UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS RUSSAS

CURSO DE GRADUAÇÃO EM ENGENHARIA DE SOFTWARE

LUAN DHARLIN LEMOS DA SILVA

PROPOSTA DE PROCESSO ÁGIL PARA PROJETOS DE DESENVOLVIMENTO DE SOFTWARE

RUSSAS

6 de julho de 2018

LUAN DHARLIN LEMOS DA SILVA

PROPOSTA DE PROCESSO ÁGIL PARA PROJETOS DE DESENVOLVIMENTO DE SOFTWARE

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Software do Campus Russas da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia de Software.

Orientadora: Prof. Dr. Anna Beatriz dos Santos Marques

RUSSAS

6 de julho de 2018

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S581p Silva, Luan Dharlin Lemos da.
Uma proposta de processo ágil para projetos de desenvolvimento de software / Luan Dharlin Lemos da Silva. – 2018.
79 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Russas, Curso de Engenharia de Software, Russas, 2018.
Orientação: Profa. Dra. Anna Beatriz dos Santos Marques.

1. Scrum (Desenvolvimento de software). 2. Guia do conjunto de conhecimentos em gerenciamento de projetos (Guia PMBOK). 3. Desenvolvimento ágil de software . I. Título.

CDD 005.1

LUAN DHARLIN LEMOS DA SILVA

PROPOSTA DE PROCESSO ÁGIL PARA PROJETOS DE DESENVOLVIMENTO DE SOFTWARE

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Software do Campus Russas da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Engenharia de Software.

Aprovada em:

BANCA EXAMINADORA

Prof. Dr. Anna Beatriz dos Santos Marques (Orientadora)
Universidade Federal do Ceará (UFC)

Prof. Dr. Valeria Lelli Leitão Dantas
Universidade Federal do Ceará (UFC)

Prof. Ms. José Osvaldo Mesquita Chaves
Universidade Federal do Ceará (UFC)

“Seu trabalho vai preencher uma parte grande da sua vida, e a única maneira de ficar realmente satisfeito é fazer o que você acredita ser um ótimo trabalho. E a única maneira de fazer um excelente trabalho é amar o que você faz.”

(Steve Jobs)

RESUMO

Atualmente, é considerável o percentual de projetos de desenvolvimento de *software* que não são concluídos com sucesso, não atendendo às expectativas dos clientes ou contrariando as restrições impostas por eles. Um dos principais causadores destes problemas é o mau uso ou, até mesmo, a inexistência de práticas de gerenciamento de projetos. Este trabalho propõe um processo ágil para projetos de desenvolvimento de *software*, permitindo a execução, o monitoramento e o controle de projetos por equipes com pouca ou nenhuma experiência neste tipo de projeto. Este processo foi definido a partir de experiência do autor e de pesquisa bibliográfica, utilizando-se de práticas de diferentes abordagens para gerenciamento e execução de projetos de desenvolvimento, dentre eles o Guia Project Management Body of Knowledge (PMBOK) 6a edição, o *Scrum*, Feature-Driven Development (FDD) e Dynamic Systems Development Method (DSDM), aplicando e adaptando no que se faz necessário e obtendo, como resultado, uma mescla entre abordagens ágeis e tradicionais. A adoção deste processo em ambiente acadêmico permitiu que as equipe execute as atividades com maior controle e fluidez, concentrando-se na resolução de problemas e cumprimento de objetivos. Conclui-se que o processo de gerenciamento definido, facilita o o controle e a organização do trabalho, melhora a comunicação e maximiza desenvolvimento de habilidades e troca de conhecimentos, aprimorando o desempenho individual e da equipe como um todo.

Palavras-chave: Scrum (Desenvolvimento de software). Guia do conjunto de conhecimentos em gerenciamento de projetos (Guia PMBOK). Desenvolvimento ágil de software.

ABSTRACT

Currently, the percentage of software development projects that are not completed successfully, not meeting customer expectations or against the constraints imposed by them, is considerable. One of the main causes of these problems is the misuse or even the inexistence of project management practices. This work proposes an agile process for software development projects, allowing the execution, monitoring and control of projects by teams with little or no experience in this type of project. This process was defined based on the author's experience and bibliographic research, using practices from different approaches to the management and execution of development projects, including the PMBOK Guide 6th edition, Scrum, FDD and DSDM, applying and adapting what is necessary and obtaining, as a result, a mixture between agile and traditional approaches. The adoption of this process in an academic environment allowed the team to carry out activities with greater control and fluidity, concentrating on problem solving and meeting objectives. It is concluded that the defined management process facilitates the control and organization of work, improves communication and maximizes skills development and knowledge exchange, enhancing individual and team performance as a whole.

Keywords: Scrum (Software Development). Project Management Body of Knowledge Guide (PMBOK Guide). Agile software development.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

CASE	Computer-Aided Software Engineering
DAO	Data Access Object
DoD	Definition of Done
DSDM	Dynamic Systems Development Method
EAP	Estrutura Analítica do Projeto
FDD	Feature-Driven Development
GP	Gerente de Projeto
MPS.BR	Melhoria de Processos do Software Brasileiro
MVC	Model-View-Controller
N2S	Núcleo de Soluções em Software
OOPSLA	Object-Oriented Programming, Systems, Languages and Applications
PMBOK	Project Management Body of Knowledge
PMI	Project Management Institute
PO	Product Owner
RAD	Rapid Application Development
TAP	Termo de Abertura do Projeto
TCLE	Termo de Consentimento Livre e Esclarecido
UFC	Universidade Federal do Ceará
UX	User Experience
XP	Extreme Programming

SUMÁRIO

1 INTRODUÇÃO

Hoje em dia é praticamente impossível falar em desenvolvimento de software sem tratá-lo como um projeto, seja pelo seu dinamismo, complexidade ou diferentes expectativas. "Um projeto é um esforço temporário empreendido para criar um produto, serviço ou resultado exclusivo"(PMI, 2017). Outras características mensuráveis são que um projeto precisa ser planejado, executado e controlado, além de desenvolvido em etapas, de forma progressiva.

Segundo Vargas (2017), o gerenciamento de projetos é um conjunto de ferramentas que permite que a empresa desenvolva um conjunto de habilidades, incluindo conhecimento e capacidades individuais, respeitando restrições de tempo, custo e qualidade. Este gerenciamento é realizado através da aplicação e integração de processos e práticas de gerenciamento definidos pela organização ou pela pessoa responsável pelo projeto, permitindo que o mesmo seja executado de forma eficiente e eficaz.

O gerenciamento como conhecemos surgiu no meio do século XX, mais exatamente no início dos anos 60, quando foi formalizado como ciência. Teve início na indústria bélica e aeroespacial americana e logo em seguida foi adotado pela construção civil e por outras áreas da engenharia (MARTINS, 2010). Em 1969, um grupo de cinco profissionais de gestão de projetos nos EUA, se reuniu para discutir as melhores práticas e assim foi fundado o *Project Management Institute (PMI)* (PMI, 2017). Hoje, de acordo com o próprio PMI (2017), ele é uma das maiores associações para profissionais do gerenciamento de projetos no mundo, contando com cerca de 700.000 associados em quase 200 países.

