes serem criativos ao mesmo tempo que eles devem receber suporte e pontos de verificação e acompanhamento de forma a mantê-los motivados. O fluxo do CBL pode ser estruturado e modificado de diferentes formas. Baseado no estudo apresentado em 2009 [58], foi adaptada a Figura 5.2 de forma a apresentar o fluxo inicialmente concebido.

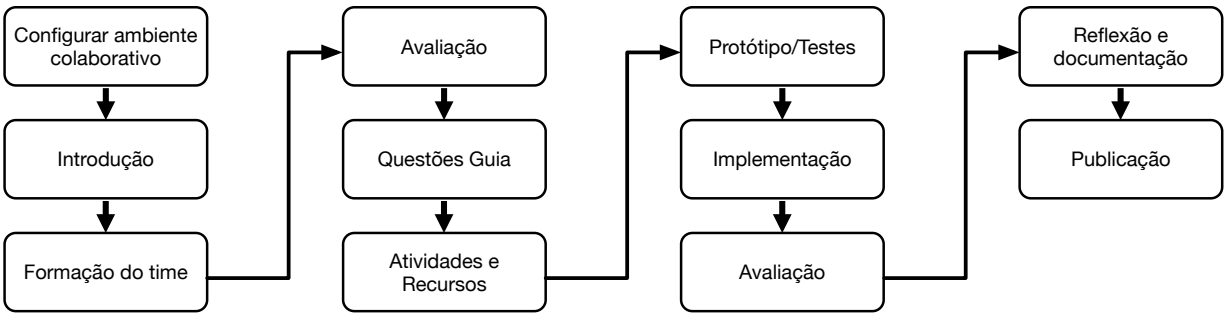

Figura 5.2: CBL - Fluxo principal

O processo apresentado na figura 5.2 é um ponto inicial e pode ser personalizado e alterado de acordo com demandas e realidade de diferentes, neste sentido o relatório técnico [58] descreveu os passos deste fluxo os quais são apresentados de forma adaptada na Tabela 5.2.

Tabela 5.2: Fluxo principal da abordagem baseada em desafios

Passo	**Descrição**
Configurar ambiente colaborativo	Deve ser fornecido um ambiente que esteja disponível a qualquer dia e hora para os estudantes, incluindo o acesso aos recursos necessários, atividades, *timeline*. Além disso plataformas web devem ser usadas o máximo possível para facilitar a comunicação e colaboração.
Introdução	Após a seleção da *big idea*, o primeiro passo é desenvolver com os estudantes o problema a ser resolvido dentro do tema.
Formação da equipe	Durante este estágio, é importante considerar os papéis e responsabilidades discutindo o desenvolvimento natural das equipes.
Avaliação	Os instrutores e as equipes discutem o que será utilizado para medir o sucesso das atividades, e adotam isso de forma que possa ser adaptado e desenvolvido de forma a garantir o sucesso dos projetos.
Perguntas guia	Após as equipes estarem formadas e terem definidos os seus desafios, começa o processo de identificação das perguntas que vão guiar as análises dos tópicos definidos como desafio por cada equipe. Estas questões vão delinear o que os estudantes pensam que precisam para saber como formular uma solução viável.
Atividades e recursos guia	As equipes buscam como encontrar as respostas das perguntas guia participando de uma variedade de atividades, conduzindo pesquisas, entrevistas e diferentes dinâmicas que os ajudem a definir a melhor solução possível.
Protótipo/Teste	Uma vez que as soluções são definidas, é possível construi-las e testa-las em pequenos grupos, permitindo que as equipes melhorem suas soluções.
Implementar	Neste passo será desenvolvido um plano de ação para colocar a implementação da solução em prática.
Reflexão	Uma boa parte do processo de aprendizado é construído quando se pensa sobre como ele está ocorrendo, analisando relacionamentos entre os conteúdos aprendidos e suas aplicações, interagindo com diferentes pessoas e desenvolvendo uma solução.
Publicação	Os estudantes devem ser encorajados a publicar o trabalho realizado.

Através da análise de como a abordagem baseada em desafios é estruturada como também seu fluxo de uso, e fazendo uma análise com os resultados encontrados no mapeamento sistemático da literatura (Capítulo 4), foram encontrados indicativos que o desenvolvimento ágil pode ser integrado a este tipo de abordagem.

A abordagem baseada em desafios busca desenvolver projetos reais, ao mesmo tempo que ela é flexível e personalizável por definição [58, 61]. Neste sentido, o desenvolvimento de aplicativos para dispositivos envolve diferentes desafios como desenvolver para uma variedade de dispositivos os quais operam em diferentes plataformas de *hardware* e *software* em constante atualização e evolução de requisitos [122, 123, 125, 127], gerando uma necessidade do gerenciamento da evolução constante destas plataformas e requisitos, como também de um *time-to-market* adequado. Dessa forma, a metodologia Scrum se apresenta como uma alternativa pois ela foi concebida com o intuito de prover flexibilidade através de mecanismos de controle, para planejar *releases* de produtos e gerenciar o desenvolvimento de acordo com o seu progresso, permitindo que os projetos e entregas

sejam alterados a qualquer momento de forma a entregar o produto mais apropriado [107]. Além disso o Scrum deixa os desenvolvedores livres para conceber as soluções criativas ao longo do projeto, como também ocorre a aprendizagem adaptação do ambiente [107]. Assim, buscou-se utilizar os principais conceitos do método Scrum, como também as principais práticas ágeis de forma geral.

5.2 Estrutura do *framework* Scrum

O método Scrum já mencionado na subseção 2.3.4 é um método adaptivo de desenvolvimento ágil mais utilizado pela indústria. Pesquisas recentes [90] realizadas com mais de 1000 participantes, 69% das empresas relataram estar utilizando Scrum. De acordo com Schwaber [107], o Scrum tem como estrutura as fases apresentadas na Tabela 5.3.

Tabela 5.3: Principais fases do *framework* Scrum

Fase	**Descrição**
Planejamento	Esta fase envolve diversas atividades, entre elas as principais que se destacam são o desenvolvimento de uma lista compreensiva de requisitos, chamada de *"product backlog"*, definir os objetivos e datas de uma ou mais *releases* e selecionar a *release* mais apropriada para o desenvolvimento imediato, de acordo com as prioridades definidas no *product backlog*.
Arquitetura/Design	Revisar os itens do *backlog* de forma a identificar mudanças necessárias, refinar a arquitetura do sistema para suportar os requisitos a serem implementados.
Desenvolvimento (*"Sprint"*)	Esta fase é um ciclo iterativo de trabalho de desenvolvimento, com Sprints iterativos até que o produto esteja pronto para distribuição. A O Sprint é um conjunto de atividades de desenvolvimento que devem ser conduzidas em um período de uma a quatro semanas. Para cada Sprint deve ser realizado uma revisão onde todas as equipes apresentam o seu progresso de trabalho, levantando e resolvendo problemas como também atualizando o *backlog*, neste momento os riscos são revisados e respostas são definidas.
Fechamento	Esta fase prepara o produto desenvolvido para release.

O Scrum é um método que facilita o gerenciamento de projetos ágeis [107], mas o que de fato faz o desenvolvimento ser ágil são as práticas ágeis utilizadas. Diferentes práticas de desenvolvimento ágil apresentadas por estudos recentes foram descritas na subseção 2.3.2. Neste sentido, o método a ser proposto visa adotar as cinco práticas ágeis mais utilizadas de acordo com o estudo recente da Version One [90], as quais são descritas na Tabela 5.4.

Tabela 5.4: Conjunto de práticas ágeis que fazem parte do método

Percentual	Prática
80%	Reunião diária
79%	Iterações curtas
79%	*Backlogs* priorizados
71%	Planejamento Iterativo
69%	Retrospectivas

Os *backlogs* priorizados permitem que os requisitos sejam ordenados de acordo com a sua relevância ao escopo corrente, focando em satisfazer primeiro os requisitos com maior importância [96]. O planejamento iterativo garante que os problemas fundamentais sejam resolvidos antes que um processo ou produto seja finalizado [110].

As iterações curtas através das Sprints (que são um conjunto de atividades de desenvolvimento), devem conduzidas em um período de uma a quatro semanas de forma a permitir um planejamento iterativo e aprendizado contínuo [107]. Através de reuniões diárias cada um dos membros da equipe responde geralmente três perguntas: o que tem sido feito desde a última reunião? o que se espera fazer até a próxima reunião? e quais são os impedimentos para atingir os objetivos? [105]. Neste sentido, ao final de cada iteração as retrospectivas permitem uma revisão de como o processo de desenvolvimento está evoluindo de forma a identificar lições aprendidas e oportunidades de melhoria [9].

Neste contexto, é importante ressaltar que o método é fundamentado em uma abordagem baseada em desafios e práticas ágeis (inspirado no Scrum). Entretanto o Scrum é apenas uma forma de organizar e gerenciar os projetos de maneira ágil. O desenvolvimento ágil de fato depende do uso de práticas ágeis de desenvolvimento e do engajamento das equipes na adoção das mesmas.

5.3 Estrutura do Método Inicial

A estrutura base do método basicamente é composta pelos seguintes estágios da abordagem baseada em desafios: *Big Idea, Essential Question, Challenge, Implementation, Solution e Evaluation* [60, 61]. Assim, analisando estes estágios com os estágios descritos na Seção 5.1 os estágios *Assessment e Publication* foram substituídos pelos estágios *Implementation e Evaluation* propostos em uma evolução do método CBL [60, 61], aspecto que vai de encontro a uma maneira de integrar práticas ágeis na proposta CBL.

Este método inclui práticas ágeis nos estágios de *Solution, Implementation e Evaluation*. O estágio de *solution* engloba o *product backlog*, o estágio de *implementation* contempla a iteração ágil, incluindo planejamento de sprint, reuniões diárias e entrega do produto incremental. Assim, o estágio de *evaluation* inclui as revisões de sprint e retrospectivas de sprint de forma a consolidar alguns dos principais conceitos do *framework* scrum, incorporando também práticas ágeis. As fases do método são apresentadas na Tabela 5.5.

Tabela 5.5: Fases da proposta inicial do método

Fase	Estágios Envolvidos	Objetivo
Visão	*Big Idea, Essential Question, Challenge*	O objetivo desta fase é montar os grupos de estudantes de acordo com afinidades e objetivos em comum, de forma a permitir um processo de aprendizado através da identificação de uma área de interesse e de um problema a ser resolvido.
Pesquisa	*Guiding Questions, Resources and Activities, Solution Proposal*	Tem como objetivo proporcionar uma oportunidade para que os membros das equipes descubram e aprendam quais os conhecimentos, recursos e atividades são necessários para propor uma solução que seja viável, permitindo também um processo de aprendizado baseado em problemas reais.
Desenvolvimento	*Product Backlog, Sprint Planning, Sprint development, Testing*	Faz com que a proposta de solução encontrada através do processo de pesquisa seja transformada em requisitos transpostos em um *product backlog*, os quais são desenvolvidos através de curtos ciclos de desenvolvimento proporcionando aprendizado e melhoria contínua.
Avaliação	*Sprint Review, Sprint Retrospective*	Permite que a cada iteração de desenvolvimento o produto desenvolvido seja avaliado em conjunto de instrutores e membros da equipe e entre membros de diferentes equipes, além de permitir também uma retrospectiva sobre o processo de aprendizado e desenvolvimento durante a iteração de desenvolvimento.

Seguindo os principais elementos teóricos da abordagem baseada em desafios, e do *framework* Scrum e das principais práticas ágeis, foi proposta a estrutura do método inicial, a qual é apresentada na Figura 5.3.

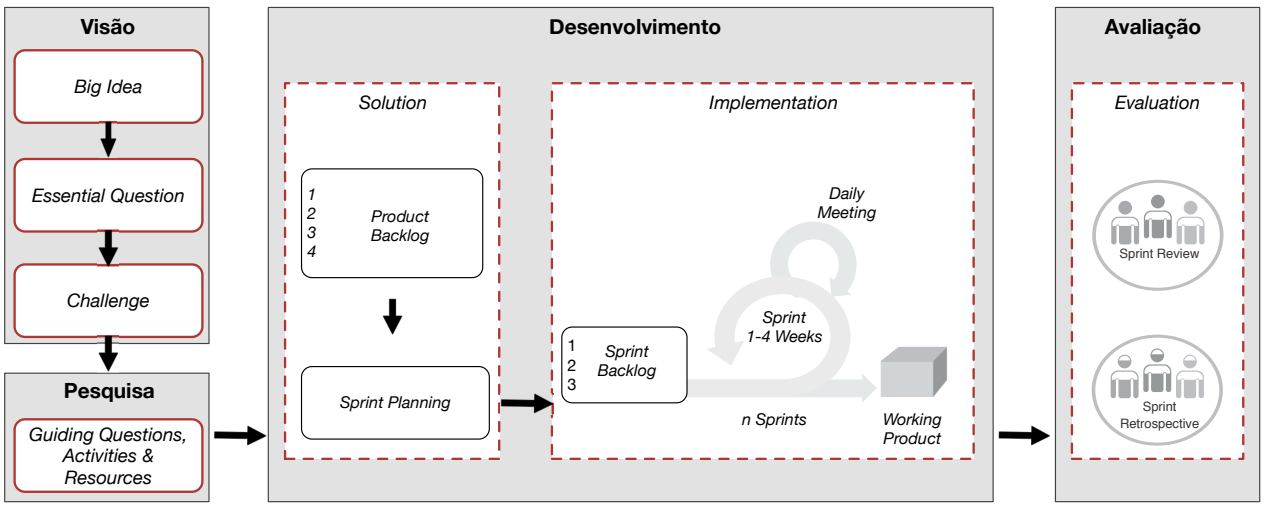

Figura 5.3: Método inicial - Abordagem Baseada em Desafios e Práticas Ágeis

Após a construção do método inicial, foi realizado um conjunto de reuniões com especialistas com o intuito de analisar a estrutura inicial do método e contribuir com eventuais melhorias antes da realização do estudo de campo.

5.3.1 Reuniões com especialistas

A estrutura do método passou por uma análise através de reuniões com um conjunto de especialistas que fazem parte de um projeto em colaboração com uma empresa multinacional, na área de ensino e desenvolvimento de aplicativos para dispositivos móveis, conforme apresentado na Tabela 5.6, a coluna Tipo de experiência significa (Academia ou Indústria).

Tabela 5.6: Reuniões com especialistas: características e experiências

Especialista	Função	Tipo de Experiência	Local	Experiência em Aplicativos	Experiência em CBL	Experiência em Ágil
# 1	Instrutor	Ambas	Brasil	2 anos	2 anos	1 ano
# 2	Instrutor	Acadêmica	Brasil	2 anos	2 anos	1 ano
# 3	Instrutor	Indústria	Brasil	6 anos	2 anos	3 anos
# 4	Instrutor	Indústria	Brasil	5 anos	2 anos	6 anos
# 5	Instrutor	Indústria	Brasil	4 anos	2 anos	3 anos
# 6	Consultor	Ambas	EUA	-	8 anos	-
# 7	Gerente Pedagógico	Ambas	Brasil	9 anos	2 anos	1 ano
# 8	Gerente de Programa	Ambas	Brasil	2 anos	2 anos	9 anos

O principal objetivo destas reuniões foi analisar possíveis contribuições necessárias ao método, através da coleta de *feedback* de especialistas. O método foi discutido com especialistas em quatro diferentes reuniões. Três dessas reuniões foram realizadas de forma presencial com especialistas do Brasil, e uma reunião com o especialista dos EUA foi feita através de video conferência.

Baseado no *feedback* coletado foram realizadas três considerações no método. A primeira foi a inclusão das reflexões em paralelo com todos os estágios do método, pois os alunos tem a liberdade de realizar e gravar reflexões sobre o aprendizado em qualquer momento do processo. A segunda consideração foi a inclusão de um passo chamado *"Research Findings"* o qual permite que os estudantes agrupem e organizem os resultados e aprendizado durante o processo de pesquisa, antes de que se inicie o estágio de desenvolvimento. A terceira, foi a inclusão de um retorno da implementação para a fase de pesquisa, pois em alguns momentos durante a implementação os alunos podem se deparar com oportunidades que não tenham sido descobertas no processo de pesquisa inicial. Isso faz com que seja necessário um retorno ao processo de pesquisa de forma a permitir uma evolução e consolidação dos conhecimentos necessários durante o processo de aprendizagem e desenvolvimento dos projetos. Dessa forma, a Figura 5.4 apresenta o método proposto.

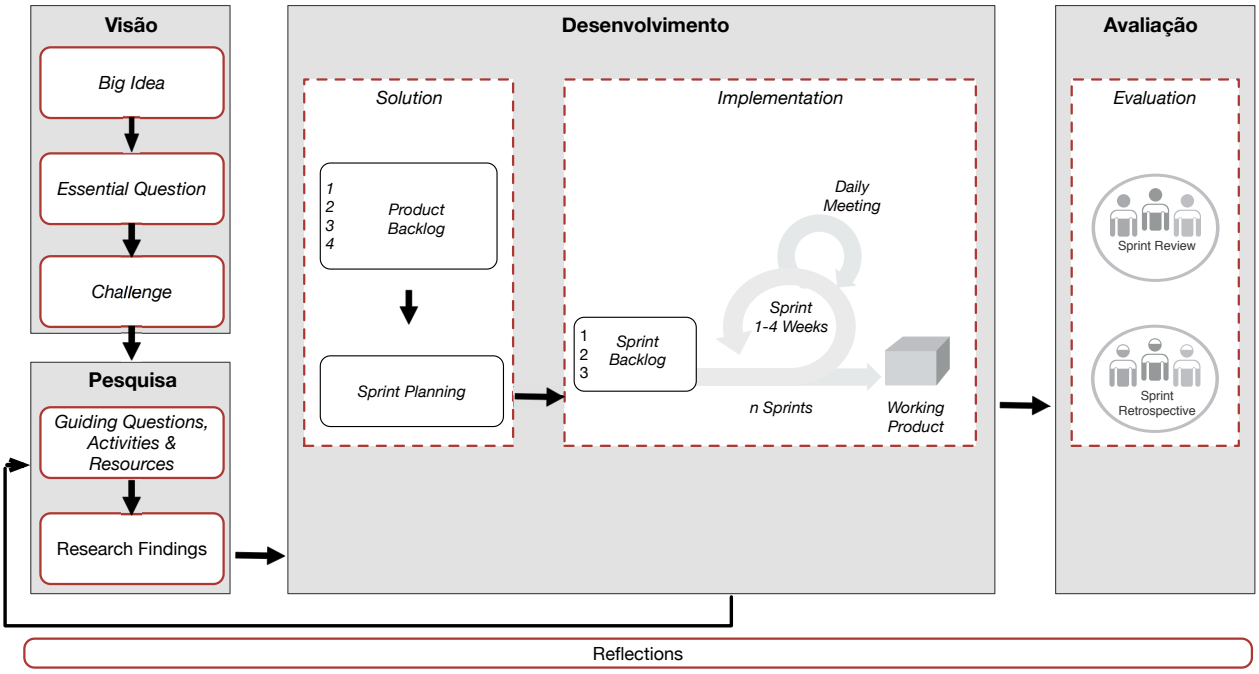

Figura 5.4: Método Proposto - Abordagem Baseada em Desafios e Práticas Ágeis

Esta análise do método forneceu contribuição para esta pesquisa, pois foi possível coletar dados para melhorar o fluxo de sequência entre os estágios do método permitindo uma maior aderência do mesmo devido ao fato de que é possível ser realizada mais pesquisa após ter se iniciado os ciclos de desenvolvimento. Além disso a questão de incluir o *research findings* proporciona uma melhor organização e facilita a transição da fase de pesquisa para a fase de desenvolvimento. Além disso, estas reuniões também promoveram uma melhor compreensão sobre o método proposto a algumas das partes interessadas pelo projeto de pesquisa proposto nesta tese. Dessa forma, o próximo passo é a definição de considerações iniciais para o uso do método proposto.

5.4 Considerações iniciais para o uso do método proposto

Algumas são as considerações para a utilização do método proposto. A transição do desafio para a solução é um elemento crítico no processo CBL, e talvez seja o mais difícil para os estudantes uma vez que eles geralmente querem mover diretamente para soluções [58]. Uma atividade chave a ser realizada na integração entre a abordagem baseadas em desafios e o desenvolvimento ágil é configurar uma *timeline* com datas marco para *Big Idea, Essential Question, Challenge, Guiding Questions, Resources e Activities* (GRA), somente após finalizar o GRA que os estudantes vão iniciar a integração com desenvolvimento ágil na solução.

Os estudantes devem usar *GRA* para organizar e documentar o trabalho, este processo é um passo crítico e deve ser efetuado antes de iniciar o estágio de práticas ágeis, porque os instrutores devem trabalhar com os estudantes de forma a identificar quando eles precisam aguardar para mover para o passo da solução, mostrando que as idéias de solução talvez gerem mais questões guia. Os

estudantes não devem mover para o estágio de Solução até que o fluxo de pesquisa (GRA) esteja satisfatoriamente completo.

O primeiro estágio entre o CBL e o desenvolvimento ágil é a transição de GRA para Solução, é nesse estágio que a integração se inicia. Neste estágio a saída do processo GRA será utilizada como a entrada para construir o *product backlog* ordenado por prioridades de acordo com as demandas de negócio/usuários.

O estágio de solução vai se repetir enquanto existirem sprints a serem realizados para o escopo de trabalho. No final do planejamento do sprint se inicia o estágio de *implementation* e esse estágio vai ocorrer de acordo com o número de sprints de desenvolvimento. No final de cada sprint será executado o estágio de *evaluation* e este estágio também irá repetir de acordo com o número de revisões de produto e lições aprendidas a serem realizadas. De forma a realizar o processo de aprendizado e melhoria continua, os estágios de *Solution, Implementation e Evaluation* vão se repetir de acordo com o número de sprints necessários para finalizar o escopo de trabalho.

6. ESTUDO DE CAMPO

A pesquisa de campo busca estudar pessoas engajadas nas suas atividades diárias [46] e pode ser utilizado em conjunto com outros métodos de pesquisa [46]. O método de estudo de campo foi conduzido através de um levantamento para a coleta e classificação de dados [10]. O levantamento é apropriado quando o foco de interesse é sobre o que está acontecendo ou como e porque algo esta acontecendo e também se aplica quando não é possível controlar as variáveis dependentes e independentes [10]. Como mencionado na Etapa 2 da subseção 3.1.2 o objetivo do estudo de campo é analisar o uso do método de forma a detectar características da amostra, mapear o ambiente de desenvolvimento, ferramentas utilizadas e práticas de desenvolvimento, entender os fatores que fazem do método uma solução viável e também identificar vantagens e desvantagens, criando um corpo de conhecimento sobre o método. A seção 6.1 descreve o protocolo utilizado, a seção 6.2 apresenta a descrição e ambiente do local onde o estudo de campo foi realizado, a seção 6.4 apresenta os resultados encontrados no estudo de campo, a seção 6.5 apresenta as lições aprendidas neste estudo de campo, a seção 6.6 apresenta as limitações na condução deste estudo e por fim a seção 6.7 apresenta as considerações finais deste estudo.

6.1 Protocolo

O objetivo do estudo de campo é verificar a viabilidade do método proposto. A análise da viabilidade serve para testar a eficácia de um processo, mas não é capaz de afastar todas as hipóteses rivais que ainda podem existir no final do estudo [110]. De acordo com Gibbs [40] a pesquisa qualitativa visa entender e descrever fenômenos de diversas formas, como por exemplo analisando experiências de indivíduos ou grupos, analisando práticas cotidianas e que podem ser tratadas analisando-se conhecimento, relatos, registro de práticas ou investigando documentos.

A fonte das informações fornecidas no estudo de campo vão ser mantidas em sigilo por motivos de acordos confidenciais. Tanto os nomes das instituições quanto dos participantes não serão divulgados.

6.1.1 Procedimentos

Este estudo de foi realizado através do uso de questionários. O fato das mesmas questões serem enviadas aos sujeitos da amostra facilita a agregação de respostas para montar uma visão sobre as respostas, mesmo que se use questões abertas onde os participantes tem permissão para escrever abertamente, as perguntas ainda fornecem um panorama para uma categorização inicial das respostas qualitativas [110]. Os tópicos levantados nos questionários não induziram qualquer resposta.

Fonte de dados

Estudantes, tópicos a serem levantados nos questionários: i) quais os desafios em desenvolver aplicativos; ii) opiniões sobre adoção de ágil em ambiente de desenvolvimento de aplicativos; iii) perfil dos estudantes em treinamento iv) viabilidade da continuação do estudo.

6.1.2 Questões de Pesquisa

As seguintes questões de pesquisa foram definidas para o estudo de campo:

(**RQ1**) *Como as diferenças em desenvolver um aplicativo para dispositivo móvel se apresentam na prática?*

(**RQ2**) *Qual a percepção sobre o uso de desenvolvimento ágil para aplicativos?*

(**RQ3**) *O uso do método proposto tem alguma vantagem ou desvantagem?*

(**RQ4**) *O método proposto é viável?*

RQ1 busca identificar as diferenças ao desenvolver um aplicativo especificamente para dispositivos móveis. A identificação destas diferenças permite um comparativo com o que foi encontrado na literatura e a identificação de novas contribuições. RQ2 busca entender as percepções sobre o uso de desenvolvimento ágil em ambientes de desenvolvimento de aplicativos. RQ3 busca identificar as possíveis vantagens e desvantagens do método proposto para o ensino e desenvolvimento em ambientes de aplicativos para dispositivos móveis. RQ4 tenta identificar se o método proposto é viável de forma a identificar as lições aprendidas durante o estudo de campo.

6.2 Descrição do Local do Estudo de Campo

O ambiente onde este estudo de campo foi realizado é um projeto em parceria com uma universidade do sul do Brasil. Este projeto engloba um treinamento que tem suporte de uma grande empresa da área de tecnologia para transformar estudantes em desenvolvedores de aplicativos iOS, onde são desenvolvidos projetos reais. O foco do treinamento é em estudantes de graduação. Este projeto é pioneiro no país e realizado em um ambiente de aprendizado único, configurado para prover uma variedade de ambientes de trabalho.

O treinamento tem duração de um ano, com turmas de aproximadamente 100 alunos que se organizam em diferentes equipes, o curso é dividido em duas partes principais: Na primeira o foco é maior na aplicação de conteúdos através de *workshops* e com o desenvolvimento de pequenos projetos de desenvolvimento de aplicativos; Na segunda o foco é em projetos mais robustos com um escopo maior para os aplicativos, onde os alunos são divididos entre diferentes equipes, através de projetos com aproximadamente seis meses de duração. Durante o treinamento, cada estudante possui seus próprios equipamentos para usar nas aulas e desenvolver os projetos. O currículo do

curso inclui os seguintes aspectos: Programação Orientada a Objetos, Componentes de interface de Usuário, Model View Controller (MVC), Data sources, Navegação, Animações e Frameworks. Além disso os estudantes foram expostos a conceitos de abordagem baseada em desafios, *framework* Scrum e práticas ágeis. O estudo de campo foi realizado no final de 2014.

Os instrutores servem como especialistas para todos os grupos, supervisionando os projetos e proporcionando *feedback* para os alunos. O tamanho dos grupos varia, pois os alunos tem liberdade para escolher os suas próprias equipes, geralmente são equipes de duas a cinco pessoas. O curso requer uma dedicação de 4 horas diárias, totalizando 20 horas por semana. O treinamento é facilitado por 6 instrutores, os quais tem experiência na indústria e academia, conforme apresentado na Tabela 6.1.

Tabela 6.1: Perfil de experiência dos instrutores participantes no estudo de campo

Instrutor	Tipo de Experiência	Tempo de Experiência
# 1	Ambas	15 anos
# 2	Acadêmica	7 anos
# 3	Indústria	11 anos
# 4	Indústria	9 anos
# 5	Indústria	5 anos
# 6	Indústria	16 anos

As equipes são apoiadas durante todo o ciclo da abordagem baseada em desafios integrada ao desenvolvimento ágil, apresentando retrospectivas ao final de cada iteração (geralmente sprints de duas semanas). Os instrutores monitoram as equipes e avaliam o aprendizado e evolução dos estudantes através das revisões e retrospectivas.

6.3 Coleta de Dados

A amostra da população participante foi dimensionada utilizando o critério de conveniência devido ao fato que os participantes foram selecionados de acordo com a sua disponibilidade. O tamanho da população foi definindo utilizando o maior número possível de pessoas disponíveis dentro de um universo de 94 estudantes, totalizando mais de 80 respostas no total. O instrumento aplicado no estudo de campo foi um questionário online, descrito no apêndice B, através da ferramenta Qualtrics[1]. Este questionário foi aplicado entre Novembro de 2014 e Dezembro de 2014.

6.3.1 Procedimentos para análise dos dados

A exploração do material de análise foi desenvolvida tomando como referência as recomendações de análise de dados apresentadas por Creswell [22]. Segundo o autor, este método tem como objetivo obter de forma sistemática, informações que permitam a realização de inferências a cerca do objeto de estudo. Para o desenvolvimento da técnica, foram seguidos os seguintes passos:

[1]O Qualtrics é um serviço de organização e coleta de dados para pesquisa via internet que possibilita criar e responder questionários - https://pucrs.qualtrics.com

Passo 01 Organizar e preparar os dados para análise.

Passo 02 Ler todos os dados para obter uma percepção geral das informações e refletir sobre o seu significado.

Passo 03 Realizar uma análise detalhada com um processo de codificação, segmentando sentenças ou parágrafos em categorias e rotulando essas categorias com um termo.

Passo 04 Utilizar este processo para gerar uma descrição do local, das pessoas ou temas para análise.

Passo 05 Utilizar passagens narrativas para comunicar os resultados da análise.

Passo 06 Realizar uma interpretação ou extrair um significado dos dados.

Concluído o processo de coleta de dados empíricos, realizou-se a organização e preparação dos dados, cujo material constituiu o *corpus* da análise. De posse dos dados, realizou-se a leitura flutuante, deixando-se invadir pelas impressões e orientações do texto.

O *corpus* da análise foi organizado em quadros síntese com uma expressão gráfica da análise textual, onde as respectivas respostas recebidas dos sujeitos foram ordenadas sequencialmente a cada um dos tópicos do instrumento de pesquisa na forma de indicadores construídos de acordo com as questões norteadoras e os objetivos deste estudo. A análise dos resultados é apresentada na Seção 6.4.

6.4 Análise de Resultados

A maioria dos respondentes são dos cursos de sistema da informação 37,14%, seguidos de Ciência da Computação 32,86% e Análise de Sistemas 14,29%. Nesse sentido, a maioria deles está no quarto semestre 31,43%, seguidos do terceiro semestre 20% e sexto semestre 15,71%. A média de idade entre os alunos da amostra é de 22,5 anos, considerando-se um desvio padrão de 3,46.

6.4.1 Dados da amostra

A maioria dos estudantes 65,71% não haviam participado de nenhum treinamento anteriormente ao curso. Dois estudantes participaram de 3 treinamentos anteriormente ao curso. Três estudantes participaram de 2 treinamentos anteriormente ao curso e 12 estudantes participaram de até um treinamento antes do curso, conforme apresentado na Figura 6.1.

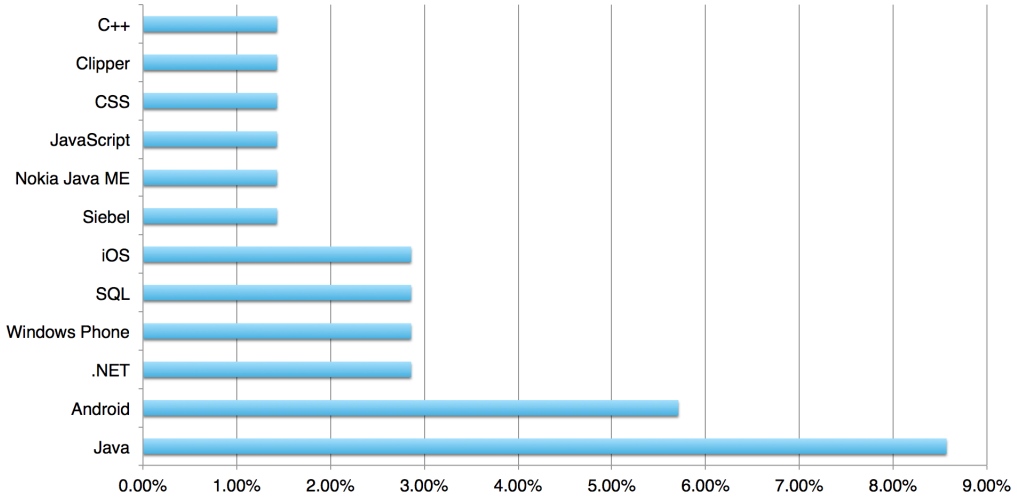

Figura 6.1: Treinamentos que os participantes tiveram antes do curso

Dos estudantes, 9 relataram treinamentos anteriormente ao curso em plataformas de desenvolvimento para dispositivos móveis (Android, iOS, Windows Phone, Nokia java me). Dentre os estudantes, 72,85% possui conhecimento anterior em pelo menos 3 linguagens de programação diferentes. A maioria 71,43% tinha um conhecimento anterior na linguagem Java, seguido por C 62,86% e C++ 38,57% e C# 37,14%. A lista de linguagens conhecidas pela amostra anteriormente ao curso está disponível na Tabela 6.2.

Tabela 6.2: Conhecimento em linguagens de programação anteriormente ao curso

Linguagem de Programação	Percentual
Java	71,43%
C	62,86%
C++	38,57%
C#	37,14%
PHP	28,57%
Javascript	18,57%
Python	17,14%
SQL	12,86%
Pascal	8,57%
Ruby, Visual Basic	7,14%
HTML, Objective C	5,71%
Assembly	4,29%
Clipper, CSS, Delphi, VHDL	2,86%
Nenhum	2,86%
Android, ASP.NET, Lisp, Scala, VB.NET	1,43%

Quanto aos conhecimentos em metodologias de desenvolvimento de software, 65,71% não tinha conhecimento algum sobre metodologia de desenvolvimento de *software* anteriormente ao curso. Dentre os alunos que já conheciam alguma metodologia, 27,14% conhecia Scrum, seguido de 12,86% XP e 8,57% Cascata, conforme apresentado na Figura 6.2.

