Sabadágil: Métodos Ágeis de Desenvolvimento de Software aplicados ao Ensino

Sabadágil: Agile Software Development applied to Education

Avelino Ferreira Gomes Filho

PPGI – Universidade Federal do Rio de Janeiro
avelino.filho@ppgi.ufrj.br

Maria Carmem Moraes

PPGI – Universidade Federal do Rio de Janeiro
mclmoraes@hotmail.com

Bruna de Barros Santos Xavier

PPGI – Universidade Federal do Rio de Janeiro
bruna.santos@ppgi.ufrj.br

Raphael Duarte Paiva

PPGI – Universidade Federal do Rio de Janeiro
raphaelpaiva@ufrj.br

Carlos Felipe Cardoso de Resende

PPGI – Universidade Federal do Rio de Janeiro
cfc@ufrj.br

Saulo Andrade

PPGI – Universidade Federal do Rio de Janeiro
sauloandrade@gmail.com

Douglas Nascimento Rehem

PPGI – Universidade Federal do Rio de Janeiro
douglasrehem@gmail.com

Rodrigo de Toledo

PPGI – Universidade Federal do Rio de Janeiro
rtoledo@dcc.ufrj.br

Resumo *Um dos grandes desafios das universidades é fazer dos alunos agentes ativos no contexto de ensino e aprendizagem e, ao mesmo tempo, promover o desenvolvimento profissional dos próprios professores. Pensando em vencer este desafio, o professor de Métodos Ágeis do Curso de Bacharelado Informática da Universidade Federal do Rio de Janeiro desenvolveu um método baseado em simulação do ambiente de desenvolvimento de sistemas utilizando diversas práticas ágeis em sala de aula que culmina com um evento anual denominado Sabadágil. Nesse artigo descreveremos o Sabadágil, evento em que os alunos em um único dia participam da construção de um software utilizando Métodos Ágeis numa simulação que os aproxima do ambiente profissional.*

Palavras-Chave: *Métodos Ágeis, Aprendizagem, Colaboração*

Abstract *A major challenge that many professors have to face is how to make their class attractive to students while transmitting the necessary knowledge. Thinking about it, the teacher of Agile Methods at the B.Sc. Informatics of the Federal University of Rio de Janeiro developed the Sabadágil. An annual event where students apply their knowledge through a practical and reflective experience in which they solve problems similar to those that occur in professional software development environment. In this article we describe this event, as it is done, its learning objectives and how it is assembled to make students learn Agile through knowledge application, experimentation and adaptation.*

Keywords: *Agile Methods, Learning, Collaboration*

1 Introdução

Passar conhecimento para os alunos e tornar essa experiência algo prático e agradável não é uma tarefa fácil. Na Universidade Federal do Rio de Janeiro, os professores da turma de Métodos Ágeis do curso de Ciência da Computação desenvolveram um evento que busca não só fazer com que os alunos aprendam, como também vivenciem diversas experiências comuns do dia-a-dia de organizações desenvolvedoras de *software*. O principal objetivo desse evento é fazer com que os alunos apliquem o conhecimento recebido durante o período na construção de produtos reais, utilizando valores, princípios e práticas de Métodos Ágeis de Desenvolvimento de Software.

O Sabadágil é um evento de um único dia em que os alunos são desafiados a criar um produto através de ciclos curtíssimos de iteração. Este evento busca proporcionar um aprendizado acelerado através da aplicação e transformação do conhecimento que os alunos fazem ao usar Métodos Ágeis em uma simulação que os aproxima do que eles vivenciarão no mercado de trabalho.

Além de apresentar o Sabadágil, o presente trabalho relaciona as teorias e métodos de aprendizagem que servem como base pedagógica para o evento. Espera-se que isso auxilie a outros educadores a criar novas formas de aprendizagem baseadas em processos sociais, experienciais e transformativos.

O artigo está organizado da seguinte forma: a Seção 2 apresenta os Métodos Ágeis utilizados, os trabalhos relacionados e as teorias de aprendizagem que servem de base pedagógica para a realização do evento; O Sabadágil é descrito na Seção 3. O método utilizado na pesquisa é apresentado na Seção 4; Na Seção 5, as suposições da pesquisa são descritas; Os resultados encontram-se na Seção 6; A discussão sobre os resultados está na Seção 7; O trabalho é concluído na Seção 8 e a Seção 9 apresenta os trabalhos futuros.

2 Referencial Teórico

2.1 Métodos Ágeis

Até o final da década de 90, o desenvolvimento de software baseava-se no Modelo Cascata [1]. Esse modelo era composto por sete etapas sequenciais que começavam pela especificação do sistema de informação e iam até a implantação do produto no cliente. Neste modelo, o Time de desenvolvimento era separado em pequenos times de especialistas (negócio, requisitos, analistas, arquitetos, programadores, testadores e operação). Cada Time de especialistas era responsável por executar uma etapa do processo. Havia a necessidade de detalhamento minucioso do trabalho a ser feito para evitar qualquer tipo de erro,

pois caso alguém introduzisse um problema, ele seria descoberto apenas em uma etapa tardia [2]. Quando a equipe de especialista responsável terminava sua parte do trabalho, ele encaminhava os resultados que serviam de insumos para a próxima etapa e para o próximo grupo de especialistas.

Entretanto o formato sequencial do modelo apresentava algumas fragilidades. As etapas de especificação e análise ocorriam no início do projeto, enquanto o conhecimento sobre o problema que o software resolveria ainda era imaturo. A compreensão do problema e a solução evoluem ao longo do projeto [2]. Além disso, a busca pelo erro zero consumia tempo e custo. Isso levou a criação de modelos iterativos e incrementais [3][4].

Com o surgimento desses novos tipos de modelos, a Engenharia de Software ganhou dinamismo e passou a incentivar o uso de métodos que conseguissem refinar o produto repetidas vezes e gerassem maior satisfação de times de desenvolvimento, cliente e usuários. Porém, diferentes desenvolvedores perceberam que o foco desses novos métodos estava sobre aspectos processuais e ferramentas que eram incapazes de satisfazer os principais objetivos da Engenharia de Software: criar, de forma organizada, produtos úteis, com alta qualidade que resolvesse problemas de clientes e usuários [5][6].

Em 2001, alguns desses desenvolvedores escreveram o Manifesto Ágil [7]. Nesse manifesto eles definiram quatro valores e doze princípios fundamentais para o desenvolvimento de software de forma iterativa, interativa e incremental, capaz de produzir produtos de alta qualidade através de ciclos curtos de planejamento, execução, *feedback* e adaptação. Segundo os autores do manifesto, o software deve atender às necessidades do negócio e ser capaz de se adequar às mudanças necessárias para manter o alto valor agregado pelo produto ao longo do tempo. O processo utilizado para tal deve estimular a colaboração entre desenvolvedores, usuários e clientes, empregar times multidisciplinares e auto-organizados e promover o planejamento adaptativo e evolutivo [8] [9] [10].

O termo Métodos Ágeis de Desenvolvimento de Software não descreve métodos específicos para construção de sistemas computacionais. De fato, esse termo serve como um "guarda-chuva" para um conjunto de práticas, ferramentas e técnicas compatíveis com os valores e princípios do Manifesto Ágil [11]. Para evitar essa confusão, alguns autores preferem utilizar o termo Desenvolvimento Ágil de Software [5][6].

Das diversas implementações de Métodos Ágeis, são de especial interesse para o presente trabalho: os Métodos *eXtreme Programming* (XP) e *Lean Startup*, o *framework Scrum*, o sistema *Lean Kanban*.

O eXtreme Programming (XP) foi originalmente descrito por Kent Beck no final dos anos 90. Segundo o autor, é extremamente importante que exista intensa colaboração entre usuários e desenvolvedores. Os ciclos de desenvolvimento, feedback, adaptação devem ser curtíssimos, se possível uma semana. Para Beck [12], ciclos curtos diminuem o risco de problemas, aceleram o processo de experimentação e validação e geram comprometimento entre desenvolvedores e usuários. O XP descreve algumas práticas fundamentais para o bom desenvolvimento do software, as mais utilizadas no Sabadágil são: ambiente informativo, programação em pares, código compartilhado, construções curtas, *User Stories* (US), programação orientada a testes e código coletivo.

O *Scrum* é o *framework* utilizado para a gestão e acompanhamento de projetos de Desenvolvimento Ágil. Como qualquer arcabouço, ele provê uma abstração conceitual sobre o que deve ser feito, sem descrever como será feito. Ele apresenta um ciclo de desenvolvimento, papéis e artefatos que auxiliam a gestão de produtos do conhecimento. A instanciação prática depende do contexto de aplicação [13]. A Seção 3 apresenta a adaptação feita para o Sabadágil.

O *Lean Kanban* é um sistema que busca trazer transparência sobre o processo de desenvolvimento, reduzir desperdício e busca embutir a qualidade do produto em cada etapa do processo [14]. Dentre as várias inovações trazidas pelo Kanban, destacam-se no Sabadágil, a corrente de valor, o sistema puxado, os limites de trabalho em andamento, filas de itens de escopo e a gestão visual. Durante o evento, as técnicas de *Lean Kanban* aparecem no processo de desenvolvimento enxuto dado o tempo que os alunos tem para implementar os produtos, nas restrições de carga de trabalho impostas pelos próprios alunos e na transparência do processo descrito em etapas de desenvolvimento e apresentado em local visível para todos os participantes.

Ao contrário daqueles já mencionados, o *Lean Startup* tem o foco no negócio e não no produto. Ele descreve uma série de práticas e técnicas empreendedoras utilizadas para validar suposições mercadológicas de forma rápida, iterativa e com *feedback* de clientes potenciais [15]. O uso desse método acontece na fase de preparação do Sabadágil para avaliar o problema e hipóteses de soluções escolhidos pelos grupos. Também é utilizado após o evento para coleta de métricas de usuários e readequação dos produtos.

2.2 Teorias e Métodos de Aprendizagem

Uma vez que os Métodos Ágeis quebram muitos paradigmas da Engenharia de Software [16], os professores da disciplina decidiram que utilizar os Métodos Tradicionais de Ensino, em que o professor passa um período

apresentando conceitos e no final aplica uma avaliação, não seria o mais adequado. Então, decidiram que o Sabadágil deveria basear-se em métodos e teorias de aprendizagem inovadoras para auxiliar os alunos a compreender, reter, avaliar e utilizar o conteúdo apresentado em sala de aula.