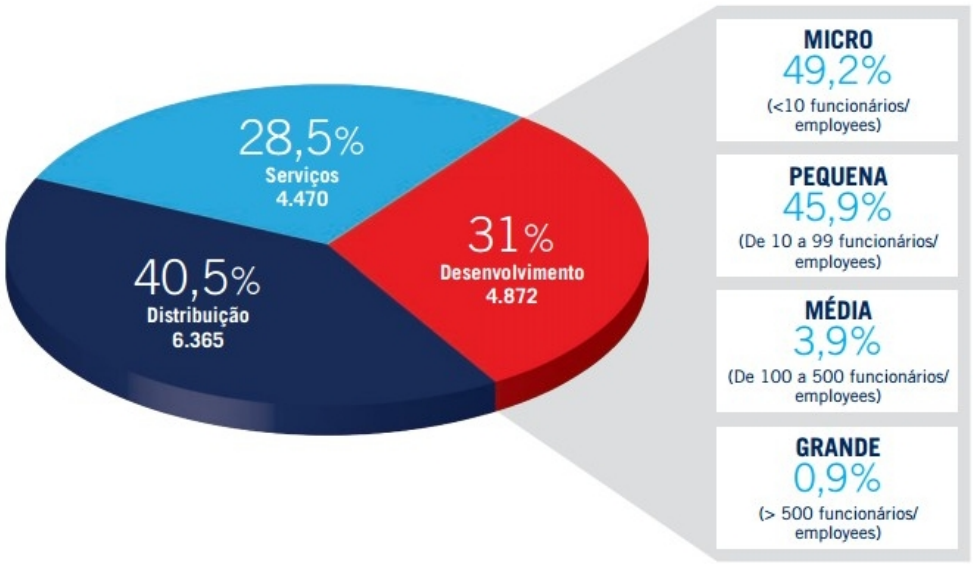
O Instituto organizou e mantém um conjunto de práticas de gerenciamento de projetos que servem como a base de conhecimento da gestão de projetos por profissionais da área. Esse conjunto de práticas está compilado no Guia PMBOK, atualmente em sua sexta edição.

Por outro lado, "em projetos complexos como o desenvolvimento de software, a incerteza e imprevisibilidade nos direcionam a utilizar uma abordagem orientada a valor, visto que a entrega final tem grande probabilidade de não ser a gerada nos primeiros momentos do projeto"(RIBEIRO; RIBEIRO, 2015b). Estas abordagens, também chamadas de ágeis, enfatizam as atividades que trazem o maior retorno de investimento ao cliente no menor intervalo de tempo possível. No entanto, para Abrahamsson *et al.* (2003), apesar dos métodos ágeis darem suporte à gerência, elas deixam lacunas que podem colocar o projeto em risco, fazendo-se necessário a adoção de práticas dos métodos tradicionais. Deste modo, fica claro a necessidade de utilizar-se uma abordagem mista, usando métodos tradicionais e ágeis um como complemento do outro.

1.1 Justificativa

De acordo com a ABES (2017), considerando-se as 4.872 empresas que atuam no desenvolvimento e produção de software no Brasil, cerca de 95% podem ser classificadas como micro e pequenas empresas, conforme ilustrado na Figura 1.

Figura 1 – Panorama brasileiro das empresas de TI



Fonte: http://central.abessoftware.com.br/Content/UploadedFiles/Arquivos/Dados%202011/ABES-Publicacao-Mercado-2017.pdf

Segundo O'Connor e Coleman (2007 apud BERNI, 2010), essas pequenas empresas de *software* precisam ser flexíveis e altamente dinâmicas, sempre buscando melhorar o *time-to-market*, que é o tempo que leva da demanda de um produto até sua entrega ao cliente. Por essa questão, estas empresas adotam um comportamento *ad hoc*, ou seja, alocam praticamente todos os seus recursos apenas no desenvolvimento, com o receio de burocratizar e engessar demais o seu processo.

Para Sommerville (2011), o sucesso de um projeto não é garantido por um bom gerenciamento, mas um empreendimento mau gerenciado pode ser, e geralmente é, o grande motivo de falhas. Assim sendo, a correta aplicação de práticas de gestão é fundamental para o sucesso de um projeto e de uma organização como um todo.

Devido à grande diversidade de abordagens existentes, saber qual delas utilizar ou até mesmo como utilizá-las torna-se um grande desafio para as organizações, principalmente para aquelas de pequeno porte e recém surgidas. Assim, devido a sua alta adaptabilidade, os métodos ágeis podem ser uma boa alternativa para gerenciamento de projetos de tais empresas

ou ainda podem ser um complemento para métodos orientados a planos. "Equipes ágeis de sucesso estão produzindo *software* de maior qualidade que atendem melhor às necessidades do usuário, com maior rapidez e com menor custo do que equipes tradicionais"(COHN, 2011). Ainda para o autor, empresas que adotam um processo tendo como base o *Framework Scrum* estão tendo grandes vantagens além das supracitadas, como maior previsibilidade e controle do projeto. Segundo Špundak (2014), projetos de desenvolvimento de *software* têm a necessidade de combinar abordagens ágeis e tradicionais.

1.2 Escopo do trabalho

Este trabalho propõe um processo de gerenciamento ágil de projetos de *software* mesclando práticas ágeis e tradicionais, com o intuito de apoiar equipes inexperientes a concluírem os projetos. Este processo é baseado no Guia de Pŕaticas Ágeis feito em conjunto pelo PMI e pela Aliança Ágil, que reúne as melhores práticas de de metodologias ágeis existentes como *Scrum* e DSDM.

Este trabalho não fará uma cobertura de todas as abordagens de gerenciamento, sejam elas orientadas a planos ou orientadas a valor e tampouco fará um comparativo entre abordagens. O escopo do trabalho refere-se a proposta de um processo misto bem como sua adoção e resultados alcançados. O mesmo foi aplicado por equipes em sala de aula da disciplina de Processos de *Software* nos final do segundo semestre de 2018.

1.3 Objetivos

O objetivo geral deste trabalho é estabelecer um processo ágil para gerenciamento de projetos de *software* no qual equipes com pouca ou nenhuma experiência possam concluir seus projetos com sucesso. Os objetivos específicos deste trabalho são:

- Propor um processo para gerenciamento de projetos de *software*;
- Documentar este processo de gerenciamento para uso em projetos reais;
- Realizar a implantação desse processo, apontando os benefícios e as dificuldades encontradas pelas equipes, bem como as lições aprendidas;
- Tornar a equipe de projeto mais eficiente e eficaz, tendo como consequência a conclusão bem sucedida dos projetos;
- Incorporar cada vez mais os princípios e práticas ágeis no dia-a-dia das pessoas envolvidas.