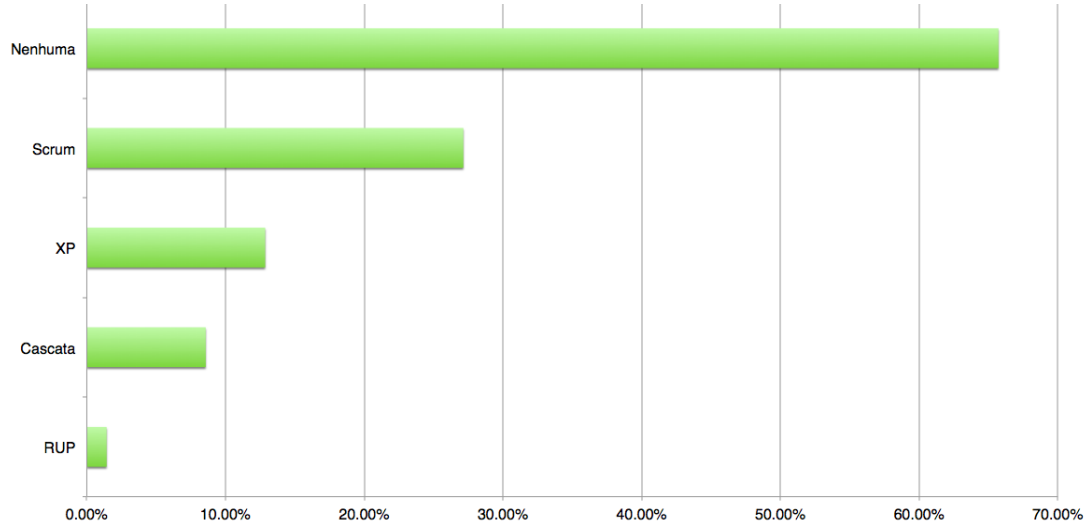

Figura 6.2: Conhecimentos em metodologias anteriores ao curso

6.4.2 Diferenças no desenvolvimento de aplicativos para dispositivos móveis

Os participantes relataram alguns desafios encontrados durante o desenvolvimento de aplicativos para dispositivos móveis, os quais são resumidos em Interface e Experiência de Usuário, Desempenho, Atualização constante, Usuários, Plataformas e Velocidade. Comparando os desafios encontrados no estudo de campo com os desafios encontrados por Gordon [42], foram encontradas semelhanças em diversos aspectos como Interface de Usuário e Usabilidade, Hardware, Manipulação de dados, cooperação de dispositivos e questões de programação. Um desafio citado por Gordon [42] que não foi encontrado no estudo de campo é a questão de Interação de aplicativos. Por outro lado o estudo de campo apresentou dois desafios adicionais, no que se refere a atualização constante de plataformas e sistemas operacionais e também no que se refere a base de usuários.

Interface de usuário/Experiência de usuário

Interface do usuário foi mencionada como um dos fatores que diferem o desenvolvimento de aplicativos móveis para aplicativos convencionais. *"Porque várias pessoas usarão o aplicativo em um celular com diferentes resoluções, tamanhos etc"* (Participante 1). Isso é um desafio, principalmente devido a diversidade de dispositivos, e também aos diferentes tamanhos e plataformas de desenvolvimento que podem ser utilizadas para desenvolver aplicativos.

> Os dispositivos móveis promovem uma serie de possibilidades de maneiras de interagir com o usuário e com o conteúdo do aplicativo de maneira dinâmica, que nenhuma outra plataforma oferece, e com isso surgem diversos desafios [...] (Participante 13).

Experiência do usuário foi citada como um dos fatores que diferem o desenvolvimento de aplicativos móveis para aplicativos convencionais. Isso é um desafio, principalmente devido a diversidade de dispositivos, e também as funcionalidades e sensores que podem ser utilizados através de um dispositivo móvel.

> Acho que a principal diferença é a questão de interfaces, não pela quantidade gigantesca de telas diferentes, mas pela forma que o aplicativo é utilizado, em um computador/mac/pinguim sempre foi utilizado teclado e mouse como forma de entrada e em um dispositivo móvel temos o teclado, mas no lugar do mouse temos o toque na tela que tem inúmeras outras representações que clique e não clique, sem falar na utilização de todo os outros sensores disponíveis, o que torna a criação da interface para integrar de forma harmônica tudo bem desafiadora (Participante 15).

Gordon [42] também menciona as diferentes formas de pensar as entradas e saídas produzidas, com foco na capacidade de resposta e interações sensíveis ao contexto como um desafio tanto para a interface de usuário quanto para a usabilidade, o que vai de encontro ao resultado encontrado. Devido ao fato dos desenvolvedores estarem acostumados a desenvolver aplicações que usam apenas o teclado e mouse da forma tradicional, o desafio apresentado é entender como os aplicativos para dispositivos mudam as opções de interface de usuário e experiência de usuário, devido a uma gama de opções de entrada e sensores. Isso é um processo desafiador porque os aplicativos geralmente são desenvolvidos para um público geral, onde deve-se prestar atenção a todos os aspectos.

Desempenho

Outro ponto relatado como uma diferença e desafio no desenvolvimento de aplicativos é a questão de desempenho no acesso a dados. *"[...] Os maiores desafios acredito que seja estar sempre buscando o melhor desempenho para o aplicativo, principalmente no acesso a dados [...]"* (Participante 2).

Lidar com características de múltiplos sensores, memória limitada e armazenamento reduzido é um desafio no desenvolvimento de aplicativos [42]. Além do desafio no acesso a dados, a questão da intensidade do uso dos aplicativos e limitações de *hardware* também são destacadas como um diferencial em desenvolver aplicativos para dispositivos móveis.

> O desenvolvimento de aplicativos implica em *software* para uso intenso e extremamente intimo ao usuário. Deve sempre estar de acordo com o padrão de qualidade esperado, mesmo com diferentes limitações e opções de *hardware* (Participante 4).

Atualização Constante

A atualização constante dos aplicativos também é citada como uma das principais diferenças e desafios *"[...] Os maiores desafios acredito que seja estar sempre atualizado, ou o mesmo ficará obsoleto muito rapidamente"* (Participante 2).

Devido a frequente atualização das plataformas de desenvolvimento, sistemas operacionais, frameworks, como também o frequente lançamento de dispositivos com diferentes tamanhos e funcionalidades. *"Como desafios, acredito que o fato de você ter que se manter sempre atualizado, pois*

no desenvolvimento mobile está sempre surgindo novidades, frameworks novos, linguagens novas" (Participante 3).

Neste contexto, questões de manutenção de aplicativos neste mundo de mudanças rápidas em plataformas móveis, também são criticas, pois usuários frequentemente atualizam os seus dispositivos e seus aplicativos, o que se torna um desafio para os desenvolvedores [127].

Usuários

Os usuários também foram apontados como um dos aspectos de diferença para o desenvolvimento tradicional em dois sentidos: primeiro na questão da base de usuários, pois um único aplicativo pode ter uma grande base de usuários, segundo pelo sentido de que uma grande quantidade de usuários também corresponde à uma grande diversidade de usuários, com diferentes expectativas, demandas e dispositivos.

> Acredito que a principal diferença de desenvolver *mobile* ao invés de outros plataformas é a proximidade com o usuário. É comum um aplicativo *mobile* ser utilizado por milhões de pessoas, enquanto um sistema desktop é diferente. No meu ponto de vista, aplicativos móveis podem ajudar e mudar as vidas das pessoas de uma forma mais direta e rápida em comparação a algum outro sistema. Os maiores desafios são promover soluções que realmente façam a diferença na vida das pessoas (Participante 9).

Proporcionar soluções práticas e rápidas para o usuário, é um desafio porque desenvolver aplicativos muitas vezes implica em automatizar ou simplificar atividades cotidianas do usuário através de aplicativos intuitivos.

> Desenvolver aplicativos para dispositivos móveis é diferente porque essa é a categoria de dispositivos mais íntima do usuário. Desenvolvemos produtos que vão fazer parte da vida das pessoas (ás vezes mais do que imaginamos durante o desenvolvimento). Os maiores desafios são: Captar as necessidades do usuário, suprir essas necessidades com o aplicativo e fazer com que o usuário tenha uma experiência ótima durante o uso (Participante 18).

Através da evolução de *softwares* e tecnologias, a dinâmica de requisitos emergentes tem se tornado um desafio no sentido de atender melhor as necessidades dos usuários [125].

Plataformas

As diferentes plataformas de *hardware* e *software* também foram apontadas como uma das diferenças e desafios em desenvolver aplicativos para dispositivos móveis, devido ao fato da quantidade de APIs existente em cada uma das plataformas de desenvolvimento, como também as diferentes funcionalidades/diferenças a nível de *hardware*. A grande diversidade de tipos e recursos desses dispositivos também gera um grande desafio para os desenvolvedores, pois precisam desenvolver o sistema de tal forma que este seja capaz de ser executado satisfatoriamente em uma vasta gama

de dispositivos. *"É interessante porque possui algumas limitações de dispositivo, e devemos ser criativos para poder contornarmos estes problemas, um bom exemplo disso é guardar os dados na nuvem. [...]"* (Participante 9).

Uma questão interessante foi que alguns participantes não encontraram diferenças significativas, entretanto relataram a questão do modelo de negócio para distribuição do aplicativo, conforme o exemplo a seguir:

> Não acredito que desenvolver para plataformas móveis seja muito diferente. Existem diferenças técnicas obviamente, mas o processo de criação do negócio é aplicável em qualquer plataforma. Talvez a parte mais distinta seja o plano de negócio, pois deve considerar as app stores e suas diferenças (Participante 21).

Questões de programação e projetos para diferentes plataformas também é um desafio no desenvolvimento de applications [42]. Nesse sentido, cada plataforma de distribuição (iOS, Android, Windows Phone) tem suas particularidades tanto para publicação dos aplicativos, pois cada uma tem um guia que deve ser seguido e também suas próprias regras de publicação, além de diferem modelos de negócio que podem usar diferentes formas de participação no lucro com as publicações dos desenvolvedores.

Velocidade

A questão da velocidade no ciclo de desenvolvimento também foi encontrada como um fator ao qual implica o desenvolvimento de aplicativos para dispositivos móveis, devido a dinâmica do mercado estes ambientes de desenvolvimento precisam de agilidade no desenvolvimento e distribuição dos aplicativos. *"Proporcionar soluções práticas e rápidas para o usuário, este é o desafio"* (Participante 6). A indústria de desenvolvimento de aplicativos para dispositivos móveis tem crescido muito como também a complexidade dos aplicativos e a velocidade no processo de desenvolvimento.

6.4.3 Uso de desenvolvimento ágil para aplicativos

Em relação ao uso de desenvolvimento ágil em ambientes de desenvolvimento de aplicativos, os participantes relataram diversos aspectos já descritos na literatura, entre eles: Desempenho, Transparência, Velocidade, Entrega, Iterações, Organização e Controle, Melhoria Contínua e Comunicação. Neste sentido, as práticas ágeis utilizadas pelos participantes, 95,24% reportou ter utilizado a prática de reuniões diárias, seguido de 76,19% que utilizaram kanban e 42,86%. A prática menos utilizada foi TDD 9,52%.

Desempenho

Quanto a opinião sobre o uso de métodos ágeis no desenvolvimento de aplicativos, foi relatada melhora no desempenho dos projetos. *"Acredito que auxilia muito na melhora do desempenho. Talvez ainda mais do que nas outras áreas de desenvolvimento. Se encaixa muito bem com desenvolvimento mobile"* (Participante 1).

O desempenho dos próprios estudantes, além dos projetos, foi mencionado por diversos autores [4,70,71] ao se utilizar o desenvolvimento de aplicativos para ensinar conteúdos relacionados a Ciência da Computação. Neste sentido o desempenho percebido pelos alunos ao utilizar o desenvolvimento ágil também pode estar sendo influenciada positivamente pelo próprio desempenho dos indivíduos.

Transparência

Transparência também foi levantada como um ponto de clareza e objetividade gerada pelo uso de desenvolvimento ágil. *"A experiência que tive com desenvolvimento e uso de métodos ágeis foi totalmente proveitosa, então na minha opinião é fundamental o uso dessa metodologia, pois deixa os processos de desenvolvimento mais claros e mais objetivos"* (Participante 2).

Os processos de desenvolvimento ágil buscam aumentar a transparência na condução dos problemas de desenvolvimento, dando poderes ao desenvolvedores para resolver os problemas técnicos [9].

Velocidade

Velocidade foi um outro fator levantado na opinião de métodos ágeis em desenvolvimento de aplicativos para dispositivos móveis, principalmente devido aos curtos ciclos de desenvolvimento de aplicativos.

> Acredito que utilizar métodos ágeis é essencial no desenvolvimento de um aplicativo móvel. A realidade mais dinâmica das app stores fazem com que a interação com o público alvo e ciclos de desenvolvimento mais curtos sejam necessários para o sucesso do aplicativo (Participante 21).

Outros participantes citam a questão de comparação do método ágil com o desenvolvimento tradicional, pois no tradicional a implementação do aplicativo só poderia ser validada no final do ciclo de desenvolvimento o que pode representar um risco para o desenvolvimento do produto.

> A metodologia ágil é fundamental, pois com isso conseguimos planejar e prototipar o produto com muito mais velocidade, ao contrário de outra metodologia. Ex.: Cascata, onde somente validamos a implementação ao final do ciclo (Participante 5).

O desenvolvimento de aplicativos para dispositivos móveis pode ser realizado em um curto período de tempo [105]. *"Métodos ágeis influenciam positivamente no desenvolvimento mobile, pois tratam-se geralmente de soluções que necessitam o desenvolvimento imediato e rápido. [...]"* (Participante 4).

Entrega

Entrega também foi apontado como um fator de opinião sobre o uso de métodos ágeis em desenvolvimento de aplicativos. *"Acredito que a utilização de métodos ágeis ajudam muito uma equipe mobile à se organizar e entregar [...]"* (Participante 3).

A questão da entrega contínua é um dos fatores que fazem parte do manifesto ágil, desde sua origem e é amplamente descrita na literatura. *"[...] O uso de métodos ágeis nos ajuda a gerenciar*

melhor as demandas e as entregas, melhora a interação entre os membros da equipe e fornece uma visão geral realista do andamento do projeto" (Participante 18).

Colocar um sistema simples rapidamente em produção, e então liberar novas versões brevemente é um conceito que é bastante utilizado em XP [12], o que também se aplica como conceito para o desenvolvimento ágil em geral.

Iterações

As diversas iterações também foram apresentadas como importantes as rápidos ciclos de desenvolvimento e melhoria incremental dos aplicativos. *"[...] As diversas iterações existentes em métodos ágeis são de extrema importância para o sucesso do aplicativo. Métodos ágeis facilitam e auxiliam muito o desenvolvimento mobile"* (Participante 9).

Planejamento iterativo em diferentes ciclos de desenvolvimento também foi mencionado como prática utilizada por 71% dos participantes da pesquisa da Version One [90]. Isso se deve principalmente devido a organização principal de múltiplas iterações de desenvolvimento, permitindo uma evolução constante do aprendizado da equipe e do desenvolvimento do produto através de múltiplas releases.

Organização e Controle

A organização em projetos de desenvolvimento de aplicativos para dispositivos móveis também foi citada como influência sob o aspecto de demanda e entrega do produto, relacionando também o controle do andamento dos projetos. *"Creio que é de extrema importância, possibilita uma melhor organização das tarefas e também um melhor acompanhamento da equipe"* (Participante 8).

Além do acompanhamento e organização dos projetos, também foi levantada a questão de engajamento e comprometimento com os projetos. *"Auxilia bastante a manter a organização e se cobrar mais no desenvolvimento geral da aplicação"* (Participante 16).

A adoção da metodologia apropriada melhora o planejamento e controle do projeto, gerando também uma melhoria na qualidade e um produto melhor [104].

Melhoria Contínua

Melhoria contínua através de constantes iterações, planejamento iterativo e revisão de iterações também foi levantado como opinião positiva do uso de métodos ágeis em desenvolvimento de aplicativos. *"Acredito que é fundamental para o desenvolvimento, principalmente pelos constantes reviews em que facilitam a solução de problemas e redefinição do escopo do projeto (se necessário) [...]"* (Participante 9).

Quanto maior a experiência em desenvolvimento ágil, mais benefícios, a busca pela melhoria contínua aumenta a excelência técnica dos membros de uma equipe [88]. O desenvolvimento ágil favorece o desenvolvimento de aplicativos, tornando o processo mais eficiente, diminuindo risco do projeto por que é possível identificar e eliminar falhas ou comportamentos indesejados com rapidez e precisão.

Comunicação

Comunicação também foi apontada como opinião positiva do uso de métodos ágeis em desenvolvimento de aplicativos. *"Muito bom, faz com que o projeto fique mais organizado, facilita a comunicação entre o grupo e nos torna mais críticos"* (Participante 10).

6.4.4 Práticas de desenvolvimento

Um dos aspectos avaliados durante o estudo de campo foi a percepção dos alunos sobre alguns dos papéis utilizados em desenvolvimento ágil. O *Scrum Master* SM e o *Product Owner* PO. Dentre os estudantes, a maioria 61,43% concordam que ter alguém como PO é importante dentro do método proposto. Entretanto 5,71% discordam que esse papel seja necessário dentro do método. No que se refere ao papel de SM 77,14% concordam que ter alguém que o papel do SM é importante dentro do método proposto. Entretanto 7,14% discordam que esse papel seja necessário.

Quanto as práticas ágeis utilizadas pelos participantes do estudo de campo, 95,24% reportou ter utilizado a prática de reuniões diárias, seguido de 76,19% que utilizaram o quadro kanban para controle das atividades em progresso e atividades concluidas e 42,86% que utilizaram ambos o planejamento iterativo e pequenas releases. Seguido de programação em par 33,33%, seguida de refatoração, integração contínua e uso de gráfico de *burn down* para controle da execução das atividades com 19,05% cada). A práticas menos utilizada foi *Test Driven Development* TDD 9,52%, conforme apresentado na Figura 6.3.

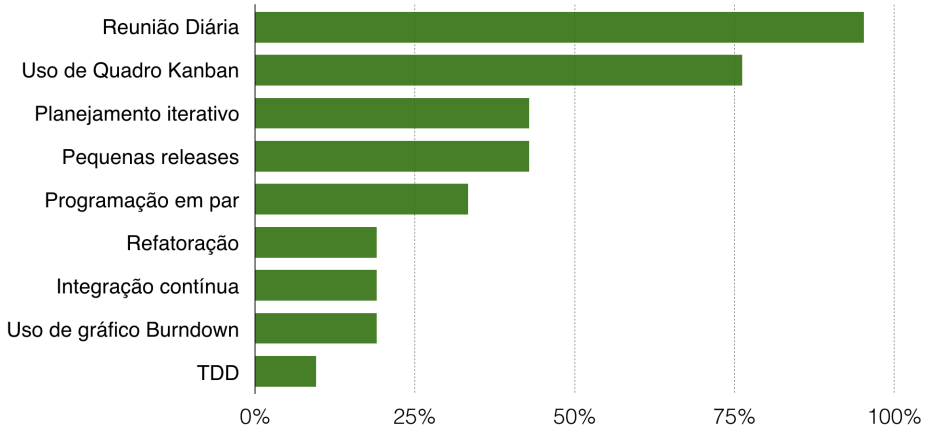

Figura 6.3: Práticas ágeis utilizadas no estudo de campo

As práticas utilizadas pelos alunos durante o desenvolvimento dos projetos são apresentadas na Figura 6.4. Dentre as práticas citadas, a mais utilizada pelos alunos foi modelagem UML 74,29%, seguida de *design patterns* 34,29%. Alguns respondentes não informaram as práticas utilizadas 8,57%.

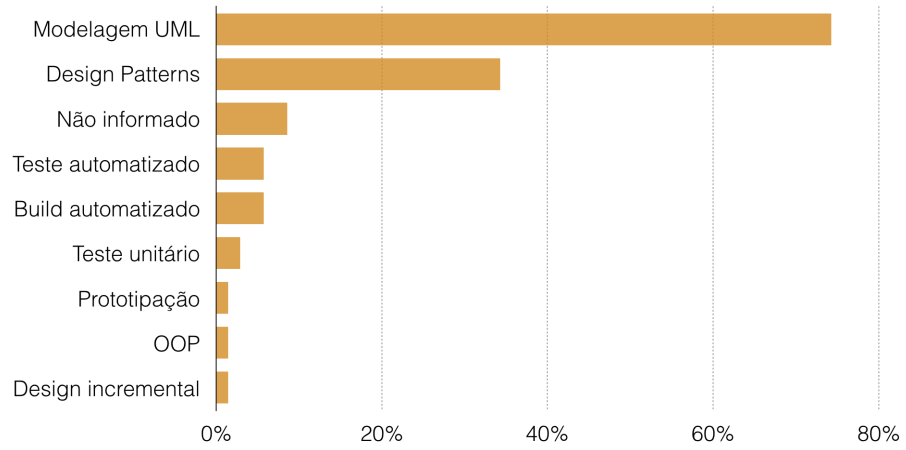

Figura 6.4: Práticas de desenvolvimento utilizadas para desenvolver aplicativos

6.4.5 Ferramentas utilizadas

No que refere-se as ferramentas utilizadas para o desenvolvimento dos projetos, elas resumem-se basicamente em controle de versão de código, desenvolvimento, organização dos projetos, comunicação e design. Devido ao curso ser de desenvolvimento iOS, a ferramenta xCode é utilizada por 100% da amostra representada. A Tabela 6.3 abaixo descreve a lista de todas as ferramentas utilizadas durante o desenvolvimento dos aplicativos.

Tabela 6.3: Ferramentas utilizadas no desenvolvimento de aplicativos

Ferramenta	Percentual
Github	81,43%
Bitbucket	61,43%
Trello	58,57%
Slack	25,71%
Photoshop	12,86%
Dropbox	11,43%
Astah	10,00%
Sourcetree	8,57%
Google Drive, Parse	7,14%
Kanbanflow, Sublime Text	5,71%
Evernote, Google Docs, Runrunit	4,29%
After Effects, Google Analytics, Pixelmator, Sketch	2,86%
Basecamp, Copy, Crashlytics, Final Cut, fuidUI, GanttProject, HipChat, Invision, Joomia, Kanban, Kanbanize, Keynote, MySQL, Navicat, OmniFocus, OmniPlan, Proto.io, Testflight, Toad, VIM, Wireshark, X-Mirage, Affinity Designer	1,43%
Não informado	1,43%

Dentre os diversos estudantes, foi relatado o uso de 44 diferentes ferramentas durante o curso. As cinco ferramentas mais utilizadas são Github 81,43% e Bitbucket 61,43% (Ferramentas de controle de versão), seguida do Trello 58,57% (ferramenta para organização de projetos ágeis), Slack 25,71% (ferramenta para facilitar a comunicação) e Photoshop 12,86% (ferramenta utilizada para *design* de aplicativos).

6.4.6 Percepção sobre diferenciais do método

Um dos questionamentos levantados neste estudo de campo, foi entender a percepção de quais os aspectos que fazem o método proposto ser diferente de outras formas de aprendizado como aula expositiva e métodos tradicionais de ensino. Neste sentido, uma das questões do estudo de campo se relaciona ao aspecto de como esse método poderia ser comparado ou como ele se diferencia de outros métodos.

As diferenças relatadas pelos participantes foi a realização de projetos reais, nos quais eles se envolvem e tem liberdade para a busca do conhecimento aumentando a sua motivação e envolvimento. Devido ao fato do método utilizar projetos reais, desafiando e estimulando os alunos, a motivação se dá entre os estudantes devido ao fato de eles terem mais liberdade de escolher o que aprender e a forma de aprender, tornando também os alunos mais engajados e envolvidos implicando na busca do conhecimento devido ao estímulo que o indivíduo tem em pensar e encontrar soluções diversas. A seguir são descritas as principais características que diferenciam o método proposto dos demais métodos reportados para ensino e desenvolvimento de aplicativos para dispositivos móveis.

Motivação

A diferença na motivação foi apontada como uma das diferenças entre o método e as outras abordagens de ensino como por exemplo métodos tradicionais.

> Para mim o método não se compara as outras formas de aprendizado. É outro paradigma, os ganhos são muito maiores, em compensação os desafios são maiores também. Exige muito mais motivação e determinação por parte dos aprendizes, diferente de métodos tradicionais onde o aluno é passivo e seu único dever é assistir ás aulas (Participante 18).

Foi relatado que o método incentiva mais o aprendizado através da motivação gerada pela liberdade de escolha dos desafios de problemas reais.

> No método tradicional, temos o professor como alguém um pouco distante na maioria das vezes, o que não acontece com o método. E também dá mais vontade de aprender e desenvolver os projetos do que em métodos tradicionais, pelo fato de que temos mais liberdade em escolher como aprender e o que aprender (Participante 2).

Diversos estudos [4,15,76,106,118,129] relatam que a motivação dos estudantes é positivamente influenciada pelo aprendizado de desenvolvimento de projetos de aplicativos para dispositivos móveis.

O uso de abordagem baseada em desafios integrada ao desenvolvimento ágil torna os membros das equipes muito mais envolvidos, proporcionando aos componentes das equipes a mesma carga de responsabilidade e de direitos, diferentes dos métodos tradicionais. O método deixa o aluno muito mais curioso, disposto e com vontade de realmente fazer as tarefas, pois ao longo do uso da metodologia fica muito claro o quanto o grupo evolui, fazendo com que os alunos percebam a complexidade daquilo que estão trabalhando.

Projetos Reais

O fato de os desafios representarem projetos reais também foi apontado como uma das diferenças entre o método e os métodos de ensino tradicionais ou métodos que os estudantes tiveram outras experiências. *"[...] Principalmente pelo fato de ser um desafio e não apenas um exercício que no método tradicional você iria resolver sem saber o porque"* (Participante 3).

Cinco estudos [7,93,112,116,120] apresentam sessões práticas como influência positiva no ensino e aprendizado de desenvolvimento de aplicativos para dispositivos móveis. Um fator interessante encontrado é que mesmo tendo projetos reais a serem implementados, alguns relatos posicionam a questão de aulas complementares como algo que agrega valor ao método.

O método incentiva o aprendizado, pois os alunos tem que buscar soluções para problemas reais em forma de desafio, iniciando um ciclo de aprendizado de técnicas e ou conteúdos mais profundos de forma independente e eficaz. *"Acho que o método tem como principal vantagem desafiar/estimular os alunos, além de possibilitar a aplicação dos conteúdos estudados [...]"* (Participante 11).

Busca do Conhecimento

A busca do conhecimento foi algo relatado também como uma diferença entre o método e os meios tradicionais de ensino. *"[...] estimula o indivíduo a pensar mais e procurar encontrar soluções por si mesmo, sem esperar a ajuda de alguém"* (Participante 9). Estudantes relatam que este tipo de abordagem exige uma maior de dedicação o que influencia a uma maior procura pelo aprendizado.

O tipo de ambiente utilizado em ambientes com pedagogia ativa tem como objetivo o desenvolver uma configuração educacional na qual o aprendizado dos estudantes seja baseado na concepção construtivista de aprendizado e suas aplicações pedagógicas [31].

Envolvimento

O envolvimento dos estudantes também foi relatado como um aspecto que existe no método proposto e que algumas vezes não é encontrado nos métodos tradicionais.

> O grande diferencial do método em relação aos outros métodos é o papel do aluno no processo, no processo o aluno possui uma participação muito mais ativa, visto que ele tem que ir em busca do próprio aprendizado, algo que também desloca o papel do professor, visto que esse deixa de ser o núcleo do processo e passa a se tornar um auxiliar do aluno no processo de aprendizado (Participante 13).

O aumento no envolvimento dos estudantes já tinha sido mencionado por Johnson e Adams [60], quando realizaram um estudo sobre a abordagem baseada em desafios e como resultados eles observaram um aumento no engajamento dos estudantes, um aumento no tempo investido para trabalhar nos desafios, aplicação criativa de tecnologia e uma maior satisfação dos estudantes o que melhorou também o aprendizado.

Devido ao estimulo para que o estudante busque o conhecimento, este tipo de abordagem gera uma curva de aprendizado que depende do interesse do aluno. Os alunos tem uma liberdade maior de aprender, com objetividade e propósito o que melhora também a motivação. Entretanto,

este método requer muita disciplina, justamente pela liberdade que ele proporciona. Este tipo de abordagem tem potencial de sucesso, dadas as condições certas de sua aplicação.

6.4.7 Características que fazem deste método uma solução viável

Diversas são as características que fazem deste método uma solução viável, são elas: Agilidade, Aprendizado contínuo, Auto-aprendizado e busca do conhecimento, Colaboração e sessões práticas, Criatividade, Foco, Liberdade, Melhoria contínua, Motivação, Organização e planejamento, Prevenção de problemas, Processo de pesquisa e Trabalho em Equipe. Todos estes fatores vão de encontro aos resultados encontrados no mapeamento sistemático da literatura, o que nos gera indicativos de que o método é viável durante a sua utilização.

Agilidade

Devido aos projetos de desenvolvimento de aplicativos para dispositivos móveis geralmente terem uma curta duração, o método permite que um projeto seja definido rapidamente *"[...] projetos para mobile são mais curtos. Com o método podemos definir um projeto rapidamente e começarmos a trabalhar nele com todas (a maioria) das questões tratadas"* (Participante 43).

Entretanto existem projetos que exigem um escopo maior, o que também é um desafio devido as atualizações de plataformas e sistemas operacionais. As constantes mudanças são um problema crítico em ambientes de desenvolvimento de aplicativos para dispositivos móveis [127], devido ao continuo fluxo de atualização de tecnologias, o que faz o desenvolvimento ágil ser muito eficiente para tratar novas tecnologias [52].

Aprendizado Contínuo

Os ambientes de desenvolvimento de aplicativos evoluem constantemente. Devido ao método ser uma junção entre CBL e desenvolvimento ágil, o processo permite o inicio de uma *"Big Idea"* e uma evolução natural que beneficia o aprendizado. *"O fato de seguir um processo com etapas que levam em conta não só o desenvolvimento da app, mas leva em conta etapas de aprendizagem e entendimento do problema e da solução a ser proposta"* (Participante 7).

O importante é que o método leva em conta as etapas de aprendizagem, entendimento do problema e proposta de solução. *"A constante procura pelo aprendizado em cada projeto para posteriores aperfeiçoamentos em projetos futuros"* (Participante 8).

O aprendizado através de desafios incentiva a criação de soluções para problemas reais melhorando o aprendizado, proporcionando um ciclo de aprendizagem e aperfeiçoamento constante.

> O mundo do desenvolvimento *mobile* é amplo e cada dia trás mais novidades. Aprender através de desafios e ser encorajado a criar soluções para problemas reais é com certeza a melhor maneira de aprender. Principalmente em desenvolvimento *mobile*. Desde que comecei a aprender programação, este foi o jeito mais fácil e rápido que já tive para aprender (Participante 28).

De acordo com resultados apresentados por McNely et al. [77], o Scrum facilita a prática de desenvolvimento coletivo através da intermediação de diferentes dinâmicas de colaboração e aprendizado contínuo.

Auto-aprendizado e Busca do conhecimento

O método provê o caminho para busca de conhecimentos através dos desafios, o que estimula o auto-aprendizado em diferentes circunstâncias.

> O fato de que ele gera alguns nortes para o grupo se basear e saber o que deve fazer, sem falar que aprendemos bastante por conta própria o que nos leva a aprender de uma forma na qual memorizamos melhor os conteúdos (Participante 27).

Foi relatado que a consolidação dos conceitos aprendidos se torna mais efetiva quando eles são aplicados em uma necessidade real. De acordo com a avaliação dos alunos isso também gera um melhor nível de aprendizado.

> A consolidação dos conceitos aprendidos se torna mais efetiva quando eles são aplicados em uma necessidade real. Além do fato do método estimular o auto-aprendizado, algo cada vez mais necessário em uma área que muda muito rapidamente (Participante 54).

O método instiga os alunos a buscarem mais conhecimento e não se limitarem, assim fazendo o aprendizado mais efetivo. Nesse sentido, dois fatores são muito importantes para fazer com que o método seja uma solução viável: a condição de definir ou analisar situações do mundo real em uma sequência lógica, bem como na busca do saber, questionando e refletindo profundamente sobre cada problema ou solução tecnológica encontrada. No desenvolvimento de aplicativos para dispositivos móveis as opções, conteúdos, estilos, e publico alvo são muito variados, e o método proporciona que cada um busque conhecimento independentemente e sobre os assunto que acha relevante e que são necessários para seus projetos. *"O método permite um aprendizado de forma diferente, fazendo com que cada um aprenda de acordo com seus interesses trazendo uma cooperação entre instrutores e alunos"* (Participante 65).

Colaboração e Sessões Práticas

A metodologia proporciona uma maior colaboração entre as equipes devido ao enfoque nas atividades práticas para desenvolver problemas reais, permitindo um aprendizado baseado na interação entre pessoas. *"É uma solução que incentiva o aprendizado e a colaboração"* (Participante 34).

Foi relatado que o enfoque nas atividades práticas e colaboração também favorece o desenvolvimento de aplicativos. *"Acho que o foco prático e colaborativo do método (permite a troca de ideias entre colegas) favorece o desenvolvimento mobile"* (Participante 15).

Este tipo de abordagem tem enfoque em atividades práticas, com disponibilidade dos instrutores o que facilita troca de conhecimento e colaboração entre todas as partes do processo, gerando um aprendizado baseado na interação e colaboração de pessoas com conhecimento em uma determinada área.

Criatividade

Este método incentiva a criatividade para procurar soluções e novas ideias para aplicativos. Aprender através de desafios e ser encorajado a criar soluções para problemas reais é uma forma de aprender. *"Criando desafios e deixando a criatividade livre força que os alunos tenham que aprender outras coisas diferentes das ensinadas no curso e assim os alunos evoluem no seu ritmo"* (Participante 46).

Aprender através de desafios e ser encorajado a criar soluções para problemas reais incentiva a criatividade para procurar soluções e novas ideias para aplicativos.

Foco

O processo de organização criativa e do desenvolvimento gerado pelo método auxilia os estudantes a terem foco na solução a ser desenvolvida. *"O método ajuda a organizarmos as nossas ideias, dando foco no que realmente queremos produzir"* (Participante 55).

Liberdade

O método não limita os alunos na busca de conhecimento, eles tem liberdade e flexibilidade para buscar o conhecimento da forma que for necessária, com os recursos necessários e a qualquer momento. *"Ter a liberdade de se organizar e programar de uma maneira mais agradável ao grupo de desenvolvimento de software. Principalmente a liberdade de poder trabalhar como e no que eu quiser"* (Participante 26)

Os alunos tem a liberdade para construir e também desconstruir suas soluções de acordo com necessidades de problemas reais. *"Criando desafios e deixando a criatividade livre força que os alunos tenham que aprender outras coisas diferentes das ensinadas no curso e assim os alunos evoluem no seu ritmo"* (Participante 47).