O modelo aplicado no evento busca da Teoria da Aprendizagem Experiencial descrita por Kolb [17] a aquisição de conhecimento através da experiência prática. Segundo o autor, o indivíduo adquire o conhecimento quando aplica conceitos ao tentar solucionar problemas práticos e reflete sobre suas ações experienciais. Pela observação sobre suas ações, essa pessoa torna-se capaz de desenvolver abstração teórica que irá auxiliá-lo a formular novas hipóteses e generalizar os resultados obtidos [18]. O formato do Sabadágil busca proporcionar ciclos de interação de desenvolvimento através do uso de Métodos Ágeis. Os alunos aplicam seu conhecimento teórico em um problema prático e no final da interação refletem sobre o que aconteceu.

Dada as necessidades de times multidisciplinares, auto-organizados e intensa colaboração entre os envolvidos em projetos Ágeis [7], é fundamental que os alunos aprendam através da colaboração. Tal colaboração acontece entre os alunos e entre alunos e os professores. Logo, a Teoria da Aprendizagem Social é de extrema relevância para o Sabadágil.

Segundo os Bandura e McClelland [19], o conhecimento adquirido pelo indivíduo não é inteiramente próprio. Na verdade esse conhecimento sofre transformações advindas das interações e filtros proporcionados pela colaboração do indivíduo com outras pessoas do ambiente. No caso, alunos e professores. Os autores afirmam que o aprendizado vem de uma reflexão interativa que acontece através do compartilhamento de experiências e ideias entre os vários membros do grupo. O maior grau de aprendizado acontece quando o grupo reflete em conjunto sobre as experiências realizadas [20].

Para auxiliar a quebra de paradigmas proporcionada pelos Métodos Ágeis, O trabalho de Mezirow [21] sobre a Teoria de Aprendizagem Transformativa também se torna uma relevante base do evento. O autor escreve que o aprendizado é um processo constante de construção do conhecimento, onde o indivíduo deve se engajar em uma análise crítica sobre as ações tomadas para resolver um problema e essa interpretação transforma e aprimora o conhecimento inicial e guia suas ações no futuro. Segundo o autor, o conhecimento inicial é incompleto, por isso é necessária a experimentação e reflexão crítica para transformá-lo [22][23]. No evento, é esperado que os alunos consigam aplicar e adaptar os conceitos que eles tem sobre Métodos Ágeis ao aplicar e refletir sobre o que aprenderam e experimentaram durante período.

É possível perceber que as teorias supracitadas são fortemente ligadas, pois todas abordam a aprendizagem como um processo de transformação do conhecimento através de um ciclo de experiência prática e reflexão sobre as ações. Então, o método de Ensino Baseado em Problemas (EBP) foi utilizado para aplicar as teorias de aprendizagem no contexto de ensino de Métodos Ágeis. Para tal, o Software desenvolvido durante o evento deve resolver um problema real, ter usuários e métricas reais e pode até ser especificado por clientes reais. No EBP, os alunos são convidados a resolver problemas continuamente, aplicar a teoria em casos práticos, refletir sobre os resultados de suas ações e transformar seu conhecimento através da análise crítica de suas experiências [24]. Os produtos escolhidos pelos alunos servem como base para todo o aprendizado da disciplina. Durante o período, a construção é dividida em pequenas etapas e os alunos devem aplicar seu conhecimento para resolver cada uma delas.

2.3 Trabalhos Relacionados

Muitas contribuições acadêmicas têm sido apresentadas sobre o tema de ensino e aprendizagem baseado na adaptação de Métodos Ágeis de Desenvolvimento de Software para construção e transformação do conhecimento. Conforme apresentado nas Subseções a seguir, é possível identificar alguma relação entre esses trabalhos e agrupá-los em cinco categorias: Ensino de Engenharia de Software, Métodos Ágeis aplicado ao ensino de outras disciplinas, Simulação, Ensino de Métodos Ágeis através de Jogos e Ensino Baseado em Projetos.

Ao contrário da maioria dos trabalhos aqui descritos, este não se resume à adaptação apenas do *Scrum* e XP para o contexto educacional. Ele se estende para outros Métodos Ágeis citados na Subseção 2.1 de forma a propiciar um ensino enxuto e puxado de ponta a ponta.

Além disso, ele busca suporte pedagógico nas teorias e métodos de aprendizagem apresentados na Subseção 2.2. Na revisão de bibliografia apenas dois trabalhos tentam fazer essa ligação, porém a apresentam superficialmente.

Conforme já mencionado na introdução, o Sabadágil é um evento que simula o ambiente profissional de desenvolvimento de software e busca a aplicação, reflexão e adaptação do conhecimento em um curto espaço de tempo. Espera-se contribuir para o corpo de conhecimento de professores que desejem utilizar abordagens inovadoras, dinâmicas, participativas e baseadas na adaptação de Métodos Ágeis de Desenvolvimento de Software para o contexto acadêmico. Embora o foco deste trabalho seja a Engenharia de Software, espera-se futuramente adaptar o método para outras disciplinas além da Ciência da Computação.

2.3.1 Métodos Ágeis aplicado ao ensino de Engenharia de Software

Esse grupo de trabalhos descreve os estudos que adaptam e utilizam práticas e técnicas de Desenvolvimento Ágil para o ensino de Engenharia de Software nos cursos de Ciência da Computação.

Um dos primeiros trabalhos sobre a aplicação dos Métodos Ágeis no contexto acadêmico foi realizado por Goldman et al. [25]. No artigo, os autores descrevem o funcionamento do Laboratório de Programação Extrema da Universidade de São Paulo (USP). Eles apresentam como a metodologia *Extreme Programming* (XP) foi aplicada no curso de graduação através de técnicas de Ensino Baseado em Problemas. Os autores concluem que o ensino de Métodos Ágeis é mais eficiente quando os alunos são expostos a problemas práticos e o resolvem colaborativamente. Algumas das modificações apresentadas pelos autores como o modelo participativo e feedback constantes são utilizados durante a disciplina de Métodos Ágeis e tem seu ápice no Sabadágil.

Carvalho, Barbosa e Silva [26] aplicam uma abordagem lúdica para ensinar os alunos de graduação e pós-graduação a criar o componente fundamental de requisitos de Métodos Ágeis, as *User Stories* (USs). Segundo os autores, abordagens inovadoras e participativas despertam o interesse dos alunos, aumentam a motivação e colaboração entre os envolvidos.

Manhic [27] apresenta um método de aprendizagem baseado na adaptação do *Scrum*. Assim como no curso aqui estudado, o autor adapta técnicas e práticas do *Scrum* para o contexto de ensino. Segundo o autor, o método auxilia a transformação de conhecimento pelos alunos e é capaz de transformar a sala de aula em um ambiente informal, participativo e colaborativo, propicia ao aprendizado.

Kropp e Meier [28] apresentam seu estudo sobre a necessidade dos novos profissionais terem competências relacionadas com os valores do Manifesto Ágil [7]. Os autores apresentam como eles adaptaram tais valores para o curso de Engenharia de Software. Assim como no presente trabalho, o método utilizado busca gerar engajamento dos alunos e que ele contribua com a capacitação destes para o mercado de trabalho.

2.3.2 Métodos Ágeis aplicado ao ensino de outras disciplinas

Professores de outras disciplinas também percebem que o Desenvolvimento Ágil de Software pode ser adaptado para atender as necessidades dessas matérias. Esses professores utilizam os valores, princípios e práticas ágeis

com o intuito de prover mecanismos de colaboração, transparência, reflexão e melhoria contínua das práticas educacionais.

Cubric [29] fez a adaptação do *framework Scrum* e do Método XP para escola de negócios. Segundo a autora, a demanda por uma forma ágil de pensar ultrapassa as barreiras da informática e é uma habilidade desejada por diversas empresas nos mais diversos segmentos. Assim como no presente trabalho, a autora relaciona as práticas ágeis por ela adaptada com as teorias de aprendizagem. Ela destaca que o Desenvolvimento Ágil é um facilitador para que os professores consigam implementar teorias de aprendizagem, em especial a Experiencial e a Colaborativa. Tal conclusão é semelhante a que o presente artigo chega, porém, aqui, são utilizadas as teorias Transformativa e Social, além da Experiencial.

Grimheden [30] escreve sua experiência ao utilizar valores, princípios e técnicas o *Scrum* em um estudo de caso no curso de Mecatrônica. O autor conclui que o *framework* pode ser adaptado e utilizado no curso. Tal método faz com que os alunos tornem-se mais preparados para o mercado de trabalho, pois ficam mais flexíveis e capazes de lidar ao constante aumento da complexidade de produtos dessa área.

Piunno et al. [31] descrevem a sua experiência ao adotar a Gestão Ágil de Projetos (GAP) [6] no Laboratório de Pesquisa Interdisciplinar (AirLab). O laboratório é responsável por desenvolver projetos interdisciplinares de três áreas da ciência: química, física e biologia. Segundo os autores, a GAP aprimora significativamente a execução desses projetos, pois facilita a colaboração entre os membros das equipes, permite a solução rápida, iterativa e incremental de problemas, auxilia a adaptação das soluções às necessidades da pesquisa e cria um sistema de *feedback* e melhoria contínua que auxiliam os times multidisciplinares a alcançarem seus objetivos científicos.

Por fim, Teixeira Filho [32] apresenta o Método Ágil para Processos de Aprendizagem (MAPPA). É um método de ensino baseado não só nos valores, princípios e práticas de Desenvolvimento Ágil, como também em métodos de aprendizagem participativos, construtivistas, construcionistas e interacionistas. Ele foi utilizado por alunos de física e eletricidade nos cursos de Engenharia da Computação e Engenharia de Telecomunicação. O método traz como principais vantagens a transparência do processo e evolução do aprendizado, gestão e uso eficaz do tempo necessário para o ensino e aplicação do conhecimento. O autor conclui que o MAPPA é capaz de melhorar o aprendizado individual e coletivo dos alunos.