1.4 Metodologia

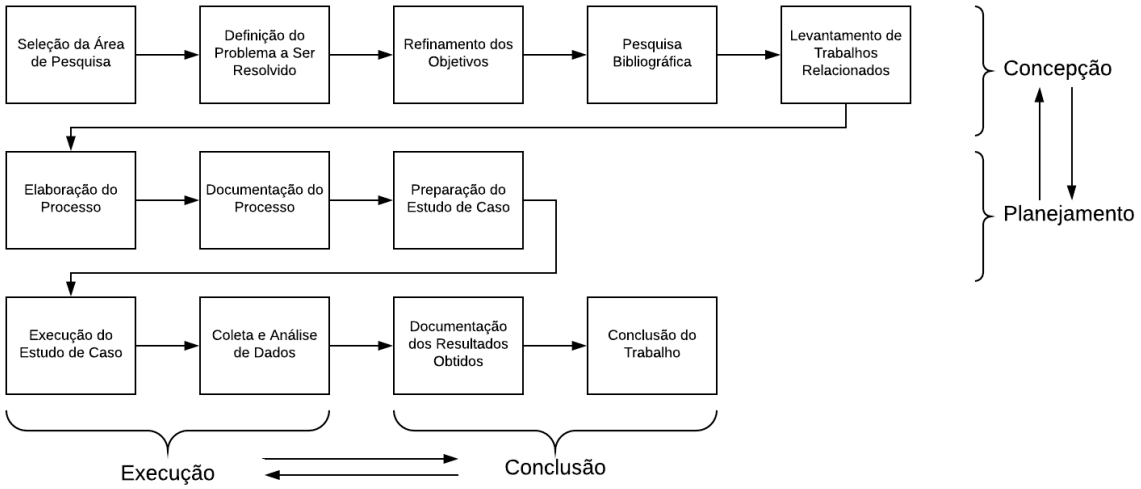
A metodologia utilizada neste trabalho foi o Estudo de Caso, cujo objetivo é o estudo empírico e aprofundado de um contexto específico, tendo caráter censitário, exploratório e prospectivo. Segundo Oliveira (2011) estudo censitário é aquele cuja população é pequena, podendo assim utilizá-la por completo como amostra.

Inicia-se pela seleção da área de pesquisa e pela definição do problema a ser resolvido, seguido posteriormente pelo estabelecimento dos objetivos do trabalho. A partir disso é realizada uma revisão bibliográfica sobre gerenciamento de projetos, processos de desenvolvimento de *software* e metodologias ágeis.

Tomando como base a problemática, os objetivos e a revisão bibliográfica, foram elencadas as práticas e as metodologias que melhor se adequavam ao problema em questão. Também foram identificados trabalhos relacionados a este estudo, observando o que eles tinham a contribuir, os resultados alcançados e o que poderia ser feito de diferente dentro do contexto de estudo.

A partir destes estudos, houve a experimentação de práticas em um contexto real para elaboração do processo para gerenciamento e execução de projetos, com o que se julgou melhor e mais adequado para o contexto de pequenos projetos de desenvolvimento de *software*. Isso permitiu avaliar os resultados do processo sob circunstâncias reais da gerência de projetos de desenvolvimento. A Figura 2 ilustra a metodologia utilizada.

Figura 2 – Metodologia do Trabalho



Fonte: Elaborado pelo Autor

Junto ao processo de elaboração ocorreu a documentação desse processo, bem como a preparação de um estudo de caso posterior, que ocorreu em ambiente acadêmico com oito equipes com diferentes projetos. Este estudo de caso foi dividido em três passos principais, que são a preparação do estudo envolvendo explicação do processo, apresentação das técnicas e artefatos, criação de *templates* e tutoriais e a entrega de um Termo de Consentimento Livre e Esclarecido (TCLE), cujo modelo pode ser visto no Apêndice D . Durante e após a execução desse Estudo de Caso houve a coleta e análise dos dados gerados pelo mesmo. A partir desses dados foram obtidos resultados que foram documentados e usados para fazer a conclusão final sobre este trabalho. Vale ressaltar a natureza iterativa entre as fases de Concepção e Planejamento e entre as fases de Execução e Conclusão. Isso se faz necessário pelo empirismo deste trabalho, que é baseado na melhoria por experimentação.

1.5 Organização do trabalho

O presente trabalho está organizado em seis capítulos dispostos da seguinte forma:

- O primeiro capítulo já apresentado introduziu o trabalho contextualizando a problemática e apresentando a justificativa do trabalho. Este mesmo capítulo definiu os principais objetivos e a metodologia utilizada.
- O segundo capítulo aborda o referencial teórico que embasa o presente trabalho, tais como Guia PMBOK e Metodologias Ágeis.
- O terceiro capítulo traz os trabalhos relacionados a este, dando uma breve descrição sobre cada um e o que os difere deste presente trabalho.
- O quarto capítulo trata da abordagem proposta, contextualizando o processo de elaboração e outros aspectos do estudo de caso. Traz o processo proposto dividido em fases e seus respectivos subprocessos, tendo estes últimos suas respectivas entradas meios e saídas, que serão explicadas brevemente à medida que aparecem no texto.
- O quinto capítulo mostra os resultados alcançados juntamente com o cronograma de atividades realizadas.
- O sexto capítulo traz as conclusões obtidas com o estudo de caso e faz menção a trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

De acordo com Kerzner (2005), a excelência no gerenciamento de projetos só é alcançada com um processo repetitivo. Nesta linha de pensamento, Tsui *et al.* (2013) defende que nenhuma metodologia específica será aplicável a todos os projetos e a todas as organizações e, portanto, o processo de decisão precisa levar em consideração as características do projeto, bem como a cultura organizacional. Assim sendo, diversas abordagens devem ser estudadas e aplicada a que melhor se encaixar no contexto ou ainda pode-se fazer uso da união de algumas abordagens, usando as práticas, processos e atividades no que for conveniente. Neste capítulo, serão tratados algumas abordagens para gerenciamento e desenvolvimento de projetos.

2.1 Guia PMBOK 6a edição

Uma destas abordagens é a proposta pelo PMI no Guia PMBOK, que é um conjunto de processos para gerenciamento de projetos que sugere boas práticas em todas as fases do projeto, do início ao fim (CRUZ, 2013). É importante salientar que o mesmo não é uma metodologia, ele próprio se define como uma referência de boas práticas para adaptação.

O Guia PMBOK define 49 processos, agrupados logicamente em 5 grupos, sendo eles o grupo de processos de Iniciação, de Planejamento, de Execução, de Monitoramento e Controle e de Encerramento. Além de grupos, os processos no Guia PMBOK também são caracterizados e separados por áreas de conhecimento. As 10 áreas e seus respectivos processos são:

- Gerenciamento da Integração do Projeto: identifica, define, combina, unifica e coordena os vários processos e atividades do gerenciamento de projetos. É a área responsável pela interligação dos processos das demais áreas. Em ambientes adaptativos essa integração é delegada a equipe, que decide como produtos e atividades devem se comunicar;
- Gerenciamento do Escopo do Projeto: inclui os processos necessários para assegurar que o projeto contemple todo o trabalho necessário, e apenas o necessário, para que o mesmo termine com sucesso. Em ambientes ágeis, o esforço é menor na definição de requisitos iniciais e maior na estruturação de meios de descoberta e refinamento de requisitos ao decorrer do projeto;
- Gerenciamento do Cronograma do Projeto: inclui os processos necessários para gerenciar o cronograma comparando com as linhas de base, que são referências para acompanhar

o andamento do projeto, para obedecer as restrições de tempo. Metodologias Ágeis utilizam ciclos curtos de trabalho, gerando assim um *feedback* mais rápido e preciso sobre o andamento do projeto. Esta é uma das práticas mais utilizadas no desenvolvimento de *software*;