Melhoria Contínua

O processo de pesquisa através das *"Guiding questions"* facilitam e propiciam o início de um ciclo de melhoria contínua. Além disso, a possibilidade de mudanças ao longo do trabalho, divisão de tarefas e elucidação do que é necessário ou não para o projeto favorecem um processo de melhoria contínua. Os ciclos dos sprints e as atividades de revisão da sprint e retrospectiva da sprint também estão aliadas ao processo de melhoria contínua.

> Aplicativos móveis precisam ser muito discutidos visando o melhor *design* e usabilidade possível pra enriquecer a experiência do usuário e método permite um ciclo constante de ideias, discussões, desenvolvimento e avaliação do resultado, que visa sempre a evolução do produto (Participante 5).

A experiência em desenvolvimento ágil favorece um aumento de benefícios, e a busca pela melhoria contínua aumenta o nível técnico dos membros de uma equipe [88]. Isso também se da no processo de utilização do método devido as constantes revisões onde é possível obter *feedback* dos

instrutores e dos colegas para melhoria do nível técnico dos membros das equipes, como também evolução dos produtos desenvolvidos.

Motivação

Foi relatado também que o método gera motivação para o desenvolvimento de aplicativos, isso se deve a aspectos como liberdade para criar e desenvolver de acordo com o interesse dos estudantes. *"Os estudantes se dedicam mais quando recebem desafios"* (Participante 20).

Isso faz com que os alunos estejam engajados para resolver problemas do mundo real através de soluções práticas. *"Fazer com que os alunos sintam-se inspirados e busquem por si próprios o conhecimento"* (Participante 64).

Pesquisas confirmam que a motivação dos estudantes tem uma influência positiva em ambientes de ensino e desenvolvimento de aplicativos para dispositivos móveis [4, 15, 76, 106, 118, 129].

Organização e Planejamento

Ocorreram diversos relatos que o método proporciona uma melhor organização dos projetos, desde a concepção inicial da ideia a ser desenvolvida. *"O fluxo do aplicativo é extremamente mais organizado, aliando um passo-a-passo à um (schedule) eficiente"* (Participante 39).

Além disso, proporciona também suporte na etapa de definição de escopo e organização do projeto como um todo, permitindo um foco ao que deve ser desenvolvido. Permitindo também uma preparação dos estudantes com práticas do mercado de trabalho.

> Pois a partir do seu uso, conseguimos nos organizar melhor, focar nosso objetivo e ao mesmo tempo estamos nos acostumando com metodologias nos preparando ao mercado de trabalho (Participante 25).

Além da organização, o planejamento também foi citado como favorecido pelo uso do método, devido a capacidade de poder gerar estimativas para o desafio proposto, definir prazos, metas e resultados além da prevenção de dificuldades. *"É possível se organizar completamente, reduzindo a perda de tempo fazendo coisas que no futuro serão alteradas por falta de planejamento"* (Participante 48).

Processo de Pesquisa

O método permite a análise de uma ideia desde sua concepção inicial até a sua solução. Isso se dá através de um processo de pesquisa que é realizado durante as atividades de *"guiding questions"* o que gera uma proposta de solução antes que a mesma seja desenvolvida.

> O que mais me ajuda são as Guiding questions, pois estudamos muito como fazer o app e de cada questão surgem várias outras, nos ajudando a prever diversas dificuldades que poderemos enfrentar e, sendo assim, buscar as devidas soluções (Participante 23).

O processo de pesquisa através das *"Guiding questions"* permite uma melhor avaliação e prevenção de problemas que podem ocorrer. *"Acredito que toda solução mobile parte de uma grande*

idéia acompanhada de um problema, a idéia é através de questões guias achar uma solução para o problema em questão" (Participante 31).

Os estudantes relatam que método é útil, pois com o seu uso eles precisam pensar em diversas situações em que o aplicativo pode estar presente, nas diversas maneiras de implementar a ideia, o que muitas vezes exige um processo de pesquisa de forma a entender o que precisa ser aprendido e também os recursos necessários para a implementação de uma idéia. *"O modo de tratar o início de uma ideia, sabendo o que é possível ser feito, como deve ser feito e, principalmente, o que deve ser feito"* (Participante 18).

Trabalho em Equipe

A forma como o método é proposto além de facilitar a colaboração, também proporciona um aprendizado baseado no trabalho em equipe, gerando um comprometimento entre os membros de cada equipe. *"O desafio de realizar o app em equipe, e o compromisso entre os integrantes"* (Participante 63).

É importante ressaltar que encontramos um total de 2,85% das respostas que discordam sobre o uso do método, que preferem o método tradicional de ensino, através de apresentação de conteúdo e exercícios com material didático.

6.4.8 Vantagens e desvantagens percebidas pelos participantes no uso do método para desenvolvimento de aplicativos

As vantagens no uso do método são diversas, que incluem agilidade, dinâmica e produtividade, controle do ciclo de desenvolvimento, estimulo a criatividade, melhoria continua através de múltiplas iterações, melhoria da parte teórica, qualidade e velocidade de desenvolvimento. Entre as principais desvantagens no uso do método, é que se ele for utilizado em um pequeno número de sprints com iterações curtas ele não é efetivo, além disso em alguns casos onde os desafios são muito rápidos pode fazer com que algumas tarefas sejam postergadas. Outra desvantagem é em relação a papéis no time, que a escolha do membro errado para assumir o papel de Scrum master pode impactar o desenvolvimento do projeto. Além disso, existem alguns pontos que pode ocorrer redundância durante o uso do método. Dessa forma, a seguir são descritas as vantagens e desvantagens percebidas pelos participantes no uso do método.

Vantagens

Uma das vantagens citadas é a agilidade ganha no desenvolvimento dos projetos. Devido ao fato deste método ser uma integração entre desenvolvimento ágil e uma abordagem de aprendizado baseada em desafios este método promove um processo contínuo de aprendizado. *"[...] a introdução do desenvolvimento ágil no CBL aumenta a chance de aprendizado por se tratar de um processo incremental"* (Participante 8).

O método permite um ambiente dinâmico e produtivo devido ao integração de abordagem de aprendizado baseado em desafios e desenvolvimento ágil. Dessa forma valoriza a aproximação,

comunicação e colaboração dos envolvidos com o processo sendo, assim, propício a uma técnica de aprendizado baseada em desafios.

"[...] permite um ambiente mais dinâmico e produtivo, desta forma podemos interagir facilmente com o grupo ou professores e resolver qualquer problema que apareça durante o desenvolvimento" (Participante 28).

Diversos relatos foram realizados sobre o método apoiar o controle do desenvolvimento dos projetos, através da organização do que precisa ser realizado, de forma a facilitar a verificação de progresso e definição de prioridades e estimativas.

> Uma vantagem é, ao listar as etapas do desenvolvimento e suas prioridades, obtêm-se uma melhor aproximação do tempo de desenvolvimento de cada etapa, o que ajuda a distinguir o que é viável fazer e o que não é conforme o tempo (Participante 34).

Devido a natureza do método ser uma combinação de aprendizado baseado em desafios e desenvolvimento ágil, ele realiza um ciclo de desenvolvimento de uma idéia o que também estimula a criatividade. Outro ponto chave foi que ocorreram relatos que o processo de pesquisa do método gera uma melhor documentação de requisitos com suas respectivas prioridades, provendo um melhor rumo ao desenvolvimento dos projetos.*"[...] processo de elaboração de Perguntas-Guia, auxilia na criação do Backlog"* (Participante 12).

A melhoria contínua também foi levantada como um ponto importante do método, seja pelas *daily meetings*, seja também pelas sprints reviews que proporcionam *feedbacks* importantes entre diferentes equipes, o que permite melhoria contínua no desenvolvimento do projeto e do produto. *"Ciclo diário de review sobre o realizado/à fazer/impedimentos. Melhoria contínua das ideias em um curto prazo de tempo"* (Participante 5).

A organização do processo também foi levantada como uma das vantagens do método devido ao suporte proporcionado deste a concepção da idéia inicial até a concretização do desenvolvimento da solução final. *"Acho que o método só vem a acrescentar no projeto, na organização, prevenir falhas, como realizar as tarefas, que ferramentas utilizar, etc"* (Participante 23).

A possibilidade de realizar diversas iterações de planejamento proporcionando o ciclo contínuo de melhoria e aprendizagem também foi citado como uma das vantagens do método. *"[...] o fato de termos varias iterações durante os desafios e também ter a liberdade de irmos em busca do conhecimento"* (Participante 41).

Outras vantagens do método que foram citadas incluem melhora na parte teórica e produção de software, aumentando a qualidade e velocidade de desenvolvimento. *"É vantajoso no sentido de que melhora a parte teórica do grupo e agiliza a produção do software, de forma que visa uma produção de software com qualidade e velocidade"* (Participante 27).

Desvantagens

Uma das desvantagens citadas pelo uso do método é que se for usado em um período muito curto de tempo, com duração total do projeto entre duas e três semanas, o método não é efetivo. Para ele ser efetivo os projetos precisam ter uma duração maior que duas a três semanas. *"[...] Uma desvantagem é se for um desafio, como por exemplo de duas semanas, pode acabar por atrapalhar o processo de um desenvolvimento muito rápido em duas semanas "* (Participante 64).

Nesse contexto, os projetos precisam ter um prazo mínimo com algumas iterações, caso contrário tarefas importantes no projeto podem perder prioridade como por exemplo o que foi citado pelo participante 51: *"Devido aos prazos curtos a fim de ter uma experiência de desafio, muitas tarefas definidas são deixadas de lado ou preteridas"* (Participante 51).

Foi relatado que a escolha do membro da equipe para assumir o papel de Scrum master pode impactar o desenvolvimento do projeto. *"[...] O que muitas vezes não se faz possível é o papel de Scrum master que pode ser assumido por um integrante do grupo que não atue bem neste papel"* (Participante 69).

Também foi relatado que o processo de criação de uma idéia desde sua concepção inicial até sua implementação exige um tempo para acontecer, e este tempo poderia ser mais focado no desenvolvimento do código dos aplicativos. *"[...] Despende tempo organizando e fazendo burocracias, que poderiam estar sendo utilizados para desenvolver e aprimorar códigos"* (Participante 24).

Outra desvantagem mencionada é uma possível redundância em alguns pontos do método. Exemplo durante o processo de pesquisa na listagem dos recursos das perguntas guia e também na possível repetição de trabalho dentro do método.

> Um ponto que não gosto é listar os recursos necessários para responder as perguntas referente ao que o app deverá ter, ou como o app mostrará certas informações. Isto porque, na minha opinião, quando analisado pelo custo-benefício, não vale a pena o tempo gasto (Participante 26).

Nesse sentido o método precisa ser bem entendido, tanto na parte de concepção e pesquisa sobre as soluções até a implementação das mesmas. *"Se não for bem entendida cada metodologia, as vezes pode se repetir o mesmo trabalho em uma ou outra parte"* (Participante 59).

6.4.9 Percepção dos participantes em relação a utilização do método

Este estudo buscou coletar a percepção dos estudantes em relação a evolução do aprendizado, através de uma auto-avaliação sobre a sua evolução no curso, com uma classificação de 0 a 10 (máximo), sobre o conhecimento em desenvolvimento de aplicativos para dispositivos móveis ANTES do curso e APÓS o curso. Os resultados são apresentados na Tabela 6.4.

Neste contexto, foi possível observar que considerando a média de valores, antes e após o treinamento, de acordo com a percepção dos estudantes o conhecimento deles aumentou em 5,23 pontos, ou seja representando uma percepção de um aumento médio de 52% no nível de conhecimento de desenvolvimento de aplicativos.

Tabela 6.4: Conhecimento em desenvolvimento de aplicativos antes e após o treinamento

Média anterior	Desvio anterior	Média posterior	Desvio posterior
2,8	2,59	8,03	1,48

Outro ponto importante, refere-se aos desafios encontrados no desenvolvimento de aplicativos móveis, os quais são resumidos em Interface e Experiência de Usuário, Performance, Atualização constante, Usuários, Plataformas e Velocidade [101]. Comparando os desafios encontrados este estudo de campo com os desafios encontrados por Gordon [42], foram encontradas semelhanças em diversos aspectos como Interface de Usuário e Usabilidade, *Hardware*, Manipulação de dados, cooperação de dispositivos e questões de programação.

Em relação ao uso de práticas ágeis em ambientes de desenvolvimento de aplicativos, este estudo de campo apresenta diversos desafios também encontrados na literatura, dentre eles: Desempenho, Transparência, Velocidade, Entrega, Iterações, Organização e Controle, Melhoria Contínua e Comunicação. Neste sentido, as principais práticas ágeis utilizadas pelos participantes, incluem reuniões diárias e kanban e as práticas ágeis menos utilizadas foram TDD e build automatizado. Além disso, foram utilizadas outras práticas de desenvolvimento de software, dentre elas modelagem UML e *design patterns*. Um dos pontos a serem trabalhados no estudo de caso é entender qual o principal uso da modelagem UML como também quais os *design patterns* mais utilizados.

No que refere-se as ferramentas utilizadas para o desenvolvimento dos projetos, elas resumem-se basicamente em controle de versão de código, desenvolvimento, organização dos projetos, comunicação e design. Devido ao curso ser de desenvolvimento iOS, a ferramenta xCode é utilizada por 100% da amostra representada.

6.5 Lições Aprendidas

Os resultados obtidos no estudo de campo mostram que o método proposto é viável. O estudo de campo apresentado neste capítulo contribui para a identificar um conjunto de lições aprendidas, as quais são descritas a seguir.

- **Lição 1: Estratégia pedagógica baseada na resolução de desafios**
  O modelo de abordagem baseada em desafios, envolveu as experiências dos alunos, estimulando o desenvolvimento de uma aprendizagem ativa e autônoma na resolução de problemas do mundo real. Ao mesmo tempo em que exige instrutores mais preparados, torna a atividade de aprendizado mais aberta e flexível. Esta estratégia não é a única que propicia um aprendizado baseado em experiências reais, mas se apresenta como uma opção para o desenvolvimento de ambientes construtivistas de aprendizagem.

- **Lição 2: Suporte de instrutores especialistas é um diferencial**
  Considerando a diversidade de desafios encontrados em ambientes de ensino e desenvolvimento de aplicativos, o apoio de instrutores especialistas é um elemento fundamental para o suporte

dos estudantes. Os instrutores ajudam os alunos a encontrarem os caminhos para resolver suas dúvidas. O contra-ponto disso é que se o instrutor não for um especialista, é possível que ocorra um desequilíbrio no processo de aprendizado.

- **Lição 3: Necessidade de postura ativa e autonomia dos estudantes**
  Geralmente, os estudantes apresentam uma postura passiva, onde o professor passa o conteúdo durante toda uma disciplina ou treinamento. Em abordagens de pedagogia ativa os alunos precisam estar engajados na execução das atividades.

- **Lição 4: A importância do processo de pesquisa**
  O processo de pesquisa permite uma melhor avaliação sobre os problemas a serem solucionados como também dos conhecimentos necessários, instigando a busca pelo conhecimento e autonomia. Nesse sentido, o processo colabora tanto para a evolução do aprendizado quanto para a evolução do produto. Diferentes práticas podem ser abordadas pelas equipes, as quais serão avaliadas em um estudo de caso.

- **Lição 5: A dinâmica dos ambientes de desenvolvimento de aplicativos**
  Os ambientes de desenvolvimento de aplicativos são dinâmicos. Diferentes plataformas, sistemas operacionais e *frameworks* facilitam o trabalho mas exigem conhecimento sobre como utilizar os recursos apropriadamente que também sofrem constantes atualizações. Dessa forma, ferramentas automatizam tarefas garantindo uma maior produtividade. Utilizar o controle de versão gerou alguns desafios no inicio dos projetos, principalmente para estudantes com menos experiência, entretanto facilitou bastante o funcionamento dos projetos. A utilização de *frameworks* e bibliotecas facilita a reutilização de componentes, ajuda a aumentar a produtividade da equipe, mas exige atualização. No meio do treinamento foi lançada uma atualização da versão do sistema operacional iOS o que exigiu flexibilidade e dinâmica das equipes para continuarem o desenvolvimento dos projetos.

- **Lição 6: Usar método ágil x agilidade das equipes**
  Quando se iniciou o projeto imaginava-se que o simples uso de uma metodologia específica implicaria em equipes ágeis. Entretanto, ser ágil não significa apenas utilizar uma metodologia, as equipes precisam pensar de forma simples e objetiva. Além disso, o que mais importa é o comprometimento com os princípios do desenvolvimento ágil, as práticas serão seguidas de forma natural. Neste sentido, é possível existir equipes auto-organizadas, aprendendo continuamente e gerando produtos incrementais através de entregas contínuas.

6.6 Limitações e ameaças a validade

Em diversos estudos existem ameaças que podem afetar a validade dos resultados. Este estudo possui algumas limitações. Primeiramente, alguns participantes possuem diferentes níveis de experiência, incluindo experiência em desenvolvimento de aplicativos e metodologias de desenvolvimento.

Além disso, nenhum dos estudantes teve contato anterior com metodologias de ensino baseadas em pedagogia ativa, motivo pelo qual eles podem ter uma percepção sobre a abordagem baseada em desafios. Neste sentido, estes fatores podem ter influenciado positivamente ou negativamente os resultados encontrados. Assim, os resultados encontrados devem ser classificados como indicativos e não podem ser generalizados.

6.7 Considerações Finais do Estudo de Campo

Através deste estudo de campo foi possível entender melhor as características da amostra envolvida no ambiente de estudo, como também entender e desmistificar o ambiente de desenvolvimento, identificando tanto as práticas de desenvolvimento utilizadas, como também as ferramentas. Além disso, foi possível ver na prática quais as diferenças em desenvolver aplicativos para dispositivos móveis, como também como se caracteriza o uso de práticas ágeis neste tipo de ambiente.

Além disso, foram encontrados os quatro diferenciais no uso deste método (Motivação, Projetos Reais, Busca do conhecimento e Envolvimento) os quais são intimamente relacionados. Através deste estudo de campo foi possível encontrar indicativos que o método é viável por causa de alguns fatores específicos, dentre eles: agilidade, auto-aprendizado e busca do conhecimento, colaboração e prática, criatividade, foco, liberdade, melhoria contínua, motivação, organização e planejamento, pesquisa e trabalho em equipe.

Foi possível observar durante o estudo de campo que acima de 97% dos participantes são favoráveis ao uso de uma abordagem baseada em desafios e práticas ágeis em ambientes de ensino e desenvolvimento de aplicativos para dispositivos móveis, entretanto ainda existe um pequeno percentual (menor que 3% que ainda prefere aprender seguindo a maneira tradicional de ensino.

Velocidade é um aspecto que foi mencionado como um dos diferenciais exigidos no desenvolvimento de aplicativos para dispositivos móveis, e também mencionado como uma das vantagens em usar desenvolvimento ágil para este tipo de ambiente. Da mesma forma, organização e controle foi mencionada como um dos aspectos importantes no uso de desenvolvimento ágil para aplicativos e também foi mencionada como um dos fatores que fazem o método proposto ser viável. Neste sentido, a melhoria contínua foi relatada como relevante no uso de ágil para desenvolvimento de aplicativos e também aparece como um dos fatores que fazem o método proposto ser viável.

Neste contexto, a motivação foi apresentada como um dos diferenciais do método e também como um dos fatores que fazem este método ser uma solução viável. No mesmo sentido, a busca do conhecimento foi apresentada como um dos diferenciais do método e também é um dos fatores que fazem este método ser uma solução viável. Portanto, este estudo de campo permitiu observar e analisar que o ensino e desenvolvimento de aplicativos funciona através de abordagem baseada em desafios e práticas ágeis.

7. ESTUDO DE CASO

Estudos de caso são adequados ao exame exploratório dos fenômenos ainda pouco estudados e que precisam ser investigados em seu ambiente de ocorrência [132]. Este estudo de caso foi realizado em quatro universidades, sendo a primeira universidade localizada na região Nordeste do Brasil, a segunda localiza-se na região Norte do Brasil, a terceira localiza-se na região Sudeste do Brasil e a quarta é localiza-se na região Sul do Brasil.

Como mencionado na Etapa 3 da subseção 3.1.3 o objetivo do estudo de caso é aplicar o método proposto em diferentes locais de pesquisa, e através da análise do uso do método proposto conseguir identificar oportunidades de melhoria, melhores práticas e recomendações de forma a descrever o método final. A seção 7.1 descreve o protocolo utilizado, como também a descrição dos locais do estudo de caso, a seção 7.2 apresenta os resultados encontrados no estudo de caso, a seção 7.3 apresenta as limitações na condução deste estudo e por fim a seção 7.4 apresenta as considerações finais deste estudo.

7.1 Protocolo

A fonte das informações fornecidas no processo de entrevista vão ser mantidas em sigilo por motivos de acordos confidenciais. Tanto os nomes das instituições quanto dos participantes não serão divulgados. Quaisquer documentos que eventualmente sejam parte desta tese não vão apresentar o nome das instituições envolvidas. Foi compartilhado um relatório de estudo de caso para as instituições envolvidas nesta pesquisa.

7.1.1 Procedimentos de campo

Antes da aplicação das entrevistas foi realizado um teste piloto com o questionário de forma a validar o instrumento de pesquisa. Para a realização das entrevistas, os pesquisadores possuíam uma cópia do manuscrito da entrevista e quando permitido o entrevistador usou gravador, caso não fosse permitido então anotações seriam realizadas durante o processo de entrevista. Os tópicos a serem discutidos foram apresentados aos entrevistados usando uma abordagem aberta para não induzir qualquer resposta. Entretanto, outros tópicos que talvez fossem considerados relevantes e não faziam parte do manuscrito poderiam ser explorados pelos pesquisadores. A duração máxima planejada para cada entrevista era de 30 minutos. Entretanto, dependendo da disponibilidade do entrevistado este tempo poderia ser extendido um pouco. Neste contexto, os materiais utilizados no estudo de caso foram: protocolos das entrevistas, bloco de anotações e gravador digital.

7.1.2 Questões de Pesquisa

As seguintes questões de pesquisa foram definidas para o estudo de caso:

(**RQ1**) *Como o desenvolvimento ágil influencia na implementação dos aplicativos?*

(**RQ2**) *Porque a plataforma de desenvolvimento pode influenciar no aprendizado?*

(**RQ3**) *Como fica o papel dos alunos e dos instrutores?*

7.1.3 Seleção das universidades e unidade de análise

A unidade de análise do estudo foi definida como sendo universidades que estejam envolvidas com ensino e desenvolvimento de aplicativos para dispositivos móveis. O que justificou a escolha das instituições, por conveniência, foi o fato de diferentes universidades participarem de um programa em comum a nível nacional no qual foi possível analisar o uso do método proposto em ambientes distintos.

7.1.4 Descrição dos locais de estudo

Semelhante ao estudo de campo, os ambientes das universidades onde foi realizado o estudo de caso são um projeto em parceria com universidades de diferentes regiões do país. O que muda em relação ao estudo de campo é o tamanho das turmas nas diferentes universidades, com turmas que variam de 50 a 100 alunos. A forma de organizar o conteúdo didático, tamanho dos grupos e dedicação diária segue a mesma descrição do estudo de campo.

A quantidade de instrutores também varia em cada um dos locais do estudo de caso, conforme descrito na Tabela 7.1.

Tabela 7.1: Número de instrutores por universidade

Universidade	Número de Instrutores
U1	7
U2	4
U3	4
U4	6

Da mesma forma que descrito no estudo de campo, no estudo de caso as equipes são suportados durante todo o ciclo da abordagem baseada em desafios integrada ao desenvolvimento ágil, apresentando retrospectivas ao final de cada iteração (geralmente sprints de duas semanas). Os instrutores monitoram as equipes e avaliam o aprendizado e evolução dos estudantes através das revisões e retrospectivas.

7.1.5 Coleta de dados

A coleta de dados foi constituída por fontes primárias através de entrevistas. Foram realizadas 32 entrevistas semi-estruturadas individuais com estudantes de diferentes equipes. A partir de um roteiro de questões o qual é apresentado no anexo D. A Tabela 7.2 apresenta a duração da coleta de dados em cada universidade.

Tabela 7.2: Coleta de dados nas quatro universidades

Universidade	Número de entrevistas	Tempo gravação	Número de transcrições
U1	8	5,75 h	8
U2	8	4,88 h	8
U3	8	5,1 h	8
U4	8	5,63 h	8

Cada universidade alocou um responsável para atuar como facilitador do processo, e para dar apoio nas entrevistas. Todas as entrevistas foram gravadas e transcritas, totalizando 183 páginas de transcrição.

7.1.6 Procedimentos para análise dos dados

A exploração do material de análise foi desenvolvida tomando como referência as recomendações de análise de dados apresentadas por Creswell [22]. Segundo o autor, este método tem como objetivo obter de forma sistemática, informações que permitam a realização de inferências a cerca do objeto de estudo. Para o desenvolvimento da técnica, foram seguidos os seguintes passos:

Passo 01 Organizar e preparar os dados para análise.

Passo 02 Ler todos os dados para obter uma percepção geral das informações e refletir sobre o seu significado.

Passo 03 Realizar uma análise detalhada com um processo de codificação, segmentando sentenças ou parágrafos em categorias e rotulando essas categorias com um termo.

Passo 04 Utilizar este processo para gerar uma descrição do local, das pessoas ou temas para análise.

Passo 05 Utilizar passagens narrativas para comunicar os resultados da análise.

Passo 06 Realizar uma interpretação ou extrair um significado dos dados.

Concluído o processo de coleta de dados empíricos, realizou-se a organização e preparação dos dados, cujo material constituiu o *corpus* da análise. De posse dos dados, realizou-se a leitura flutuante, deixando-se invadir pelas impressões e orientações do texto.

O *corpus* da análise foi organizado em quadros síntese com uma expressão gráfica da análise textual, onde as respectivas respostas recebidas dos sujeitos foram ordenadas sequencialmente a

cada um dos tópicos do instrumento de pesquisa na forma de indicadores construídos de acordo com as questões norteadoras e os objetivos deste estudo. A análise dos resultados é apresentada na Seção 7.2.

7.2 Análise de Resultados

Em relação à análise de dados, todas as entrevistas foram gravadas, transcritas e analisadas posteriormente, através de análise de conteúdo segundo Cresewell [22]. Após a realização das transcrições, uma leitura cuidadosa foi realizada, de modo a buscar a familiarização do pesquisador com os dados antes de iniciar a organização das categorias.

7.2.1 Dados da amostra

Os indivíduos participantes do estudo vem de diferentes regiões do brasil e diferentes cursos, principalmente de Ciência da Computação, Sistemas da Informação e Engenharia da Computação, conforme apresentado na tabela 7.3.

Tabela 7.3: Dados da amostra de participantes no estudo de caso

Univ.	Região	Participante	Idade	Curso	Sem.	Experiência (anos)	Tipo
U1	Nordeste	#1	23	Ciência da Computação	9	2	Acadêmica
U1	Nordeste	#2	27	Ciência da Computação	4	1	Indústria
U1	Nordeste	#3	21	Ciência da Computação	8	4	Acadêmica
U1	Nordeste	#4	26	Sistemas da Informação	6	4	Ambas
U1	Nordeste	#5	20	Ciência da Computação	4	2	Ambas
U1	Nordeste	#6	27	Sistemas da Informação	6	3	Ambas
U1	Nordeste	#7	23	Ciência da Computação	5	5	Acadêmica
U1	Nordeste	#8	23	Design	10	3	Indústria
U2	Norte	#9	19	Engenharia da Computação	4	3	Ambas
U2	Norte	#10	20	Ciência da Computação	6	4	Acadêmica
U2	Norte	#11	22	Sistemas da Informação	8	4	Indústria
U2	Norte	#12	21	Ciência da Computação	7	5	Ambas
U2	Norte	#13	25	Sistemas da Informação	4	1	Indústria
U2	Norte	#14	24	Engenharia da Computação	9	1,5	Indústria
U2	Norte	#15	22	Ciência da Computação	6	1	Acadêmica
U2	Norte	#16	23	Ciência da Computação	9	2	Acadêmica
U3	Sudeste	#17	26	Ciência da Computação	5	1	Acadêmica
U3	Sudeste	#18	20	Engenharia Elétrica	4	1,5	Acadêmica
U3	Sudeste	#19	25	Ciência da Computação	4	4	Acadêmica
U3	Sudeste	#20	20	Ciência da Computação	6	0,5	Indústria
U3	Sudeste	#21	27	Ciência da Computação	7	2	Acadêmica
U3	Sudeste	#22	20	Ciência da Computação	3	0,8	Acadêmica
U3	Sudeste	#23	20	Engenharia Elétrica	4	3	Acadêmica
U3	Sudeste	#24	21	Sistemas da Informação	5	1	Acadêmica
U4	Sul	#25	40	Análise de Sistemas	6	15	Indústria
U4	Sul	#26	21	Engenharia da Computação	7	3	Acadêmica
U4	Sul	#27	22	Design de Games	4	1	Acadêmica
U4	Sul	#28	22	Jogos Digitais	6	1,5	Indústria
U4	Sul	#29	21	Sistemas para Internet	5	1	Indústria
U4	Sul	#30	23	Engenharia da Computação	10	1	Acadêmica
U4	Sul	#31	23	Engenharia da Computação	6	4	Ambas
U4	Sul	#32	24	Sistemas da Informação	8	8	Indústria

Na tabela 7.3, a coluna *Sem.* representa o semestre que o aluno esta cursando na universidade. Em relação as experiências anteriores que os participantes tinham, metade dos participantes 50% tinha experiência acadêmica em torno de 2,26 anos. Outros 31,25% dos participantes tinham experiência na indústria em torno de 3,65 anos e 18,75% tinham experiência com ambas as áreas em torno de 3,5 anos. Quanto a experiência anterior com desenvolvimento de aplicativos, 28,13% dos participantes afirmaram ter tido alguma forma de experiência. Dentre estes, 12,50% tiveram experiência com Android, 9,30% com iOS, e 3,13% com Windows Phone e 3,13% com Web apps. No que refere-se a experiência anterior com desenvolvimento ágil, a maioria dos participantes 65,62% não tinha experiência anterior, destes 6,25% tem experiência com método tradicional de desenvolvimento.

Dos participantes que já conheciam desenvolvimento ágil, 18,75% tiveram algum tipo de experiência acadêmica, e 15,63% tiveram experiência da indústria. Neste contexto, a maioria dos participantes já tinha conhecimento anterior em diferentes linguagens de programação. Java 71,88%, C 59,38%, Javascript 50%, C++ 40,63%, PHP 37,5%, C# 28,13%, seguido de outras como Python e HTML, conforme apresentado na figura 7.1.

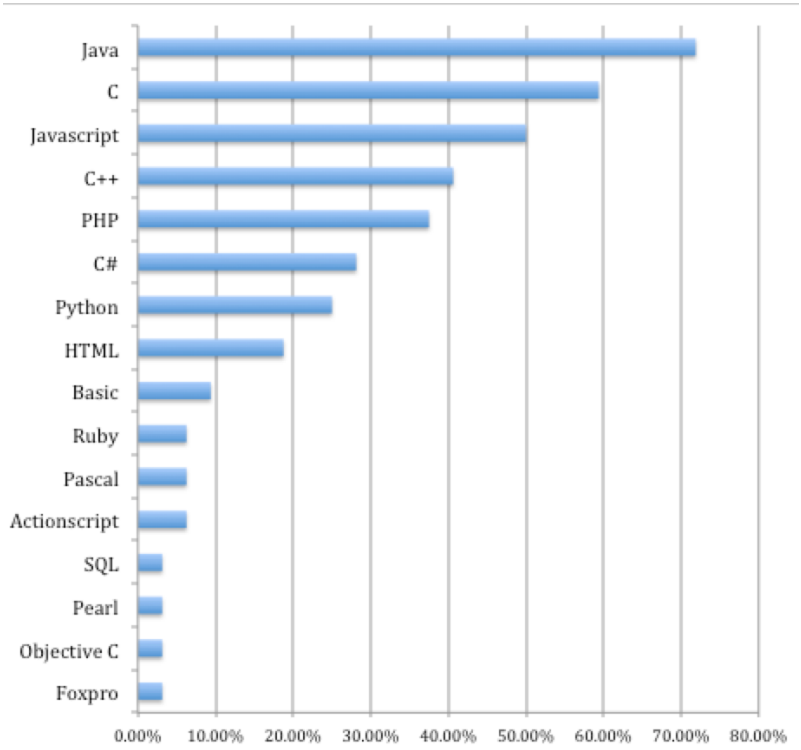

Figura 7.1: Conhecimentos de linguagens de programação anteriores ao treinamento

7.2.2 Influência do desenvolvimento ágil nos projetos de aplicativos

O desenvolvimento ágil mostrou uma influência positiva nos projetos de aplicativos sob diferentes perspectivas que incluem a rapidez na identificação de problemas, o *feedback* constante, organização, velocidade de desenvolvimento, transparência, comunicação, flexibilidade, desempenho e qualidade.

Rapidez na identificação de problemas

A rapidez na identificação de problemas facilita a correção e melhoria no ciclo de desenvolvimento dos projetos de aplicativos. O desenvolvimento ágil e se mostrou melhor quando comparado a experiências anteriores que participantes tiveram com o desenvolvimento tradicional, o qual dificulta um pouco a mudança durante o processo de desenvolvimento. De acordo com Avritzer *et al.* [9] os processos ágeis são destinados a aumentar a transparência na condução do desenvolvimento e esforços na resolução de problemas.

> Para *software* tem de ser algo mais ágil, rápido, interativo. O Scrum, junto com outro tipo de metodologia poder ser o ideal. Elas influenciam bem a medida do que você consegue realmente trabalhar em cima dela, o lean startup, você também pode mensurar e avaliar em cima dele. Para um projeto de empresa, um Scrum fica mais claro para as pessoas estarem focadas. O método cascata, outras mais rígidas, não conseguem guiar ou modificar um pouco o caminho se tiver alguma falha durante o processo. Influencia muito mais na rapidez, identificação de problemas e a partir dai você consegue consertar ou modificar o caminho do desenvolvimento. (Participante 1)

Feedback constante

O *feedback* constante influencia positivamente, pois a partir do desenvolvimento iterativo é possível se obter *feedback* de funcionalidades efetivamente entregues de forma a facilitar a identificação de melhorias que podem ser aplicadas nas iterações seguintes.