2.3.3 Simulação de ambiente profissional

Outra forma de ensino é a simulação de ambiente pro-

fissional. Nessa grupo estão os trabalhos que buscam simular virtualmente um ambiente profissional para o ensino de Métodos Ágeis.

Gkritsi [33] aborda a construção de software via simulação virtual para permitir diversas experiências sem a necessidade de envolver produtos ou pessoas reais. Tal abordagem também é utilizada por Rodriguez, Soria e Campo [34] no Virtual Scrum. Os autores apresentam um ambiente virtual colaborativo no qual eles simulam as iterações do processo de desenvolvimento comparando os Métodos Tradicionais frente aos Métodos Ágeis. Os autores concluem que o segundo conjunto de métodos é mais eficiente e que a ferramenta de simulação por eles desenvolvida auxilia o aprendizado e aplicação de Desenvolvimento Ágil de Software em um contexto acadêmico.

Entretanto, conseguir boas simulações em ambiente virtual é uma atividade complexa, pois todas as possíveis situações devem ser codificadas no software o que não abre espaço para as imprevisibilidades comuns em um ambiente real [35]. O Sabadágil não é restrito a situações previamente planejadas justamente porque a casualidade o aproxima de um ambiente real de trabalho.

2.3.4 Ensino baseado em jogos

O uso de jogos lúdicos também tem sido muito explorado para aperfeiçoar o processo de aprendizagem. Segundo os autores que utilizam essa forma de ensino, esses jogos são capazes de gerar aprendizado em um curto espaço de tempo aplicando o conhecimento teórico em casos práticos [36][37]. Todavia, nesses casos, os alunos não tem a experiência completa de gerar um produto e vê-lo sendo utilizado.

2.3.5 Ensino Baseado em Projetos

Também há trabalhos que não são especificamente sobre o ensino de Métodos Ágeis, porém são importantes, pois utilizam o Método de Ensino Baseado em Projetos em contexto acadêmico. Destaca-se o artigo de Wange- nheim, Carvalho e Battistella [38] que apresenta uma dinâmica que busca ensinar alunos de Engenharia de Software a lidar com situações emergentes em projetos. Segundo os autores, a abordagem prática e colaborativa tem impacto positivo na motivação, experiência do usuário e aprendizagem dos alunos.

3 Sabadágil

A disciplina de Métodos Ágeis oferecida pelo Departamento de Ciência da Computação. Ela é eletiva, tem carga horaria de sessenta horas, vale quatro créditos e o único pré-requisito para cursá-la é que o aluno tenha sido

aprovado em Fundamento de Engenharia de Software (FES). Ao contrário de outras disciplinas, as aulas são ministradas por dois professores. As turmas tem em média vinte alunos por período.

O Sabadágil é um evento de um único dia que acontece na parte final do período. Ele proporciona aos alunos uma experiência transformadora através da aplicação colaborativa de Métodos Ágeis de Desenvolvimento de Software na construção de um produto real.

Além dos alunos e professores da disciplina, o evento conta com a presença de profissionais de mercado, convidados para realizar os papéis de *Product Owner* e *Scrum Master* e *Agile Coach*. Isso é feito com o intuito de aproximar os alunos de pessoas com experiência profissional capazes de trazer conhecimento de mercado sobre Métodos Ágeis.

3.1 Preparação do Sabadágil

A preparação para o Sabadágil acontece durante as 52 horas de aula que antecedem o evento. Durante o período, os alunos desempenham o papel de *Product Owner* (PO). O PO é responsável por especificar e priorizar as funcionalidades e avaliar o produto desenvolvido [13].

O EBP começa logo na primeira aula do período. Nella os alunos fazem um *brainstorming* para escolher alguns problemas por eles percebidos que possam ser resolvidos por software. Qualquer problema pode ser apresentado, não há filtros. Após o *Brainstorming*, os alunos escolhem quais desses problemas não seguirão adiante. A eliminação depende do resultado do debate entre os defensores e aqueles que não gostaram da proposta. A eliminação é feita democraticamente por votação.

Depois da eliminação, os problemas escolhidos passam pelo Processo de Validação do Problema [15]. Este avalia se o problema existe de fato, se é possível resolvê-lo por software e se há aceitação do futuro produto por usuários reais. Os alunos dividem-se em duplas para que o conhecimento sobre esse processo seja construído colaborativamente. Diversas técnicas são apresentadas para que os alunos reflitam e apliquem aquela que for mais pertinente ao caso. No final, os alunos devem descrever todas as etapas do processo no *Validation Board* [39] e apresentar o resultado da validação para a turma. A turma poderá fazer comentários, críticas e sugestões sobre as técnicas utilizadas, hipóteses avaliadas, suposições feitas e do resultado da análise.

Após as apresentações, os alunos abrem mais um debate para defender, complementar ou questionar as validações realizadas. No final, apenas três problemas são escolhidos para servir de base para o EBP. A construção do produto é dividida em pequenas etapas que permitam a aplicação do conhecimento teórico para solucionar cada

parte do problema. A divisão em pequenas etapas é essencial para permitir maior número de experiências e reflexões possível. Essa reflexão não só permite o aprendizado, como também propicia a rápida adaptação do curso às necessidades, questionamentos e anseios dos alunos.

Uma vez que os problemas já tenham sido escolhidos, os alunos se juntam em três grupos, um para cada problema e começam a pensar em uma proposta de solução. As propostas são apresentadas no formato de *Elevator Pitch* [40] em sala e debatidas pelos alunos e professores.

Após a definição da proposta, os alunos descrevem os possíveis usuários dos seus produtos no formato de *Persona*. As *Personas* fornecem diferentes perspectivas sobre as funcionalidades que irão compor o *software* [41][42]. As *personas* escolhidas também são debatidas pelos alunos e professores com o intuito de aperfeiçoá-las ou, até mesmo, eliminar aquelas que se mostrarem desnecessárias.

Conhecendo o produto e os possíveis usuários, os alunos especificam os requisitos do *software* em pequenas funcionalidades descritas no formato de *User Stories* (USs) [26][42]. Estas têm como objetivo descrever a funcionalidade em uma linguagem compreensível tanto para o usuário final quanto para o Time de desenvolvimento. Para criá-las, os alunos utilizam o jogo *Prune The Product Tree* [43] que oferece uma maneira prática e divertida de contribuir para o processo de tomada de decisão, olhando para o conjunto de funcionalidades que compõem o produto de uma forma holística. A lista contendo todos as História de Usuários do projeto é chamada de *Product Backlog* (PB).

Uma vez que os alunos já tenham definido o produto, os possíveis usuários e suas funcionalidades, profissionais de mercado são convidados a desempenhar o papel de PO e os alunos passam a desempenhar apenas o papel de Time de desenvolvimento.

A primeira coisa que o novo PO deverá fazer após conhecer os desenvolvedores, os produtos e suas funcionalidades é estimar o valor de cada US. A estimativa reflete a prioridade da US, quanto maior o valor, mais prioritária. Isso é feito utilizando a técnica *Buy a feature* [43] na qual o PO tem um número fixo de cem moedas e deve distribuí-las entre as USs do PB.

O novo PO tem liberdade para alterar o PB conforme desejar. Ele pode criar novas USs, alterar as existentes ou excluir algumas se achar conveniente. A alteração é bem vinda, pois simula um contexto comum de mudança de requisitos das organizações. Vale ressaltar que a colaboração entre PO e Time é fundamental para a correta compreensão do problema e solução.

O Time de desenvolvimento recebe o PB priorizado e

estima o esforço necessário para implementar cada US. Essa estimativa é feita por cada grupo em uma reunião de *Planning Poker* [13] [44].

O *Product Owner* calcula a taxa de retorno do investimento de cada US que é dada pela Equação 1.

$$RoI = \frac{V}{E}$$

Equação 1: Retorno sobre o Investimento.

RoI é o Retorno esperado sobre o investimento (*Return of Investment*). **V** é o Valor que o *Product Owner* espera ganhar com disponibilização da funcionalidade no *software*. **E** é o Esforço estimado pelo Time para implementar a US. O RoI serve como um indicador de importância das US: quanto mais alto for, mais importante ela é.

Ao calcular o RoI, o PO em conjunto com o Time identifica o *Minimum Viable Product* (MVP). Ries [15] define o MVP como sendo o conjunto mínimo de funcionalidades que permite a avaliação de uma suposição sobre o comportamento de clientes ou usuários. A identificação do MVP é de extrema importância, pois espera-se que no final do Sabadágil cada grupo tenha implementado todas as USs contidas nele. Durante o evento, o PO e os alunos podem, colaborativamente, incluir, alterar e excluir USs conforme a experiência por eles adquirida demonstre a necessidade de modificação do MVP.

Também vale ressaltar que durante o período letivo, os alunos têm algumas seções de *Coding Dojo* [45] onde eles aprendem técnicas de codificação ágil e experimentam a linguagem de programação que utilizarão no evento.

Toda preparação é feita através da aplicação de conhecimento. No final de toda aula, os alunos fazem a Retrospectiva com o intuito de refletir sobre o que aprenderam. Eles discutem sobre o que foi bom, o que pode melhorar, o que eles aprenderam, o que não aprenderam e o que eles gostariam de ter aprendido.

3.2 Os Produtos

No período estudado, 2014.1, os alunos apresentaram vinte problemas. Desses, três foram eliminados no *brainstorming* inicial de ideais. Após o debate de Validação de Problemas, os alunos descartaram mais quatorze deles. Os problemas e produtos que serviram de insumo para o Sabadágil são apresentados a seguir.

O **Caronas** é um aplicativo integrado ao Facebook onde os usuários podem oferecer ou tomar caronas apenas de seus amigos ou dos amigos de seus amigos e com isso, espera-se promover uma cultura de compartilhamento de recursos (carro) ao mesmo tempo reduzir o risco de

assaltos e sequestros nesse modelo de transporte.

O **Tô na Fila** é um aplicativo móvel colaborativo de mapeamento de filas. Seu principal objetivo é fazer com que as pessoas evitem filas demoradas quando vão a um restaurante, banco, ou eventos culturais. O usuário do aplicativo pode ver como estão as filas ao redor dele e pode dizer o estado da fila em que ele está (cheia, média ou vazia).