- Gerenciamento dos Custos do Projeto: inclui os processos envolvidos em planejamento, estimativas, orçamentos, financiamentos, gerenciamento e controle dos custos, de modo que o projeto possa ser terminado dentro do orçamento aprovado. Para ambientes ágeis, o PMI (2017) recomenda a não tentar fazer cálculos de custo detalhado devido a grande variabilidade dos requisitos. Deve-se fazer fazer uma estimativa *just-in-time*, ou seja, somente naquilo que será feito no momento, na hora certa. Ainda de acordo com o PMI (2017) quando se fizer necessários ajustes e o orçamento for muito restrito, escopo e cronograma são ajustados com uma frequência maior para não desrespeitar as restrições de custo;
- Gerenciamento da Qualidade do Projeto: inclui os processos para incorporação da política de qualidade da organização com relação ao planejamento, gerenciamento e controle dos requisitos de qualidade do projeto e do produto, com o objetivo de atender as expectativas das partes interessadas. Para ambientes ágeis, simples atividades pré-datadas para inserção e averiguação da qualidade não são suficientes. O PMI (2017) sugere verificações frequentes, sempre que possível, nos pacotes mínimos de trabalho;
- Gerenciamento dos Recursos do Projeto: inclui os processos para identificar, adquirir e gerenciar os recursos necessários para a conclusão bem-sucedida do projeto. Apesar de terem as estimativas de recursos menos precisas, os ambientes ágeis têm o benefício da auto-organização, onde os membros da equipe maximizam a colaboração e consequentemente o uso de recursos físicos e humanos;
- Gerenciamento das Comunicações do Projeto: inclui os processos necessários para assegurar que as informações do projeto sejam planejadas, coletadas, criadas, distribuídas, armazenadas, recuperadas, gerenciadas, controladas, monitoradas e finalmente organizadas de maneira oportuna e apropriada. Ambientes altamente dinâmicos exigem comunicação rápida e precisa para que a equipe possa se adaptar à mudança de forma eficaz e eficiente, assim sendo, deve-se dar maior fluidez a comunicação, removendo barreiras burocráticas e documentação desnecessária. O PMI (2017) sugere a demonstração de artefatos de forma clara, para que todos tenham a mesma visão do andamento do projeto. Outro ponto

relevante levantado pelo PMI (2017) é a necessidade do agrupamento de todos os membros da equipe sempre que possível;

- Gerenciamento dos Riscos do Projeto: inclui os processos de condução do planejamento, identificação e análise de gerenciamento de risco, planejamento de resposta, implementação da resposta e monitoramento de risco em um projeto. É natural projetos de requisitos dinâmicos estarem sujeitos a maiores riscos. Para mitigar esse problema o PMI (2017) sugere maior frequência nas revisões e uma disseminação do conhecimento e habilidades entre os membros da equipe, para que todos entendam e consigam trabalhar em cima dos riscos. Outro ponto relevante é que nos ambientes adaptativos, o documento de requisitos é um artefato vivo, que pode, e deve, sofrer mudanças ao longo do projeto, facilitando assim, a incorporação de mudanças para mitigar e eliminar riscos;
- Gerenciamento das Aquisições do Projeto: inclui os processos necessários para comprar ou adquirir produtos, serviços ou resultados externos à equipe do projeto. Quando se trata de incorporação de externos ao projeto é um pouco mais complicado. O PMI (2017) sugere um compartilhamento de riscos e recompensas com terceiros que estendam a equipe de projeto;
- Gerenciamento das Partes Interessadas do Projeto: inclui os processos exigidos para identificar as pessoas, grupos ou organizações que podem impactar ou serem impactados pelo projeto.

Todos os processos, grupos de processos e áreas do conhecimento são organizados no Anexo A.

O Guia PMBOK funciona na maioria dos projetos, na maior parte do tempo (PMI, 2017). Isso significa que apesar de essas práticas não serem as únicas, elas podem ajudar eficientemente na diminuição dos problemas do dia a dia dos projetos, além de aumentar muito as suas chances de sucesso. Como cada projeto é único, faz-se necessário uma adaptação, ou seja, a seleção oportuna dos processos, das entradas, técnicas, ferramentas saídas e fases do ciclo de vida.

2.2 Metodologias Ágeis

De acordo com Prikladnicki (2014) os métodos ágeis surgiram na década de 90, introduzindo uma nova visão sobre como desenvolver *software*, sendo que o que diferencia estes dos outros métodos é o enfoque maior nas pessoas e não em processo. Segundo (ALMEIDA, 2017

apud LAANTI *et al.*, 2011) as pessoas que trabalham com esses métodos elencam benefícios como aumento da efetividade do desenvolvimento, ganho de qualidade do produto, maior organização do trabalho, aumento da autonomia dos times de desenvolvimento além de permitir encontrar erros e defeitos mais cedo.

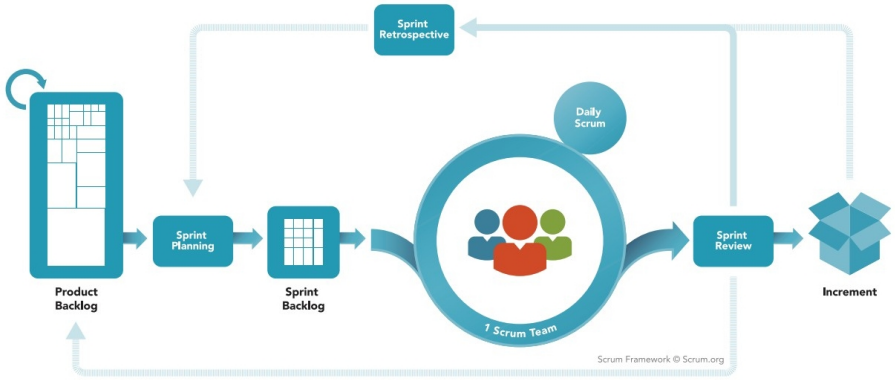
Em um estudo feito por Bermejo *et al.* (2014), no Brasil, as organizações desenvolvedores de *software* com as maiores taxas de sucesso são as mesmas que têm taxas mais elevadas de incorporação dos princípios ágeis. Isso não quer dizer que estes princípios são os únicos responsáveis pelo sucesso de projeto, podem até mesmo não ser os principais fatores de sucesso, porém, se fazem necessários para a conclusão bem sucedida de projetos de desenvolvimento de *software*.

2.2.1 Scrum

O conceito do *Scrum* e sua aplicabilidade para o desenvolvimento de software foi desenvolvido por Ken Schwaber e Jeff Sutherland na conferência Object-Oriented Programming, Systems, Languages and Applications (OOPSLA) em 1995 em Austin, Texas (SCRUM STUDY , 2013). Mas eles não foram os primeiros a usar o termo *Scrum* no ambiente empresarial. De acordo com Prikladnicki (2014), eles basearam-se em um estudo publicado em um artigo por Hirotaka Takeuchi e Ikujiro Nonaka, que defendiam que equipes pequenas e multidisciplinares produziam melhores resultados e então fizeram pela primeira vez uma analogia com esse termo.

O *Scrum* é um dos métodos ágeis mais conhecidos do mundo e sua aplicabilidade se dá em diversas áreas. Ele não resolve todos os problemas, mas organiza o trabalho para que a equipe tenha as melhores condições de solucionar estes problemas. O ciclo de vida *Scrum* está ilustrado na Figura 3.