> Eu vejo um valor muito grande quando se tem o cliente próximo, porque, por exemplo, a metodologia cascata, que o cliente pede algo e tu vai desenvolvendo, chega ao final e entrega depois de muito tempo e o cliente vê que não é o que ele pediu. A metodologia ágil mantém sempre em contato com o cliente, a cada *release* faz uma reunião com o cliente, vê se é aquilo mesmo que ele precisa. O desenvolvimento de *software* fica mais assertivo por estar mais perto do cliente. Também a agilidade que tem para mudar, se não for bem esse o caminho, temos a margem para fazer melhor. (Participante 31)

O processo de aprendizagem de habilidades de programação requer *feedback* constante [4], por isso a participação dos instrutores nas atividades que incluem revisão de sprint e retrospectiva de sprint é importante. O *feedback* constante também ajuda no processo de desenvolvimento dos projetos. Através de práticas como entrega contínua, as equipes podem obter um *feedback* contínuo para o produto em desenvolvimento.

Velocidade

Ocorreram relatos sobre a melhora na velocidade do desenvolvimento, principalmente onde existe o comprometimento da equipe em concretizar o projeto. Velocidade também foi mencionada como

um fator que além de ser um desafio, influencia e é determinante em projetos de desenvolvimento de aplicativos para dispositivos móveis conforme descrito nas subseções 6.4.2 e 6.4.3.

> Para projeto de desenvolvimento de aplicativos de *software* influencia, porque é mais dinâmica, mais flexível e mais rápida. Flui melhor com o emprego dessa metodologia. Para acelerar o projeto, para a equipe estar mais unida em relação ao desenvolvimento e para concretizar o projeto, acho que serve bem nessa situação. (Participante 14)

A velocidade também é influenciada pela melhora na organização das iterações, influenciando o desempenho da equipe positivamente. *"Depois que começamos a utilizar Scrum, começamos a nos organizar melhor, tiveram as Sprints. Conseguimos melhorar bastante o desempenho da equipe. Influenciou na velocidade do desenvolvimento e na organização da equipe"*. (Participante 16)

Organização

O desenvolvimento ágil facilita a organização do projeto de desenvolvimento, pois os membros das equipes tem visibilidade sobre o que cada um esta trabalhando, além do que as múltiplas iterações permitem um planejamento mais modularizado e iterativo.

> [...] influencia na organização da equipe. No primeiro *challenge* que tivemos aqui, trabalhamos sem utilizar nenhum método e foi muito ruim. Cada um foi pro seu lado e depois viu que tinham três pessoas trabalhando na mesma coisa. Depois que começamos a utilizar Scrum, começamos a nos organizar melhor. Conseguimos melhorar bastante o desempenho da equipe. Influenciou na velocidade do desenvolvimento e na organização da equipe. (Participante 16)

Organização foi mencionada como um dos diferenciais que este método proporciona, quando comparado a outros métodos comuns de ensino, conforme descrito na subseção 6.4.6

Transparência

O direcionamento e o compartilhamento de informações entre as equipes facilitam o processo de transparência, pois o trabalho a ser feito fica claro para as equipes. *"Primeiramente é lembrar exatamente o que tem de ser feito em certo tempo, porque se não o pessoal empurra com a barriga. Também aumenta bastante a comunicação com a equipe, porque às vezes achamos que está tudo certo, mas algo tá impedindo um colega e você poderia ajudar nisso"*. (Participante 7)

> Você consegue entender mais fácil o que está dando de errado, fica mais claro para grupo todo se tem uma pessoa atrasando a outra, fica mais fácil de corrigir as coisas por conta do processo ser interativo. Se começou de um jeito e teve de mudar o processo é um pouco mais rápido. (Participante 3)

Transparência também foi citada como um dos diferenciais deste método na subseção 6.4.6. A falta de transparência é um aspecto que pode comprometer o projeto e impactar no desempenho da

equipe. Neste sentido, o uso de CBL e Ágil através do método proposto, facilitam e proporcionam a transparência entre os indivíduos.

Comunicação

O desenvolvimento ágil influenciou positivamente na integração das equipes melhorando também a base de comunicação e trocas de experiências entre as equipes.

"[...] ajuda para a equipe estar mais unida em relação ao desenvolvimento e para concretizar o projeto, acho que serve bem nessa situação, e comunicação também, entre o time, para saber o que cada um está fazendo, no que pode ajudar os outros, quais as dificuldades, quais as conquistas, tudo isso." (Participante 14)

Comunicação foi apresentada por Scharff *et al.* [105] como algo a ser tratado na introdução de Scrum em ambientes de sala de aula. Configurar uma forma de observar e monitorar as comunicações da equipe, de forma a aumentar a consciência das equipes sobre o andamento dos projetos. Além disso, comunicação é um dos fatores que se apresentou como um dos diferenciais deste método na subseção 6.4.6.

Entrega

O desenvolvimento ágil evita o uso de burocracia, facilitando a equipe em manter o foco na entrega de produtos que agreguem valor aos seus usuários.

"Com os métodos ágeis você tem um tempo fechado, os Sprints, e você tem de ter aquela coisa pronta pra fechar o sprint. Você consegue lançar o seu produto. Ajuda na entrega." (Participante 18)

O uso do Kanban se mostrou como uma prática ágil que ajuda a dar visibilidade no que precisa ser realizado como também influencia na entrega dos produtos. *"[...] o Kanban é simples e diz o que tem de ser feito, você fica encarando o papel e só sai da cadeira depois de fazer aquilo. Ele agiliza porque pressiona o aluno a terminar a atividade"*. (Participante 5)

De acordo com o relatório de uma pesquisa realizada com mais de 1000 participantes e publicado pela Version One em 2015 [90], quando os respondentes foram perguntados o seu sucesso em iniciativas ágeis, o indicador mais citado foi *on-time delivery* dos projetos.

Flexibilidade

Acredita-se que os métodos de desenvolvimento ágil melhoram a flexibilidade do desenvolvimento de software, proporcionando meios para se adaptar às mudanças nos requisitos e ambiente, e também através do aprendizado de experiências de desenvolvimento [96]. A flexibilidade foi citada pelos participantes como um dos benefícios no uso de desenvolvimento ágil para projetos de aplicativos.

"O desenvolvimento ágil influencia projetos de desenvolvimento de aplicativos de software, porque é mais dinâmico e flexível e mais rápido". (Participante 14)

Desempenho

A melhoria no desempenho dos projetos através das equipes também é influenciada pelo uso de desenvolvimento ágil através da organização e entrega incremental.

"Eu acho que aumenta primeiro a produtividade, e tu torna o processo mais organizado de forma que tu consegue modularizar, e deixa mais organizado". (Participante 26)

A melhoria no desempenho das equipes também foi citada como um dos fatores que são diferenciais nesta integração entre CBL e práticas ágeis, conforme descrito na subseção 6.4.6.

Melhoria continua

A possibilidade de melhoria continua através das iterações onde ocorre o aprendizado incremental também foi mencionada como uma das influências positivas do uso de desenvolvimento ágil para produzir aplicativos.

> [...] o desenvolvimento ágil me orientou a estruturar como seriam feitos meus projetos, eu fiz muita coisa em cima de Scrum. Quando eu ia fazer a sprint eu sabia quanto cada tarefa tinha demorado e eu consegui fazer um cronograma melhor para as próximas iterações. É possível definir metas cabíveis e que conseguiriam ser feitas no tempo correto. (Participante 22)

"[...] fica mais fácil de ver por qual caminho ir e como vai aprender o que não sabe, é mais fácil aprender de pouco em pouco. Obvio que às vezes erramos para mais ou para menos, mas você consegue fazer coisas mais realistas". (Participante 27)

A melhoria contínua no processo de desenvolvimento e aprendizagem foi citada tanto no estudo de campo através do entendimento sobre o uso de desenvolvimento ágil no desenvolvimento de aplicativos na subseção 6.4.3, como também como um dos diferenciais do métodos na subseção 6.4.6.

Planejamento Iterativo

O planejamento iterativo permite uma evolução continua no aprendizado e desenvolvimento dos projetos.

"Ele permite, por exemplo, eu estar trabalhando com coisas que nunca tinha visto na vida. Essa etapa de interações permite ir vendo o que estou fazendo certo ou errado. Se eu não sei, vejo como vou fazer. Eu planejo passo a passo. Vejo o que preciso saber aqui, e dou mais um passo e planejo novamente". (Participante 19)

Estimativas

A organização do *product backlog* e dos *sprints backlog*, através de atividades corretamente descritas também facilita o processo de estimativas através de técnicas ágeis como *planning poker*, o qual é uma técnica de estimativa baseada em consenso. Esta técnica a equipe lê as *user stories* do *product backlog* e descreve as funcionalidades necessárias de forma a se realizar as estimativas.

> [...] baseado na experiência que eu tive aqui, ele deixa muito bem organizadas todas as tarefas que eu tenho de fazer para o *software* funcionar bem. E ainda consigo estipular bem o prazo para terminar. Porque antigamente, eu estipulava um prazo na minha cabeça e muitas vezes eu ultrapassava esse prazo do desenvolvimento. Pegava meio mal. Em alguns casos raros eu acabava terminando antes do prazo. Eu não tinha esse controle para definir em quanto tempo eu ia terminar o *software* para entregar. Agora aqui, o exemplo foi o último *challenge* que tivemos. Nós utilizamos o Scrum para definir o *backlog* de todas as atividades. Então, nós estipulamos os prazos muito mais fechados e certos do que antes. Enfim, em resumo, acho que a metodologia ágil ajudou muito para definir tempo de projeto e para definir quais atividades precisam ser feitas. (Participante 12)

Usuários

Um grande desafio no desenvolvimento de aplicativos é atingir as expectativas de uma grande base de usuários que muitas vezes são clientes que não estão próximos da equipe, que na realidade vão estar baixando os aplicativos das plataformas de distribuição.

> [...] é possível criar um aplicativo se existir uma equipe boa identificada para gerar uma MVP no primeiro mês, ao utilizar ágil é possível ir ajustando vendo gaps durante as semanas, durante a sprint 1 vamos dizer assim, se fosse um processo tradicional a primeira semana poderia ser perdida com alguém as vezes longe da equipe pensando no que seria o produto dai vem com algum documento, sempre podendo entregar algo diferente do que o cliente quer. E eu acho que o ágil, principalmente para *mobile* a equipe normalmente sabe o que que realmente é necessário, o que esta sendo acessado através de análises do *analytics*. Então, eu não vejo uma equipe funcionar bem sem ágil para *mobile*, pelo curto tempo do desenvolvimento de aplicativos é uma equipe que conhece muito bem os produtos que desenvolve, o cliente é o nosso usuário final essa que é a diferença em *mobile*, normalmente o pessoal acha que os usuários internos da empresa é que são os clientes. Mas na verdade os clientes são os usuários que vão estar baixando os apps da loja, quem está escrevendo os review dos aplicativos [...] (Participante 25)

De acordo com o relatório da Version One 2015 [90], quando os respondentes foram perguntados o seu sucesso em iniciativas ágeis, a satisfação dos clientes e usuários figura como importante para 44% dos respondentes.

Qualidade

De acordo com Melo *et al.* [103] dentre as principais razões que direcionam as empresas a adotar métodos ágeis, a melhora na qualidade de *software* é citada por 83% das respostas.

> O desenvolvimento ágil permite um *feedback* muito mais simples e rápido. E dá tempo de corrigir mais fácil. Como detectamos desvios de erros mais rápido,

> influencia na qualidade. Porque você faria um trabalho imenso de um mês e entregar para o cliente e poderia estar tudo errado. Mas quando têm trabalhos menores, você corrige o erro. Pode ser que a percepção da velocidade não seja imediata, mas com certeza deixa mais rápido. (Participante 2)

Com o *feedback* e melhoria constante, o desenvolvimento ágil permite a melhoria da qualidade do produto através da comunicação clara, rápida detecção de problemas e correção de eventuais desvios nos projetos de desenvolvimento.

7.2.3 Práticas de desenvolvimento

Conforme ilustrado na figura 7.2, diferentes práticas ágeis foram adotadas pelos participantes durante o desenvolvimento dos projetos. Builds automatizados 15,62%, uso de gráfico de*Burndown* 21,87%, Reunião diária 81,25%, *Collocation* 81,25%, Integração Contínua 34,37%, uso do quadro Kanban 68,75%, Programação em Par 46,87%, Refatoração 25%, TDD 3,1%, Pequenas *releases* 43,75% e Planejamento Iterativo 59,37%.

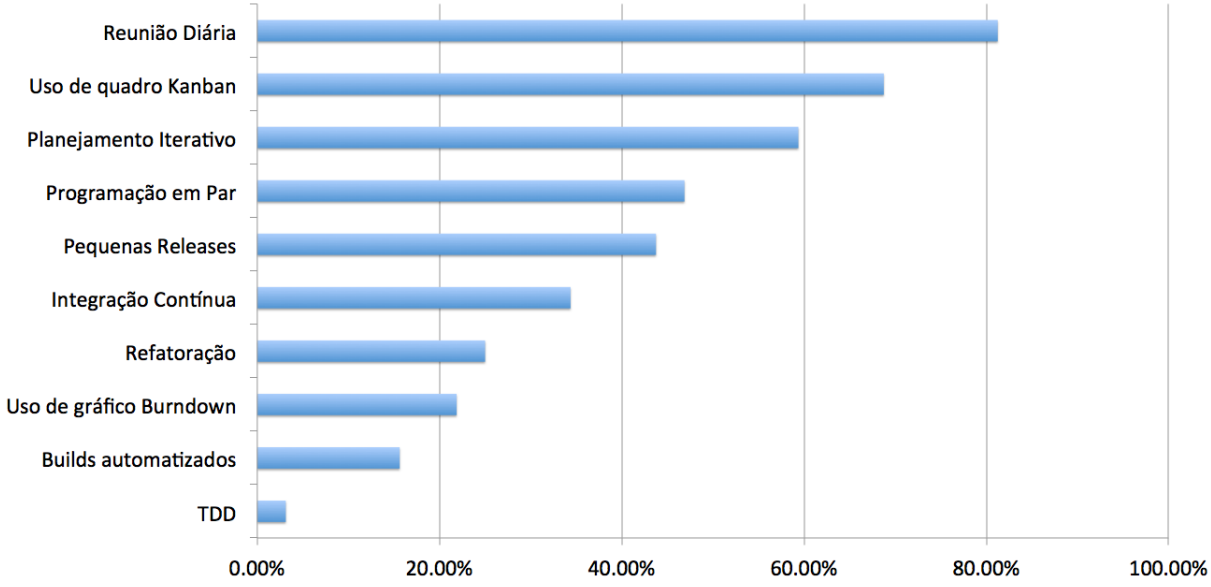

Figura 7.2: Práticas ágeis identificadas no estudo de caso

No que refere-se a adoção da prática de *Burndown*, duas equipes deixaram de adotar durante o andamento do projeto, o principal motivo foi que eles estavam errando muito as estimativas e acabavam ficando frustrados ao realizar o *Burndown*. Neste sentido, faltou a estas equipes o entendimento do conceito de melhoria contínua, que as vezes até por ter pouco tempo de experiência a equipe pode errar as estimativas nas primeiras iterações, mas com o andamento do projeto as estimativas tendem a ser mais assertivas.

A reunião diária, é a prática ágil adotada pela maioria dos participantes. *"Reunião diária sempre para localizar onde você está no projeto"*. (Participante 20). Integração contínua foi adotada por 34,37% dos participantes, entretanto ocorreram alguns relatos de conflito na storyboard. *"[...]*

tivemos um problema com o Git, na parte do storyboard." (Participante 1). *"Se eu estou mexendo ninguém mais pode mexer. Se acontecer, dá um conflito. Coisa bem crítica. Então estar trabalhando junto e toda hora poder inspecionar o código do seu colega ajuda demais"*. (Participante 17)

O uso do quadro de Kanban para controlar o progresso das atividades foi adotado por 68,75% dos participantes, inclusive alguns citaram o uso de ferramentas para apoiar o uso do Kanban, como Trello ou Tiger. Além disso foi citado que o uso do quadro de Kanban facilita que uma pessoa com determinada habilidade escolha as tarefas que ela pode contribuir melhor. *"Costumamos usar o Trello, para separar tarefas em o que já foi feito, o que está sendo feito e o que será feito. Se você tiver fazendo algo e notar que eu mudei o status da tarefa, você sabe que pode ajudar. Melhorar a distribuição dos trabalhos, a pessoa com habilidade em alguma área pegar uma tarefa correta para ela"*. (Participante 22)

Programação em Par adotada por 46,87% se mostrou como uma prática que ajuda em momentos de aprendizado, e também em momentos de desenvolver funcionalidades mais críticas. *"[...] principalmente se alguém estiver com dificuldades."* (Participante 28), *"Utilizamos, principalmente para situações que sabemos que será muito difícil, para não parar. Utilizamos muito no começo, quando estávamos aprendendo frameworks."* (Participante 19)

Refatoração foi adotada por 25% dos participantes, algumas vezes em situações de incremento de aprendizado *"[...] refactoring se usa bastante porque algumas vezes aprendemos algo novo e lembramos que poderíamos mudar algo que já existe e melhorá-lo."* (Participante 25)

Pequenas *releases* foi adotado por 43,75% dos participantes *"Mandamos versões pequenas para as pessoas testarem e pedimos o feedback."* (Participante 20) *"[...] descobrimos que vale a pena desenvolver a interface e testar com as pessoas e ver se funciona."* (Participante 24)

Conforme ilustrado na figura 7.3, outras práticas de desenvolvimento foram utilizadas nos projetos, dentre elas: Protótipo 62,5%, UML 65,63%, Modelagem ER 46,88% e Teste Unitário 9,37%. Além das práticas ágeis descritas, os estudantes também trabalhavam em forma de *collocation*, que significa trabalhar fisicamente próximo a equipe. *"Collocation, por enquanto foi essencial, mesmo que tenha e-mail, e estamos cheios de ferramentas que conseguem controlar a equipe inteira, o simples fato de você virar a cadeira para o lado e falar que algo não vai dar faz um diferencial enorme"*. (Participante 17).

O uso de protótipos foi aplicado pela maioria dos participantes 62,5%, principalmente no sentido de melhorar a coleta de requisitos e análise de fluxo de navegação e interface de usuário.

"Prototipação funciona como um coletor de requisitos. Estamos desenvolvendo agora um projeto que envolve hardware, então se eu chegar no cliente e mostrar algo físico funcionando ele pode achar bacana e aí volta para nós e podemos implementar mais coisas no aplicativo e projeto." (Participante 14)

Grande parte dos participantes 65,62%, utilizam modelagem UML, entretanto eles não utilizam todos os diagramas e documentos, pois isso geraria um processo rígido, a maioria dos participantes adota alguns dos principais diagramas existentes, como diagramas de caso de uso, diagrama de classes, diagrama de atividades e diagrama de fluxo.

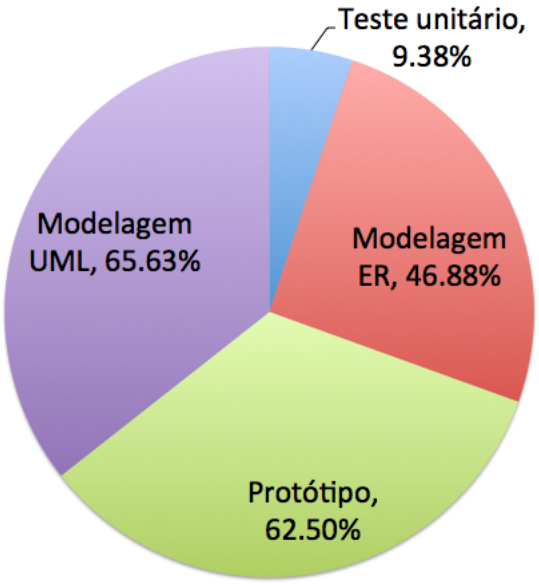

Figura 7.3: Outras práticas utilizadas durante o desenvolvimento

"Estamos usando um pouco de UML, a criação de cenários, casos de uso." (Participante 1)

"[...] nós fazemos, diagrama de caso de uso, diagrama de classe, diagrama de atividade e diagrama de fluxo." (Participante 11)

As cinco principais ferramentas utilizadas durante os projetos são XCode, o qual é utilizado por 100% dos participantes, devido a ser a IDE de desenvolvimento para iOS, seguido de GitHub 62,5% para controle de versionamento de código, Photoshop 53,12% para edição de imagens de aplicativos e desenho de interface Trello 50% o qual é utilizado como ferramenta para controle de projetos ágeis, seguido de Illustrator 43,75% utilizado para *design* de aplicativos (ícones, tipografia, etc). Além disso, diversas outras ferramentas utilizadas são listadas na tabela abaixo, Slack para comunicação 15,62%, o Sketch 15,62% (que faz o que ambos Photoshop e Illustrator fazem), conforme apresentado na tabela 7.4.

Tabela 7.4: Ferramentas utilizadas pelos participantes do estudo de caso

Ferramenta	%
Xcode	100
GitHub	62,5
Photoshop	53,12
Trello	50
Illustrator	43,75
Sketch, BitBucket	25
Parse, Source Three	18,75
Slack	15,62
Astah, Tiger	12,5
Google Docs, Sublime	9,37
Facebook, After Effects, Marvel, Star UML	6,25
Invision, Matlab, Paper, Draw.io, Bizagi, Google Calendar, Skype, SVN, Jira, Pixelmator	3,12

Diferentes são as ferramentas e práticas de desenvolvimento utilizadas para desenvolver aplicativos para dispositivos móveis. O método proposto possui diferentes fases que serão descritas nas próximas subseções, dentre elas a fase de visão apresentada na subseção 7.2.4, a fase de pesquisa apresentada na subseção 7.2.5, a transição para a fase de desenvolvimento apresentada na subseção 7.2.6, a fase de desenvolvimento apresentada na subseção 7.2.7, e por fim a fase de avaliação apresentada na subseção 7.2.8. Para cada uma destas fases buscou-se identificar as oportunidades de melhoria, melhores práticas e recomendações de forma a gerar o método final que será descrito no próximo capítulo.

7.2.4 Fase da visão

A fase da visão, é a primeira fase do método proposto e é onde são gerados os conceitos fundamentais necessários para se iniciar o aprendizado baseado em desafios. Esta fase engloba o processo da definição do desafio e diferentes práticas como a montagem da equipe, busca de interesses em comum, *brainstorm*, pesquisa de campo, entre outras são descritas no decorrer desta seção.

Prática 1: Montagem da Equipe

As equipes são definidas geralmente por afinidades em torno de um tema especifico. Todas as universidades participantes do estudo de caso dão liberdade para as equipes, mas algumas adotam dinâmicas particulares. Por exemplo a universidade U4 faz uma dinâmica de grupo onde cada um dos alunos apresenta suas paixões e temas de interesse, fazendo com que os alunos se conheçam melhor e formem as equipes de acordo com suas afinidades. A universidade U1 busca montar as equipes de forma interdisciplinar, agrupando por exemplo alunos com conhecimento em Computação, *design* e alguma outra área de interesse. O número de integrantes de cada equipe é decidido pelos alunos, entretanto as universidades geralmente recomendam equipes de 3 a 5 alunos, de forma a evitar equipes sobrecarregadas com um ou dois alunos, como também evitar equipes muito dispersas com mais de 5 alunos.

Prática 2: *Brainstorm*

As equipes inicialmente buscam analisar temas que fazem parte dos seus problemas cotidianos, ou que influenciem alguma familia, estes temas geralmente envolvem uma diversidade de problemas que geralmente englobam: Saúde, Educação, Transporte, Finanças, etc. Qualquer membro das equipes tem liberdade para propor diferentes temas. As equipes buscam assuntos que eles realmente queiram trabalhar e se sintam engajados para resolver um problema real. Algumas equipes buscam selecionar duas a três idéias que sejam de interesse da maioria, a partir dai tentam elaborar uma questão de algum problema que faça parte da ideia, gerando uma discussão geral de idéias e abordagens. Outras equipes tentam *"bombardear"* as ideias gerando perguntas para dúvidas ou problemas, buscando apresentar perguntas que possam *"quebrar"* uma ideia, e depois fazem uma previa do que vão trabalhar e definem um desafio (que não é necessariamente um aplicativo ainda).

> Basicamente aqui temos uma grande ideia, qualquer pessoa escolhe o tema, esportes, saúde, educação, tentamos escolher duas ou três ideias e pegar o melhor de cada um até chegar a um ponto em comum , para ter paixão de todos os membros do grupo. A partir dai tentamos elaborar uma questão que abrange toda a ideia e tentamos extrair o problema a ser resolvido para propor o desafio. (Participante 4)

Prática 3: Identificação dos indivíduos e Pesquisa de campo

Uma vez que é realizado o *brainstorm* inicial, as equipes procuram entender os interesses que pessoas teriam naquela área, para poder formar as essential questions reais. Identificação de certo problema para resolver, buscando trazer isso para a realidade das pessoas, na parte de identificação da persona, o contexto dela, idade, profissão, o relacionamento dela com o problema. Algumas equipes realizam um trabalho de campo inicial, de forma a tentar obter um *feedback* sobre o problema que estão tentando resolver dentro de uma determinada área, buscando refinar a proposta de um desafio a ser resolvido, como também obter opiniões externas para ter uma visão sob diferentes perspectivas do desafio a ser proposto. Algumas equipes utilizam técnicas como mind map para documentar o processo criativo de definição da *big idea, essential question e challenge*.

Prática 4: Apresentações e *feedback*

Algumas universidades, como por exemplo a U4 possuem um *timeline* proposto com todas as entregas a serem realizadas pelas equipes durante o ano. Este *timeline* prevê uma apresentação onde os as equipes tem a oportunidade de compartilhar quais são suas *big ideas, essential question e challenges* de forma a poder receber *feedbacks* construtivos de outras equipes, como também de instrutores.

> [...] é feita uma apresentação de 5 minutos todos os grupos, mostrando qual a *big idea* que esse grupo vai implementar, porque sem o *timeline* pessoas passariam um mês dizendo que estão na *big idea* ainda, até antes tem um negócio interessante é que as pessoas meio que se apresentam com um feeling das áreas que elas gostam, isso eu achei legal, porque da pra ter uma idéia de quais equipes você consegue montar através do mesmo interesse [...] acho que esse tempo foi ideal para não chegar no meio do desenvolvimento e cortar. (Participante 25)

Nessas apresentações, muitas equipes recebem *feedbacks* valiosos que vão desde o refinamento do problema, discussão sobre o escopo do desafio proposto, até mesmo casos onde outras pessoas conhecem o mercado, e sabem que alguma idéia já falhou por alguma razão particular, fazendo que ocorra uma reflexão e análise sobre o assunto durante a fase de pesquisa que se inicia após estas definições iniciais de *big idea, essential question e challenge*.

Além das práticas identificadas, os participantes também levantaram algumas oportunidades de melhoria, as quais são apresentadas a seguir:

Oportunidade 1: *Lean*

Uma das sugestões de um dos participantes foi utilizar o conceito de Lean ao invés de Scrum como base do método. *"o lean apesar de já ser uma variação do PDCA, acho que da forma como ele foi mostrado, ele se encaixa tão bem quanto. Enxergo que seria um Scrum, com a forma de interação do lean"*. (Participante 1)

Oportunidade 2: Retorno da avaliação

Outra oportunidade de melhoria levantada para o método foi a inclusão de um retorno do processo de avaliação, após a retrospectiva, para as guiding questions, pois podem ocorrer situações que após a finalização de uma iteração, revisão e retrospectiva, surjam questões que precisão ser pesquisadas no processo de guiding questions.

Oportunidade 3: Retorno do *sprint planning*

Outra sugestão foi que durante o sprint planning pode ocorrer a necessidade de voltar para as guiding questions, porque quando as equipes estão no planejamento de uma sprint eles discutem muitas coisas, mas alguém lembra de alguma guiding question, então volta-se voltar para a pesquisa. Mesmo que as equipes já tenham um *product backlog definido*, pode acontecer a necessidade de se pesquisar sobre alguma demanda especifica.

Oportunidade 4: Retorno do *research findings*

Algumas equipes relataram que o Research finds é um ponto muito importante do projeto, pois ele é o último passo de transição entre a pesquisa e o processo de desenvolvimento, as vezes quando a pesquisa está concluída e se inicia a transição do processo de desenvolvimento, pode ocorrer a necessidade de se retornar para o processo de pesquisa através de guiding questions. Pois em algumas ocasiões onde estão discutindo o planejamento do *product backlog* pode ocorrer a necessidade de voltar no research finds para verificar o que as equipes tinham encontrado.

"[...] Na transição que é algo que você tem de trabalhar bastante o research findings para a criação do backlog. Não foi tão trivial essa transição daqui para a hora de criar as features no backlog". (Participante 6)

Outros relatos de participantes confirmaram que durante a definição da *big idea, essential question e challenge* podem ocorrer mudanças até que se finalize esta etapa e se mova realmente para o processo de pesquisa.

Alguns dos participantes relataram alguns desafios que incluem uma dificuldade de abstração quando começaram a utilizar o método, pois montar uma equipe, realizar brainstorm e definir um problema a ser resolvido é algo extremamente desafiador para pessoas que estão aprendendo, principalmente porque 100% dos participantes não tinham experiência prévia com métodos de pedagogia ativa.

> [...] é meio abstrato, você não sabe como vai chegar naquilo, como vai lidar com uma situação. Geralmente achamos que conseguimos pensar no produto final e

> na primeira etapa você não quer o produto final, não quer pensar na solução. Você quer pensar em algo maior, numa área de atuação mais abrangente. Isso é complicado. (Participante 22)

Ocorreram também alguns relatos de participantes que em uma determinada universidade receberam a *big idea* especificada, um dos questionamentos para trabalhos futuros é entender como isso pode influenciar na definição do problema e do desafio, como também o quanto isso pode impactar no engajamento e entrega das equipes.

Encontramos também um relato de uma equipe que pensa em algo que gostariam de fazer como aplicativo, identificam uma necessidade das pessoas e definem a *big idea* (fazendo o ciclo inverso), um dos questionamentos para trabalhos futuros é verificar porque isso acontece e entender qual a influência no ciclo de montagem da idéia e desenvolvimento do produto.

7.2.5 Fase de pesquisa

O processo de pesquisa é um dos processos mais críticos do ciclo CBL antes de se definir a solução, muitas vezes durante este processo os estudantes além de definirem suas guiding questions, activities e resources, classificando-as de acordo com suas necessidades técnicas e de negócio, eles podem descobrir que o problema que estão tentando resolver não é viável. O processo de pesquisa engloba práticas como debate entre equipes, procura de especialistas na área, participar de eventos sobre determinados temas, assistir documentários, trabalho de campo, pesquisa de mercado, etc.

Prática 5: Montagem e priorização do processo

Um dos primeiros passos que as equipes buscam fazer neste processo é a montagem de uma lista com todas as guiding questions, buscando também identificar todas as atividades e recursos necessários para responder as guiding questions. Essa lista inclui todos os conhecimentos necessários, tecnologias necessárias, especialistas necessários, perguntas sobre stakeholders, mercado, e outras, independente da complexidade do desafio a ser pesquisado. Após estas definições iniciais são definidas as prioridades das perguntas a serem respondidas, e se buscam os recursos necessários para responder as perguntas com maior prioridade.

"[...] a dinâmica que os instrutores passavam era para que fizéssemos nossas perguntas e depois trocássemos de time. A outra equipe olha suas perguntas e tenta visualizar outras perguntas enquanto você visualiza as deles". (Participante 13)

Prática 6: Documentários e eventos sobre o tema

Algumas equipes buscam participar de eventos sobre o tema, que vão desde *workshops* até conferências em determinadas áreas, estes eventos muitas vezes além de ser um recurso por si só, eles também geram contatos que podem se tornar recursos durante o processo de pesquisa e aprendizado. Além do fato de participar de eventos, algumas equipes também assistem documentários sobre o assunto em pesquisa, o que promove um olhar sobre uma outra perspectiva, pois os documentários já podem apresentar resultados de pesquisas relacionadas.

"[...] fazemos a pesquisa primeiramente acadêmica, para ver o que tem na área, se já estudaram. Aí, tentamos achar alguma oportunidade através dessas pesquisas, que elas apontam nas conclusões dos artigos. Vemos se as oportunidades responderam as guiding questions". (Participante 4)

Prática 7: Análise de mercado

A prática de análise de mercado envolve a busca de aplicativos que tentam resolver problemas semelhantes, de forma a identificar suas oportunidades de melhoria e potenciais de inovação como concorrência. Outra parte da análise do mercado envolve o entendimento sobre o público alvo, e o mercado que se busca atingir com o aplicativo.

> [...] fazemos pesquisa de mercado, vimos o que tinha de diferente, procuramos bastante por tendências alinhadas com as guiding questions, alguma inovação, mas que ainda não tenha pegado, então tentamos descobrir porque ainda não pegou. Isso ajudou bastante na definição do produto. A persona, quem seriam os personagens afetados pela nossa aplicação, modelagem da persona, a análise de mercado, convergimos isso e sintetizamos. (Participante 6)

Uma vez que o *challenge* é definido, as equipes não buscam pensar diretamente uma solução específica. Nessa etapa de pesquisa se pergunta de tudo, na visão de todos os *stakeholders*. O foco é estruturar o trabalho de uma forma melhor, independente da complexidade exigida.

Ao final do processo de pesquisa os estudantes vão conseguir montar uma proposta da solução a ser desenvolvida. Além disso, após finalizar o processo de pesquisa e seguir para as etapas seguintes, pode ocorrer a necessidade de se retornar ao processo de pesquisa para refinar suas necessidades de forma a ajustar a solução proposta.

7.2.6 Transição para a fase de desenvolvimento

O processo de transição para o desenvolvimento é um processo muito crítico, pois é neste estágio que se aplica toda a descoberta do processo de pesquisa para gerar os requisitos que farão parte da solução a ser desenvolvida. Neste sentido, é necessário visualizar de fato o que precisará ser implementado a nível de solução para atender os requisitos encontrados. Seja ferramenta ou serviços extras ou integrações. Nesta etapa é montado o *backlog*, baseado em tudo que foi pesquisado ou em discussões, as funcionalidades são colocadas em uma lista, a partir da qual serão definidas prioridades. É analisado o que irá gerar mais valor para o usuário, seguindo do planejamento das sprints. A cada sprint os requisitos serão implementados por ordem de prioridade.