Por fim, o **Joga Fácil** é uma aplicação *web* onde pessoas podem marcar e participar de eventos esportivos. Com foco principal no futebol, aqueles que tanto gostam de marcar com os amigos (e que sempre precisam completar o Time) quanto os que simplesmente querem jogar encontram uma forma simples e fácil de se juntar para “jogar uma pelada”.

3.3 A Dinâmica

Nesse período a disciplina contou com dezenove alunos que foram divididos em dois grupos de seis pessoas e mais um grupo de sete pessoas. Três profissionais externos foram convidados para desempenhar os papéis simulâneos de *Product Owner* e *Scrum Master* cada um em um grupo. Além disso, um quarto profissional foi convidado para, juntamente com os dois professores da matéria, desempenhar o papel de *Agile Coach* e auxiliar os *Product Owners* e *Scrum Masters*.

Importante ressaltar que embora esse seja um evento facultativo, realizado sempre em um sábado, ele conta com um altíssimo percentual de presença dos alunos. Nesta edição, todos os alunos inscritos na disciplina compareceram e ficaram até o final do evento. Foram registrados apenas dois atrasos.

A duração do evento é de oito horas com uma pausa para almoço e ocorre no mesmo local onde são ministradas as aulas. Reserva-se uma sala para cada Time e nelas eles contam com espaço para as estações de trabalho e quadro branco.

O tempo é dividido em três *Sprints* (iterações) de duas horas cada. A *Sprint* começa com cinco minutos da reunião *Sprint Planning 1* em que o Time e o PO decidem quais *User Stories* serão implementadas naquela iteração. Após isso, eles passam para a *Sprint Planning 2* que é outra reunião, agora de dez minutos em que o Time discute tecnicamente como as USs selecionadas serão implementadas. Em seguida começa a fase de desenvolvimento em que o Time transforma a US em software.

Para simular a *Daily Meeting* [13], de trinta em trinta minutos, todos param para fazer uma pequena reflexão sobre o andamento do trabalho. Nessa reflexão cada membro do Time deve dizer o que ele já fez, o que ele fará, se há algo que o atrapalha. Como o conhecimento se

transforma durante a construção do produto, em algumas *Daily Meetings* os alunos também aproveitam para sugerir melhorias no processo de construção ou no produto. Finda essa reunião, o Time volta para o desenvolvimento.

Vinte minutos antes do final da *Sprint*, o Time apresenta o *Software* para o PO. Esse avalia a entrega e fornece *feedback* sobre o produto.

Depois da entrega e revisão, o Time se reúne mais uma vez por quinze minutos para fazer a reunião de Retrospectiva [13]. Essa reunião é de extrema importância, pois discute sobre todo o aprendizado que eles tiveram durante a *Sprint*. A Retrospectiva permite que eles compartilhem suas experiências, conversem sobre o que foi feito e como eles podem aperfeiçoar o trabalho que ainda será realizado. É também o momento em que os alunos conversam sobre os *insights* que tiveram sobre a aplicação de conceitos aprendidos durante as aulas. A Retrospectiva auxilia a consolidação da aprendizagem do grupo através da reflexão constante sobre a experiência prática.

No final da última *Sprint*, todos os grupos apresentam os produtos para a turma, professores e convidados. Após essa apresentação, cada grupo retorna para a sua sala para debater sobre o aprendizado que eles tiveram durante o dia.

No fechamento do evento, os alunos são convidados a apresentar quais itens de aprendizagem foram utilizados durante o dia e o quanto eles acham que o conhecimento sobre aquele item evoluiu através da experiência colaborativa. O item de aprendizado é parte do conteúdo apresentado na sala de aula. Por exemplo, nas aulas cujo tema é *Scrum*, os alunos são apresentados aos itens: *Product Backlog*, papéis do *Scrum*, cerimônias do *Scrum*, *Burn Down Chart*, *Sprint*, entre outros. Já na aula sobre *Lean Kanban* os itens são: *Work In Progress*, *Lead Time*, *Touch Time*, Transparência do processo de desenvolvimento, representatividade das *User Stories*, etc.

A Figura 1 apresenta o Ciclo de Desenvolvimento que acontece antes e durante o Sabadágil. Com esse formato de tempo reduzido, é fomentado o foco na entrega e nas práticas de produtos enxutos [15].

Figura 1: Ciclo de Desenvolvimento do Sabadágil.

A Aprendizagem Social é estimulada durante todo o evento. A codificação do produto é feita em duplas auto-organizadas, essas as duplas mudam durante o dia. Há reuniões de 30 em 30 minutos, no final de cada *Sprint* e no final do evento para discutir sobre as ações que o grupo tomou e como essas ações influenciam no resultado.

Durante o evento, é comum que os alunos se encontrem em situações difíceis como impasses, falta de objetividade, problemas de comunicação, decisões controversas, entre outras. Nesse momento os convidados externos e os professores são de fundamental importância para levar os alunos à reorientação do conhecimento. Eles auxiliam os alunos através do Diálogo Socrático [46]. Ao invés de dar a resposta que resolve o problema, eles conduzem os alunos através de perguntas para que eles cheguem à solução. Tal postura, permite que os grupos façam uma autoanálise de suas ações e encontrem soluções próprias.

3.4 Pós-Sabadágil

O Sabadágil termina quando a aplicação desenvolvida é disponibilizada para usuários reais. Os alunos passam as próximas semanas obtendo métricas, informações e feedback desses usuários. Esses dados auxiliam o aperfeiçoamento do produto que conta com uma apresentação final na última aula do período.

4 Método de Pesquisa e Análise

Para avaliar essa proposta, foi aplicado o desenho proposto por Pimentel para estudos de caso [47]. Inicialmente, os pesquisadores entraram em contato com os professores responsáveis pelo Sabadágil para compreender a dinâmica do evento e da disciplina de Métodos Ágeis. Em seguida, os pesquisadores buscaram o referencial teórico nas bases da *IEEE Xplore*, *ACM DL*, *Elsevier's Science Direct*, *Scielo* e *Scopus*. O *Google Scholar* também foi utilizado como fonte complementar da bibliografia.

O seguinte comando de busca foi utilizado: "*Agile*" AND "*Software*" AND "*Development*" AND "*Project*" AND ("*Education*" OR "*Learning*") e os campos de sobre os quais as pesquisas foram realizadas foram: título, *Abstract* e palavras-chave. No caso da *Scielo*, o comando de busca foi traduzido para o português. Além dos resultados dessa pesquisa, fontes de grande relevância sobre Desenvolvimento Ágil, Aprendizagem e Metodologia Científica também foram utilizadas.

A pesquisa retornou diversas contribuições. Estas foram sistematicamente analisadas através da leitura do título, abstract, introdução, conclusão e finalmente a leitura completa do artigo. As mais significativas são

apresentadas na Seção 2.

Após o levantamento bibliográfico e a compreensão do evento pelos pesquisadores, a preparação do estudo de caso foi feita. Foram realizadas quatro reuniões com os professores da disciplina. Nessas reuniões, definiu-se a forma como os grupos seriam observados, a divisão dos observadores, as variáveis que seriam observadas e reuniões de avaliação posteriores ao evento.

Os pesquisadores tiveram acesso aos artefatos que descreviam tanto o problema, quanto a proposta de solução (Lista de ideias, *Validation Boards*, *Elevator Pitches* e o *Product Backlog*).

Na penúltima aula antes do evento, os alunos apresentaram os problemas e propostas de solução para os observadores. Estes se apresentaram e explicaram o trabalho científico que ocorreria durante o evento.

Inicialmente foi definido que cada grupo teria dois observadores, porém, devido a problemas de agenda isso não foi possível. Cada grupo foi acompanhado por pelo menos um observador.

Os observadores procuraram alimentar as variáveis apresentadas na Seção 5 e colher informações empíricas sobre o comportamento do grupo e a dinâmica do evento.

Após o Sabadágil, os observadores se reuniram duas vezes com os professores para avaliar os resultados obtidos e concluir a pesquisa.

Para avaliação dos resultados, foi aplicado o Método de Análise das Quatro Áreas de Domínio da Agilidade [48]. O método busca evidências de aplicação de conhecimento em quatro áreas de domínio que um Time de Desenvolvimento Ágil deve cobrir: Negócio, Cultural, Organizacional e Técnica.

A área de Negócio busca verificar o conhecimento sobre assuntos ligados ao produto de software. Como ele foi imaginado, valor que eles proporcionaram aos usuários finais, planos de desenvolvimento e entregas, etc.

A área Cultural evidencia as relações humanas dentro e fora do Time de desenvolvimento. Nela busca-se compreender como a colaboração acontece, como é feita a solução de conflitos e se os alunos são capazes de aplicar valores e princípios ágeis [7]. Também avalia se eles desenvolveram uma cultura ágil através da aplicação dos itens de aprendizado recebidos durante as aulas para resolver os diversos problemas que se apresentam durante o evento.

A Área de Domínio da Agilidade Organizacional avalia a forma de trabalho do Time. Como ele se organiza para dividir o trabalho, qual o processo de desenvolvimento adotado, como este é organizado, entre outros.

Por fim, a Área Técnica procura evidências que des-

crevam como o Time de desenvolvimento está criando o software. Se eles usam automação, ferramentas adequadas para codificação e gestão, etc.

Cada área possui um conjunto de variáveis associadas definidas pelos pesquisadores e professores com base nos itens de aprendizagem apresentados em sala de aula. Estas variáveis possuem duas informações, o valor desejado e o realizado. O valor desejado é definido pelos professores e indica que o item de aprendizado foi totalmente compreendido pelo grupo e aplicado durante o evento. O valor realizado é atribuído pelos pesquisadores através da análise dos dados coletados durante e após o evento.

As variáveis receberam um identificador que é composto pela primeira letra maiúscula da Área de Domínio da Agilidade e um número sequencial positivo. Há casos em que o valor da variável depende da medição de outras variáveis. Nestas ocasiões, são utilizadas letras minúsculas, sequenciais, começando de “a” para identificar a dependência. Por exemplo, o resultado da variável N1 apresentada na subseção 5.1 depende da divisão dos valores de N1b por N1a.

Após atribuição dos valores, o Gráfico de Barras sobre Aplicação do Conhecimento é montado. Esse gráfico representa percentualmente o quanto os alunos conseguiram aplicar o conhecimento de cada área durante o evento.