Figura 3 – Ciclo de Vida Scrum



Fonte: https://www.scrum.org/resources/what-is-scrum

A partir do *Backlog* do Produto é feita uma seleção de requisitos a serem trabalhados de acordo com a prioridade atribuída. Partindo disso, o Time *Scrum* realiza essas atividades em um ciclo com duração pŕe-determinada resultando em um incremento do produto. Vale ressaltar a diferença entre Time *Scrum* e Time de Desenvolvimento. Este último é composto pelos desenvolvedores do projeto, enquanto o primeiro é composto pelo Time de Desenvolvimento mais os papéis de *Scrum Master* e do Product Owner (PO).

Diferente do que geralmente acontece, *Scrum* não é uma sigla, e sim um termo herdado de uma jogada de um esporte chamado *rúgbi*, onde é imprescindível a participação de todos os membros do time de forma cooperativa para se obter êxito. Outro aspecto que deve ser ressaltado é que, embora muitos profissionais da área em questão coloquem o *Scum* como metodologia, segundo seus próprios criadores, ele é um *framework* (SCHWABER; SUTHERLAND, 2016). O que o caracteriza como *framework* é que ele fornece uma estrutura básica para adaptação e expansão, sem definir detalhadamente práticas específicas. Esta abordagem foi pensada levando em consideração que cada contexto tem suas variáveis e peculiaridades. Deste modo, não há como descrever um passo a passo universal a ser seguido. Por este motivo o *Scrum* pode ser aplicado em diferentes projetos, com diferentes complexidades nos mais variados contextos.

Torres (2013) o define como uma abordagem empírica baseada em controle de processos, que permite maior flexibilidade, adaptabilidade, produtividade e controle de escopo de projeto. Corroborando com este pensamento, Sabbagh (2013) diz: ”O *Scrum* é embasado no empirismo e utiliza uma abordagem iterativa e incremental para entregar valor com frequência e, assim, reduzir os riscos do projeto.” Os dois autores levantam a questão do empirismo, que é a chamada Teoria *Scrum* sobre o qual o *framework* é baseado, que será detalhada a seguir.

2.2.1.1 Teoria Scrum

Dizer que o *Scrum* se baseia no controle empírico de processos significa que ele é baseado na experiência adquirida com a aplicação e observação ao invés de em um plano inicialmente detalhado. De acordo com Schwaber e Sutherland (2016), autores do guia oficial do *Scrum*, três pilares apoiam a implementação de controle de processo empírico, são eles:

- Transparência : Aspectos relevantes devem estar visíveis aos responsáveis pelos resultados e deve-se garantir que todos tenham a mesma visão do que se trata. Por exemplo o *backlog* priorizado ou o Gráfico *Burndown* são artefatos que mostram de forma sucinta o progresso

do time de desenvolvimento.

- Inspeção: deve-se inspecionar os artefatos e o progresso para detectar desalinhamentos. Porém, esta inspeção não deve atrapalhar a própria execução das tarefas. Um bom exemplo é um quadro de tarefas visível a todos os *stakeholders*, possibilitando assim a inspeção por indivíduos externos ao time de desenvolvimento..
- Adaptação: qualquer aspecto que desvie do que foi acordado deve ser reajustado. Geralmente estes aspectos são identificados nos eventos *Scrum* que serão detalhados neste trabalho.

A Scrum Study (2013) trata o controle de processos empíricos como um princípio do *Scrum*, juntamente com outros cinco, que são:

- Auto-organização: o *Scrum* parte do mesmo pressuposto que a Teoria Y idealizada por Douglas McGregor na década de 60, que diz que as pessoas fazem o trabalho porque gostam do que fazem e não apenas por obrigação. Por este motivo usa-se o conceito de times auto-organizáveis, que permitem as pessoas explorarem suas principais habilidades. Mas isso não quer dizer que o time trabalhará da forma que quiser, processos obrigatórios ainda devem ser seguidos. Em outras palavras, é dado ao time de desenvolvimento o que fazer e o time decide como fazer.
- Colaboração: este princípio é muito mais do que apenas dividir tarefas. Entre membros do Time *Scrum* deve-se ter a consciência de que a tarefa que está sendo desenvolvida por um influencia diretamente na execução das tarefas dos demais. Outro fato importante é que um membro pode, e deve dar suporte ou até assumir em totalidade a tarefa de outro membro que está com dificuldade. Colaboração também diz respeito a interação entre o Time *Scrum* e os demais *stakeholders* do projeto, um traço marcante das metodologias ágeis. O *framework* traz consigo várias práticas que ajudam na implantação dessa colaboração, como por exemplo quadro de tarefas, painéis informativos e as chamadas *war room*, que seria colocar a equipe em um mesmo ambiente de trabalho e envolvê-los nas tomadas de decisão, induzindo à comunicação direta e face-a-face.
- Priorização baseada em valor: priorizar o trabalho a ser realizado não é novidade, até os modelos de gerenciamento mais antigos realizam esta atividade. O diferencial do *Scrum* está no critério de priorização, que trabalha a questão do escopo do produto de forma semelhante à metodologia DSDM quanto a necessidade de entrega daquilo que tem maior valor mais cedo. Essa priorização é feita pelo PO (papel que será definido mais adiante),

que após receber os requisitos de negócio do usuário, os transforma em estórias de usuário as quais posteriormente serão colocadas no *Backlog* Priorizado do Produto, levando em consideração, além do valor de negócio, os riscos envolvidos e as dependências entre atividades.

- *Time-boxing*: Este é um conceito-chave do *Scrum*, que trata o tempo como uma das restrições mais importantes. Ele indica que práticas e atividades dentro do *framework* devem ter uma duração previamente acordada, o trabalho a ser realizado deve estar definido e os objetivos devem ser claros. De acordo com Scrum Study (2013), este princípio assegura que os envolvidos não gastem tempo a mais ou a menos que o necessário em um trabalho específico. Alguns eventos e práticas *time-boxing* serão apresentadas mais adiante.
- Desenvolvimento iterativo com priorização de funcionalidades: como visto anteriormente, o *Scrum* tem como objetivo a entrega de valor o quanto antes. Para que isso seja possível é necessário um desenvolvimento de produto que trabalhe aos poucos, com ciclos repetitivos. Esta prática é chamada de desenvolvimento iterativo, algo comum para todas as abordagem mencionadas neste trabalho, inclusive o Guia PMBOK, que usa a nomenclatura de de ondas sucessivas. Basicamente trata-se de pegar partes ou funcionalidades do produto que tem maior valor para o cliente e planejar uma forma de entregar essas partes ou funcionalidades no menor intervalo de tempo possível. Deste modo, além de incorporar mudanças com maior facilidade, gera um *feedback* constante, que auxilia no alinhamento do produto desenvolvido com o produto esperado.

”A estrutura do *Scrum* busca aproveitar a maneira como as equipes realmente trabalham, dando a elas as ferramentas para auto-organizar-se e, o mais importante, aprimorar rapidamente a velocidade e a qualidade de seu trabalho” (SUTHERLAND, 2016). Tal estrutura fornece um ciclo de vida básico, com alguns papéis, práticas e artefatos, deixando liberdade para as equipes incorporarem o que mais acharem necessário.