"Quando chegamos aqui tem que ver se o backlog bate com nosso challenge e guiding questions". (Participante 4)

Através do *research synthesis* é montado um *backlog*, esse *backlog* é ordenado e as principais funcionalidades são escolhidas para desenvolvimento nas sprints. As sprints são planejadas sempre seguindo a ordem de prioridades do *backlog*. O papel do *Product Owner* (PO) para algumas equipes é exercido por um cliente externo real, em alguns outros casos específicos o papel do PO é desempenhado por um instrutor ou conjunto de instrutores.

Prática 8: Protótipo

Após ter concluído a pesquisa, gerado o *research synthesis* e montado o seu *product backlog* com as principais funcionalidades priorizadas, algumas equipes buscam criar um protótipo para validar como seria a solução da proposta pré definida, o objetivo do protótipo é validar primeiramente com a equipe, e após ajustes validar com o público externo, após estes *feedbacks* o *product backlog* é refinado e o Mínimo Produto Viável (MVP) é definido.

> Quando viemos das *guiding question*, viemos com tudo pronto, com as perguntas que vamos transformar em requisitos, vêm com a complexidade, tudo armazenado em uma planilha. Uma vez criado tudo isso, aí vamos transformar as perguntas em algo entregável, como podemos entregar a solução de um problema para um cliente de forma concreta, ou seja, abstrair aquela pergunta. As pesquisas, as *researches findings* servem para isso, irmos atrás de informação, para saber como transformamos algo abstrato em algo concreto, que estará junto da implementação. A partir do momento que eu transformo isso aqui em algo entregável, eu modifico e transformo o problema em um sistema e depois disso levo no cliente. (Participante 14)

Prática 9: Modelagem e criação do ambiente

Outras equipes buscam realizar a modelagem de classes e modelagem entidade-relacionamento (ER), definição de arquitetura do aplicativo, seguido de atividades como criação do ambiente e repositório de código, e após isso partem para a criação do *storyboard* de forma a visualizar as principais funcionalidades do aplicativo. Um ponto interessante encontrado no relato dos participates é que eles não tem medo de errar e melhorar continuamente durante os projetos dos aplicativos, o que é uma contribuição direta do uso da abordagem de desenvolvimento ágil.

"[...] na minha pouquíssima experiência dos projetos aqui, essa é a melhor parte do método ágil. [...] dificilmente teremos a solução de primeira, pode dar errado, mas temos a chance de melhorar durante a próxima iteração". (Participante 17)

7.2.7 Fase de desenvolvimento

O processo de desenvolvimento basicamente segue o ciclo ágil e as práticas já descritas na sub-seção 7.2.2, como também as ferramentas e práticas descritas na sub-seção 7.2.3.

O uso da *daily meeting* foi relatado pelos participantes que ajuda as pessoas a entregar mais no geral e que facilita muito para a equipe visualizar o trabalho que está sendo feito. Os membros das equipes procuram se comunicar o máximo possível tanto para visibilidade do que esta ocorrendo no desenvolvimento do projeto, como também para obter *feedback* e controlar o escopo do que foi planejado para entrega no final de cada sprint.

O *feedback* constante durante as iterações também facilita tanto o desempenho da equipe no desenvolvimento do produto, como também a melhoria contínua no processo de trabalho de cada equipe, obtido através do aprendizado contínuo das lições aprendidas em cada iteração de desenvolvimento.

> Começamos a montar de acordo com o que as pessoas querem e estruturar o mínimo que temos de entregar para que as pessoas possam utilizar sem quebrar a experiência do usuário com o app. Bom, vemos o que dá para fazer de partida, o que dá para separar, começamos a quebrar em blocos. Por exemplo, não podemos fazer uma persistência de dados se não temos um bloco de dados. Então damos prioridades. Começamos a separar em iterações que são próprios sprints. Fazemos uma estimativa de quantas sprints teremos de revisar até a entrega. Começamos a pensar o que conseguimos fazer em cada sprint. Se a sprint der errado, vamos começar a pensar nos próximos, porque se falhamos é porque tem algo errado e não podemos continuar assim. Fazemos lista do que tivemos dificuldade, reclamações um do outro. Mas de repente esquecemos de uma feature que veio em mente no meio da sprint. Voltamos no começo, às questões, fazemos a pesquisa. Se for realmente importante, vemos como podemos encaixar na próxima sprint, se precisa ser agora ou pode ser mais para frente. Aí fica nesse ciclo até sair algo. (Participante 19)

Algumas equipes relataram não utilizar 100% das práticas ágeis, fazendo customização de algumas atividades como definição de histórias. Algumas atividades estavam ficando longas, por exemplo uma tarefa não estava sendo possível fazer em um dia. Mas seguiam fazendo os planejamentos de sprint, procurando identificar o que cada membro da equipe tinha de conhecimento, no que já trabalhou, dividindo assim responsabilidades. Relataram também aplicar os processos de revisão de sprint e retrospectiva sempre que possível para avaliar os objetivos atingidos e ações de melhoria para o produto e para o processo de desenvolvimento.

Algumas equipes relataram também dificuldades de utilizar a técnica de *burndown*, provavelmente devido a pouca experiência e dificuldade no processo de estimativas através de técnicas como por exemplo *Planning poker*. Outras equipes relataram que alguns membros da equipe não atualizavam o kanban, ou não participavam da daily meeting, o que dificultava bastante a comunicação da equipe e também a visualização do que faltava ser feito dentro de uma determinada sprint de desenvolvimento.

7.2.8 Fase de avaliação

O processo de avaliação faz parte da iteratividade do método, e é realizado constantemente, basicamente o processo de avaliação é composto de 3 partes: *Reflections, Sprint Review e Sprint Retrospective*. As *reflections* podem ocorrer em qualquer momento e são um fator fundamental especialmente durante o aprendizado, já as sprint review e sprint retrospective são realizadas ao final de cada iteração de desenvolvimento.

A *reflection* se mostrou com algo que influencia bastante no processo de aprendizado, e ela pode ser utilizada como material de apoio durante o processo. Entretanto existem relatos de um pouco de resistência em relação as *reflections* sob o ponto de vista de burocracia ou sob o ponto de vista que os membros das equipes podem vir a ser avaliados pelo conteúdo das *reflections*, o que é uma concepção errada, pois as *reflections* são muito importantes como material de auxilio a equipe de instrutores de forma a planejar e direcionar melhor o processo de aprendizado. Outro ponto relatado em relação as *reflections* é que tem pouca utilidade durante o ciclo de desenvolvimento do projeto, o que faz sentido, pois durante o desenvolvimento do projeto as cerimônias de revisão de sprint e retrospectiva de sprint, é que vão guiar a avaliação das iterações de desenvolvimento.

> Quando faço uma *reflection* pessoal, de vez em quando eu assisto de novo. É uma forma de lembrar o que realmente melhorei desde a última vez que fiz. Por exemplo, se tenho uma para saber o que melhorou no meu processo de desenvolvimento de iOS, falo lá que aprendi a linguagem bem, mas estou com dificuldade em tal coisa. Aí quando chego à próxima vez de fazer, vejo que já melhorei naquele ponto. É mais uma questão pessoal de saber que estou melhorando, por isso que eu torno a ver as minhas. Eu estou tentando agora criar um hábito de fazer *reflections* mais frequentemente, porque isso ajuda em vários processos. (Participante 12)

Durante a revisão da sprint é possível realizar uma demonstração do *working product* disponível no final de cada iteração, em algumas universidades a prática utilizada é a demonstração para a equipe de instrutores, em outras universidades são realizadas apresentações para todos os outros grupos, o que permite uma maior interação e *feedbacks* oriundos de diferentes equipes, ajudando muito na evolução do produto.

Por outro lado, as retrospectivas permitem uma avaliação melhor da equipe e do processo de desenvolvimento em si. Estas lições aprendidas no final de cada sprint permitem um processo de melhoria contínua, pois faz cada um dos membros das equipes refletir sobre suas atividades permitindo assim um aprendizado constante.

> O *sprint review* ajuda a ver as vezes se está adiantado ou não e principalmente a ter *feedback*, pois as vezes quando você apresenta surgem questionamentos que ajudam a repensar a sua solução, pois as pessoas te retornam muito *feedback* construtivo e temos várias oportunidades de *feedback*. E a cada vez você vai melhorando mais tanto a apresentação quanto o código em si. (Participante 25)

Foram identificadas oportunidades de melhoria durante o processo de avaliação, a primeira delas é em relação a *reflection*, pois existem indivíduos que não entendem o quanto as *reflections* podem ajudar os instrutores a melhorarem o processo de aprendizado.

> [...] As *reflections* acho que tínhamos um cuidado de fazer porque sabíamos que tinham instrutores que veriam aquilo e tomariam uma decisão. Por um curto

> tempo não veríamos elas e ficava de uma forma aberta, não sei se era essa ideia das *reflections*, ou todo mundo ver e ficar sabendo o que cada um falou, mas nós tentávamos buscar esses pontos fortes e fracos que faríamos em uma *reflection* [...] (Participante 13)

As retrospectivas de sprint também poderiam ser melhor utilizadas por algumas equipes em especifico, porque geralmente no final da sprint a equipe está cansada não aproveitando da melhor forma possível o momento de aplicar as lições aprendidas durante a iteração de desenvolvimento. Outro ponto relatado por alguns participantes é que a revisão de sprint poderia ser melhor utilizada pelos instrutores, porque os instrutores ao verem muitos projetos no mesmo dia, podem deixar de dar *feedbacks* importantes para algum grupo em especifico, pois é o momento onde as equipes estão apresentando tudo o que eles conseguiram aprender e implementar no produto.

A maioria dos grupos utilizou corretamente o processo de avaliação, através das *reflections* durante o aprendizado, e das revisões de sprint e retrospectivas de sprint, as revisões de sprint realmente se mostraram como uma oportunidade de diversos grupos trabalharem o *feedback* e implementarem melhor os seus produtos. Já a retrospectiva se mostrou como um processo de lições aprendidas durante o ciclo de desenvolvimento através de diferentes iterações do projeto.

Um dos pontos a serem avaliados no futuro é como fazer as *reflections* serem vistas como algo burocrático, como também serem vistas como algo que não faz parte da avaliação dos estudantes. Pois a *reflection* é uma ferramenta muito importante na melhoria de processo de suporte ao aprendizado dos alunos.

7.2.9 Recomendações para uso do método inicial

Um dos aspectos que se buscou durante as entrevistas com os participantes foi a identificação de contra-medidas que possam proporcionar recomendações para o uso do método. Neste sentido procurou-se identificar recomendações para o uso do método ou em forma de ferramentas e práticas. Dessa forma, dentre as várias mudanças propostas, esta sessão esta organizada por práticas que devem começar a ser feitas, práticas que devem continuar sendo feitas e práticas que não devem mais ser utilizadas.

É de extrema importância para o sucesso do aprendizado e desenvolvimento dos projetos que a equipe se sinta engajada e esteja motivada para resolver um problema gerando o desafio proposto. Neste mesmo sentido, a forma de interação dentro do grupo também seria algo que alguns participantes mudariam, eles teriam uma postura mais direta e profissional.

"No método não mudaria nada. Mudaria que agora aprendi a fazer sozinha, eu era meio preguiçosa, perguntava quem sabia fazer, pedia pra vir me ajudar. Mudaria a postura. E a parte de entender as coisas melhores. Se os professores falam e não entendo, agora eu vou atrás e pesquiso, antes ficava assim mesmo porque achava que não ia entender de outro jeito." (Participante 20)

Alguns participantes descreveram como recomendação a aplicação do controle de versão de código no inicio do desenvolvimento, pois durante o desenvolvimento aumentaria o volume de trabalho

e número de erros e perda de arquivos, pois isso exige uma curva de aprendizado maior, podendo impactar no processo de desenvolvimento.

Outro aspecto relatado pelos participantes seria dar mais valor as *Reflections*, tanto de forma gravada como também em arquivo texto.

Além disso a busca de colocar os membros da equipe a desenvolver as atividades com as quais tem mais motivação, pois algumas vezes é melhor colocar uma pessoa que não tem tanto conhecimento mas esta motivada e se comunique bem com a equipe.

Dentre os relatos do que os participantes gostariam de continuar fazendo, eles relatam que gostariam de ter conhecido o método antes, para trabalhar em outros projetos que tiveram antes de ter participado do programa. *"Eu voltaria para implementar esse método nos trabalhos antigos. Deu errado porque não utilizamos uma metodologia. No método acredito que não mudaria."* (Participante 19)

Como oportunidades de melhoria, alguns membros de equipes concluíram que possivelmente poderiam ter focado mais no processo de aprendizado do que no processo de desenvolvimento dos projetos.

> Eu acho que entrei muito focado em tentar aprender o desenvolvimento, em aprender código e programar, só que no meio do projeto acabei fazendo design, pelo tempo que tinha para entregar e por já saber. Hoje talvez eu tivesse sido um pouco menos responsável pelo projeto, ver o projeto como educação e menos como trabalho. O programador aprender um pouco mais de design, e eu um pouco mais de programação. O produto ficou ótimo, mas o aprendizado ficou meio deficiente. (Participante 8)

Outros participantes reconheceram que pularam etapas importantes no método, o que gerou problemas e desafios que eles tiveram muita dificuldade em resolver. Se estes participantes tivessem a oportunidade de voltar atrás eles seguiriam mais fielmente o método de forma a ter uma proposta de solução viável antes de iniciar o processo de desenvolvimento e codificação. Além disso, novamente alguns participantes relataram que ao invés de pensar em um problema a ser resolvido pensaram logo na solução o que também gerou dificuldade durante a implementação do projeto.

Alguns respondentes relataram que quanto menos rotatividade tiver entre as equipes, mais fácil é o processo de comunicação e aprendizado.

> Tivemos muita rotatividade nas equipes, eu acho que isso atrapalha, pois as vezes tu pega uma equipe nova mas pode ter uma idéia boa e não gosta da equipe, você não investe tanto energia para fazer dar certo. O modelo é fantástico a forma como os professores abordam está ok, mas a abordagem de rotatividade não fica legal. Então como recomendação para o método eu recomendaria evitar rotatividade. Por exemplo eu estava em uma equipe que teve uma idéia muito legal que eu estava junto e agora eles estão trabalhando legal em algo que eu gostaria de fazer parte e isso acaba desmotivando, porque eu ajudei na elaboração mas não pude

> participar no desenvolvimento. Acaba afetando a motivação e talvez também o aprendizado. Quando fui para o outra equipe eu mal participei, tu meio que topa qualquer coisa. Vira uma entrega normal sem engajamento. (Participante 25)

Além disso, outra recomendação para uso do método foi evitar ciclos curtos de desenvolvimento com poucas sprints, pois além de ter a questão de engajamento com equipes, um tempo muito curto não permite uma aplicação profunda das facilidades que o método promove. E isso algumas vezes pode dificultar o aprendizado e entrega dos diferentes equipes.

Como forma de consolidar as oportunidades de melhoria, recomendações e melhores práticas, também foram analisadas ferramentas e práticas que os participantes descreveram como importantes e utilizados além das ferramentas e práticas que de alguma forma não foram utilizadas corretamente. Neste sentido a subseção 7.2.10 apresenta a perspectiva de importância e utilidade de ferramentas e práticas.

7.2.10 Relação de importância e utilização de ferramentas e práticas

Este estudo de caso buscou também analisar as ferramentas e práticas que são importantes e efetivamente utilizadas, como também de coisas que são importantes e não estão sendo corretamente utilizadas. O objetivo desta análise é complementar as melhores práticas e oportunidades de melhoria no uso do método, a Figura 7.4 apresenta as principais práticas citadas como importantes e utilizadas.

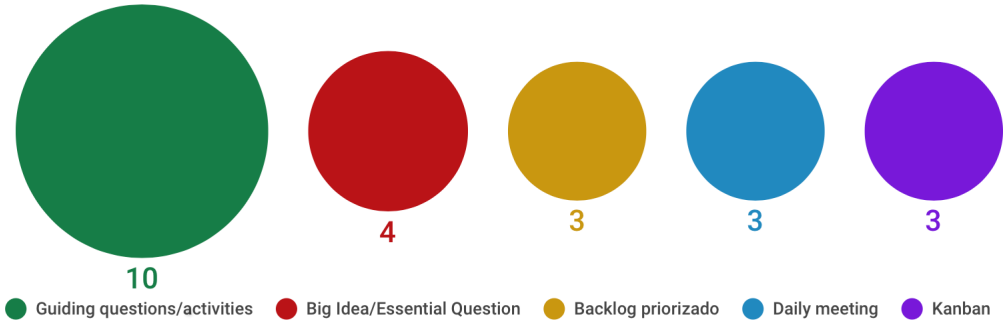

Figura 7.4: Ferramentas e práticas importantes e utilizadas

O processo de pesquisa, através do uso de *guiding questions, activities e resources* apareceu como o item mais importante e utilizado pelos participantes, seguido do processo de definição do desafio que inclui a *big idea, essential question e challenge*. Nesse sentido, os estágios do processo de abordagem baseada em desafios que ajudam tanto no processo de aprendizagem de desenvolvimento quanto na concepção do produto foram definidos como os mais importantes e mais utilizados. Seguido do processo de aprendizado e concepção as práticas ágeis também foram definidas como importantes e utilizadas, entre elas o uso de um *product backlog* priorizado, as reuniões diárias e o *kanban* figuram entre as práticas mais importantes e utilizadas de acordo com os participantes. Além disso, outros fatores também foram citados como importantes e utilizados,

entre eles o trabalho em equipe, colaboração e *feedback* constante, incluindo também práticas como casos de uso, protótipo, integração contínua, *sprint planning, reflections* e ferramentas.

"Guiding questions. Nesse último challenge, meu grupo tinha uma ideia e mudou totalmente com as questions que fizemos. Não fazíamos ideia de como funcionava. Ir atrás das pessoas e fazer as perguntas certas dá uma visão melhor da área e você consegue planejar melhor o aplicativo." (Participante 18)

Além das práticas importantes e utilizadas, outros participantes citaram práticas que eles consideram importantes mas que não utilizaram corretamente, estas práticas estão disponíveis na Figura 7.5.

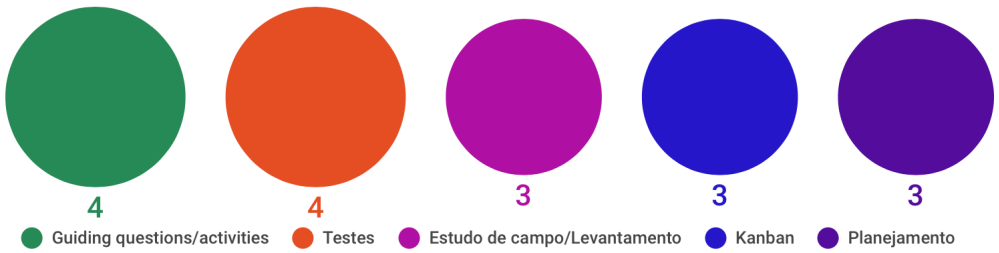

Figura 7.5: Ferramentas e práticas importantes e não utilizadas corretamente

Dentre as práticas importantes e não utilizadas corretamente pelos participantes foi citado o processo de pesquisa *guiding questions, activities e resources*. Isso nos leva a concluir que o processo de pesquisa realmente é o item mais importante durante o aprendizado de desenvolvimento e também concepção do produto. Entretanto tem alguns participantes que entendem que não utilizaram o processo de pesquisa da melhor forma possível.

"[...] as nossas guiding questions poderiam ser melhoradas. Eu acho que a forma de pesquisa poderia ter sido melhor implementada, nosso formulário poderia ter sido mais próximo da big idea." (Participante 25)

O uso de testes também foi citado como importante e não utilizado corretamente, muitos participantes da pesquisa entendem a importância do uso de testes para os seus aplicativos, o que inclui desde testes unitários até testes funcionais, isso será abordado em trabalhos futuros.

"A parte que não fizemos foi teste dos aplicativos. Porque se tivéssemos feito o teste antes, muita coisa poderia ter sido melhorada. Isso é ruim porque se o usuário perceber ele não vai mais querer baixar o aplicativo." (Participante 20)

Outras práticas que os participantes classificaram como importantes e não utilizaram corretamente englobam o estudo de campo e levantamento de informações, o planejamento e o uso da técnica de Kanban. Além disso os participantes também citaram algumas práticas que poderiam ser melhor utilizadas as quais incluem o uso do *Planning Poker, Burndown, Sprint planning, Pair programming*, protótipo, controle de versão e ferramentas.

"Especificamente pair programming, eu usei mas foi uma das que menos usei, acho muito importante porque acho que é algo muito ágil, na maneira que tem alguém ali codificando com você, você tem uma idéia nova e pode dar um aprimoramento na hora." (Participante 26)

Neste contexto podemos concluir que as práticas ágeis são importantes dentro do contexto do método, entretanto algumas equipes mesmo entendendo a sua importância ainda não utilizam todas elas.

Algumas práticas foram consideradas não importantes, entretanto utilizadas. Uma delas é a obrigatoriedade de um número mínimo de guiding questions, algumas universidades obrigaram as equipes a ter um número mínimo de questões no processo de pesquisa, isso se mostrou como algo que não deveria fazer parte das práticas do processo. Outra prática que foi citada mais no sentido técnico, foi evitar o uso de muitas animações nos aplicativos, aparentemente isso ocorreu em uma equipe especifica.

Em relação as práticas não importantes e também não utilizadas foram citadas reuniões longas, que não agregam valor nenhum ao processo e devem ser evitadas ao máximo, além do uso completo de UML, ou seja ao tentar utilizar todo o processo UML vai diminuir e dificultar a agilidade do processo de aprendizado e desenvolvimento de aplicativos.

Dessa forma podemos concluir que o uso do método e práticas envolvidas é considerado importante pela maioria dos participantes, embora poucos participantes tenham sugerido alterações no método em si, alguns deles entendem que tem algumas práticas que deveriam ser melhor utilizadas durante os próximos projetos. Neste sentido, encontramos indicativos que o método foi efetivamente utilizado pelos participantes, de forma a contribuir no seu processo de aprendizado, e no seu processo de concepção e produção de aplicativos que resolvem reais.

7.2.11 Influência da plataforma de desenvolvimento no aprendizado

Durante o mapeamento sistemático da literatura, foram encontrados quatro estudos que discutem a influência de plataformas de desenvolvimento em ambientes de aprendizado de desenvolvimento de aplicativos para dispositivos móveis [44, 86, 87, 111]. Neste contexto, o estudo de caso buscou mapear relatos sobre a influência ou não da plataforma de desenvolvimento durante o aprendizado. De acordo com alguns relatos, o fato de ser a plataforma iOS simplifica o entendimento e aprendizado, devido a fatores como documentação simplificada, interface intuitiva e a nova linguagem de programação swift.

> Sim, com certeza, o iOS acho mais que o android, na época do objective C diria que não porque era uma linguagem oriunda do C e você percebia várias pessoas com dificuldade de aprender e eu já peguei vários aplicativos pela metade porque as pessoas começaram e sumiram deixando os clientes na mão, porque era difícil até para o aprendizado. Agora com o swift é fantástico, o swift facilita muito, ainda mais o pessoal mais novo que é familiarizado com scripts, eles aprendem estão voando já eu que venho do objective C, percebo que eles aprenderam a base de objective C, mas vejo que eles aprendem muito mais fácil o swift e muito mais

> rápido. *Story board* que é a possibilidade de desenhar a interface, então tudo isso acaba que dando uma visão já do aplicativo e não é só ela te dar um diagrama de como vai funcionar a navegação, mas você pode também conectar com o código, isso tudo acho que da uma visão do que funciona e está sendo chamado e eu digo que ajuda e mais do que o Android. (Participante 25)

Por outro lado, outros participantes relataram que a plataforma não influencia no aprendizado, que o que faz diferença mesmo é o aprendiz no processo, considerando também que cada plataforma terá diferentes práticas e recomendações.

"No aprendizado, acho que não influencia, mas vai ter influência de práticas. No quesito aprender você consegue aprender bem em qualquer um. Mas vai ter influência da plataforma que usar." (Participante 17)

Outros participantes apresentaram comparativos de experiência com diferentes plataformas de desenvolvimento de aplicativos, alguns relataram desafios no trabalhar e entender a arquitetura do Android, outros que a vantagem do Android é ter uma plataforma mais aberta, o que ajuda muito, entretanto existe diferença em trabalhar no Android Studio ou no XCode. A documentação do Android é muito prática, mas a plataforma da um pouco mais de trabalho que o iOS. Por outro lado, os participantes que tinham experiência prévia com Windows Phone, relataram que ele é padrão para todos que desenvolvem em C#, isso é interessante. No iOS os principais relatos de facilidades se referem a documentação, uso padrão de MVC e delegates, e a nova linguagem de programação Swift, relatando que quando migraram do Objective C para o Swift foi muito fácil de aprender, rodar e testar. Além disso, o iOS disponibiliza também o Storyboard que facilita muito no *design* dos aplicativos.

Devido ao fato dos ambientes de estudo serem focados em desenvolvimento iOS, pode ocorrer um viés de preferência pela plataforma iOS, motivo pelo qual classificamos estes indicativos inconclusivos no que se refere a influência da plataforma no aprendizado de desenvolvimento de aplicativos.

7.2.12 Percepção dos participantes em relação ao método tradicional de ensino

Na utilização do método proposto as aulas não são expositivas, são realizadas algumas apresentações e *workshops* para mostrar o rumo de determinados conteúdos e a partir dai os alunos tem autonomia para buscar o conhecimento da forma que preferirem. Os instrutores são mais guias do que professores. É interessante, mas ao mesmo tempo surgem desafios onde alguns alunos podem perder de aprender alguns conceito. Entretanto, *"o aprendizado é muito mais rápido do que qualquer escola de pedagogia tradicional."* (Participante 5)

Ao comparar este método com os métodos tradicionais, os participantes relatam que este método os ensina a aprender, a buscar o conhecimento. Entretanto na visão dos participantes o professor deixa de ser o ícone, ele passa a atuar mais como um mentor, fazendo perguntas que farão as equipes refletirem onde querem chegar. Obviamente os instrutores tem que ter um conhecimento avançado de forma a poder dar suporte de acordo com as demandas de diferentes equipes.

Analisando a postura do papel do aluno e do instrutor neste método, a postura de ambos é mais ativa do que no tradicional, pois ao invés dos alunos apenas acompanharem a exposição de conteúdo pelos instrutores, os alunos precisam ter atitude, buscar o que não conhecem, o que faz com que os alunos se estimulem em buscar o conhecimento.

> Acho que no método tradicional é muito chato, porque uma coisa que motiva a aprender é a finalidade, onde aplicar. O método tradicional peca nisso porque você não consegue ver a aplicação e o CBL é tudo isso, torna a atividade mais interessante. Tem um feedback. Para o lado eu como aluno vejo isso. O professor não tem mais aquela figura autoritária, ele define a tarefa e trabalha mais como um auxiliar, um Scrum Master. Tirando as barreiras se necessário, é mais um mediador. (Participante 6)

O fato do método expor uma situação e as pessoas buscarem o conhecimento ajuda a manter a pessoa engajada. O instrutor foca mais em guiar um caminho. O conhecimento que o aluno está buscando vem com *feedback* imediato. *"esse é um ponto gigantesco em questão de diferença com o tradicional"*. (Participante 7)

Os estudantes de forma geral estão acostumados com a estrutura professor, aluno. Neste sentido, muitas vezes os alunos não percebem que o professor na verdade é uma fonte de conhecimento que é um recurso que pode ser utilizado. Com o método os participantes percebem isso, eles tem acesso normalmente, compartilhando experiências, tanto o instrutor quanto o aluno são aprendizes no processo. Outra diferença é que este tipo de abordagem não segue os processos de avaliação tradicionais. A partir do momento que se quebra esse paradigma, os alunos entendem que precisam procurar as coisas e investigar o que os motiva. Os participantes relatam que o nível de liberdade durante o uso do método é ótimo, fazendo com que eles não se limitem na busca do conhecimento.

> Para os alunos acredito que a visão muda, saímos da visão acadêmica do professor na frente te ensinado e você fica meio limitado e tem pouca abertura para aprender outras coisas, mas tem essa questão, aprende o que o professor ensinou, aprende um pouco mais, mas não tem motivo para aprender mais ainda, porque o professor pede só aquilo. Fico um pouco limitado, de fato. Aprende o cronograma que foi dado na sala, no tradicional. Aqui não, aqui os professores tem um conhecimento x, mais ou menos e nós também, alguns tem um nível, outros não. Com os desafios que recebemos, os que já tem conhecimento acabam melhorando aquilo e aqueles que não tem já buscam mais, pesquisando, aprendendo, vendo com quem sabe; Dependendo dos desafios trabalhados aqui, alguns desenvolvem habilidades de umas ferramentas maiores do que de outros. Então, sempre temos informação para trocar, coisas que um trabalhou mais já, fica essa diferença de conhecimento, é positivo por um lado, porque você tem a quem recorrer, aquele cara que dedicou mais tempo para aquilo enquanto você estava lidando com outras questões. Fica um leque bem grande de conhecimento, não fica limitado ao que o professor pede. Aqui temos mais abertura. (Participante 13)

Um dos pontos levantados como potencial limitação do método, é quando existem alunos tímidos. De acordo com alguns participantes o método pode de alguma forma excluir, porque a pessoa fechada, muito tímida, aprende na forma tradicional, mas tem dificuldade com a abordagem baseada em desafios porque não consegue se comunicar com as pessoas. *"[...] um caso aqui dentro, que uma pessoa estava tendo dificuldade para se expressar. Ela foi acumulando as dificuldades e dois dias antes de entregar, não tinha conseguido fazer quase nada. Dificulta para quem tem problemas na comunicação."* (Participante 19)

Neste sentido, os participantes relataram que gostam do método e se sentem a vontade durante o seu uso, eles acreditam que os instrutores tem que cobrar com um *timeline*, a forma como os professores trabalham e os prazos definidos ajuda o pessoal a se organizar. Os alunos tem que buscar o conhecimento, os professores explicam como funciona o processo e nos entregáveis eles avaliam se o aluno tem alguma dificuldade ou não e provêem *feedback*, através de um processo transparente. *"Acho que o papel dos alunos é muito mais descobridor, de curiosidade e querer mesmo aprender, no momento que eu procurar algo, posso aprender coisas que eu não estava esperando"*. (Participante 32)

7.2.13 Avaliação da percepção do conhecimento adquirido

Os alunos foram convidados a emitir um parecer sobre o seu aumento de conhecimento sobre desenvolvimento de aplicativos para dispositivos móveis, fazendo um comparativo com o conhecimento que eles tinham antes de entrar no programa sem utilizar o método e o conhecimento obtido após o programa utilizando o método. A auto-avaliação dos participantes em relação aos conhecimentos adquiridos em desenvolvimento de aplicativos é apresentada na Tabela 7.5.

Tabela 7.5: Auto-avaliação de conhecimento dos participantes do estudo de caso

Univ.	Região	Participante	Idade	Curso	Semestre	Nota antes	Nota depois
U1	Nordeste	#1	23	Ciência da Computação	9	2	7
U1	Nordeste	#2	27	Ciência da Computação	4	1	7,5
U1	Nordeste	#3	21	Ciência da Computação	8	3	8
U1	Nordeste	#4	26	Sistemas da Informação	6	3	7
U1	Nordeste	#5	20	Ciência da Computação	4	1	8
U1	Nordeste	#6	27	Sistemas da Informação	6	1,5	7
U1	Nordeste	#7	23	Ciência da Computação	5	0	7
U1	Nordeste	#8	23	Design	10	6	8
U2	Norte	#9	19	Engenharia da Computação	4	4,5	7,5
U2	Norte	#10	20	Ciência da Computação	6	0	9
U2	Norte	#11	22	Sistemas da Informação	8	4	10
U2	Norte	#12	21	Ciência da Computação	7	5	8,5
U2	Norte	#13	25	Sistemas da Informação	4	3	9
U2	Norte	#14	24	Engenharia da Computação	9	5	8
U2	Norte	#15	22	Ciência da Computação	6	0	7
U2	Norte	#16	23	Ciência da Computação	9	1	7
U3	Sudeste	#17	26	Ciência da Computação	5	3	10
U3	Sudeste	#18	20	Engenharia Elétrica	4	2	9
U3	Sudeste	#19	25	Ciência da Computação	4	0	7
U3	Sudeste	#20	20	Ciência da Computação	6	1	10
U3	Sudeste	#21	27	Ciência da Computação	7	0	5
U3	Sudeste	#22	20	Ciência da Computação	3	1	6
U3	Sudeste	#23	20	Engenharia Elétrica	4	0	5
U3	Sudeste	#24	21	Sistemas da Informação	5	0	8,5
U4	Sul	#25	40	Análise de Sistemas	6	7	9
U4	Sul	#26	21	Engenharia da Computação	7	4	9
U4	Sul	#27	22	Design de Games	4	0	7
U4	Sul	#28	22	Jogos Digitais	6	6	9
U4	Sul	#29	21	Sistemas para Internet	5	0	7
U4	Sul	#30	23	Engenharia da Computação	10	5	8
U4	Sul	#31	23	Engenharia da Computação	6	0	5
U4	Sul	#32	24	Sistemas da Informação	8	7	8,5

Em média, dentro de uma escala de 0 a 10, os alunos se auto-avaliaram com uma nota 2,37 anteriormente ao treinamento, e após o treinamento a média passou para 7,76, mostrando uma percepção de aprendizado média de 5,39 pontos, triplicando o aumento do conhecimento de acordo com a percepção dos participantes. Os números desses diferentes casos vão de encontro ao estudo de campo realizado anteriormente nesta pesquisa, com um aumento médio na percepção do aprendizado de no mínimo 5 pontos (de dez). Auto-avaliação estão disponíveis na Tabela 7.6.

Tabela 7.6: Estatística da auto-avaliação de conhecimento

	Nota antes	Nota depois
Média	2,37	7,76
Mínimo	0	5
Máximo	7	10
Desvio Padrão	2,32	1,36

Baseado nos dados da amostra, é possível observar que os participantes que se auto-avaliaram em um aumento de conhecimento de no mínimo 5 pontos, são alunos a partir do quarto semestre, não foi possível estabelecer uma correlação entre cursos, pois a maioria dos participantes da amostra é de Ciência da Computação e Sistemas de informação.