Esse gráfico é importante, porém insuficiente para descrever o estado do conhecimento e as lacunas de aprendizagem dos grupos. Por isso gráficos de teia são montados para representar visualmente estas informações.

Além das variáveis observadas nas Quatro Áreas de Domínio da Agilidade, também foi obtido comentários dos alunos sobre o que eles achavam do evento. Esse retorno foi importante para avaliar a disseminação da cultura ágil e a satisfação do aluno ao participar do evento.

Para tal, foi disponibilizado para cada grupo o Quadro de *Feedback* e Melhorias. Esse quadro foi dividido em três partes. Na primeira, os alunos colavam *Post-its* para indicar os fatos que eles estavam percebendo do uso de Métodos Ágeis durante o desenvolvimento. Alguns exemplos do que os alunos escreveram nesse quadro são: “O Task Board ajudou a organizar o trabalho”, “A apresentação no final da *sprint* gerou comprometimento”, entre outros. Na segunda parte, os alunos poderiam escrever sugestões de melhoria (processo de trabalho, ferramentas, práticas, organização, etc.). Essa parte do quadro foi utilizada para alimentar a variável C1a apresentada na Subseção 5.2.

Por fim, na terceira parte, os alunos utilizaram a téc-

nica de *Start, Stop e Continue* [52][53], para identificar alternativas de ações que deveriam ser iniciada para melhorar o trabalho.

Além disso, na última aula do período os alunos fizeram considerações sobre o que eles gostaram do Sabadágil e o que eles acham que podia melhorar. Eles também disponibilizaram acesso dos pesquisadores aos grupos do Facebook para que esses obtivessem relatos do andamento do trabalho e dos comentários sobre o evento.

5 Suposição

Nesse trabalho utilizaremos o conceito de suposição no lugar de hipótese. Segundo Dias e da Silva [49] pesquisas que analisam qualitativamente um questionamento científico validam suposições e não hipóteses. Isso acontece porque os métodos aplicados e a avaliação realizada oferecem indícios de que a prática estudada funciona. Entretanto, é difícil generalizar os resultados dessas pesquisas para outros contextos.

Assim como nos trabalhos de [26][27][34][36][37][38]. A suposição do presente artigo avalia o quanto o Sabadágil é capaz de contribuir com o processo de aprendizado de desenvolvimento de software ao realizar uma experiência prática que simula situações e problemas similares àqueles que o futuro profissional enfrentará no ambiente corporativo.

Ele tenta descobrir o quanto os alunos são capazes de aplicar os itens de aprendizagem apresentados durante o período letivo em uma simulação do futuro ambiente profissional; o quanto o evento promove a Cultura Ágil de colaboração, transparência e adaptação; e a satisfação do aluno ao participar do evento.

As subseções a seguir apresentam as variáveis observada em cada Área de Domínio da Agilidade, suas composições, possíveis valores e valores desejados.

5.1 Negócio

N1 mede a capacidade do Sabadágil em proporcionar a construção de um software com as suas funcionalidades essenciais em um único dia. É o resultado da divisão de N1b (Total de USs do MVP implementadas) por N1a (Total de USs no MVP). N2 é o uso de metas claras e bem definidas para lançamento de produto no término da *Sprint (release)*. A meta auxilia o Time a ter um objetivo durante cada iteração [12][13]. N3 indica se os grupo é capaz de responder às mudanças de escopo do produto que acontecem durante o evento. É o resultado da divisão de N3b (Total de USs modificadas e implementadas) por N3a (Total de USs modificadas).

É importante que o Time tenha consciência que a priorização das USs vem do *Product Owner* e que ele pode

mudar a prioridade se achar necessário. Por isso, N4 é o percentual que indica se as USs implementadas correspondem àquelas que o *Product Owner* indicou como mais prioritárias. É dada pela divisão de N4a pelo total de US implementadas N1b mais N3b.

ID	Variável	Unidade de Medida	Valor Desejado
N1a	Quantidade de <i>User Stories</i> no MVP.	Unidade	100%
N1b	Total de <i>User Stories</i> implementadas.	Unidade	
N2	Uso de meta para lançamento do produto.	Unidade	1 para cada Sprint
N3a	Quantidade de <i>User Stories</i> modificadas	Unidade	100%
N3b	Quantidade de <i>User Stories</i> modificadas e implementadas.	Unidade	
N4a	Quantidade de US prioritárias implementadas	Unidade	Todas as US implementadas.

Tabela 1: Variáveis observadas na Área de Domínio da Agilidade de Negócio.

5.2 Cultural

C1 verifica os alunos são capazes de fazer melhoria contínua e adaptar o conhecimento teórico para situações inesperadas que acontecem durante o evento a fim de garantir o resultado. É o resultado da divisão de C1b (Total de melhorias implementadas) por C1a (Total de melhorias levantadas e explicitadas).

A variável C2 foi a mais debatida durante as reuniões de preparação do estudo. Ela busca avaliar se os alunos desenvolveram a capacidade de aplicar conceitos ágeis vistos em sala de aula durante o Sabadágil. Durante o período, os alunos são expostos a noventa itens de aprendizado. Esses itens vão desde os valores e princípios ágeis, passando pelas práticas e técnicas dos métodos, frameworks e terminando em métricas de atração e retenção de usuários. Após uma série de debates entre os pesquisadores e professores da disciplina, concluiu-se que os alunos não teriam como aplicar todos os itens de aprendizado no evento, pois alguns foram aplicados durante a preparação e outros seriam aplicados apenas após o Sabadágil. Então, foi definido que os alunos poderiam identificar e aplicar até sessenta e cinco itens durante evento.

Para obter informações que expusessem a aplicação desses itens de aprendizagem, foi realizada a Dinâmica

“O que aprendemos” no final do evento. Os alunos escreveram os itens de aprendizado em *post-its* e os colocam em uma escala qualitativa o quanto eles acham que seu conhecimento sobre cada item evoluiu. A Figura 2 apresenta o resultado dessa dinâmica para um dos grupos estudados.

Figura 2: Itens de Aprendizado levantados pelo Grupo Caronas na dinâmica O que aprendemos.

Cada *post-it* foi coletado, identificado, lido e classificado como item de aprendizado. A classificação aconteceu pela revisão cega de dois pesquisadores. O primeiro lia o conteúdo do *post-it* e classificava-o como um item de aprendizado em uma planilha do Google Drive. O segundo, sem saber o resultado do primeiro, também lia o *Post-it* e o classificava em outra planilha. Um terceiro pesquisador foi o responsável por juntar as planilhas e verificar as diferenças. Caso existissem, ele mesmo as resolvia com a sua classificação.

Dado que não foram definidos parâmetros claros para que os alunos valorassem a evolução do conhecimento sobre o item de aprendizagem durante o evento, a avaliação de C2 levou em consideração apenas se o item foi escrito ou não pelos alunos.

Já a variável C3 avalia se o Sabadágil proporciona o uso de princípios ágeis de compartilhamento do conhecimento, times multidisciplinares, auto-organização e colaboração. C4 verifica se o evento proporciona o uso de princípios ágeis de respeito e solução de conflitos pelo Time. É o resultado da divisão de C4b (quantidade de conflitos solucionados pelo Time) por C4a (quantidade de conflitos);

ID	Variável	Unidade de Medida	Valor Desejado
C1a	Melhorias contínuas levantadas.	Unidade	100%

C1b	Melhorias contínuas implementadas.	Unidade	
C2	Quantidade de Itens de Aprendizado utilizados.	Unidade	100%
C3	É possível perceber que os alunos estão compartilhando conhecimento durante o evento.	Sim ou Não	Sim
C4a	Conflitos durante o evento.	Unidade	100%
C4b	Conflitos solucionados pelos próprios alunos.	Unidade	

Tabela 2: Variáveis observadas na Área de Domínio da Agilidade Cultural.

5.3 Organizacional

O1 indica se os alunos conseguem trabalhar colaborativamente, compartilhar o código e recursos. Espera-se que o número de computadores utilizados seja a metade do número de alunos para que eles façam a programação em pares [12] e compartilhem mais facilmente o conhecimento. Um número superior de computadores indica que há alunos trabalhando individualmente, o que pode dificultar a colaboração. Um número baixo de computadores pode indicar que há alunos que não estão participando da codificação.

O2 busca compreender a forma como os grupos se auto-organizam [7]. Ela é formada pelo valor composto de quatro variáveis booleanas. Cada uma contribui com 25% do valor de O2. O2a indica se os grupos conseguem compartilhar código-fonte entre as duplas para que o conhecimento circule não só na dupla, como em todo o grupo. O2b julga se o evento é capaz de proporcionar um ambiente em que os alunos sejam auto-organizados como é esperado em um Time ágil. O2c verifica se os alunos utilizaram *User Stories* para definir o trabalho a ser feito. Já O2d apresenta quando a troca de pares é realizada;

O3 avalia quantas vezes os alunos trocam de pares de programação e O4 avalia se o Sabadágil cria um ambiente de aprendizado agradável e estimulante em que todos os alunos participam ativamente da criação do software;

O5 indica se os alunos organizaram o projeto de software através de um processo de desenvolvimento. Ela é composta por quatro variáveis, cada uma contribui com 25% para o valor de O5. São elas: O5a - verifica se os alunos apresentaram um processo bem definido, dividido em etapas de agregação de valor ao Software e utilizam técnicas aprendidas em sala para dividir o trabalho a ser realizado; O5b, O5c e O5d medem a transparência do

processo. Se os alunos conseguem utilizar práticas ágeis capazes de tornar o trabalho transparente para os membros do grupo e *Scrum Master*, *Product Owner* e *Agile Coaches* [50].