2.2.1.2 Papéis

Para Schwaber e Sutherland (2016) os times *Scrum* são auto-organizáveis e multifuncionais e foram assim projetados para aperfeiçoar a criatividade, flexibilidade e produtividade. Um time multifuncional consegue resolver os problemas que lhes são dados sem necessitar de ajuda de membros externos e pelo fato de serem auto-organizáveis, podem decidir internamente

qual a melhor forma de solucionar estes problemas. Os três papéis existentes em um time *Scrum* são:

- PO: ou dono do produto, como próprio nome já diz, é a pessoa responsável por entender do negócio do produto, decidindo o que será feito para maximizar a entrega de valor. É a voz do cliente no Time *Scrum*.
- *Scrum Master*: é um facilitador, o responsável pela correta aplicação dos conceitos *Scrum* propiciando um ambiente para conclusão do projeto com sucesso. Ele quem garante o alinhamento com o *framework* e ajuda membros externos a entender o que está sendo feito.
- Time de desenvolvimento: são os profissionais que fazem o trabalho para gerar uma entrega funcional ao cliente. Um ponto importante é o tamanho do time de desenvolvimento, que deve ser composto de três a nove pessoas, pois menos que isso pode resultar em falta de habilidade para alguns problemas e mais do que isso gera um certo grau de complexidade para um controle de processos empírico. Não se deve confundir time de desenvolvimento com Time *Scrum*, sendo que este último é composto pelo time de desenvolvimento, PO e *Scrum Master*.

2.2.1.3 Eventos

Os eventos *Scrum* tem o intuito de estabelecer uma rotina, onde os eventos sempre acontecem no mesmo intervalo de tempo, não atrasando o desenvolvimento com reuniões e pausas desnecessárias. Uma característica desses eventos, além de serem *time-boxing* conforme explicado anteriormente, é que também são usados para inspecionar e adaptar o que for necessário. Os eventos definidos no *framework* são:

- *Sprint*: são ciclos repetitivos com duração entre uma e quatro semanas que contém outros eventos *Scrum* e resultam na maioria das vezes em uma entrega funcional ao cliente. Todo o trabalho de uma *Sprint* deve ser planejado, e durante sua execução apenas este trabalho deverá ser feito, nada a mais ou a menos.
- Reunião de Planejamento da *Sprint*: o trabalho a ser realizado no próximo ciclo é definido durante esta reunião. Para cada iteração há uma Reunião de Planejamento da *Sprint*. É neste evento que se decide o que será feito e como será feito.
- Reunião Diária: encontro diário com duração curta, de no máximo quinze minutos, com todos e somente os membros do time de desenvolvimento para levantar problemas e alinhar o trabalho. São ressaltadas questões como o trabalho realizado desde a última

Reunião Diária o que será feito por cada membro até a próxima Reunião e as dificuldades encontradas. Não deve ser visto como um relatório diário do trabalho, mas sim uma forma de alinhar o progresso do trabalho em relação ao objetivo da *Sprint*.

- Revisão da *Sprint*: reunião que acontece ao fim de cada *Sprint* que tem como participantes o time *Scrum* junto com os outros *stakeholders*. O intuito desse evento é inspecionar o que foi entregue, adaptar o *backlog* do produto e planejar mudanças futuras.
- Retrospectiva da *Sprint*: ocorre logo após a Revisão da *Sprint*, mas somente com o time *Scrum*. Trata-se de uma auto-avaliação, onde os envolvidos levantam pontos a ser melhorados e destacam pontos fortes a serem mantidos.

2.2.1.4 Artefatos

Os resultados dos trabalhos do time *Scrum* geram os chamados Artefatos *Scrum*. Os principais são:

- *Backlog* do Produto: é a lista dos requisitos, assim como todas as informações para seu entendimento. Como requisitos sempre mudam ou novos aparecem, é comum chamar o *Backlog* de artefato vivo, que está em constante alteração;
- *Backlog da Sprint*: é parte do *Backlog* do Produto que será trabalhado em uma *Sprint* para gerar uma entrega de funcionalidade. Depois de definido, somente o time de desenvolvimento pode alterar este artefato.
- Incremento: é a soma dos itens do *backlog* do produto concluídos. A cada *Sprint* um novo incremento é gerado;
- Definição de Pronto: ou Definition of Done (DoD), é um artefato que acorda os critérios para uma atividade ser considerada concluída. Tem o intuito de definir um vocabulário comum para que todos tenham uma visão do que é uma atividade pronta, se é simplesmente terminada, ou terminada, testada e validade e assim por diante;
- Estórias de Usuários: nada mais são que uma funcionalidade vista do ponto de vista de quem necessita desta funcionalidade, geralmente o usuário final. Deve deixar claro para quem, o que, e por que esta funcionalidade está sendo criada. Assim, os desenvolvedores poderão entender melhor e mais rapidamente as necessidades do cliente ou usuário;
- Gráfico *Burndown*: é uma representação visual do trabalho planejado e do trabalho realizado da *Sprint*, e deve ser atualizado e acompanhado em tempo real, e não apenas no fim do ciclo;

- *Taskboard*: ou quadro de tarefas, é usado mas não é originário do *Scrum*, e sim do sistema Kanban da Toyota. Trata-se de uma demonstração visual do progresso de trabalho, destacando o que foi feito e o que ainda falta fazer.

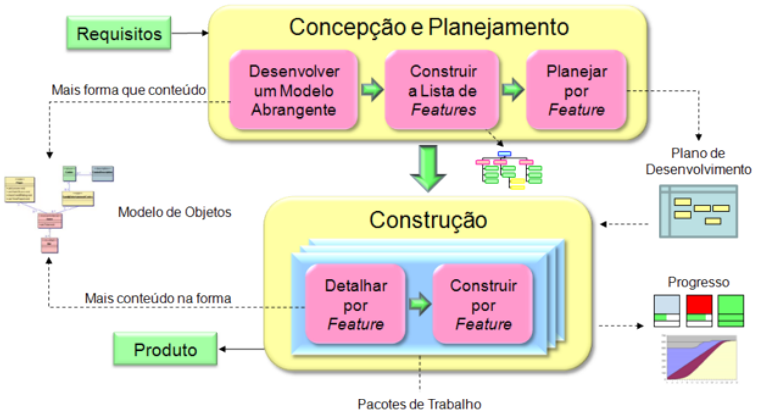
2.2.2 *FDD - Feature Driven Development*

O Desenvolvimento Guiado por Funcionalidade é uma metodologia ágil de gerenciamento e desenvolvimento de *software*, sendo composto basicamente por duas fases e cinco processos. Na fase de Concepção e Planejamento é feita uma triagem nos requisitos focada em funcionalidades. Na fase de Construção acontece um maior detalhamento das funcionalidades, especificando mais exatamente como o trabalho deve ser feito. Os cinco processos são:

- Desenvolver um modelo abrangente;
- Construir a Lista de funcionalidades;
- Planejar por Funcionalidade;
- Detalhar por Funcionalidade;
- Construir por Funcionalidade.

A estrutura do FDD está ilustrada na Figura 4.