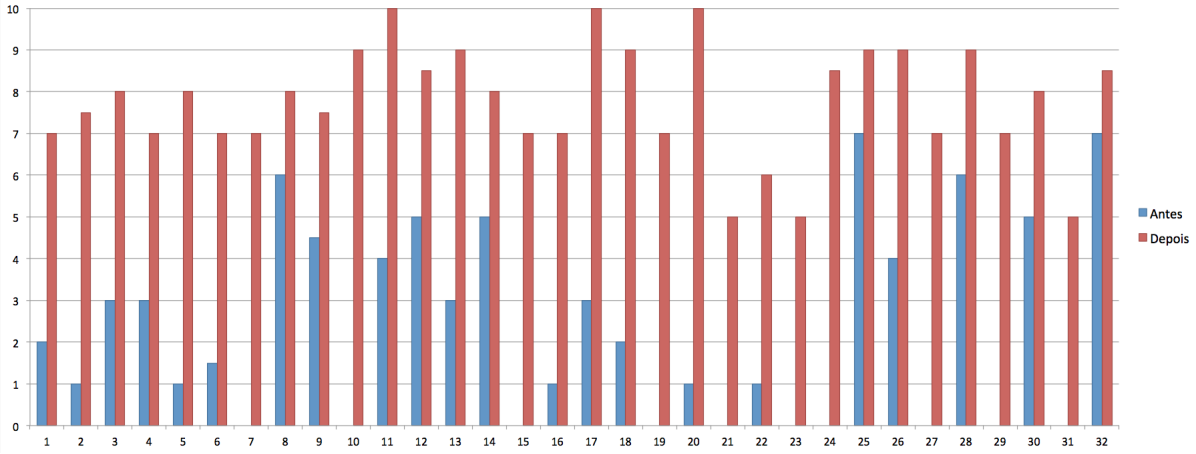

Figura 7.6: Histograma da auto-avaliação de conhecimento

Tabela 7.7: Análise do histograma da auto-avaliação de conhecimento

Aumento de Conhecimento	Participantes	Semestre
>= 5 pontos	1, 3, 6, 21, 22, 23, 31	Acima do quarto
>= 6 pontos	2, 11, 13, 16	Acima do quarto
>= 7 pontos	5, 7, 10, 15, 17, 19, 20, 24, 27, 29	Entre quarto e sexto

A partir da análise qualitativa das entrevistas e da análise da Tabela 7.7, pode-se concluir que existe uma relação entre a percepção do aumento do conhecimento e possivelmente um aumento real de conhecimento para estudantes a partir do quarto semestre. Entretanto, isso deve ser classificado como um indicativo e não deve ser generalizado devido aos dados da amostra da população de participantes neste estudo.

7.3 Limitações e ameaças a validade

Em diversos estudos existem ameaças que podem afetar a validade dos resultados. Este estudo possui algumas limitações semelhantes as encontradas no estudo de campo. Primeiramente, alguns participantes possuem diferentes níveis de experiência, incluindo experiência em desenvolvimento de aplicativos e metodologias de desenvolvimento. Além disso, nenhum dos estudantes teve contato anterior com metodologias de ensino baseadas em pedagogia ativa. Neste sentido, estes fatores podem ter influenciado positivamente ou negativamente os resultados encontrados, de forma a minimizar esta limitação o estudo foi realizado em diferentes universidades. Entretanto, preocupa-se com a generalização dos resultados. O método proposto baseou-se na literatura e nas opiniões de especialistas na área. Isto pode ter resultado em uma solução que funcionou nestes locais de estudo, mas não pode ser generalizado para outros locais. Além disso, normalmente ambientes acadêmicos não simulam totalmente as condições existentes em um ambiente de desenvolvimento da indústria. No que refere-se a relação entre a teoria e a observação, encontrou-se indicativos através dos resultados obtidos de forma qualitativa que existe uma relação dos resultados com a base teórica deste estudo.

7.4 Considerações Finais do Estudo de Caso

O desenvolvimento ágil apresenta uma influencia positiva nos projetos de desenvolvimento de aplicativos, especialmente devido à característica que estes projetos tem um curto ciclo de vida de desenvolvimento, o que gera uma necessidade de entrega contínua e velocidade no ciclo de desenvolvimento. As perspectivas positivas do desenvolvimento ágil englobam rapidez na identificação de problemas, o *feedback* constante, organização, velocidade de desenvolvimento, transparência, comunicação, flexibilidade, desempenho e qualidade. Fatores foram identificados durante a análise da influência o desenvolvimento ágil, entre eles a influência positiva da organização dos projetos ágeis no desempenho da equipe e na velocidade do desenvolvimento do projeto. Além disso, a influência positiva da transparência proporcionada pelo desenvolvimento ágil na comunicação e controle do desenvolvimento do projeto. Adicionalmente, a rapidez na identificação dos problemas ajuda no processo de melhoria contínua, facilitando a correção nos desvios do projeto de forma efetiva e transparente.

Analisando o aspecto de práticas de desenvolvimento, as principais práticas utilizadas durante a execução dos projetos foram reuniões diárias, *collocation*, kanban, programação em par, pequenas *releases* e planejamento iterativo. Além das práticas ágeis, também foram aplicadas outras práticas de desenvolvimento durante o desenvolvimento dos projetos de aplicativos, dentre elas: protótipo, modelagem ER, modelagem UML e testes unitários. Entretanto, de acordo com os resultados encontrados neste capítulo, pode-se observar que a área de testes ainda recebe pouca atenção por parte das equipes durante os projetos, essa é uma questão a ser analisada em estudos futuros.

No que se refere a influência das plataformas de desenvolvimento no aprendizado (iOS, Android, Windows Phone) não foi possível estabelecer uma conclusão, existem indicativos de algumas

características da plataforma iOS que facilitam o desenvolvimento, entretanto outros participantes classificaram que o que influencia no aprendizado mesmo é o interesse e disposição do aluno. De acordo com um estudo realizado por Miranda *et al.* [81] embora seus resultados identifiquem um maior grau de compatibilidade para a plataforma Android, os entrevistados informaram que a complexidade do desenvolvimento de aplicativos para a plataforma iOS é menor do que para o Android, após os primeiros passos para obter algum conhecimento básico de trabalho. Os resultados dos autores também informam que a plataforma Android está presente em aparelhos de diferentes marcas e modelos (Samsung, Sony, Motorola, LG etc) e que cada fabricante modifica o sistema com a intenção de personalizá-lo para os seus dispositivos.

Diferentes práticas foram identificadas no processo de definição do desafio, estas práticas contemplam desde a montagem da equipe, até a realização de *brainstorm* para definição da big idea, essential question e*challenge*, como também a inclusão de pesquisa de campo para validar se o problema a ser resolvido pelo desafio é viável. Além disso os participantes também demonstraram que procuram pessoas com interesses em comum para a montagem de suas equipes. Outras práticas interessante abordada por algumas equipes é a análise de aplicativos existentes, a busca do entendimento do contexto dos indivíduos e utilização de *mind maps* para organização dos seus desafios. Entretanto, algumas oportunidades de melhoria foram identificadas, como por exemplo a possibilidade de usar Lean ao invés de Scrum, criar uma opção de retorno mo método a partir do *sprint planning* para o processo de *guiding questions* e também uma opção de retorno do *research findings* para as *guiding questions*. Um dos aspectos que chamou a atenção no processo de definição do desafio, é que um site em especifico passa a *big idea* definida para as equipes, gerando um questionamento sobre o quanto isso pode impactar no engajamento dos estudantes. Isso é um aspecto que será abordado em estudos futuros.

O processo de pesquisa também se mostrou como algo dinâmico e com diferentes práticas adotadas pelos participantes entre elas: assistir documentários sobre o tema de interesse, participação em eventos do tema de interesse, reuniões com os especialistas, pesquisa de campo, análise de mercado e concorrência, busca de tendências e inovação, além da definição de prioridades nas perguntas a serem respondidas.

A transição da pesquisa para o desenvolvimento também envolve diferentes práticas como por exemplo: validação do *product backlog* com os resultados do research finds, criação de protótipo, definição de MVP. Definição da arquitetura do aplicativo, modelagem de classes e modelagem ER. Criação do ambiente de desenvolvimento e *story board*. O processo de desenvolvimento e práticas já foi descrito anteriormente, mas engloba diferentes práticas essenciais como reuniões diárias, definição de tarefas o mais curtas possíveis, MVP, foco na melhoria e entrega contínua. Algumas oportunidades de melhoria foram encontradas para o processo de desenvolvimento, dentre elas o uso mais efetivo do gráfico *burn down* e do quadro Kanban.

Quanto ao processo de avaliação, ele basicamente engloba o uso de *reflections, sprint review e sprint retrospective*. As *reflections* se mostraram como uma ferramenta efetiva durante o processo de aprendizado, entretanto não se aplicam durante o ciclo de desenvolvimento do produto. As

revisões de sprint se mostraram como uma oportunidade que ajuda na obtenção de *feedback* do produto e melhoria do mesmo. Por outro lado, as retrospectivas se mostraram como aliadas no processo de melhoria contínua do trabalho em equipe, onde as pessoas podem compartilhar um *feedback* construtivo sobre o trabalho realizado pela equipe no seu dia-a-dia.

Em questões de comunicação, Avritzer *et al.* [9] apresentaram um estudo sobre como processos ágeis podem melhorar produtividade em projetos de desenvolvimento de software. Neste estudo, um dos principais pontos de comunicação foi chamado de *"Comunicação Informal"*, o qual se refere a comunicações informais e espontâneas que ocorrem em membros em uma equipe trabalhando fisicamente próxima. Pois este tipo de comunicação é muito rico em manter os membros da equipe informados sobre o que esta acontecendo diariamente. Os autores relatam que este tipo de comunicação é muito difícil de se realizar em equipes que trabalham distribuído. Aspecto o qual pudemos confirmar como um facilitador na configuração de nosso estudo, devido a maioria dos participantes confirmar que trabalham em *collocation* e que isso melhora a comunicação ao ponto que algumas equipes as vezes nem chegam a fazer a reunião diária.

Outro fator encontrado no estudo de caso é a questão do uso de programação em par, a qual é frequentemente utilizada quando os alunos precisam trabalhar em atividades mais complexas ou quando estão tentando aprender algo novo. Esta característica também é citada por Avritzer *et al.* [9], de acordo com a experiência dos autores a programação em par funciona melhor quando as pessoas são agrupadas porque tem habilidades complementares, ou quando uma pessoa mais experiente atua como mentor de uma pessoa menos experiente.

A instabilidade de requisitos é um problema crítico em ambientes de desenvolvimento de aplicativos para dispositivos móveis, e este problema é endereçado através do uso de desenvolvimento ágil. Esta é uma das principais vantagens do uso de desenvolvimento ágil neste tipo de ambiente. As abordagens ágeis são particularmente efetivas para lidar com requisitos instáveis devido a curtos ciclos de *feedback* e revisão, permitindo que o trabalho seja concentrado nas funcionalidades de maior prioridade e incrementalmente desenvolvendo um sistema que incorpora as funcionalidades realmente desejadas [9].

A utilização do MVP foi outra descoberta durante o estudo de caso. MVP é definido como o conjunto mínimo de funcionalidades que permite uma ação e aprendizado sobre os clientes ou usuários [18]. Diferentes equipes buscam desenvolver seus aplicativos com o foco inicial em MVP de forma a garantir um aprendizado sobre seus usuários e um melhor incremento no seu produto.

Quanto ao fato de usar Lean ao invés de Scrum, sugerido como oportunidade de melhoria, talvez não seja uma boa alternativa devido ao fato de que a natureza iterativa da abordagem original do Lean é diferente do modelo Scrum, pois cada sprint do lean serve para validar as funcionalidades que um cliente deseja pagar, e não apenas construir um produto incremental e sem desperdício [18]. Uma vez que esta abordagem está sendo utilizada com o intuito de buscar melhorar o suporte ao aprendizado de desenvolvimento de aplicativos, a base do modelo Scrum é melhor indicada.

O método aplicado em quatro diferentes universidades se apresentou como influência positiva no engajamento, aprendizado e colaboração entre os estudantes. A combinação de uma abordagem

baseada em desafios com práticas ágeis se apresenta como uma alternativa efetiva em ambientes de ensino e aprendizado de desenvolvimento de aplicativos para dispositivos móveis. Diferentes práticas e oportunidades de melhoria foram identificadas neste estudo, as quais foram aplicadas no método final que é descrito o próximo capítulo que apresenta a evolução do método.

8. EVOLUÇÃO DO MÉTODO

Este capítulo apresenta o método final. Este método é a principal contribuição desta tese. A seção 8.1 apresenta a construção do método, a seção 8.2 descreve a estrutura do método, a seção 8.3 analisa os efeitos de desenvolvimento de aplicativos com os resultados do uso do método e por fim a seção 8.4 apresenta as considerações finais.

8.1 Processo de construção do método final

O método final foi construído baseado nos resultados das etapas definidas no desenho de pesquisa (Seção 3.1). O desenho de pesquisa engloba 3 Etapas: Etapa 1, Etapa 2 e Etapa 3. Na etapa 1, foram coletadas informações para a preparação da proposta inicial do método. Na etapa 2, foi construída a proposta inicial do método e foi realizado um estudo de campo para analisar o uso do método de forma a entender os desafios, vantagens, desvantagens e lições aprendidas no uso da proposta inicial do método. Na etapa 3 foi realizado um estudo de caso para identificar as oportunidades de melhoria, recomendações e melhores práticas no uso do método de forma a propor o método final. A adoção destes estudos resultou um conjunto de contribuições que foram utilizadas para propor o método final, conforme Figura 8.1.

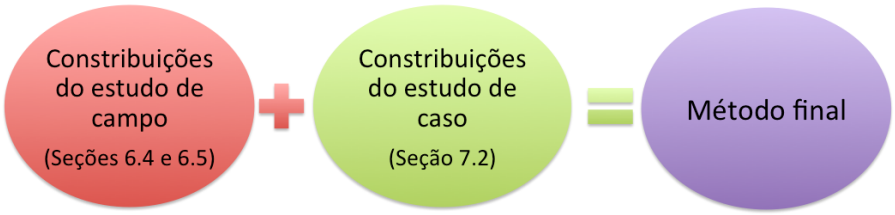

Figura 8.1: Processo de construção do método final

No estudo de campo, foram coletados dados de 94 participantes através de questionários (Seção 6.4), os quais resultaram em um melhor entendimento do ambiente de desenvolvimento, identificação de vantagens e desvantagens no uso do método como também um conjunto de lições aprendidas. Estas informações permitiram planejar e analisar a implementação do método proposto através de um estudo de caso em quatro universidades.

Através do estudo de caso, foram realizadas 32 entrevistas em diferentes universidades de forma a identificar as oportunidades de melhoria, melhores práticas e recomendações no uso do método proposto. Neste sentido, o método final inclui todas as práticas, recomendações e melhorias. As melhorias no fluxo do método basicamente incluem opções de retorno da fase de pesquisa durante o aprendizado para a fase de visão e definição do problema que o aplicativo busca resolver; a opção de retorno do planejamento da sprint na fase de desenvolvimento para a fase de pesquisa; e a opção de retornar da fase de avaliação para a fase de pesquisa, conforme apresentado na Subseção 8.1.1 a seguir.

8.1.1 Oportunidades de melhoria aplicadas no fluxo do método

As melhorias aplicadas no fluxo do método basicamente referem-se a opções de retorno de determinados estágios do método para o processo de pesquisa, e englobam:

Alteração 1: Retorno entre a fase de pesquisa e a definição do problema na fase de visão

A opção de retornar do processo de pesquisa para a essential question foi sugerida pelos participantes, pois em alguns processos de pesquisa pode-se chegar a conclusão que o problema que esta se buscando resolver e o desafio proposto precisam ser reformulados, pois a pesquisa pode indicar que eles foram inicialmente mal definidos.

Alteração 2: Retorno entre a o planejamento da sprint na fase de desenvolvimento e a fase de pesquisa

A opção de retornar do estágio de planejamento da sprint para o processo de pesquisa foi mencionada pelos participantes, pois em algumas situações especificas, mesmo que a equipe já tenha planejado o *product backlog* pode ocorrer o caso de durante o planejamento da sprint e discussão de atividades especificas podem surgir dúvidas que não foram respondidas suficientemente no processo de pesquisa anterior, fazendo com que a equipe faça pesquisa sobre algum item em especifico.

Alteração 3: Retorno entre o final da fase de avaliação e a fase de pesquisa

Alguns participantes também citam que pode ocorrer um ciclo entre o final de uma iteração e o processo de pesquisa. Ou seja, ao se finalizar determinada iteração de desenvolvimento e se realizar as cerimônias de revisão de sprint e retrospectiva de sprint, podem-se encontrar pontos e oportunidades de melhoria que precisam ser pesquisados, necessitando assim uma opção de retorno do processo de avaliação para o processo de pesquisa.

Além das três alterações no fluxo do método, também foram sugeridas algumas oportunidades de melhoria em práticas que podem ser melhor utilizadas durante a aplicação do método. Dentre estas práticas podemos destacar o uso da *reflection*, pois existem indivíduos que não entendem o quanto as *reflections* podem ajudar os instrutores a melhorarem o processo de aprendizado, isso é um fator muito importante no processo de aprendizado e que deve ser melhor entendido pelas equipes que fazem parte do processo de aprendizagem.

Outra prática que pode ser melhor utilizada em projetos futuros refere-se a retrospectiva de sprint, pois poderiam ser melhor utilizadas por algumas equipes em especifico, porque geralmente no final da sprint a equipe está cansada não aproveitando da melhor forma possível o momento de aplicar as lições aprendidas durante a iteração de desenvolvimento.

8.2 Estrutura do método final

O método final contempla cinco fases conforme apresentado na Figura 8.2, são elas: Visão, Pesquisa, Solução, Desenvolvimento e Avaliação.

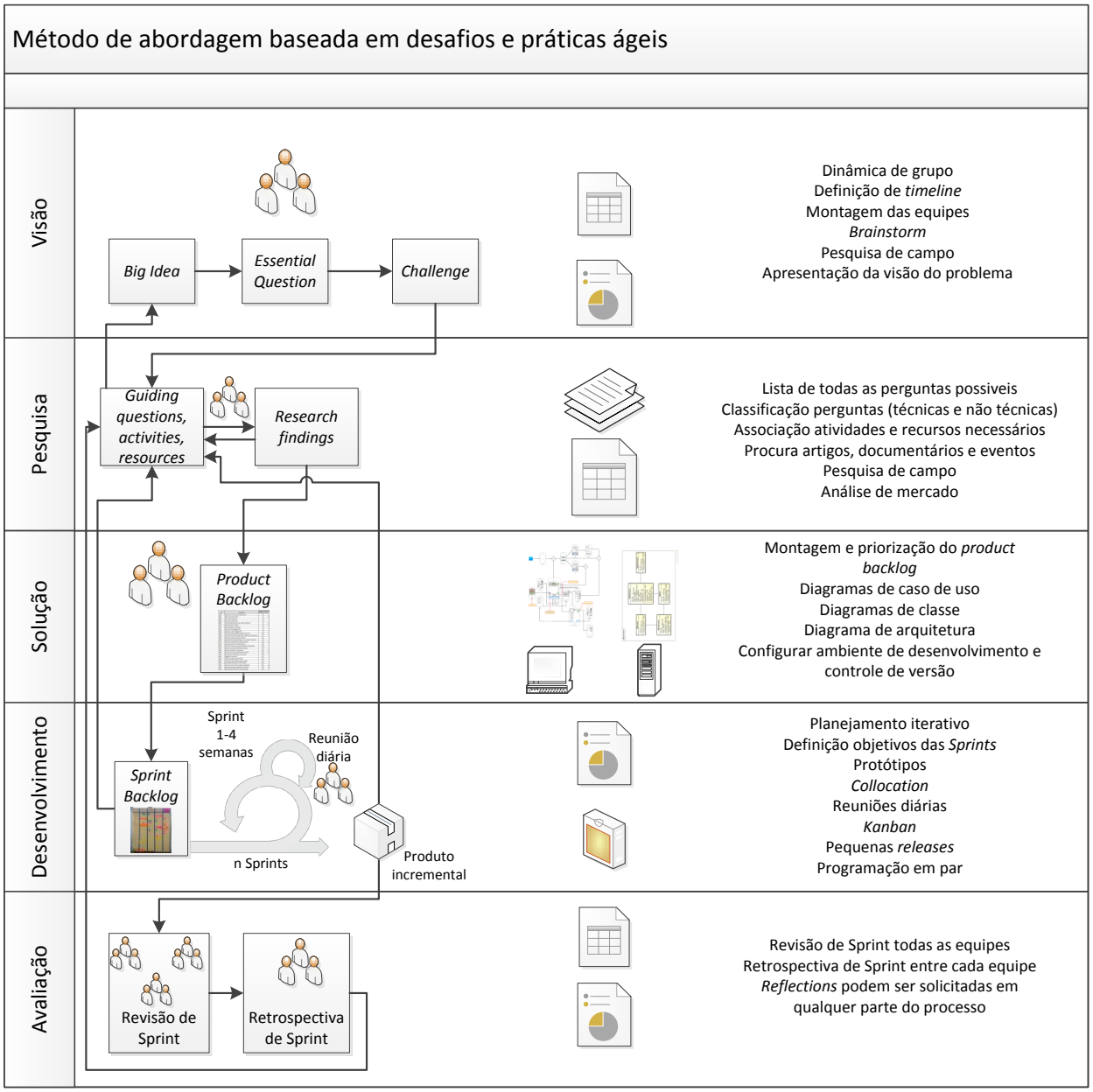

Figura 8.2: Versão final do método

O método inicia na fase da visão, a qual é fundamental para a definição da idéia a ser desenvolvida através da resolução de um problema real. Esta é a fase onde as equipes são formadas através da definição de uma idéia na qual os estudantes se sintam engajados. O processo da fase de visão é apresentado na próxima subseção 8.2.1.

8.2.1 Fase de visão

A fase de visão basicamente engloba as dinâmicas de grupo, montagem dos grupos e definição do desafio. O processo desta fase é apresentado na Figura 8.3.

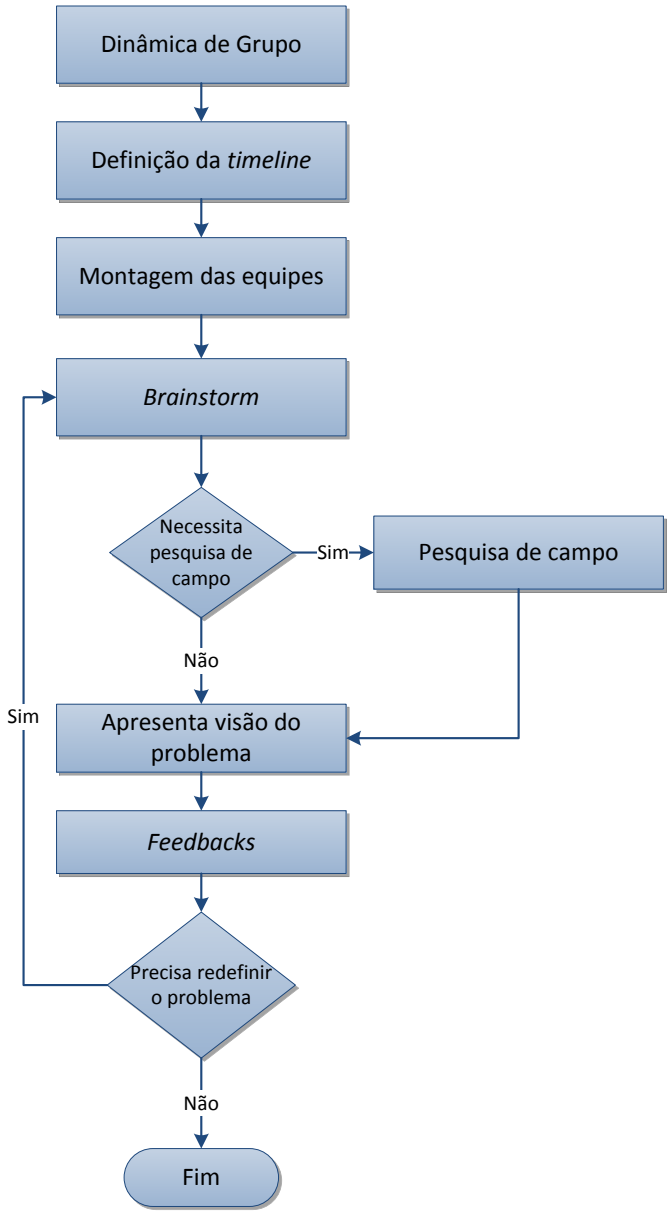

Figura 8.3: Fase de visão

Geralmente quando os estudantes começam a trabalhar no mesmo ambiente, eles ainda não se conhecem. A realização de dinâmicas de grupo, onde cada um dos alunos se apresenta e compartilha quais os seus costumes, habilidades e tópicos de interesse facilita muito na montagem das equipes. Neste sentido, quando eles começam a trabalhar juntos e se integram em torno de objetivos geralmente em comum. Dessa forma, as equipes são montadas geralmente por afinidades em torno de um tema especifico. Uma outra abordagem de dinâmica de grupo pode ser de forma interdisciplinar, agrupando alunos com conhecimentos distintos como por exemplo Computação, Design e alguma outra área de interesse. O número de integrantes de cada equipe deve ser decidido pelos alunos, entretanto pode ser recomendado o número de integrantes de cada equipe.

A definição de uma *timeline* deve ser feita entre o grupo de instrutores para dar uma direção aos alunos. Esta atividade tem uma influência muito grande para guiar os conteúdos e atividades a serem trabalhados, como também apresentar para os alunos uma comunicação clara e objetiva sobre as expectativas a serem atingidas através da abordagem do método proposto. Uma *timeline* com todas as entregas a serem realizadas para um conjunto de objetivos de aprendizado deve ser compartilhada.

As equipes inicialmente buscam analisar temas que fazem parte dos seus problemas cotidianos, ou que influenciem a comunidade da qual eles fazem parte. Estes temas geralmente envolvem uma diversidade de problemas que englobam diferentes áreas como por exemplo: Saúde, Educação, Entretenimento, Mobilidade urbana, Produtividade, Finanças, Esportes, etc. Qualquer membro de uma equipe tem liberdade para propor diferentes temas. As equipes devem buscar por assuntos que realmente queiram trabalhar e se sintam engajados para resolver um problema real. Outras práticas podem incluir a votação em idéias ou até mesmo a seleção de duas a três idéias que sejam de interesse da maioria, a partir dai tentam elaborar uma questão de algum problema que faça parte da ideia, gerando uma discussão geral de idéias e abordagens. É possível também se realizar uma análise de fortalezas e fraquezas de diferentes idéias em uma lista, gerando perguntas para dúvidas ou problemas, de forma a clarificar o problema da melhor forma possível.

Uma vez que é realizado o *brainstorm* inicial, as equipes precisam entender os interesses que pessoas teriam naquela área, para poder formar as *essential questions*. Deve se identificar um certo problema a ser resolvido. Dentro desta prática pode ser realizada uma pesquisa de campo inicial, de forma a tentar obter um *feedback* sobre o problema que será resolvido dentro de uma determinada área, de forma a refinar a proposta de um desafio, como também obter opiniões externas para ter uma visão sob diferentes perspectivas do desafio a ser proposto. Podem ser utilizadas ferramentas especificas para documentar este processo, que inclui formulários, planilhas, documentos e mapas mentais.

A apresentação da visão do problema encontrado para o grande grupo de instrutores e alunos proporciona uma diversidade de *feedbacks* construtivos que colaboram com o processo de ensino e aprendizado. Estas apresentações permitem as equipes de alunos a compartilharem quais são suas *big ideas, essential questions e challenges* de forma a proporcionar um conjunto de *feedbacks* para as equipes. Em algumas ocasiões, dependendo do *feedback* recebido pode ser necessário reavaliar o problema a ser resolvido, e neste caso a equipe voltaria a realizar sessões de *brainstorm*. Após o término das atividades da fase de visão, se inicia a fase de pesquisa.

8.2.2 Fase de pesquisa

A fase de pesquisa basicamente engloba a listagem das perguntas guia (*guiding questions*), a classificação das perguntas, a associação de atividades e recursos, pesquisa de campo, consulta a especialistas, busca de artigos, eventos ou documentários e análise de mercado. O processo envolvido na fase de pesquisa é apresentado na Figura 8.4.

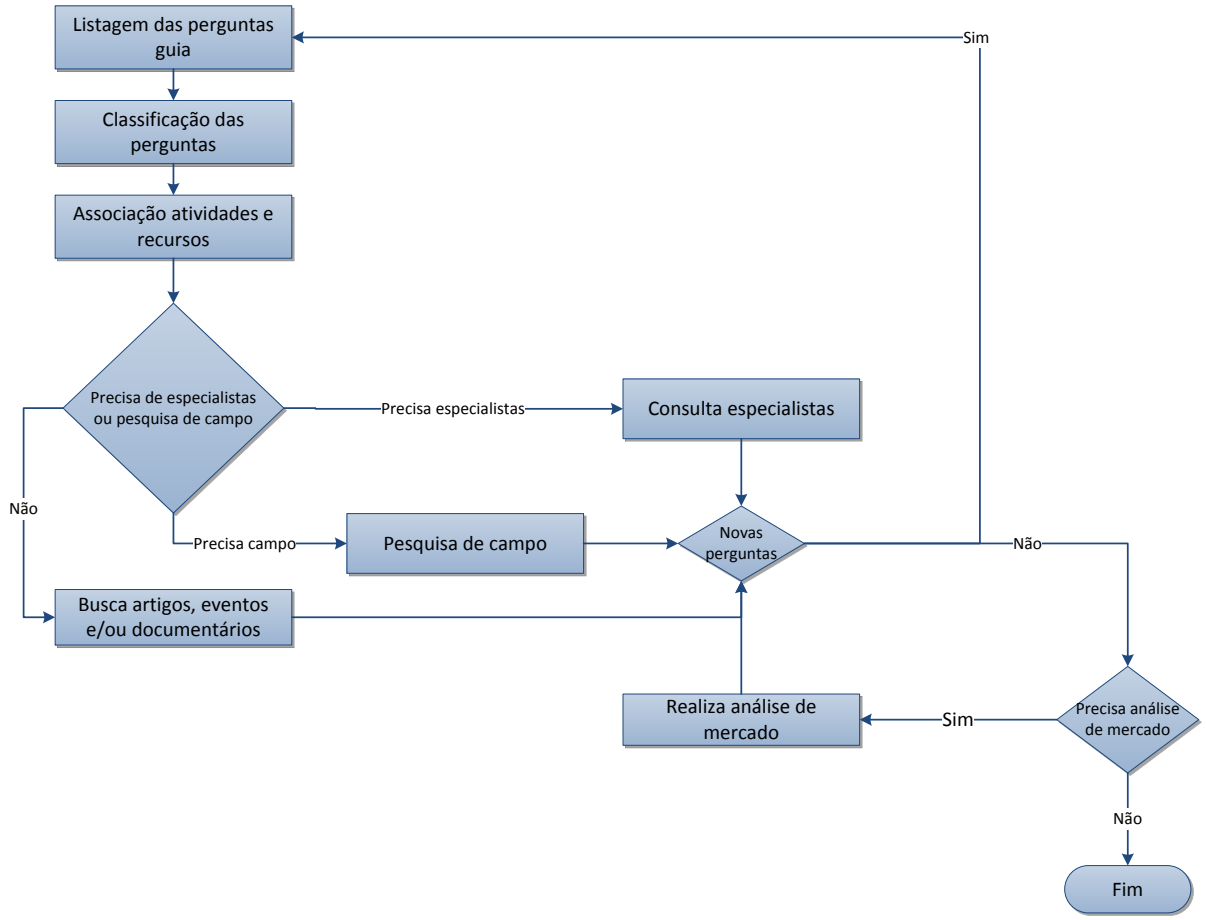

Figura 8.4: Fase de pesquisa

A organização e priorização do processo de pesquisa é uma prática muito importante e pode ser realizada através do uso de diferentes ferramentas. O primeiro passo é a criação de uma lista com todas as perguntas que vão guiar o processo de aprendizado para resolver o desafio proposto. Uma vez que esta lista é criada, as perguntas devem ser classificadas em perguntas técnicas e não técnicas de forma a facilitar a organização dos recursos e atividades necessários para resolver o processo de pesquisa. Buscar identificar todas as atividades e recursos necessários para responder as perguntas guia é um passo muito importante no processo. Devem ser listados todos os conhecimentos necessários, tecnologias necessárias, especialistas necessários, perguntas sobre *stakeholders*, mercado, e qualquer outro conhecimento ou recurso necessário independente da complexidade do desafio a ser pesquisado. Após estas definições iniciais são definidas as prioridades das perguntas a serem respondidas, de forma a se buscar os recursos necessários para responder as perguntas com maior prioridade.

Após a realização das definições iniciais do processo de pesquisa, deve ser avaliado se é necessário realizar pesquisa de campo ou consultar especialistas. Outras práticas que podem ser adotadas incluem a busca de artigos técnicos ou científicos sobre o problema a ser resolvido, como também a participação em eventos sobre o tema, que vão desde *workshops* até conferências em determinadas

áreas. Estes eventos muitas vezes além de ser um recurso por si só, também geram contatos que podem se tornar recursos durante o o ciclo do processo de pesquisa e aprendizado. Além do fato de participar de eventos, também é possível buscar documentários sobre o assunto em pesquisa, o que promove um olhar sobre uma outra perspectiva, pois os alguns documentários podem apresentar resultados de pesquisas relacionadas.

Uma vez que os recursos e atividades essenciais ao problema de pesquisa tenham sido devidamente associados, pode ser necessário uma análise de mercado antes de definir a solução que engloba o aplicativo a ser desenvolvido. Dessa forma, a análise de mercado envolve a busca de aplicativos que resolvem problemas semelhantes, de forma a identificar oportunidades de melhoria e potenciais de inovação como concorrência. Outra parte da análise do mercado envolve o entendimento sobre o público alvo, e o mercado que se busca atingir com o aplicativo. Existem diferentes técnicas de análise de mercado mais especificamente relacionadas com empreendedorismo o que não é escopo deste trabalho e por isso fica como sugestão de trabalhos futuros. A partir dos resultados da fase de pesquisa é realizada uma síntese de forma a propor uma solução ou conjunto de soluções a serem implementados.