ID	Variável	Unidade de Medida	Valor Desejado
O1a	Quantidade de alunos no grupo	Unidade	50%
O1b	Quantidade de Computadores utilizados.	Unidade	
O2a	Algum trecho do código foi alterado por mais de uma dupla.	Sim ou Não	Sim
O2b	O trabalho foi dividido pelo grupo sem a necessidade de intervenção do <i>Scrum Master</i> .	Sim ou Não	Sim
O2c	Uso de <i>User Stories</i> para descrever escopo.	Sim ou Não	Sim
O2d	Momento para troca de pares.	IS = Início da Sprint; DS = Durante a Sprint; ST = Sem Troca	IS ou DS
O3	Frequência de troca dos pares.	$\frac{Qtde.Troca\ Par}{Qtde.Sprints} \times 100$	100%
O4	Percentual de alunos trabalhando individualmente	$\frac{Qtde.Aluno\ Indiv.}{Qtde.Alunos} \times 100$	0%
O5a	Definição de um processo de desenvolvimento de software.	DA - Definido pelos alunos; AE - Definido pelos alunos em colaboração com os agentes externos <i>Scrum Master</i> ou <i>Agile Coaches</i> ; DE - Definido pelos agentes externos; ND - Não Definido;	DA ou AE

O5b	Uso do <i>Definition of Done</i> (DoD).	Sim ou Não	Sim
O5c	Uso do <i>Definition of Ready</i> (DoR).	Sim ou Não	Sim
O5d	Gestão Visual para mapear processo.	Sim ou Não	Sim

Tabela 3: Variáveis observadas na área de domínio Organizacional.

5.4 Técnica

T1 avalia se os alunos utilizaram ferramentas de Desenvolvimento Ágil apresentadas durante as aulas; T2 avalia se os alunos utilizaram técnicas de automação de testes e TDD [51]; T3 testa se os gráficos de completude de trabalho foram atualizados pelo grupo. T4 informa a quantidade de USs que foram transformadas em tarefas para facilitar o andamento e acompanhamento do trabalho. Por fim, T5 descreve se os alunos utilizaram testes de aceitação para garantir que o produto de software entregue era o que estava especificado pelo *Product Owner*. Espera-se que cada *User Story* implementada tenha pelo menos um teste de aceitação relacionado. O valor de T5 é dado pela divisão de T5a pela soma de N1b mais N3b.

ID Per.	Variável	Unidade de Medida	Valor Desejado
T1a	A linguagem de programação apresentada em aula pelos próprios alunos foi utilizada.	Sim ou Não	Sim
T1b	Utilizaram Integração Contínua.	Sim ou Não	Sim
T1c	Utilizaram Integração Contínua	Sim ou Não	Sim
T1d	Utilizaram <i>Task Board</i> .	Sim ou Não	Sim
T2	Quantidade de testes automatizados implementados.	Unidade	Pelo menos 1 Por US implementada
T3	Atualização de Gráficos para a completude do Projeto.	$\frac{Qtde.Atualizac.}{Qtde.Sprints}$	100%
T4	Quebra de <i>User Stories</i> (US)	$\frac{Qtde.US\ Quebrada}{Qtde.US\ Desenv.}$	100%

	em Tarefas.		
T5a	Testes de Aceitação realizados.	Unidade	Pelo menos 1 Por US implementada

Tabela 4 Variáveis observadas da Área de Domínio da Agilidade Técnica.

6 Resultados

Os resultados estão divididos em duas partes. A primeira é a avaliação feita pelas evidências coletadas pelos observadores e a segunda é a avaliação feitas pelos próprios alunos.

6.1 Dados Observados

A Tabela 5 apresenta os dados coletados pelos observadores.

ID Per	Variável	Caronas	Tô na Fila	Joga Fácil
N1a	Quantidade de User Stories no MVP.	5	5	4
N1b	Total de User Stories implementadas	2	3	2
N2	Uso de meta para lançamento do produto	2	3	3
N3a	Quantidade de User Stories modificadas	1	2	3
N3b	User Stories modificadas e implementadas.	1	2	1
N4a	Quantidade de US prioritárias implementadas	3	5	3
C1a	Melhorias contínuas levantadas	22	25	11
C1b	Melhorias contínuas implementadas	26	31	11
C2	Quantidade de Itens de Aprendizado	39	32	21
C3	Compartilhar conhecimento durante o evento.	Sim	Sim	Sim
C4a	Conflitos durante o evento	1	0	0

C4b	Conflitos solucionados pelos próprios alunos	1	0	0
O1a	Quantidade de alunos no grupo	7	6	6
O1b	Quantidade de Computadores utilizados	3	3	3
O2a	Código foi alterado por mais de uma pessoa	Sim	Sim	Sim
O2b	O trabalho foi dividido pelo grupo	Sim	Sim	Sim
O2c	Uso de User Stories para descrever escopo	Sim	Sim	Sim
O2d	Momento para troca de pares	IS	IS	IS
O3	Frequência de troca dos pares	100%	100%	100%
O4	Alunos trabalhando individualmente	0%	0%	0%
O5a	Definição de um processo de desenvolvimento de software	AE	DA	DA
O5b	Uso do <i>Definition of Done</i> (DoD)	Sim	Sim	Sim
O5c	Uso do <i>Definition of Ready</i> (DoR)	Não	Sim	Não
O5d	Gestão Visual para mapear processo	Sim	Sim	Sim
T1a	Linguagem apresentada foi utilizada	Sim	Sim	Sim
T1b	Controle de Versão	Sim	Sim	Sim
T1c	Integração Contínua	Não	Não	Não
T1d	Utilizaram <i>Task Board</i>	Sim	Sim	Sim
T2	Testes Automatizados Implementados	0	0	0
T3	Atualização de Gráficos para a completude do	100%	100%	0%

	Projeto			
T4	Quebra de User Stories em Tarefas	100%	100%	100%
T5a	Testes de Aceitação realizados	4	2	3

Tabela 5: Dados coletados

6.2 Feedback dos Alunos sobre o evento

Cinquenta e oito fatos de aprendizagem foram explicitados. Os alunos realizaram cinquenta e sete sugestões de melhorias para os produtos e para o processo de desenvolvimento dos Time. A Tabela 6 apresenta quantitativamente os dados de feedback obtidos.

Equipe	Fatos	Sugestões de melhorias	Start	Stop	Continue
Caronas	22	22	9	6	11
Joga fácil	25	25	21	8	2
Tô na fila	11	11	2	9	0
TOTAL	58	57	32	23	13

Tabela 6: Observações levantadas durante o evento pelos alunos.

Já a Tabela 7 apresenta quantitativamente como os alunos descreveram sua percepção de aplicação dos itens de aprendizagem durante o evento na dinâmica “O que aprendemos”. Tal tabela foi utilizada para preencher a variável C2. Vale ressaltar novamente que C2 não utilizou a gradação Bom, Médio e Ruim.

Área de Domínio	Bom	Médio	Ruim
Negócio	14	7	1
Cultural	17	5	0
Organizacional	12	12	2
Técnica	8	6	8
TOTAL	51	30	11

Tabela 7: Feedback dado pelos alunos sobre a aplicação dos itens de aprendizagem durante o Sabadágil.

7 Análise

Conforme descrito na Seção 4, este trabalho utiliza o Método de Avaliação das Quatro Áreas de Domínio da Agilidade. A Figura 3 apresenta a aplicação do conhecimento.

Figura 3: Desempenho dos grupos em cada Área do Domínio da Agilidade.

O gráfico é criado a partir da média aritmética da soma dos valores das variáveis já computados dividido pelo total de variáveis contidas na Área. Por exemplo, o desempenho do Grupo Caronas na Área Negócio é dado pela equação:

$$N_{Caronas} = \frac{N1 + N2 + N3 + N4}{4} \times 100$$

Os valores foram:

$$N_{Caronas} = \frac{0,4 + 0,67 + 1 + 1}{4} \times 100$$

Logo, $N_{Caronas} = 77\%$.

Além de apresentar o desempenho dos grupos, o gráfico possui uma quarta coluna que indica a média do desempenho. Essa média ajuda os professores a identificar indícios de problemas em grupos que ficaram abaixo dela.

Dando continuidade à análise, os gráficos de teia de aranha permitem uma visão mais detalhada sobre os dados. Esses gráficos são compostos por cinco anéis. O primeiro representa o valor zero (0%), o segundo anel representa 20%, o terceiro 40%, o quarto 60%, o início do quinto anel 80% e o final do mesmo é o valor máximo de 100%.

Duas limitações se impõem nessa etapa do método. A primeira está relacionada à utilização de variáveis exclusivamente booleanas (sim ou não), pois estes só podem assumir os valores 0% ou 100%. Tais valores podem distorcer o gráfico principalmente quando assumem o valor zero, pois dão a impressão que os alunos não tem conhecimento algum sobre o item de aprendizado, o que pode não ser verdade. Outra limitação é a quantidade de variáveis observadas em cada área. A diferença aparece na suavidade do perímetro do gráfico. Áreas com mais variáveis podem dar a impressão de que os alunos possuem mais domínio do que aquelas com menos variáveis.

Cada gráfico representa a avaliação de aplicação de conhecimento de um grupo. A Figura 4 é a avaliação do grupo responsável pelo desenvolvimento do aplicativo Caronas. A Figura 5 pertence ao grupo Tô na Fila. A Figura 6 representa o conhecimento do grupo responsável pelo Joga Fácil.

Figura 4: Avaliação sobre a aplicação de conhecimento durante o Sabadágil para o Grupo Caronas.

Figura 5: Avaliação sobre a aplicação de conhecimento durante o Sabadágil para o Grupo Tô na Fila.

Figura 6: Avaliação sobre a aplicação de conhecimento durante o Sabadágil para o Grupo Joga Fácil.

O propósito de gerar os gráficos não é fazer a comparação entre os grupos e sim avaliar o estado do conhecimento dos alunos. As subseções a seguir apresentam a análise dos resultados pelas Áreas de Domínio da Agilidade e o *feedback* dos alunos.

7.1 Negócio

Os alunos foram capazes de aplicar uma grande parte do conhecimento apresentado em sala de aula na Área de Domínio da Agilidade de Negócio.

Os grupos estimaram que poderiam entregar quatro ou cinco USs do MVP (N1), entretanto eles não conseguiram implementar todas, pois os POs convidados modificaram as USs de acordo com as necessidades por eles percebidas (N3). Interessante perceber que, embora não tenham implementado o MVP planejado inicialmente, os alunos conseguiram implementar quatro ou cinco USs durante o evento. Esse cenário é bastante interessante, pois simula um ambiente real de mudança de requisitos durante o projeto [2][7][53].

Os grupos conseguiram utilizar metas de release para cada *Sprint* (N2). O grupo Caronas não utilizou na primeira *Sprint*, entretanto eles perceberam que a falta de foco prejudicou o Time e passaram a adotá-la.