Figura 4 – Estrutura do FDD



Fonte: https://commons.wikimedia.org/wiki/File:Fdd.png

O FDD recomenda uma série de boas práticas oriundas da Engenharia de Software, como:(RIBEIRO; RIBEIRO, 2015a)

- Modelagem de Domínio do Objeto: a equipe deve entender e explicar o domínio do problema a ser resolvido;
- Propriedade individual (código): cada área do código deve ter um único proprietário, assim se garante a consistência, desempenho e integridade conceitual;

- Times dinâmico: pequenas equipes, formadas dinamicamente de acordo com as características de cada projeto.
- Inspeções: são revisões no código que ajudam a garantir a boa qualidade;
- Gerenciamento da Configuração: envolve controle de alterações e gerenciamento de versões do código fonte, onde versões anteriores podem ser resgatadas;
- Construções Regulares: entregas pequenas e constantes, o time incrementa o produto com a nova funcionalidade desenvolvida logo após sua conclusão;
- Visibilidade: controle e acompanhamento do progresso e dos resultados baseado em funcionalidades desenvolvidas, facilmente acessadas.

2.2.3 DSDM - Dynamic Systems Development Method

A Metodologia de Desenvolvimento de Sistemas Dinâmicos (DSDM) é uma metodologia de desenvolvimento iterativa e incremental (RIBEIRO; RIBEIRO, 2015a), tendo como base o envolvimento constante do usuário. Apesar do nome desenvolvimento, a DSDM é uma estrutura ágil muito usada para o gerenciamento e entrega de projetos, ajudando a fornecer resultados de forma rápida e eficaz para projetos de diferentes tamanhos sendo uma extensão do Rapid Application Development (RAD), que é uma metodologia focada em projetos com grande restrição financeira e temporal.

Um ponto importante desta metodologia, e o principal diferencial em relação a maioria das outras abordagens ágeis, é a necessidade de acordo de uma base para o projeto, como premissas, restrições e requisitos principais, algo muito parecido com o Termo de Abertura do Projeto (TAP) do Guia PMBOK. De acordo com o DSDM Consortium (2014) deixar claro e acordar as bases para o projeto a partir das perspectivas de negócios reduz a probabilidade de surpresas desagradáveis em projetos.

Segundo Sbrocco (2011), esta metodologia é dividida em cinco fases sendo elas:

- Estudo de viabilidade: como o próprio nome diz, esta é a fase de verificar se o projeto pode ser desenvolvido usando a DSDM;
- Estudo de negócio: é a fase que busca identificar as características do projeto, bem como as regras de negócio e todos os processos envolvidos;
- Modelo de interação funcional: fase de planejamento do trabalho a ser realizado a cada ciclo de trabalho, construindo protótipos para avaliá-los e aproximar o produto real do planejado;

- Projeto e construção da iteração: é a fase onde o sistema é realmente construído, e os requisitos são validados pelos clientes e usuários;
- Implantação: etapa de colocar o sistema em uso real bem como entrega de manual do usuário e treinamento caso necessário.

A DSDM descreve sua filosofia como sendo: "O melhor valor comercial surge quando os projetos estão alinhados para objetivos comerciais claros, quando entregam frequentemente e envolvem a colaboração de pessoas motivadas e capacitadas"(DSDM CONSORTIUM , 2014). Essa filosofia baseia-se em uma versão modificada do Princípio de Pareto, defendendo que que 80% do valor de uma aplicação pode ser entregue pela implementação bem sucedida de 20% dos requisitos mais críticos (PRESSMAN, 2011). E para decidir quais requisitos farão parte destes 20% é feita uma priorização por meio da técnica MoSCoW, que pode ser aplicada a requisitos, atividades, tarefas ou casos de uso, embora geralmente seja aplicadas a histórias de usuários. De acordo com Office of Government Commerce (2011), aplicando esta técnica, consegue-se, por exemplo, classificar e priorizar funcionalidades com base no seu valor de negócio. Onde as letras significam:

- M - (*Must have*) deve ter: são aquelas funcionalidades que são imprescindíveis para a aplicação. Sem elas, o sistema não funcionará;
- S - (*Should have*) deveria ter: são aquelas funcionalidades de grande importância no escopo do projeto, de grande valor para o cliente, mas que se não forem desenvolvidas não comprometerão o sistema;
- C - (*Could have*) poderia ter: são funcionalidades que seria bom tê-las, mas que agregam pouco valor para o cliente. Geralmente são implementadas ao final dos projetos caso haja tempo e recursos;
- W - (*Won't Have this time*) não terá agora: são funcionalidades que não serão desenvolvidas por enquanto, pois não agregam nenhum valor.

De acordo com Ribeiro e Ribeiro (2015a), além da filosofia, a metodologia tem como base oito princípios que a sustentam. Para a DSDM Consortium (2014), não seguir à risca os princípios coloca em risco a filosofia da DSDM e diminui as chances do projeto ser bem sucedido. Esses princípios são:

- Concentre-se na necessidade do negócio: toda tomada de decisão deve estar alinhada com os objetivos do projeto, que por sua vez não é a finalidade, e sim o meio para se chegar a um produto ou resultado esperado por uma organização.

- Entregar a tempo: cumprir o cronograma é muitas vezes o fator de sucesso mais importante de um projeto. A melhor forma de atender a este princípio é tornar o trabalho *Time-Boxing*, ou seja, ciclos com duração e trabalho pré-determinados.
- Colaborar: equipes que trabalham de forma cooperativa são mais eficientes e eficazes do que um grupo de indivíduos que trabalham de forma independentes. Essa colaboração faz com que a equipe tenha uma maior compreensão sobre o escopo do projeto e do produto e uma maior velocidade. Além disso, gera um compartilhamento de conhecimentos e habilidades, gerando algo similar o que defende a Teoria *Gestalt*, que as equipes sejam superior à soma de suas partes.
- Nunca comprometa a qualidade: no início do projeto faz-se um acordo a respeito da qualidade a ser entregue, ou seja, as necessidades e expectativas dos clientes e usuários que serão sanadas. Todo o trabalho deve ser direcionado para alcançar esse nível de qualidade.
- Construa incrementalmente de acordo com as bases estabelecidas: a principal diferença da DSDM em relação às outras metodologias ágeis é a preocupação com o estabelecimento de bases sólidas antes do início do desenvolvimento. Deve-se fazer uma análise de negócio de necessidades, de premissas e restrições que afetarão o projeto. Depois que essas bases estiverem estabelecidas, a metodologia defende a prática de entrega incremental, que fornecerá resultados reais em pequenos intervalos de tempo. De acordo com o DSDM Consortium (2014), a entrega incremental gera um *feedback* para uso em iterações subsequentes e gera benefícios reais para o cliente antes mesmo da entrega do produto final.
- Desenvolva iterativamente: a DSDM, como no *Scrum*, adota o desenvolvimento iterativo, progredindo através de tentativas sucessivas de refinamento, para prover um ambiente de colaboração entre equipe de desenvolvimento e os outros *stakeholders*, incentivando um *feedback* constante. Como dito anteriormente, este tipo de processo insere mudanças facilmente, permitindo que a equipe desenvolva uma solução mais próxima do que o negócio realmente precisa. Como tem uma abordagem adaptada no Princípio de Pareto, a DSDM inicia pela implementação bem sucedida dos requisitos que agreguem maior valor de negócio ao cliente, distribuindo as demais funcionalidades nas iterações subsequentes.
- Comunique-se de forma contínua e clara: a má comunicação ou a ausência dela, geralmente ocasiona a falha de projeto. Em muitos casos, a burocracia e documentação excessiva atrasam as equipes de desenvolvimento. Esta abordagem prega por uma comunicação

face-a-face, sem barreiras burocráticas.