8.2.3 Fase de solução

A fase de solução basicamente engloba a montagem do *product backlog* a partir dos resultados encontrados na fase de pesquisa, e posterior criação dos diagramas e modelagem da arquitetura da solução e também configuração do ambiente de desenvolvimento da solução. A definição do processo que faz parte da fase de solução é apresentado na Figura 8.5.

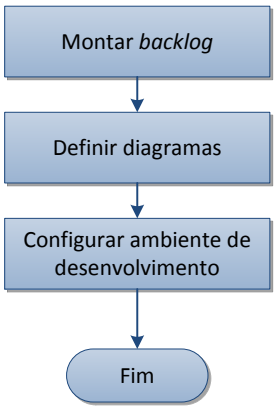

Figura 8.5: Fase de solução

O primeiro passo na fase de solução é entender os resultados obtidos nas fases anteriores do processo de forma a definir um *product backlog*. Dentro dos ambientes de ensino e aprendizado de desenvolvimento de aplicativos pesquisados, algumas vezes este processo pode envolver participantes externos que auxiliam como *product owner* PO, e para os casos que não possuem este ator externo, o papel pode ser desempenhado por algum membro da equipe ou até mesmo por algum instrutor.

Uma vez que o *product backlog* foi definido e priorizado, é realizada a modelagem da solução através de diferentes técnicas e ferramentas, que incluem diagramas de modelagem UML e diagramas de modelagem ER [1]. Realiza-se também a definição da arquitetura a ser utilizada pelos projetos dos aplicativos. A partir do momento que os principais diagramas da solução a ser implementada são criados, iniciam-se as atividades de configuração do ambiente de desenvolvimento. A configuração do ambiente de desenvolvimento engloba atividades como: a criação do ambiente e repositório de código, definição de *scripts* para *builds* automatizados e definição pelo uso ou não de integração contínua. Após a realização dos passos que definem a solução e ambiente de desenvolvimento se inicia a fase de desenvolvimento dos aplicativos.

8.2.4 Fase de desenvolvimento

A fase de desenvolvimento engloba todo o ciclo de desenvolvimento, que inclui o planejamento de *releases*, definição dos objetivos das sprints, planejamento das sprints, construção de protótipos, e as iterações desenvolvimento ágil realizando entregas contínuas. A base do processo envolvido nesta fase é apresentada na Figura 8.6.

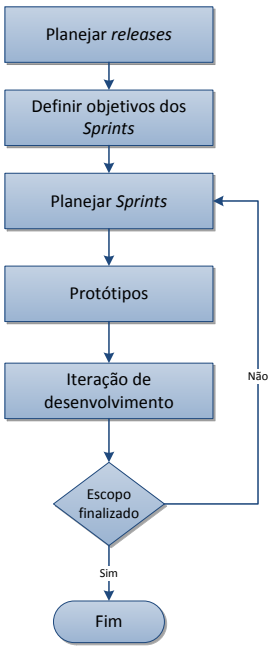

Figura 8.6: Fase de desenvolvimento

O processo da fase de desenvolvimento inicia-se através de um planejamento das *releases* do aplicativo a ser desenvolvido. Basicamente o planejamento das *releases* engloba a definição da quantidade de *releases* a serem desenvolvidas como também as datas de disponibilização de cada uma delas. As releases de aplicativos usualmente são disponibilizadas em plataformas de distribuição conhecidas como *application stores*, entretanto podem existir aplicativos para fins específicos (como

[1] Dependendo da natureza do aplicativo pode ser desnecessário o uso de modelagem ER.

por exemplo área de saúde) que são aplicativos instalados em dispositivos específicos, sem utilizar as plataformas tradicionais de distribuição.

A partir do momento que as *releases* dos aplicativos foram definidas, deve ser realizada uma definição dos objetivos a serem atingidos em cada sprint. Ou seja determinar o que se planeja conquistar ao final de cada iteração do ciclo de desenvolvimento. Uma vez que os objetivos das sprints foram planejados, recomenda-se que sejam criados protótipos que possibilitem obter um *feedback* sobre os aplicativos a serem desenvolvidos, de forma a determinar um melhor planejamento dos sprints. Cada sprint é planejada antes do inicio de cada iteração de desenvolvimento. Cada iteração de desenvolvimento envolve atividades de código e testes utilizando diferentes técnicas e ferramentas apresentados nos resultados do estudo de caso na Seção 7.2. Neste sentido, o ciclo de desenvolvimento engloba o planejamento das sprints, desenvolvimento, testes, de forma a gerar entregas contínuas de um produto, neste caso aplicativos para dispositivos móveis.

8.2.5 Fase de avaliação

A fase de avaliação basicamente abrange a revisão do incremento do produto após cada iteração de desenvolvimento, a retrospectiva da iteração de desenvolvimento, documentação de funcionalidades e melhorias, documentação de lições aprendidas e reflexões sobre o aprendizado durante o ciclo de desenvolvimento. O processo desta fase é apresentado na Figura 8.7.

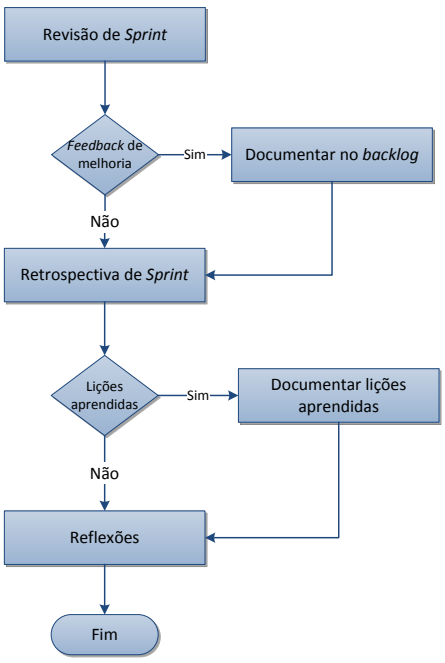

Figura 8.7: Fase de avaliação

O processo da fase de avaliação é importante em diferentes aspectos. Primeiramente sobre a perspectiva de reflexão e melhoria do aprendizado, tanto em termos de conhecimento em desenvolvimento de aplicativos, como também em termos de trabalho em equipe e processos de desen-

volvimento, gerando um processo de aprendizado contínuo. Em segundo lugar, por proporcionar a melhoria contínua do aplicativo a ser desenvolvido.

A revisão da sprint permite que o incremento do produto desenvolvido em cada iteração seja avaliado de forma a proporcionar um conjunto de *feedbacks* de melhoria do produto. Os *feedbacks* de melhoria do produto devem ser documentados no *product backlog*. Assim como a retrospectiva da sprint permite um olhar crítico sobre a forma de trabalho ajudando a identificar o que está sendo feito corretamente e deve continuar sendo feito, o que seria interessante de começar a ser feito, e o que está sendo feito de maneira incorreta que deve ser feito de outra maneira ou parar de ser feito. As lições aprendidas durante as retrospectivas de sprint devem ser documentadas de forma a permitir ao time possuir um histórico das lições e melhorias no processo de desenvolvimento do aplicativo.

Uma vez que são realizadas todas as etapas da fase de avaliação, deve ser permitido também um momento de reflexão a ambos alunos e instrutores de forma a avaliar tudo o que foi aprendido em termos de conhecimento, processos e produto, para solidificar o processo de aprendizado. Uma vez que os principais processos do método final foram apresentados, a próxima seção apresenta uma análise dos efeitos no desenvolvimento de aplicativos através do uso do método em ambos estudo de campo e estudo de caso.

8.3 Efeitos no Desenvolvimento de Aplicativos

Os resultados obtidos durante os estudo de campo e estudo de caso, apresentaram dados em relação aos efeitos no desenvolvimento de aplicativos identificados no Capítulo 4. Alguns destes efeitos foram confirmados como fatores de influência no aprendizado e desenvolvimento de aplicativos para dispositivos móveis, enquanto outros se mostraram inconclusivos, conforme apresentado na Tabela 8.1.

As abordagens de pedagogia ativa descritas na literatura [4, 31, 36, 39, 75, 104], e utilizadas nesta pesquisa através da abordagem baseada em desafios [58, 60, 61], se mostraram como um fator de influência positiva sobre o aprendizado e desenvolvimento de aplicativos para dispositivos móveis.

A construção de um aplicativo é um ciclo iterativo no qual primeiramente é realizado um *design* do aplicativo, então ele é codificado e testado para validar os requisitos de forma a prover *feedback* para o time, caso o *feedback* seja positivo pode ser realizada a distribuição de uma release, caso negativo o produto incremental pode ser ajustado e liberado em uma próxima *release* [50]. Neste sentido, o uso de desenvolvimento ágil [2, 50, 96, 105, 106, 124], através do método que une a abordagem baseada em desafios com desenvolvimento ágil através de práticas aplicadas pelas equipes, se mostrou como fator de influência positiva e sucesso durante o aprendizado e desenvolvimento dos projetos.

Tabela 8.1: Análise de efeitos do uso do método no desenvolvimento de aplicativos

Efeito	Positivo	Inconclusivo
Pedagogia ativa	✓	-
Desenvolvimento ágil	✓	-
Atitude	✓	-
Geração de código	-	✓
Colaboração	✓	-
Confiança	✓	-
Paradigma de desenvolvimento	✓	-
Plataforma de desenvolvimento	-	✓
Facilidade de aprendizado	✓	-
Motivação	✓	-
Desempenho	✓	-
Sessões práticas	✓	-
Protótipo	✓	-
Gerenciamento de risco	-	✓
Design simples	✓	-
Pequenas *releases*	✓	-
Test Driven Development	-	✓
Utilidade	✓	-
Whole team	✓	-
Carga de trabalho	-	✓

Através dos estudos realizados nesta pesquisa foi possível encontrar diferentes exemplos de participantes que demonstraram atitude e comprometimento total com o processo de aprendizagem e com o desenvolvimento de seus projetos, o que vai de encontro a literatura onde Ahmad e Gestwicki [4] relataram atitude como um fator positivamente influenciado pelo aprendizado de desenvolvimento de aplicativos para dispositivos móveis.

O fator geração de código como um influência no desenvolvimento de aplicativos resultou em inconclusão devido ao fato do contexto dos aplicativos ser desenvolvimento nativo nos estudos realizados. Entretanto encontraram-se indicativos que o uso do StoryBoard que facilita a movimentação de componentes visuais e uma pequena geração de código como fatores que influenciam e facilitam o aprendizado de desenvolvimento iOS especificamente. Neste sentido, seis estudos [27, 28, 82, 109, 126, 130] relacionaram a geração de código através de abordagens multiplataforma como uma influência positiva no desenvolvimento de aplicativos, o que não conseguiu-se confirmar através desta pesquisa.

Colaboração também foi citada como um fator que influencia positivamente a motivação de estudantes, pois a colaboração entre as equipes oferece oportunidades para auto-crítica e auto-aperfeiçoamento, bem como de avaliação pelos pares [76]. Este fator de sucesso se confirmou durante a análise de recomendações e práticas que são importantes para a aplicação do método.

O nível de confiança dos participantes em relação ao seu aprendizado foi demonstrado neste estudo através de diferentes relatos e também confirmado através da auto-avaliação dos alunos quanto ao aumento do seu conhecimento em desenvolvimento de aplicativos. Dois estudos [6, 7] relatam confiança como um fator positivamente influenciado pelo aprendizado de desenvolvimento de aplicativos para dispositivos móveis.

Em 2012, Huy e Do vanThanh [55] realizaram uma pesquisa para oferecer *guidelines* sobre quais paradigmas de desenvolvimento são mais aplicáveis para desenvolvimento de aplicativos para dispositivos móveis, indicando que a escolha final de um paradigma de desenvolvimento de aplicativos móveis depende das preferências de contexto de aplicativos e desenvolvedores. Esta afirmação se confirmou durante este estudo, pois os ambientes de desenvolvimento estudados influenciaram a escolha do paradigma de desenvolvimento nativo para a elaboração dos projetos.

Alguns estudos relataram que as plataformas de desenvolvimento ajudam a produzir uma influência positiva no desenvolvimento de aplicativos para dispositivos móveis [44, 86, 87, 111]. Esta influência foi analisada através do estudo realizado, e também diversos relatos de comparação entre plataformas, dependendo da experiência anterior dos participantes. Entretanto esta influência esta classificada como inconclusiva devido ao fato de que os ambientes de desenvolvimento estudados focam em uma plataforma especifica (iOS).

Abadi *et al.* [1] estudaram o uso de ferramentas que facilitam geração de código, reportando que este tipo de abordagem facilita o aprendizado melhorando o trabalho em ambientes de desenvolvimento de aplicativos para dispositivos móveis. Da mesma forma que relatado nas plataformas de desenvolvimento, detectamos que o aprendizado é influenciado pelas ferramentas que se utilizam nos ambientes de desenvolvimento, comunidade e documentação disponíveis. Entretanto o que faz a diferença no processo de aprendizado é o aprendiz.

O impacto positivo na motivação dos estudantes através do aprendizado e desenvolvimento de projetos de aplicativos foi mencionado em diferentes estudos [4, 15, 76, 106, 118, 129]. Esse fator de sucesso é gerado através do engajamento dos estudantes que é uma das bases do método proposto, e foi relatado em diferentes situações como influência através do engajamento dos participantes em resolver problemas do mundo real, sendo influenciado positivamente pelo processo de ensino e aprendizado de desenvolvimento de aplicativos.

Foram encontrados indicativos que o desempenho dos participantes foi impactado positivamente pelo aprendizado de desenvolvimento, tanto pelo uso das práticas da abordagem em desafios como também pelo uso das práticas de desenvolvimento ágil, as quais compõe o método proposto. Três estudos [4,70,71] apresentam resultados sobre a influência positiva do aprendizado de desenvolvimento de aplicativos para dispositivos móveis no desempenho dos estudantes.

Ambos estudo de campo e estudo de caso focaram mais no desenvolvimento dos projetos de forma prática, embora que os ambientes estudados também proporcionem sessões práticas sobre conteúdos específicos, mostrando que este fator tem um efeito positivo em ambientes de ensino e desenvolvimento de aplicativos. Além disso, cinco estudos [7, 93, 112, 116, 120] apresentam sessões práticas como influência positiva no ensino e aprendizado de desenvolvimento de aplicativos para dispositivos móveis.

Quanto ao uso de protótipos, Sa e Carriço [24] apresentaram um estudo sobre os conceitos de usuário com o foco em metodologias de *design* onde o engajamento da equipe com protótipos nos estágios iniciais dos projetos é um fator que contribui para o desenvolvimento de aplicativos para dispositivos móveis. O uso de protótipos foi declarado pelos participantes desta pesquisa como um

fator relevante e com influência positiva, tanto no estudo de campo, como também no estudo de caso realizado em diferentes regiões do brasil, o que confirma protótipos como fator de sucesso.

Kakkar *et al.* [62] estudaram os desafios no desenvolvimento tradicional de aplicativos e apresentaram diferentes aspectos sobre gerenciamento de risco em ambientes de desenvolvimento de aplicativos para dispositivos móveis, onde o gerenciamento de risco é apontado como um fator importante para projetos de desenvolvimento de aplicativos. Entretanto, não foram encontrados indicativos específicos sobre gerencia de risco nos ambientes de desenvolvimento. Isso pode se dever ao fato dos ciclos de desenvolvimento serem ágeis e o gerenciamento de risco já estar implícito durante as diferentes práticas e cerimônias do processo ágil, de acordo com Schwaber [107] nas cerimônias de revisão de Sprint os riscos são revisados e respostas apropriadas são definidas.

Durante esta pesquisa foram encontrados indicativos sobre práticas de *design* em diferentes equipes, o *design* foi identificado como fator de extrema importância no universo de desenvolvimento de aplicativos quando comparado com o desenvolvimento tradicional. Isso se deve basicamente ao fato de que ao desenvolver aplicativos para dispositivos (diferentes tamanhos e menores telas) é necessário repensar a usabilidade e experiência do usuário, o que exige um processo de *design* mais apurado. Isso confirma os resultados apresentados por Pedersen *et al.* [94], no qual é descrito que o *design* simples tem uma influencia e impacto positivo em projetos de desenvolvimento de aplicativos para dispositivos móveis.

Pequenas *releases* ajudam a equipe de desenvolvimento a entregar com frequência versões iterativas e incrementais de software. Isto ajuda a manter o cliente e os desenvolvedores trabalhando juntos de forma a compartilhar e resolver problemas encontrados durante o desenvolvimento [94]. Este fator de sucesso se confirmou tanto no estudo de campo, como também no estudo de caso em diferentes regiões do pais. A utilização de desenvolvimento ágil e geração de pequenas *releases* é algo determinante no universo de desenvolvimento de aplicativos.

Test Driven Development (TDD) é uma abordagem de desenvolvimento de software, na qual unidades de *software* são desenvolvidas em pequenas partes. Uma pequena parte do código contribui para verificar os requisitos de funcionalidades implementada, gerando um impacto positivo em projetos de desenvolvimento de aplicativos [94]. Alguns poucos relatos sobre o uso de TDD foram encontrados nesta pesquisa durante estudo de caso, esta técnica aparentemente está sendo pouco difundida e aplicada entre as equipes, o que nos leva a classificar este fator como inconclusivo.

Liu [67] relatou que a utilidade do conteúdo a ser aprendido como fator que influencia o engajamento dos estudantes, onde a utilidade da prática para os estudantes na resolução de problemas ainda é a forma preferida de cursos de programação. Assim como também Sykes [118] relatou utilidade como um fator de impacto positivo em ambientes de ensino e aprendizado de desenvolvimento de aplicativos para dispositivos móveis. A utilidade é uma das premissas do uso de abordagem baseada em desafios, partindo do principio que os aprendizes devem trabalhar em uma idéia que os engage e os motive a resolver um problema real, o que confirma utilidade como um fator de sucesso em ambientes de ensino e desenvolvimento de aplicativos.

Whole team é uma estratégia de projeto na qual a equipe compartilha entre si os objetivos de projeto e a responsabilidade para atingi-los. Pedersen *et al.* [94] descreve esta estratégia como promissora em ambientes de desenvolvimento de aplicativos para dispositivos móveis. Devido ao fato do método ser uma junção entre abordagem de desafios e práticas ágeis, esta premissa de equipe única compartilhando a responsabilidade de atingir os objetivos estabelecidos pelo próprio time, é verdadeira. Neste sentido confirmamos isto como fator de sucesso neste tipo de ambiente.

Um estudo relatou a influência negativa da carga de trabalho durante o aprendizado de desenvolvimento de aplicativos. De acordo com Sung e Samuel [117] uma carga de trabalho muito pesada dificulta que os estudantes tenham tempo para entender e apreciar detalhes importantes no desenvolvimento de aplicativos. Neste sentido, esta pesquisa não monitorou e mediu a carga de trabalho dos estudantes, entretanto pode-se perceber no comportamento de diferentes equipes que os participantes trabalharam mais do que em disciplinas comuns, o que gera um indicativo que durante o desenvolvimento dos projetos ocorreu uma carga de trabalho acima da média, entretanto não impactou motivação, aprendizado e entregas. O que nos faz classificar este fator de sucesso como inconclusivo.

8.4 Considerações Finais do Método

O processo inicial de construção do método foi iniciado a partir da base teórica e apresentado no Capítulo 5, e a partir da proposta inicial foram realizados dois estudos. O primeiro foi um estudo de campo onde foi possível analisar de forma qualitativa os fatores que o fazem ser um método viável e também entender as vantagens e desvantagens no seu uso, além de coletar um conjunto de lições aprendidas. O segundo foi um estudo de caso realizado em quatro universidades distintas em diferentes regiões do Brasil permitindo identificar oportunidades de melhoria, melhores práticas e recomendações, gerando o método final apresentado neste capítulo.

As oportunidades de melhoria, melhores práticas e recomendações que compõe o método final foram descritas no decorrer deste capítulo. Neste sentido o método final tem cinco fases principais: Visão, Pesquisa, Solução, Desenvolvimento e Avaliação. Cada uma destas fases possui sua estrutura de processo e seus artefatos, fazendo com que os resultados desta pesquisa possam ser utilizados em estudos futuros e também avaliados em outros tipos de ambientes de desenvolvimento, pois esta pesquisa aponta indicativos que este método pode ser aplicado em outros tipos de ambiente, entretanto isso exige estudos adicionais para ser confirmado. Outro ponto importante em relação a abordagem do método final é a avaliação do conhecimento dos participantes envolvidos nos estudos.

De acordo com Hoffmann [53] as novas concepções de aprendizagem propõem fundamentalmente situações de busca contínua de novos conhecimentos, questionamento e crítica sobre as ideias em discussão e mobilização dos conhecimentos em variadas situações-problema. Estas concepções foram encontradas nos resultados dos estudos realizados nesta pesquisa e vão de encontro ao método apresentado neste capítulo. Dessa forma a visão do educador/avaliador ultrapassa a concepção de um observador que acompanha o processo e verifica se foram alcançados resultados

esperados, passando a ser alguém que provoca, questiona, exigindo melhores soluções [53]. *"[...] os melhores instrumentos de avaliação são todas as tarefas e registros feitos pelo professor que o auxiliam a resgatar uma memória significativa do processo, permitindo uma análise abrangente do desenvolvimento do aluno"* [53]. A aprendizagem dos estudantes é o maior objetivo, e portanto, ela é acompanhada e estimulada, por meio de situações que os desafiam.

Relatos sobre fatores com influência em ambientes de ensino e desenvolvimento de desenvolvimento de aplicativos foram encontrados nos estudos realizados nesta pesquisa. Dos 20 fatores encontrados na literatura, os estudos classificaram 15 fatores com influência positiva, restando cinco que foram classificados como inconclusivos, conforme descrito na Seção 8.3.

Dessa forma, este capítulo apresentou o método final, suas fases e processos envolvidos para a sua utilização baseado nas lições aprendidas, melhores práticas, oportunidades de melhoria e recomendações encontradas durante os estudos qualitativos exploratórios descritos nos capítulos anteriores (Capítulo 6 e Capítulo 7). No próximo capítulo serão apresentadas considerações finais e contribuições deste trabalho, como também as perspectivas de trabalhos futuros relacionados com esta pesquisa.

9. CONSIDERAÇÕES FINAIS

O crescimento no número de projetos de desenvolvimento de aplicativos é uma realidade, devido ao aumento na demanda por mobilidade e conectividade. Atualmente diferentes instituições tem buscado formas inovadoras de capacitar alunos e profissionais para esta crescente demanda de mercado, visando gerar mão-de-obra qualificada e rica em experiências com desenvolvimento de software, o que também é um desafio. Neste sentido, o uso de uma abordagem baseada em desafios proporciona uma oportunidade como apoio a esta demanda.

O objetivo geral desta tese, conforme apresentado na Subseção 1.1.1 desta pesquisa, foi alcançado com a proposta do método de abordagem baseada em desafios e práticas ágeis, contribuindo com a literatura da área de ensino e desenvolvimento de *software* ao fornecer um método aplicado em ambientes de ensino e desenvolvimento de aplicativos. As motivações para o desenvolvimento deste método foram a oportunidade identificada a partir do crescimento da prática do desenvolvimento de aplicativos para dispositivos móveis e os constantes desafios de instituições em realizar programas efetivos de treinamento neste contexto. A partir da revisão teórica realizada, esta é uma abordagem pioneira neste cenário, agregando contribuições tanto para a teoria quanto para a prática.

O objetivo específico de explorar a base teórica foi atingido e apresentado nos Capítulos 2 e 4. O objetivo específico de explorar os ambientes de ensino e desenvolvimento e identificar vantagens, desvantagens e lições aprendidas foi atingido e apresentado no Capítulo 6. O objetivo específico de identificar características e recomendações para o uso da abordagem baseada em desafios para ensino e desenvolvimento de aplicativos para dispositivos móveis foi atingido e apresentado nos Capítulos 5, 6 e 7. No Capítulo 5 foi apresentada a proposta inicial do método através de um conjunto de características da abordagem baseada em desafios e práticas ágeis. No Capítulo 6 foram apresentados resultados do primeiro estudo desenvolvido, onde se buscou identificar os desafios associados no uso do método, como também suas vantagens e desvantagens e viabilidade, proporcionando um conjunto de lições aprendidas de forma a identificar os fatores de sucesso. No Capítulo 7 foram apresentados resultados do segundo estudo desenvolvido, gerando um conjunto de melhorias para o método, melhores práticas e recomendações para seu uso. E por fim, o Capítulo 8 apresenta a evolução do método e a visão de cada uma de suas fases, obtida através do aprendizado e recomendações resultantes de ambos Estudo de campo e Estudo de caso.

9.1 Contribuições desta Tese

Esta pesquisa contribuiu para aprofundar o estudo na área de abordagens não tradicionais de ensino e desenvolvimento de projetos, com foco específico na abordagem baseada em desafios e práticas ágeis aplicado ao desenvolvimento de aplicativos para dispositivos móveis. Durante o desenvolvimento desta tese foi possível aprofundar o entendimento nas principais abordagens não tradicionais de ensino e principais práticas de desenvolvimento ágil, caracterizando a proposta inicial do método

e suas práticas. Como resultado, gerou-se um conjunto de oportunidades de melhoria, melhores práticas e recomendações de forma a facilitar o futuro uso deste método em outros ambientes de desenvolvimento.

Do ponto de vista de contribuição teórica, esta pesquisa contribui para a área de Engenharia de Software no sentido de identificar as implicações técnicas em ambientes de ensino e desenvolvimento de aplicativos. Além disso, contribui no sentido de avançar os estudos atualmente existentes e prover um método de apoio ao ensino e desenvolvimento de aplicativos fundamentado em estudos de base qualitativa e exploratória. Esta tese também gerou publicações de artigos científicos em conferências internacionais qualificadas.

Do ponto de vista prático, o método propõe ajudar instituições que buscam modelos diferenciados e inovadores para o ensino e desenvolvimento de projetos. Este método tem sido utilizado em diferentes estados do Brasil e possivelmente será utilizado em breve em outros países. Para o pesquisador, esta tese contribuiu principalmente no aprendizado e aperfeiçoamento acadêmico, através de um rigoroso estudo do estado da arte e da utilização de um método de pesquisa através de estudos que seguiram protocolos de pesquisa. Além disso, esta pesquisa também contribuiu em proporcionar um engajamento entre o pesquisador e o meio acadêmico.

9.2 Limitações da Pesquisa

As principais ameaças a validade desta pesquisa estão relacionadas ao processo metodológico. A seleção e aplicação dos métodos foram conduzidos pelo pesquisador que obteve um entendimento sobre recomendações da literatura, definindo então a estratégia para conduzir esta pesquisa. Para garantir o rigor do processo metodológico esta pesquisa adota como guia recomendações de diferentes estudos. Durante todo o planejamento e execução, outro pesquisador revisou o processo. Esta tese utilizou protocolos através dos diferentes estudos para guiar a execução desta pesquisa buscando assim reduzir estas limitações.

O estudo de campo adotado nesta pesquisa foi aplicado em apenas uma universidade. Entretanto, o estudo de caso adotado nesta pesquisa foi realizado em quatro universidades de regiões distintas do país. A amostra da população analisada nos estudos faz parte de um programa de pesquisa e desenvolvimento, o qual é liderado por uma das maiores empresas de tecnologia do mundo. Foi adotado um mapeamento sistemático da literatura para obter um melhor entendimento sobre o estado da arte na área relacionada com o problema desta pesquisa. Como parte de qualquer MSL, existem ameaças a validade do processo tais como seleção de estudos, extração de dados, classificação incorreta e também um viés potencial do pesquisador. Para reduzir o viés do autor, dois pesquisadores revisaram a interpretação dos dados coletados. Nos casos de discordância entre os revisores, as diferenças eram discutidas até se obter um consenso.

Uma outra ameaça em potencial seria o viés do pesquisador sobre a pesquisa. De forma a minimizar os efeitos de viés, resultados preliminares foram publicados em conferências da área. A proposta inicial foi apresentada na *20th Annual Conference on Innovation and Technology in*

Computer Science Education (ITiCSE 2015). Além disso resultados iniciais do estudo de campo foram apresentados na *18th International Conference on Enterprise Information Systems (ICEIS 2016)*.

Portanto, o método é baseado na teoria e em estudos qualitativos e exploratórios. Os dados obtidos da teoria podem gerar equívocos de interpretação. Em relação aos estudos qualitativos, todos os métodos aplicados tem suas próprias particularidades o que pode interferir diretamente ou indiretamente nos resultados obtidos durante esta pesquisa.

9.3 Trabalhos futuros

Identifica-se oportunidades de continuação desta pesquisa através de diferentes estudos para entender e avaliar o uso deste método não limitado ao desenvolvimento de aplicativos, de forma a aplicá-lo em diferentes ambientes de desenvolvimento como Web, Desktop e outros. Outro fator a ser explorado é realizar um estudo para entender a resistência de indivíduos no uso das práticas de teste, os indicativos mostram que isso é algo que não está relacionado com o método em si, mas com o fato de que os estudantes que estão se formando como desenvolvedores demonstrarem pouco interesse em atividades relacionadas com testes de aplicativos.

Existe a possibilidade de realizar um estudo da aplicação deste método em um ambiente de trabalho de conclusão de curso, de forma a avaliar o comportamento do método e suas práticas em um contexto diferente dos avaliados neste estudo. Este método poderá ser extendido para criar um modelo de avaliação e análise pedagógica no apoio ao ensino e aprendizado em diferentes ambientes de desenvolvimento, de forma a não só enriquecer a aprendizagem dos indivíduos mas também ajudar diferentes instituições a monitorar e medir a aplicação do método apresentado nesta pesquisa.

Uma outra contribuição que poderia ocorrer é a utilização do método em ambientes de ensino e aprendizado de desenvolvimento de aplicativos em outros países. Baseado nos resultados obtidos através deste estudo, a empresa parceira desta pesquisa tem interesse em aplicar o método em iniciativas semelhantes que serão implementadas em outros países.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] A. Abadi, Y. Dubinsky, A. Kirshin, Y. Mesika, I. Ben-Harrush, and U. Hadad. Nitrogen: Rapid development of mobile applications. In *SPLASH 2013 - Proceedings of the 2013 Companion Publication for Conference on Systems, Programming, and Applications: Software for Humanity*, pages 15–16, Indianapolis, IN, USA, 2013.

[2] P. Abrahamsson, A. Hanhineva, H. Hulkko, T. Ihme, J. Jäälinoja, M. Korkala, J. Koskela, P. Kyllönen, and O. Salo. Mobile-D: An Agile Approach for Mobile Application Development. In *Companion to the 19th Annual ACM SIGPLAN Conference on Object-oriented Programming Systems, Languages, and Applications*, OOPSLA '04, pages 174–175, Vancouver, BC, Canada, 2004.

[3] O. M. T. I. Agency. Mobile App Backlog Is Directly Damaging Revenue in the Enterprise. Technical report, OutSystems, 2014.

[4] K. Ahmad and P. Gestwicki. Studio-based learning and app inventor for android in an introductory CS course for non-majors. In *Proceedings of the 44th ACM Technical Symposium on Computer Science Education (SIGCSE 2013)*, pages 287–292, Denver, Colorado, USA, 2013.

[5] D. R. Albert Endres. *A Handbook of Software and Systems Engineering. Empirical Observations Laws and Theories*. The Fraunhofer IESE Series on Software Engineering. PEARSON Addison Wesley., 9th edition, 2003.

[6] M. Alonso Jr., S. Hug, and H. Thiry. Work in progress: Recruiting computing students through in-command CS-0: An introduction to computing through mobile application development. In *Proceedings of ASEE Annual Conference and Exposition*, page 6p, Vancouver, BC, Canada, 2011.

[7] P. Alston. Teaching Mobile Web Application Development: Challenges Faced and Lessons Learned. In *Proceedings of the 13th Annual Conference on Information Technology Education*, SIGITE '12, pages 239–244, Calgary, Alberta, Canada, 2012. ACM.

[8] L. d. G. C. Anastasiou and S. G. Pimenta. *Docência no ensino superior*. Cortez, São Paulo, Brazil, 4 ed. edition, 2010.

[9] A. Avritzer, F. Bronsard, and G. Matos. *Improving Global Development Using Agile*. Springer Berlin Heidelberg, 2010.

[10] E. Babbie. *Métodos de pesquisas de survey*. UFMG, Belo Horizonte, 2005.

[11] F. Balagtas-Fernandez, J. Forrai, and H. Hussmann. Evaluation of User Interface Design and Input Methods for Applications on Mobile Touch Screen Devices. In *Proceedings of the 12th IFIP TC 13 International Conference on Human-Computer Interaction: Part I*, INTERACT '09, pages 243–246, Berlin, Heidelberg, 2009. Springer-Verlag.

[12] K. Beck. *Programação extrema (XP) explicada : acolha as mudanças*. Bookman, Brasil, 2004.

[13] H. Beckman, N. Coulter, S. Khajenoori, and N. Mead. Collaborations: closing the industry-academia gap. *Software, IEEE*, 14(6):49–57, Nov 1997.

[14] B. Bergvall-Kareborn and D. Howcroft. Persistent problems and practices in information systems development: a study of mobile applications development and distribution. *Information Systems Journal*, 24(5):425–444, 2014.

[15] B. Burd, J. a. P. Barros, C. Johnson, S. Kurkovsky, A. Rosenbloom, and N. Tillman. Educating for Mobile Computing: Addressing the New Challenges. In *Proceedings of the Final Reports on*

Innovation and Technology in Computer Science Education 2012 Working Groups, ITiCSE-WGR '12, pages 51–63, Haifa, Israel, 2012.

[16] L. Cao, K. Mohan, P. Xu, and B. Ramesh. A Framework for Adapting Agile Development Methodologies. *European Journal of Information Systems*, 18(4):332–343, 2009.

[17] A. S. Carter and C. D. Hundhausen. A Review of Studio-based Learning in Computer Science. *Journal of Computer Science and Colleges*, 27(1):105–111, Oct. 2011.

[18] L. C. Cheng. The mobile app usability inspection (maui) framework as a guide for minimal viable product (mvp) testing in lean development cycle. In *Proceedings of the The 2th International Conference in HCI and UX on Indonesia 2016*, CHIuXiD '16, pages 1–11, Jakarta, Indonesia, 2016. ACM.

[19] K. Conboy and B. Fitzgerald. Method and Developer Characteristics for Effective Agile Method Tailoring: A Study of XP Expert Opinion. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 20(1):47–77, July 2010.

[20] L. Corral. Using Software Quality Standards to Assure the Quality of the Mobile Software Product. In *Proc. of the 3rd Annual Conference on Systems, Programming, and Applications: Software for Humanity (SPLASH'12)*, pages 37–40, Tucson, AZ, USA, 2012.