Os grupos também respeitaram os acordos com relação às prioridades definidas pelos POs em conjunto com o Time (N4).

7.2 Cultural

Os grupos parecem conhecer e aplicar conhecimentos da Área de Domínio da Agilidade Cultural de forma

bastante similar. Todos os grupos conseguem identificar melhorias no processo de desenvolvimento de software (C1a) e aplicar essa melhoria (C1b). O fato de existirem mais melhorias implementadas do que identificadas para os grupos Caronas e Tô na Fila deve-se ao fato de que os alunos pararam de escrever os pontos de melhoria no quadro na 2ª *Sprint*. Esses grupos decidiram alterar seu processo de Reunião de Retrospectiva. No final da mesma, cada integrante já era designado para aplicar uma das melhorias por eles conversadas. Coube aos observadores anotar essas melhorias para o estudo. Para manter o caráter científico, os observadores não interferiram no trabalho realizado, logo não colocaram as melhorias no quadro designado para tal. A anotação ficou apenas no material de pesquisa. Para saber se as melhorias por eles identificadas foram implementadas, foram coletados dados do *Task Board*, servidor de configuração e observação da reunião de Retrospectiva da 3ª *Sprint*.

Também é possível perceber que os grupos não conseguem aplicar todos os possíveis conhecimentos ensinados durante o período (C2). Em média, os grupos utilizaram 47% do conteúdo. Essa informação está sendo analisada para que seja possível compreender corretamente o fenômeno. Uma suposição que pode justificar o fato é a superposição de conteúdo. Por exemplo, durante o período, os alunos aprendem a acompanhar o projeto através do *Burn Down Chart* e do *Comulative Flow Chart*. Cada grupo escolhe o gráfico que melhor lhe agrada e uma vez escolhido, eles não experimentam o outro gráfico. Outra suposição que parece justificar também a baixa quantidade de testes automatizados (T2) e falta de Integração Contínua (T1c) é a dificuldade de construir e manter um servidor de Integração Contínua que roda os testes automáticos e integra as funcionalidades do software.

Um fato muito interessante é que os alunos parecem ter interesse de compartilhar conhecimento sobre todo o desenvolvimento (C3). Além de discutirem muito sobre problema, software, processo durante as reuniões, eles sempre trabalharam em duplas ou trios (O1) e, segundo anotações de todos os observadores, as duplas discutiam bastante sobre o que eles estavam implementando. Algumas duplas chegaram a adaptar e utilizar a Técnica Pomodoro [55] trocando o “piloto” da dupla de cinco em cinco minutos. Os observadores também reportaram que no final do dia, as duplas pareciam mais descontraídas, menos formais e mais colaborativas.

Apenas um conflito aconteceu durante todo o evento (C4a). Uma dupla do grupo Caronas teve uma discussão séria no final da 2ª *Sprint*. Os alunos, sem intromissão de pessoas externas ao grupo, conseguiram resolver o impasse (C4b) e no final do evento, os membros da dupla reconheceram os excessos e disseram que todo o ocorrido havia sido um grande aprendizado.

7.3 Organizacional

A Área Organizacional é aquela que os alunos mais conseguiram aplicar os conceitos passados em sala de aula.

Os grupos se auto-organizaram para trabalhar desenvolver as USs (O2c) através da técnica de programação em pares (O1). As duplas não eram estáticas, elas trocaram sempre no início da *Sprint* (O2d e O3). De acordo com as observações e modificações realizadas no servidor de configuração, o conhecimento sobre a implementação pode ser compartilhado entre os integrantes do grupo (O2a) sem a necessidade de pessoas externas intervir no grupo (O2b). Nenhum aluno trabalhou individualmente (O4).

Os grupos também organizaram seu processo de desenvolvimento de software (O5a). Apenas o grupo caronas utilizou a ajuda do PO para criar o processo. Todos utilizaram gestão visual (O5d) e o *Definition of Done* (O5b) para descrever os critérios que o grupo utiliza para avaliar se cada US foi totalmente implementada. Apenas o grupo Tô na Fila utilizou o *Definition of Ready* (O5c) para indicar que uma US estava pronta para ser desenvolvida.

7.4 Técnica

A Área Técnica foi aquela que apresentou o pior resultado entre as quatro áreas estudadas. A média de aplicação do conhecimento dessa área foi de 64%.

Dessa área, a Garantia da Qualidade do Software foi o tópico de Engenharia de Software mais afetado. Os alunos não implementaram testes automatizados (T2) e não realizaram Integração Contínua (T1c). A falta dessa integração levou os grupos a terem problemas quando as diversas duplas tentavam juntar as funcionalidades dos softwares. Segundo os professores, embora esse não seja um resultado ideal, foi considerado um aprendizado valioso, pois criou uma situação em que os alunos perceberam a importância da Garantia da Qualidade desde o início do desenvolvimento e não apenas no final.

Conforme descrito anteriormente, o problema não pode ser solucionado durante o evento, porque criar e manter um servidor de integração não é uma tarefa trivial. Para as próximas turmas, os professores planejam apresentar serviços de Computação na Nuvem que atendam à essa necessidade.

Os alunos realizaram testes de aceitação das USs entregadas (T5). Isso gerou um fato interessante, pois eles garantiam o que entregavam o que o PO queria (T5), porém não garantiam a qualidade da implementação (T2).

Ademais, com exceção da Integração Contínua, os

alunos utilizaram as ferramentas e técnicas apresentadas em sala de aula durante o evento (T1b, T1d e T4). Também utilizaram a linguagem de programação que escolheram e apresentaram nas seções de *Coding Dojo* (T1a).

Com exceção do grupo Joga Fácil, os grupos mantiveram seus gráficos de completude de trabalho atualizados a cada *Sprint* (T3).

7.5 Feedback dos alunos sobre o evento

Pelas informações apresentadas na Tabela 6 é possível perceber que os alunos têm uma boa percepção de como as práticas apresentadas durante a disciplina os auxiliam na construção do Software. Também identificaram uma quantidade significativa de melhorias no processo de desenvolvimento. Além disso, a técnica de *Start, Stop and Continue* apresenta indícios de que eles refletiram sobre o resultado da aplicação de conhecimento em cada *Sprint*.

Já a Tabela 7 mostra que a maioria dos alunos acredita ter um bom conhecimento sobre a aplicação dos itens de aprendizado após o Sabadágil.

Por fim, o evento recebeu boas críticas na última aula da disciplina e nas redes sociais. Um aluno pediu: “poderia ter mais um Sabadágil”, outro disse “Gente, só queria deixar aqui o quanto adorei fazer esse trabalho com cada um de vocês!”. Outro escreveu que “Confesso que foi difícil sair da sala de aula ontem. A experiência de fazer a matéria e especialmente ter feito com esse grupo, no meu último período da faculdade, foi incrível”.

8 Conclusão

Esse trabalho apresentou o Sabadágil, um evento de um único dia que busca simular o contexto real de desenvolvimento de software, onde os alunos são convidados a ter experiências práticas sobre os diversos conceitos apresentados em sala de aula. Além de descrever os Métodos Ágeis que servem como base para o evento, ele também apresenta as teorias de aprendizagem que servem como suporte pedagógico ao mesmo.

O Sabadágil é parte da disciplina de Métodos Ágeis, a qual este artigo não é suficiente para retratar todos os detalhes. Para tal é recomendada a leitura de Gomes Filho, Resende e Toledo [56].

Ao apresentar o evento, buscou-se relacionar como ele relaciona as teorias e métodos de aprendizagem com suas atividades. Com isso, espera-se fornecer insumos para outros professores que desejem utilizar práticas inovadoras possam instanciar tais teorias em seu próprio contexto de ensino.

No tocante ao método, essa pesquisa utilizou um es-

tudo de caso realizado com dezenove alunos divididos em três grupos. A análise foi feita através do Método das Quatro Áreas de Domínio da Agilidade para auxiliar a visualização dos resultados e a identificação das lacunas de aprendizagem dos alunos. Técnicas de *feedback* também foram utilizadas para explicitar a reflexão que os alunos fizeram ao aplicar o conhecimento durante o evento. Por fim, foram coletados comentários dos alunos para avaliar o quanto o Sabadágil é capaz de criar um ambiente de aprendizado lúdico, participativo e engajador.

A partir dos dados aqui apresentados, é possível concluir que existem indícios de que o Sabadágil é eficiente no seu propósito de proporcionar um ambiente de aprendizado onde os alunos podem aplicar, experimentar, refletir e readequar seu conhecimento em um contexto que simula o futuro ambiente profissional.

9 Trabalhos Futuros

Inicialmente, o grupo buscará encontrar as possíveis causas que influenciaram o rendimento insatisfatório da Área Técnica. Os professores experimentarão serviços de Computação na Nuvem para Integração Contínua. Além disso, pretendem fazer mais seções de *Coding Dojo* para transmitir a importância da Garantia da Qualidade de Software desde o início do projeto.

A pesquisa continuará acompanhando as próximas turmas para avaliar os resultados das mudanças feitas pelos professores e o desempenho dos novos alunos. Nessas novas turmas, seria interessante alterar o desenho experimental para pré-teste e pós-testes ($O_1 \times O_2$) ou até mesmo fazer uma turma com o desenho experimental com grupo de controle ($O_1 \times O_2$, $O_3 \times O_4$) e avaliar o conhecimento dos grupos que participaram do Sabadágil e aqueles que não participaram do Evento.

Além da mudança do desenho experimental, é necessário transformar as variáveis booleanas em escala de valor para que elas não tenham tanta influência no gráfico. A variável C2 também deve ser revista para que ela atenda ao problema de escolha dos grupos descrita na Subseção 7.2.

Por fim, seria importante encontrar outros professores de Engenharia de Software que estivessem dispostos a aplicar o método aqui apresentado para evitar alguns vieses de pesquisa e proporcionar uma generalização maior dos resultados. Ademais, assim como apresentado no final do trabalho de Teixeira Filho [32], o método aqui apresentado pode ser adaptado e aplicado em outros cursos além da Ciência da Computação.