[21] L. Corral, A. Sillitti, G. Succi, A. Garibbo, and P. Ramella. Evolution of mobile software development from platform-specific to web-based multiplatform paradigm. In *Proceedings of the 10th SIGPLAN symposium on New ideas, new paradigms, and reflections on programming and software*, pages 181–183. ACM, 2011.

[22] J. W. Creswell. *Projeto de pesquisa : métodos qualitativo, quantitativo e misto*. Artmed, Porto Alegre, 3. ed. edition, 2010.

[23] I. Dalmasso, S. Datta, C. Bonnet, and N. Nikaein. Survey, comparison and evaluation of cross platform mobile application development tools. In *9th International Conference on Wireless Communications and Mobile Computing (IWCMC 2013)*, pages 323–328, Cagliari, Italy, July 2013.

[24] M. de Sá and L. Carriço. Low-fi Prototyping for Mobile Devices. In *CHI '06 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '06, pages 694–699, Montreal, Quebec, Canada, 2006.

[25] L. de Souza Mariz, A. França, and F. da Silva. An Empirical Study on the Relationship between the Use of Agile Practices and the Success of Software Projects that Use Scrum. In *Brazilian Symposium on Software Engineering (SBES)*, pages 110–117, Sept 2010.

[26] J. Dehlinger and J. Dixon. Mobile application software engineering: Challenges and research directions. In *Workshop on Mobile Software Engineering*, volume 2, pages 29–32, 2011.

[27] P. E. Dickson. Cabana: A Cross-platform Mobile Development System. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, SIGCSE '12, pages 529–534, Raleigh, North Carolina, USA, 2012. ACM.

[28] P. E. Dickson. Teaching Mobile Computing Using Cabana. *Journal of Computing Sciences in Colleges*, 27(6):128–134, June 2012.

[29] C.-K. Diep, Q.-N. Tran, and M.-T. Tran. Online Model-driven IDE to Design GUIs for Cross-platform Mobile Applications. In *Proceedings of the Fourth Symposium on Information and Communication Technology*, SoICT '13, pages 294–300, Danang, Vietnam, 2013.

[30] J. DiMarzio. *Android a Programmers Guide*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 2009.

[31] F. Dochy, M. Segers, P. Bossche, and K. Struyven. Students' Perceptions of a Problem-Based Learning Environment. *Learning Environments Research*, 8(1):41–66, 2005.

[32] S. Donizetti Zorzo, L. de Ponte, and D. Lucredio. Using Scrum to Teach Software Engineering: A Case Study. In *Frontiers in Education Conference, 2013 IEEE*, pages 455–461, Oklahoma City, Oklahoma, USA, Oct 2013.

[33] R. P. dos Santos, P. S. M. dos Santos, C. M. L. Werner, and G. H. Travassos. Utilizando experimentação para apoiar a pesquisa em educação em engenharia de software no brasil. *Fórum de Educação em Engenharia de Software*, page 55, 2008.

[34] T. Dybå and T. Dingsør. Empirical studies of agile software development: A systematic review. *Information and Software Technology*, 50(9:10):833 – 859, 2008.

[35] E. Fernandez and D. M. Williamson. Using Project-based Learning to Teach Object Oriented Application Development. In *Proceedings of the 4th Conference on Information Technology Curriculum*, CITC4 '03, pages 37–40, Lafayette, Indiana, USA, 2003.

[36] M. Fetaji and B. Fetaji. Analyses of mobile learning software solution in education using the task based learning approach. In *Proceedings of the ITI 2009 31st International Conference on Information Technology Interfaces, ITI '09.*, pages 373–378, June 2009.

[37] H. K. Flora and S. V. Chande. A Review and Analysis on Mobile Application Development Processes using Agile Methodologies. *International Journal of Research in Computer Science*, 3(4):9–18, 2013.

[38] R. Francese, C. Gravino, M. Risi, G. Scanniello, and G. Tortora. Using Project-Based-Learning in a mobile application development course-An experience report. *Journal of Visual Languages and Computing*, 31, Part B:196 – 205, 2015. Special Issue on {DMS2015}.

[39] P. Gestwicki and K. Ahmad. App Inventor for Android with Studio-based Learning. *Journal of Computing Sciences in Colleges*, 27(1):55–63, Oct. 2011.

[40] G. Gibbs. *Análise de dados qualitativos*. Artmed, Porto Alegre, 2009.

[41] P. Gokhale and S. Singh. Multi-platform strategies, approaches and challenges for developing mobile applications. In *Circuits, Systems, Communication and Information Technology Applications (CSCITA), 2014 International Conference on*, pages 289–293, April 2014.

[42] A. J. Gordon. Concepts for Mobile Programming. In *Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education*, ITiCSE '13, pages 58–63, Canterbury, England, UK, 2013.

[43] D. Granlund, D. Johansson, K. Andersson, and R. Brännström. A Case Study of Application Development for Mobile and Location-Based Services. In *Proceedings of International Conference on Information Integration and Web-based Applications and Services*, IIWAS '13, pages 658–662, Vienna, Austria, 2013.

[44] T.-M. Grønli, J. Hansen, and G. Ghinea. Android vs Windows Mobile vs Java ME: A Comparative Study of Mobile Development Environments. In *Proceedings of the 3rd International Conference on Pervasive Technologies Related to Assistive Environments*, PETRA '10, pages 45:1–45:8, Samos, Greece, 2010.

[45] T.-M. Grønli, J. Hansen, G. Ghinea, and M. Younas. Mobile application platform heterogeneity: Android vs windows phone vs ios vs firefox os. In *Advanced Information Networking and Applications (AINA), 2014 IEEE 28th International Conference on*, pages 635–641, May 2014.

[46] R. Hall. *Applied Social Research : Planning, Designing and Conducting Real-World Research*. palgave macmillan, 2008.

[47] K. Haller. Mobile Testing. *SIGSOFT Software Engineering Notes*, 38(6):1–8, Nov. 2013.

[48] A. Hammershoj, A. Sapuppo, and R. Tadayoni. Challenges for mobile application development. In *14th International Conference on Intelligence in Next Generation Networks (ICIN)*, pages 1–8, Berlin, Germany, Oct 2010.

[49] E. Hasnain. An Overview of Published Agile Studies: A Systematic Literature Review. In *Proceedings of the 2010 National Software Engineering Conference*, NSEC '10, pages 3:1–3:6, Rawalpindi, Pakistan, 2010.

[50] A. Heredia, R. Esteban-Santiago, J. Garcia-Guzman, and A. de Amescua. Mass-Market Application Development Using Agile Techniques: How Agile Are We Really? In F. McCaffery, R. O'Connor, and R. Messnarz, editors, *Systems, Software and Services Process Improvement*, volume 364 of *Communications in Computer and Information Science*, pages 259–269. Springer Berlin Heidelberg, 2013.

[51] R. Hernández Sampieri, C. Fernández Collado, and M. d. P. B. Lucio. *Metodologia de Pesquisa*. McGraw-Hill, São Paulo, 5 ed. edition, 2013.

[52] J. Highsmith and A. Cockburn. Agile software development: the business of innovation. *Computer*, 34(9):120–127, Sep 2001.

[53] J. Hoffmann. *Avaliar para promover : as setas do caminho*. Mediação, Porto Alegre, 14 edition, 2011.

[54] S. Huang and D. Distante. On Practice-Oriented Software Engineering Education. In *19th Conference on Software Engineering Education and Training Workshops, 2006. CSEETW'06.*, pages 15–15. IEEE, 2006.

[55] N. P. Huy and D. vanThanh. Evaluation of Mobile App Paradigms. In *Proceedings of the 10th International Conference on Advances in Mobile Computing and Multimedia*, MoMM '12, pages 25–30, Bali, Indonesia, 2012.

[56] A. Ikonen, A. Piironen, K. Saurén, and P. Lankinen. CDIO Concept in Challenge Based Learning. In *Proceedings of the 2009 Workshop on Embedded Systems Education*, WESE '09, pages 27–32, Grenoble, France, 2009.

[57] A. Inc. Apple Classrooms of Tomorrow - Today. Technical report, Apple Inc., Cupertino, CA, USA, 2008.

[58] A. Inc. Challenge Based Learning: A Classroom Guide. Technical report, Apple Inc., Cupertino, CA, USA, 2009.

[59] D. Inel and A. G. Balm. Concept Cartoons Assisted Problem based Learning Method in Science and Technology Teaching and Students Views. *Procedia - Social and Behavioral Sciences*, 93(0):376 – 380, 2013. 3rd World Conference on Learning,Teaching and Educational Leadership.

[60] L. Johnson and S. Adams. Challenge Based Learning: The Report from the Implementation Project. Technical report, The New Media Consortium, Austin, TX, USA, 2011.

[61] L. F. Johnson, R. S. Smith, J. Smythe, and R. K. Varon. Challenge-Based Learning: An Approach for Our Time. Technical report, The New Media Consortium, Austin, TX, USA, 2009.

[62] K. Kakkar, R. Shah, and M. Kakkar. Risk analysis in mobile application development. In *Confluence 2013: The Next Generation Information Technology Summit (4th International Conference)*, pages 429–434, Sept 2013.

[63] S. B. Kaleel and S. Harishankar. Applying Agile Methodology in Mobile Software Engineering: Android Application Development and its Challenges. Technical report, Department of Computer Science, Ryerson University, 2013.

[64] B. Kitchenham. The current state of evidence-based software engineering. In *International Conference on Evaluation and Assessment in Software Engineering.*, 2007.

[65] G. A. Lewis, N. Nagappan, J. Gray, D. Rosenblum, H. Muccini, and E. Shihab. Report of the 2013 ICSE 1st International Workshop on Engineering Mobile-enabled Systems (MOBS 2013): 12. *SIGSOFT Software Engineering Notes*, 38(5):55–58, Aug. 2013.

[66] B. Liu, W. Hsu, and Y. Ma. Integrating Classification and Association Rule Mining. In *Fourth International Conference on Knowledge Discovery and Data Mining*, pages 80–86, New York, NY, USA, 1998.

[67] X. Liu. Test-Run of the "App-Driven Approach"in Teaching A Mobile Programming Course. In *Proceedings of the Western Canadian Conference on Computing Education*, WCCCE '14, pages 9:1–9:4, Richmond, BC, Canada, 2014.

[68] C. Luis and A. Marrero. Real object mapping technologies applied to marine engineering learning process within a CBL methodology. *Procedia Computer Science*, 25:406–410, 2013.

[69] F. Macias, M. Holcombe, and M. Gheorghe. A Formal Experiment Comparing Extreme Programming with Traditional Software Construction. In *Proceedings of the Fourth Mexican International Conference on Computer Science*, pages 73–80, Sept 2003.

[70] Q. H. Mahmoud, T. Ngo, R. Niazi, P. Popowicz, R. Sydoryshyn, M. Wilks, and D. Dietz. An Academic Kit for Integrating Mobile Devices into the CS Curriculum. In *Proc. of the 14th Annual ACM SIGCSE Conference on Innovation and Technology in Computer Science Education (ITiCSE'09)*, pages 40–44, Paris, France, 2009.

[71] Q. H. Mahmoud and P. Popowicz. A mobile application development approach to teaching introductory programming. In *Frontiers in Education Conference (FIE)*, pages T4F–1 – T4F–6, Arlington, VA, USA, Oct 2010.

[72] C. Marcangelo and C. Gibbon. Problem Based Learning Evaluation Toolkit. Technical report, Higher Education Academy Health Sciences and Practice Subject Centre, Cumbria, UK, 2009.

[73] C. Marin, J. Hargis, and C. Cavanaugh. iPad learning ecosystem: Developing challenge-based learning using design thinking. *Turkish Online Journal of Distance Education*, 14(2):22–34, 2013.

[74] M. T. Masetto. *Competência Pedagógica do Professor Universitário*. Summus Editorial, São Paulo, 1 ed. edition, 2003.

[75] A. Massey, V. Ramesh, and V. Khatri. Design, development, and assessment of mobile applications: the case for problem-based learning. *IEEE Transactions on Education*, 49(2):183–192, May 2006.

[76] V. Matos and R. Grasser. Building Applications for the Android OS Mobile Platform: A Primer and Course Materials. *Journal of Computing Sciences in Colleges*, 26(1):23–29, Oct. 2010.

[77] B. J. McNely, P. Gestwicki, A. Burke, and B. Gelms. Articulating Everyday Actions: An Activity Theoretical Approach to Scrum. In *Proceedings of the 30th ACM International Conference on Design of Communication*, SIGDOC '12, pages 95–104, Seattle, Washington, USA, 2012.

[78] N. Mead, V. Viswanathan, and D. Padmanabhan. Incorporating Security Requirements Engineering into the Dynamic Systems Development Method. In *Computer Software and Applications, 2008. COMPSAC '08. 32nd Annual IEEE International*, pages 949–954, Turku, Finland, July 2008.

[79] B. Meyer. Software Engineering in the Academy. *Computer*, 34(5):28–35, May 2001.

[80] P. Middleton and D. Joyce. Lean Software Management: BBC Worldwide Case Study. *IEEE Transactions on Engineering Management*, 59(1):20–32, 2012.

[81] M. Miranda, R. Ferreira, C. R. B. de Souza, F. Figueira Filho, and L. Singer. An Exploratory Study of the Adoption of Mobile Development Platforms by Software Engineers. In *Proceedings of the 1st International Conference on Mobile Software Engineering and Systems*, MOBILESoft 2014, pages 50–53, New York, NY, USA, 2014. ACM.

[82] P. Miravet, I. Marín, F. Ortín, and A. Rionda. DIMAG: A Framework for Automatic Generation of Mobile Applications for Multiple Platforms. In *Proceedings of the 6th International Conference on Mobile Technology, Application and Systems*, Mobility '09, pages 23:1–23:8, Nice, France, 2009.

[83] J. K. Muppala. Teaching Embedded Software Concepts Using Android. In *Proceedings of the 6th Workshop on Embedded Systems Education*, WESE '11, pages 32–37, New York, NY, USA, 2011.

[84] L. Naismith, P. Lonsdale, G. Vavoula, and M. Sharples. Literature Review in Mobile Technologies and Learning. Technical report, University of Birmingham, 2013.

[85] M. Nauman and M. Uzair. SE and CS Collaboration: Training Students for Engineering Large, Complex Systems. In *20th Conference on Software Engineering Education and Training, 2007. CSEET'07.*, pages 167–174. IEEE, 2007.

[86] M. Nebeling, C. Zimmerli, and M. Norrie. Informing the Design of New Mobile Development Methods and Tools. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '13, pages 283–288, Paris, France, 2013.

[87] T. A. Nguyen, S. T. A. Rumee, C. Csallner, and N. Tillmann. An experiment in developing small mobile phone applications comparing on-phone to off-phone development. In *in Proceedings of the First International Workshop on User Evaluation for Sofware Engineering Researchers. IEEE Press*, pages 9–12, Piscataway, NJ, USA, 2012.

[88] C. O. Melo, V. Santos, E. Katayama, H. Corbucci, R. Prikladnicki, A. Goldman, and F. Kon. The evolution of agile software development in Brazil. *Journal of the Brazilian Computer Society*, 19(4):523–552, 2013.

[89] T. K. O'Mahony, N. J. Vye, J. D. Bransford, E. A. Sanders, R. Stevens, R. D. Stephens, M. C. Richey, K. Y. Lin, and M. K. Soleiman. A Comparison of Lecture-Based and Challenge-Based Learning in a Workplace Setting: Course Designs, Patterns of Interactivity, and Learning Outcomes. *Journal of the Learning Sciences*, 21(1):182–206, 2012.

[90] V. One. 9th Annual State of Agile Survey. Technical report, Version One, 2015.

[91] A. Oulasvirta, M. Wahlström, and K. Anders Ericsson. What Does It Mean to Be Good at Using a Mobile Device? An Investigation of Three Levels of Experience and Skill. *International Journal of Human-Computer Studies*, 69(3):155–169, Mar. 2011.

[92] V. P. Pauca and R. T. Guy. Mobile Apps for the Greater Good: A Socially Relevant Approach to Software Engineering. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, SIGCSE '12, pages 535–540, Raleigh, North Carolina, USA, 2012.

[93] B. R. Payne. Teaching Android and iOS Native Mobile App Development in a Single Semester Course. *Journal of Computing Sciences in Colleges*, 30(2):176–183, Dec. 2014.

[94] O. Pedersen, M. Kristiansen, M. Kammersgaard, and J. Hosbond. Mobile Systems Development: Exploring the Fit of XP. In *Proceedings of the 15th International Conference on Information Systems*

Development-New Methods and Practice for the Networked Society (ISD 2006), volume 1 of *Advances in Information Systems Development: New Methods and Practice for the Networked Society*, pages 215–224, Budapest, Hungary, 2007. Springer.

[95] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson. Systematic Mapping Studies in Software Engineering. In *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering*, EASE'08, pages 68–77, Swinton, UK, 2008. British Computer Society.

[96] V. Rahimian and R. Ramsin. Designing an agile methodology for mobile software development: A hybrid method engineering approach. In *Second International Conference on Research Challenges in Information Science. RCIS 2008*, pages 337–342, Marrakech, Morocco, June 2008.

[97] T. Reichlmayr. Working towards the student Scrum - Developing Agile Android applications. *ASEE Annual Conference and Exposition, Conference Proceedings*, 2011.

[98] A. Ribeiro and A. R. da Silva. Survey on cross-platforms and languages for mobile apps. In *Proceedings of the 2012 Eighth International Conference on the Quality of Information and Communications Technology*, QUATIC '12, pages 255–260, Lisbon, Portugal, 2012.

[99] G.-C. Roman, G. P. Picco, and A. L. Murphy. Software Engineering for Mobility: A Roadmap. In *Proceedings of the Conference on The Future of Software Engineering*, ICSE '00, pages 241–258, Limerick, Ireland, 2000.

[100] N. Salleh, E. Mendes, and J. Grundy. Empirical Studies of Pair Programming for CS/SE Teaching in Higher Education: A Systematic Literature Review. *IEEE Transactions on Software Engineering*, 37(4):509–525, July 2011.

[101] A. Santos, J. Kroll, A. Sales, P. Fernandes, and D. Wildt. Investigating the Adoption of Agile Practices in Mobile Application Development. In *Proceedings of the 18th International Conference on Enterprise Information Systems (ICEIS 2016)*, pages 490–497, Rome, Italy, April 2016.

[102] A. Santos, A. Sales, P. Fernandes, and M. Nichols. Combining Challenge-Based Learning and Scrum Framework for Mobile Application Development. In *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE'15)*, pages 189–194, Vilnius, Lithuania, July 2015.

[103] V. Santos, A. Goldman, and H. Roriz Filho. The Influence of Practices Adopted by Agile Coaching and Training to Foster Interaction and Knowledge Sharing in Organizational Practices. In *46th Hawaii International Conference on System Sciences (HICSS)*, pages 4852–4861, Jan 2013.

[104] S. Sarif and N. Shiratuddin. m^d-Matrix: An assistive learning tool in blended project-based learning for mobile development course. In *Proceedings of ASCILITE 2009 - The Australasian Society for Computers in Learning in Tertiary Education*, pages 923–927, Sydney, Australia, 2009.

[105] C. Scharff and R. Verma. Scrum to Support Mobile Application Development Projects in a Just-in-time Learning Context. *Proceedings - International Conference on Software Engineering*, pages 25–31, 2010.

[106] C. Scharff, A. Wasilewska, J. Wong, M. Bousso, I. Ndiaye, and C. Sarr. A model for teaching mobile application development for social changes: Implementation and lessons learned in senegal. In *International Multiconference on Computer Science and Information Technology, 2009. IMCSIT '09.*, pages 383–389, Wisla, Poland, Oct 2009.

[107] K. Schwaber. SCRUM Development Process. In *Proceedings of the 10th Annual ACM Conference on Object Oriented Programming Systems, Languages, and Applications (OOPSLA)*, pages 117–134, 1995.

[108] K. Schwaber. *Agile Project Management with Scrum*. Microsoft Press, USA, 1st edition, 2004.

[109] K. L. Sewell and J. Ringenberg. Accelerating K-12 Interest in Computer Science Using Mobile Application-based Curriculums. In *2012 ASEE Annual Conference*, page 13p, San Antonio, Texas, June 2012. ASEE Conferences.

[110] F. Shull, J. Carver, and G. H. Travassos. An empirical methodology for introducing software processes. In *Proceedings of the 8th European Software Engineering Conference Held Jointly with 9th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, ESEC/FSE-9, pages 288–296, Vienna, Austria, 2001.

[111] Simon and D. Cornforth. Teaching Mobile Apps for Windows Devices Using TouchDevelop. In *Proceedings of the Sixteenth Australasian Computing Education Conference - Volume 148*, ACE '14, pages 75–82, Darlinghurst, Australia, 2014. Australian Computer Society, Inc.

[112] G. W. Skelton, J. Jackson, and F. C. Dancer. Teaching Software Engineering Through the Use of Mobile Application Development. *Journal of Computing Sciences in Colleges*, 28(5):39–44, May 2013.

[113] J. Song, H. Choi, J. Baker, and A. Bhattacherjee. Mobile Application Development Platform Adoption: A Grounded Theory Investigation. In *19th Americas Conference on Information Systems, AMCIS*, Chicago, Illinois, USA, August 2013.

[114] S. Song. Lessons learned from Mobile Computing Application Development with Android. In *ASEE Annual Conference and Exposition, Conference Proceedings*, page 8p, San Antonio, TX, USA, 2012.

[115] J. Stapleton. DSDM: Dynamic Systems Development Method. In *Proceedings of Technology of Object-Oriented Languages and Systems*, pages 406–406, Nanjing, China, June 1999.

[116] C. Stringfellow and D. Mule. Smartphone Applications As Software Engineering Projects. *Journal of Computing Sciences in Colleges*, 28(4):27–34, Apr. 2013.

[117] K. Sung and A. Samuel. Mobile Application Development Classes for the Mobile Era. In *Proceedings of the 2014 Conference on Innovation Technology in Computer Science Education*, ITiCSE '14, pages 141–146, Uppsala, Sweden, 2014.

[118] E. Sykes. New Methods of Mobile Computing: From Smartphones to Smart Education. *TechTrends*, 58(3):26–37, 2014.

[119] W.-K. Tan and H.-H. Teo. Training Students to Be Innovative Information Systems Developers: Synergizing Project-based Learning with Problem-based Learning. In *Proceedings of the Special Interest Group on Management Information System's 47th Annual Conference on Computer Personnel Research*, SIGMIS CPR '09, pages 19–32, Limerick, Ireland, 2009.

[120] C.-C. Teng and R. Helps. Mobile Application Development: Essential New Directions for IT. In *Seventh International Conference on Information Technology: New Generations (ITNG)*, pages 471–475, Las Vegas, Nevada, USA, April 2010.

[121] J. W. Thomas. A review of research on project-based learning. Technical report, Autodesk Foundation, San Rafael, CA, 2000.

[122] K. Tracy. Mobile Application Development Experiences on Apple iOS and Android OS. *Potentials, IEEE*, 31(4):30–34, July 2012.

[123] B. Unhelkar and S. Murugesan. The enterprise mobile applications development framework. *IT Professional*, 12(3):33–39, May 2010.

[124] V. Uskov. Mobile software engineering in mobile computing curriculum. In *Interdisciplinary Engineering Design Education Conference (IEDEC)*, pages 93–99, Santa Clara, CA, USA, March 2013.

[125] B. Vogel. Towards Open Architecture System. In *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering*, ESEC/FSE 2013, pages 731–734, New York, NY, USA, 2013. ACM.

[126] A. Wagner, J. Gray, J. Corley, and D. Wolber. Using App Inventor in a K-12 Summer Camp. In *Proceedings of the 44th ACM Technical Symposium on Computer Science Education*, SIGCSE '13, pages 621–626, Denver, Colorado, USA, 2013.

[127] A. I. Wasserman. Software Engineering Issues for Mobile Application Development. In *Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research*, FoSER '10, pages 397–400, Santa Fe, New Mexico, USA, 2010.

[128] R. Wieringa, N. Maiden, N. Mead, and C. Rolland. Requirements Engineering Paper Classification and Evaluation Criteria: A Proposal and a Discussion. *Requirements Engeneering*, 11(1):102–107, Dec. 2005.

[129] D. Wolber. App Inventor and Real-world Motivation. In *Proceedings of the 42Nd ACM Technical Symposium on Computer Science Education*, SIGCSE '11, pages 601–606, Dallas, TX, USA, 2011.

[130] S. Xanthopoulos and S. Xinogalos. A Comparative Analysis of Cross-platform Development Approaches for Mobile Applications. In *Proceedings of the 6th Balkan Conference in Informatics*, BCI '13, pages 213–220, Thessaloniki, Greece, 2013.

[131] C. Xu. Classroom flipping as the basis of a teaching model for the course Mobile Application Development. *World Transactions on Engineering and Technology Education*, 11(4):537–540, 2013.

[132] R. K. Yin. *Estudo de Caso Planejamento e Métodos*. Bookman, Brasil, 5th edition, 2015.

A. COMPLEMENTO PROTOCOLO ESTUDO DE CAMPO

A. Reuniões para levantamento das questões	
Participantes:	Alan Santos, Afonso Sales, Paulo Fernandes
Data:	Julho de 2014
Local:	PUCRS
B. Reuniões para revisão do questionário	
Participantes:	Alan Santos, Afonso Sales, Josiane Kroll, Instrutores
Data:	Agosto de 2014
Local:	PUCRS
C. Autorização da empresa e universidade participantes	
Participantes:	Gerente Pedagógico, Gerente de Programas, Coordenador de Programa
Data:	Setembro de 2014
Local:	Porto Alegre & São Paulo, Brasil
D. Pré-teste	
Participantes:	Instrutures e 3 alunos
Data:	Outubro de 2014
Local:	Porto Alegre, Brasil
E. Aplicação dos questionários online	
Participantes:	***Universidade X***: alunos do programa de desenvolvimento
Data:	*Dezembro de 2014*
Local:	*Porto Alegre, Brasil*

Outros recursos utilizados

- Qualtrics (coleta de dados)
- Microsoft Excel (tabulação e análise de dados)

B. QUESTIONÁRIO ESTUDO DE CAMPO

Bloco 1 - Informações do Participante

1. Qual a sua Idade ?
2. Qual o seu curso ?
3. Qual semestre ?

Bloco 2 - Experiência Anterior

4. Quais treinamentos você teve anteriores ao curso ?
5. Quais linguagens de programação você conhecia anteriormente ao curso ?
6. Quais metodologias você conhecia antes do curso ?

Bloco 3 - Engenharia de Software

7. Porque desenvolver aplicativos para dispositivos móveis é diferente ?
8. Quais os desafios no desenvolvimento mobile ?
9. Comente a sua opinião sobre o uso de métodos ágeis no desenvolvimento de aplicativos ?
10. Quais ferramentas foram utilizadas durante o desenvolvimento dos projetos ?
11. Foram aplicadas práticas de engenharia de software ? Quais ?
12. Qual a importância do papel do Product Onwer dentro do projeto ?
13. Qual a importância do papel do Scrum master dentro do projeto ?
14. Quais as práticas ágeis utilizadas ?

Bloco 4 - Viabilidade e Aprendizado

15. Na sua opinião quais são os fatores que fazem deste método uma solução viável para um ambiente de aprendizado de desenvolvimento mobile?
16. Como este método pode ser comparado com as outras formas de aprendizado ? (aula expositiva, métodos tradicionais de ensino, etc)
17. Por favor cite vantagens ou desvantagens no uso deste método para desenvolvimento mobile.
18. Em uma nota de 0 a 10 (máximo), qual o seu conhecimento em desenvolvimento mobile ANTES do curso?
19. Em uma nota de 0 a 10 (máximo), qual o seu conhecimento em desenvolvimento mobile APÓS o curso?

C. COMPLEMENTO PROTOCOLO ESTUDO DE CASO

A. Reuniões para levantamento das questões e estruturação do guia para a entrevista	
Participantes:	Alan Santos, Afonso Sales, Paulo Fernandes
Data:	Fevereiro de 2015
Local:	PUCRS
B. Reuniões para revisão do guia para a entrevista	
Participantes:	Alan Santos, Afonso Sales, Josiane Kroll, Instrutores, Gerente Pedagógico
Data:	Março de 2015
Local:	PUCRS
C. Autorização da empresa e universidades participantes	
Participantes:	Gerente Pedagógico, Gerente de Programas, Coordenador Universidade 1, Coordenador Universidade 2, Coordenador Universidade 3, Coordenador Universidade 4
Data:	Junho de 2015
Local:	Porto Alegre, Manaus, São Paulo, Recife : Brazil
D. Pré-teste	
Participantes:	Instrutures e 4 alunos
Data:	Julho de 2015
Local:	Porto Alegre, Brasil

E. Entrevistas Universidade 1	
Participantes:	alunos do programa de desenvolvimento
Data:	*Setembro de 2015*
Local:	*Recife, Brasil*
F. Entrevistas Universidade 2	
Participantes:	alunos do programa de desenvolvimento
Data:	*Setembro de 2015*
Local:	*Manaus, Brasil*

G. Entrevistas Universidade 3	
Participantes:	alunos do programa de desenvolvimento
Data:	*Setembro de 2015*
Local:	*São Paulo, Brasil*
H. Entrevistas Universidade 4	
Participantes:	alunos do programa de desenvolvimento
Data:	*Setembro de 2015*
Local:	*Porto Alegre, Brasil*

Outros recursos utilizados

- Microsoft Excel (tabulação e análise de dados)
- Gravador Digital, Papel, Caneta, Sala de reunião

D. QUESTIONÁRIO ESTUDO DE CASO

Bloco 1: Informações Contextuais

1. Qual a sua idade?
2. A quanto tempo em anos você desenvolve software? Você tem alguma experiência prévia da indústria?
3. Qual o seu curso de graduação? Em que semestre você está?
4. Quais linguagens de programação você conhecia anteriormente?
5. Comente um pouco sobre sua experiência anterior com desenvolvimento de aplicativos para dispositivos móveis?
6. Você teve contato anterior com métodos de pedagogia ativa como Problem-Based Learning, Challenge-Based Learning ou outros?
7. Comente um pouco sobre a sua experiência com desenvolvimento ágil, esse é o seu primeiro contato?

Bloco 2 - Influência de métodos ágeis e engenharia de software

8. Como o desenvolvimento ágil influencia na implementação dos projetos?
9. Quais as práticas ágeis que você e o seu time adotam durante o projeto? (build, reuniões, etc)
10. Quais as práticas de engenharia de software adotadas durante os projetos? (Ex: modelagem, protótipo, teste, etc)
11. Quais as ferramentas utilizadas para desenvolver os projetos?
12. Você acredita que a plataforma de desenvolvimento (iOS, Android, Windows Phone, etc) pode influenciar o aprendizado de desenvolvimento mobile? Porquê?

Bloco 3 - Experiência com o método

13. Como é o processo desde a concepção da ideia até a definição do desafio?
14. Como é o processo de pesquisa (guiding questions e activities)?
15. Como é a transição do processo de pesquisa para o inicio do desenvolvimento?
16. Como as práticas ágeis são utilizadas no desenvolvimento?
17. Como é realizado o processo de evaluation (Sprint Review/Retrospectivas)? Como as reflections colaboram para o processo?
18. Como você vê o papel dos alunos e o papel dos instrutores neste método (postura, busca do conhecimento, etc) ?

Bloco 4 - Auto-avaliação

19. Em uma nota de 0 a 10 (máximo), qual o seu conhecimento em desenvolvimento mobile ANTES desta experiência?

20. Em uma nota de 0 a 10 (máximo), qual o seu conhecimento em desenvolvimento mobile APÓS esta experiência?

Bloco 5 - Recomendações

21. Se você pudesse voltar no tempo e arrumar o que não funcionou, o que você mudaria no método ou na sua forma de trabalhar ?

22. Pensando nos projetos que você realizou utilizando o método, cite:

a. Algo que você considera importante e utilizou

b. Algo que você considera importante mas não utilizou

c. Algo que você não considera importante mas utilizou

d. Algo que você não considera importante e não utilizou

O uso de aprendizagem baseada em desafios e desenvolvimento ágil em ambientes de desenvolvimento de aplicativos para dispositivos móveis

Faculdade Informática/PUCRS
Avenida Ipiranga, 6681 – 90619-900 Porto Alegre– RS
Tel: (51) 3320-3558

Termo de Consentimento Livre e Esclarecido

A PUCRS, através do **grupo de em processamento paralelo e distribuído** da Faculdade de Informática**,** agradece a sua atenção e a inestimável contribuição que prestarão para o auxílio do avanço da pesquisa na área de Ciência da Computação.

O objetivo desta pesquisa é investigar questões relacionadas a adoção da aprendizagem baseada e desafios e desenvolvimento ágil em ambientes de desenvolvimento de aplicativos para dispositivos móveis. Para isto, os participantes são convidados a responderem um questionário. Durante a atividade os participantes serão orientados por um ou mais pesquisadores.

Lembramos que o objetivo deste estudo **não é avaliar o participante**, **mas sim** avaliar a prática que o participante usará durante o estudo de caso. O uso que se faz dos registros efetuados durante o experimento é **estritamente** limitado a atividades acadêmicas de pesquisa e desenvolvimento, garantindo-se para tanto que:

1. O anonimato dos participantes será preservado em todo e qualquer documento divulgado em foros científicos (tais como conferências, periódicos, livros e assemelhados) ou pedagógicos (tais como apostilas de cursos, *slides* de apresentações, e assemelhados).
2. O anonimato do local do estudo será preservado em todo e qualquer documento divulgado em foros científicos (tais como conferências, periódicos, livros e assemelhados) ou pedagógicos (tais como apostilas de cursos, *slides* de apresentações, e assemelhados).
3. Todo participante que se sentir constrangido ou incomodado durante a realização da atividade pode interromper a sua participação e estará fazendo um favor à equipe se registrar por escrito as razões ou sensações que o levaram a esta atitude. A equipe fica obrigada a descartar os dados do participante para fins da avaliação a que se destinaria.

Declaro que estou de pleno acordo com os termos acima. _____/_____/2015.

_____
Assinatura do participante

_____
Alan Santos
Pesquisador Responsável

Nome do Participante: _____
Pesquisadores Responsáveis: Me. Alan Santos e Dr. Paulo Fernandes
Contato: alan.santos @ pucrs.br e paulo.fernandes@pucrs.br
Faculdade de Informática - PUCRS