Referências

- [1] W. Royce. Managing the development of large software systems. In *Proceedings of the XXVI IEEE WESCON*. Páginas 328-338, 1970.
- [2] B. Boehm. Software Engineering Economics. *IEEE Transactions on Software Engineering*. 10 (1): 4 – 21, 1984.
- [3] C. Larman, V. R. Basili. Iterative and Incremental Development: A Brief History. *IEEE Computer*. 36 (6): 47-56, 2003.
- [4] B. Boehm. A Spiral Model of Software Development and Enhancement. *IEEE Computer*. 21 (5): 61-72, 1988.
- [5] A. Cockburn. Agile Software Development. 2nd Edition. Addison-Wesley. Upper Saddle River, NJ, 2006.
- [6] J. Highsmith. Agile Project Management. 2nd Edition. Addison-Wesley. Upper Saddle River, NJ, 2009.
- [7] K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries et al. The agile manifesto. <http://agilemanifesto.org/>, 2001.
- [8] K. Conboy, B. Fitzgerald. Toward a conceptual framework of agile methods: a study of agility in different disciplines. In *Proceedings of the 2004 ACM workshop on Interdisciplinary software engineering research*. Páginas 37-44, 2004.
- [9] H. Ming, J. Verner, L. Zhu, M. A. Babar. Software quality and agile methods. In *Proceedings of the 28th Annual International Computer Software and Applications Conference*. Páginas 520-525, 2004.
- [10] P. Abrahamsson, J. Warsta, M. T. Siponen, J. Ronkainen. New directions on agile methods: a comparative analysis. In *Proceedings of the 25th International Conference on Software Engineering*. Páginas 244-254, 2003.
- [11] B. Boehm. Get ready for agile methods, with care. *IEEE Computer*. 35 (1): 64-69, 2002.
- [12] K. Beck. Extreme programming explained: embrace change. US Edition. Addison-Wesley Professional. Boston, MA, 2000.
- [13] K. Schwaber, J. Sutherland. The scrum guide. 2011 Edition. Scrum.org. Boston, MA. 2011.
- [14] D. J. Anderson. Kanban: Successful evolutionary change for your technology business. 1st Edition. Blue Print Press. Sequim, WA. 2010.
- [15] E. Ries. The Lean Startup: How Constant Innovation Creates Radically Successful Businesses. Kindle Edition. Penguin Books Limited. New York, NY, 2011.
- [16] A. F. Gomes Filho, C. F. C. de Resende, R. Toledo. Usando Métodos Ágeis para Ensinar Métodos Ágeis. In *Proceedings do 5º Workshop Brasileiro de Métodos Ágeis (WBMA)*. Páginas 1-12, 2014
- [17] D. A. Kolb. The Process of Experimental Learning. In *Experiential learning: Experience as the source of learning and development*. 1st Edition. Páginas 21-38. Prentice-Hall. Englewood Cliffs, NJ. 1984
- [18] M. Keen, S. Mahanty. Learning in sustainable natural resource management: Challenges and opportunities in the Pacific. *Society and Natural Resources*. 19 (6): 497-513, 2006.
- [19] A. Bandura, D. C. McClelland. Social learning theory. 1st Edition. Prentice-Hall. New York, NY, 1977.
- [20] D. Armitage, M. Marschke, R. Plummer. Adaptive co-management and the paradox of learning. *Global Environmental Change*. 18 (1) 86-98, 2008.
- [21] J. Mezirow. Learning to think like an adult. In *Learning as transformation: Critical perspectives on a theory in progress*. 1st Edition. Páginas 3-33. Jossey-Bass. San Francisco, CA. 2000.
- [22] J. Mezirow. How critical reflection triggers transformative learning. In *Fostering critical reflection in adulthood*. 1st Edition. Páginas 1-20. Jossey-Bass. San Francisco, CA. 1990.
- [23] J. Mezirow. Transformative learning: Theory to practice. In *New directions for adult and continuing education*. 1997 (74): 5-12, 1997.
- [24] D. F. Wood. Problem Based Learning. *British Medical Journal*. 326 (7384): 328 – 330, 2003.
- [25] A. Goldman, F. Kon, P. J. S. Silva, J. W. Yoder. Being extreme in the classroom: Experiences teaching XP. *Journal of the Brazilian Computer Society*. 10 (2): 5-21, 2004.
- [26] L. C. Carvalho, M. W. Barbosa, V. B. Silva. Proposta e avaliação de uma abordagem lúdica para o ensino de Histórias de Usuário e Scrum. *Revista de Gestão e Projetos*. 5 (3): 44-58, 2014.
- [27] V. Manhic. Teaching Scrum through team-project work: students' perceptions and teacher's

- observations. 26 (1): 96-119, 2010.
- [28] M. Kroop, A. Meier. New Sustainable Teaching Approaches in Software Engineering Education: How agile methods influence teaching software engineering. In: *Proceedings of IEEE Global Engineering Education Conference (EDUCON)*. Páginas 1019-1022, 2014
- [29] M. Cubric. An agile method for teaching agile in business schools. In: *The International Journal of Management Education*. 11 (3): 119-131, 2013.
- [30] M. E. Grimheden. Can agile methods enhance mechatronics design education? In: *Mechatronics*. 23 (8): 967-973, 2013.
- [31] P. A. E. Piunno, C. Boyd, V. Barzda, C. C. Gradinaru, U. J. Krull, S. Stefanovic, B. Stewart. The Advanced Interdisciplinary Research Laboratory: A Student Team Approach to the Fourth-Year Research Thesis Project Experience. In: *Journal of Chemical Education*. 91 (1): 655-661, 2014.
- [32] N. Teixeira Filho. Estudo de Impacto do uso de Metodologias Ágeis na melhoria do planejamento e acompanhamento do processo de ensino e aprendizagem em sala de aula. [Dissertação de Mestrado]. Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE), Universidade Estadual do Ceará (UFCE). Fortaleza – CE. 2012.
- [33] A. Gkritsi. Scrum Game: An Agile software management game. [Dissertação de Mestrado] School of Electronics and Computer Science, University of Southampton. Southampton – Inglaterra. 2011.
- [34] G. Rodriguez, A. Soria, M. Campo. Virtual Scrum: A teaching aid to introduce undergraduate software engineering students to Scrum. In: *Computer Applications in Engineering Education*. 23 (1): 147-157, 2015.
- [35] J. A. Valente. Diferentes usos do Computador na Educação. In: *Computadores e Conhecimento: repensando a educação*. 2ª Ed. NIED, Unicamp. Campinas – SP. 1998
- [36] C. Wangenheim, R. Savi, A. Borgatto. SCRUMIA—An educational game for teaching SCRUM in computing courses. *Journal of Systems and Software*. 86 (10) 2675-2687, 2013.
- [37] M. Paasivaara, V. Heikkilä, C. Lassenius, T. Toivola. Teaching Students Scrum using Lego Blocks. In: *Proceedings of the 36th International Conference on Software Engineering*. Páginas 382-391, 2014.
- [38] Battistella, Paulo E. Wangenheim, Christiane G. Carvalho, Osório P. Ensinar a Gerência de Equipes em Disciplinas de Gerência de Projetos de Software. *Revista Brasileira de Informática na Educação*. 21 (1): 16-22, 2013.
- [39] The Startup Machine. Validation Board. <https://www.leanstartupmachine.com/validationboard/>. 2014.
- [40] G. A. Moore. Crossing the Chasm: Marketing and Selling Technology Project. Collins Business Essentials Series. Revised Edition. Harper-Collins. New York, NY, 2009.
- [41] E. Derby. Agile Retrospectives: Making Good Teams Great. Kindle Edition. Pragmatic Bookshelf. Frisco, TX. 2006.
- [42] M. Cohn. User stories applied: For agile software development. Kindle Edition. Addison-Wesley Professional. Boston, MA, 2004.
- [43] L. Hohmann. Innovation Games: Creating Breakthrough Products Through Collaborative Play. 1st Edition. Addison-Wesley Professional. Boston, MA, 2006.
- [44] V. Mahnic, T. Hovelja. On using planning poker for estimating user Stories. In: *Journal of Systems and Software*. 85 (9): 2086-2095, 2012
- [45] C. Delgado, R. Toledo, V. Braganholo. Uso de Dojos no Ensino Superior de Computação. In: *Workshop sobre Educação em Computação (WEI)*. Páginas 1-10, 2012.
- [46] A. H. Neto, A. Direne. Uma abordagem dialógica para a aquisição de habilidades táticas em jogos educacionais. In: *Revista Brasileira de Informática na Educação*. 18 (1): 59-75, 2010.
- [47] M. Pimentel. Estudo de caso em sistemas colaborativos. In: *Sistemas Colaborativos*. 1ª Ed. Campus. Rio de Janeiro, RJ, 2012.
- [48] R. Toledo. Até onde vai a agilidade?. <http://www.knowledge21.com.br/ate-onde-vai-agilidade>, 2014.
- [49] D. S. Dias, M. F. da Silva. Como escrever uma monografia: manual de elaboração com exemplos e exercícios. 1ª Edição. Atlas. São Paulo, SP, 2010.
- [50] L. Adkins. Coaching Agile Teams: A Companion for Scrum Masters, Agile Coaches, and Project Managers in Transition. Addison-Wesley

Professional. Boston, MA, 2010.

- [51] K. Beck. Test-driven Development: By Example. Addison-Wesley Professional. Boston, MA, 2003.
- [52] Boston University. Getting Feedback from Students. <http://www.bu.edu/ceit/teaching-resources/start-stop-continue/>. 2014.
- [53] D. Leffingwell. Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise. 1st Edition. Addison-Wesley Professional. Boston, MA, 2011.
- [54] S. Harichandan, N. Panda, A. A. Acharya. Scrum Testing With Backlog Management in Agile Development Environment. In *International Journal of Computer Science and Engine and Engineering*. 2 (3): 187-192, 2014.
- [55] A. S. Amorim Júnior, C. A. R. S. Amorim. “Coaching” para equipes de Tecnologia da Informação. In *Sistemas de Informação & Gestão de Tecnologia*. 4 (1): 1-10, 2010.
- [56] A. F. Gomes Filho, C. F. C. de Resende, R. de Toledo. Usando Métodos Ágeis para Ensinar Métodos Ágeis. In *Anais do 5º Workshop Brasileiro de Métodos Ágeis (WBMA)*.. Páginas 1-12, 2014.