- Demonstrar controle: como já dito, o projeto precisa ser controlado. Este controle deve estar presente em todo projeto e além disso, precisa ser demonstrado. Em abordagens orientadas à planos, isso é facilmente visível. Mas em abordagens ágeis não é tão simples. A DSDM talvez seja a mais descritivas das metologias ágeis, pois estabelece bases que sustentarão o planejamento e a execução do projeto .

3 TRABALHOS RELACIONADOS

Há um número considerável de trabalhos relacionados a este trabalho. De modo geral, o que o difere de grande parte, é que a maioria baseia-se em materiais datados do começo do século, onde ocorreu a explosão das metodologias Ágeis. Este, por sua vez, foi embasado bibliograficamente por trabalhos no estado da arte, como por exemplo o Guia PMBOK já na sua última versão, lançada durante o período de escrita deste trabalho.

Outra grande diferença é que os outros trabalhos, na sua maioria, fazem um estudo bibliográfico, seja exploratório ou comparativo entre metodologias ágeis e tradicionais. Já este propõe uma abordagem mista e sua aplicação prática, em um estudo de caso real. Os principais trabalhos serão descritos a seguir.

3.1 An Iterative and Agile Process Model for Teaching Software Engineering

Em seu artigo, Alfonso e Botia (2005) usam um processo ágil em um ambiente acadêmico, mas para ensino de Engenharia de Software, explicando o passo a passo, interferindo juntamente com os alunos e comparando com processos anteriormente estudados. Diferente deste trabalho, que deixa com os alunos a responsabilidade da adoção do processo, para maior fidelidade dos resultados da sua implantação. Outra diferença a ser mencionada é o fato do processo deste autor ser direcionado para projetos de desenvolvimento de *software*, seja acadêmico ou profissional.

3.2 Modelagem de um Novo Processo de Desenvolvimento de Software com Base em Metodologias Ágeis

No seu artigo, Costa (2010) faz um trabalho que se assimilou bastante com o presente trabalho. Ambos propõem um novo processo ágil para desenvolvimento de *software*. Mas há alguns pontos de divergência bem marcantes, como por exemplo, o presente trabalho não se limita a adoção de práticas ágeis, mas utiliza-se do Guia PMBOK, classificado como práticas tradicionais.

A principal diferença entre os trabalhos é o fato do autor do presente trabalho ter colocado sua proposta de processo em execução, organizando um estudo de caso em projetos acadêmicos. A elaboração do processo também pode ser destacada como um diferencial, pois o supracitado baseou-se na literatura, não utilizou-se de experimentação, como fez o autor do

presente trabalho.

3.3 Fatores de escolha entre metodologias de desenvolvimento de software tradicionais e ágeis

O nome deste artigo é sugestivo. Nele, Almeida (2017) faz uma revisão bibliográfica e busca identificar os fatores de cada tipo de metodologia que possam ser mais adequado do que outros. Ele elenca fatores habilitadores e inibidores relacionados aos cenários formados por características da organização, das pessoas e dos projetos de desenvolvimento de software para a escolha da aplicação de metodologias ágeis ou de metodologias tradicionais. Em seguida, o autor faz um estudo de caso através de questionários respondidos por profissionais da área, e de seus conhecimentos e experiências ágeis e tradicionais.

No trabalho supracitado o autor sugere uma mescla entre práticas de abordagens ágeis e tradicionais baseados na necessidade do projeto e da organização em questão. Porém o seu escopo termina aí, em uma sugestão.

O presente trabalho, diferentemente, após estudo bibliográfico e empírico, define um processo com o que foi julgado práticas mais adequadas, e mostra os resultados reais de sua aplicação em ambientes controlados, dentro do meio acadêmico.

3.4 Mixed agile/traditional project management methodology – reality or illusion?

No seu artigo, Špundak (2014) faz uma análise comparativa e um estudo sobre a viabilidade da adoção do Guia PMBOK e o *framework Scrum* em um mesmo projeto. O autor defende o uso de mais de uma abordagem, pois apenas uma metodologia pode não ser capaz de suprir todas as necessidades do projeto. No entanto, este é um trabalho teórico, sem aplicações de um estudo de caso prático. Este presente trabalho propõe uma abordagem para ser aplicado em um contexto real e suscetível a aferições. Outra diferença é que trata de mais abordagens, e não se limita ao Guia PMBOK e ao *Scrum*.

3.5 Method for adaptation and implementation of agile project management methodology

No seu artigo, Rasnacis e Berzisa (2017) sugerem um método para implantação de uma metodologia ágil genérica, partindo do pressuposto que o sucesso de qualquer atividade

depende do fator humano.

O autor divide este processo de incorporação de uma abordagem ágil em quatro fases, que são: Fase de Preparação; Fase de Análise dos Funcionários; Fase de Seleção da Metodologia Ágil; Fase de Adaptação da Metodologia Ágil e a Fase de Implementação da Metodologia Ágil. E em seguida é feito um estudo de caso, que mostra que este processo de adaptação funciona para equipes que tem entre dez e dezesseis membros.

O artigo citado propõe um passo a passo genérico para se identificar e aplicar uma metodologia ágil qualquer em uma organização. Por outro lado, o presente trabalho propõe uma abordagem que mescla práticas de diferentes abordagens resultando em um processo para adoção.

3.6 An Agile Process Model for Inclusive Software Development

No seu trabalho, Bonacin *et al.* (2009) propõe um processo para desenvolvimento de *software* mostrando a instanciação do processo, bem como seus resultados e discussões a respeito de possíveis ajustes, baseado em práticas ágeis, algo muito similar com este trabalho. Porém, o trabalho citado, que se autodenomina inclusivo, é focado acessibilidade e usabilidade de sistemas, enquanto o presente trabalho é mais genérico, buscando apenas dar suporte para equipes inexperientes concluírem seus projetos com sucesso.

3.7 Gerenciamento ágil de projetos: proposta e avaliação de método para gestão de escopo e tempo

Trata-se de uma tese de mestrado onde Conforto (2009) faz um estudo sobre os princípios ágeis e uma rápida comparação com abordagens tradicionais. Posteriormente é proposta uma metodologia ágil de gerenciamento de tempo e escopo para criação de produtos inovadores. Foi aplicado a duas empresas, uma com um único grande projeto complexo e outra com sete projetos pequenos. Após a implantação foi feito um estudo estatísticos para saber a viabilidade e benefícios encontrados com a implantação do gerenciamento ágil.

A grande diferença entre a tese citada e o presente trabalho, é que este último é voltado exclusivamente para pequenos projetos de desenvolvimento de software. Outras diferenças que valem ser ressaltadas é que este trabalho não faz uma comparação entre abordagens ágeis e tradicionais, e sim propõe uma mescla onde práticas ágeis cobrem os pontos fracos das

tradicionais e vice-versa. Além disso, o presente trabalho não aborda apenas do escopo e cronograma como a tese citada, mas também as partes interessadas, riscos e qualidade entre outras áreas do projeto.

3.8 Scrum com Equipes Inexperientes

Em seu trabalho, França *et al.* (2010) realizaram um estudo de caso a fim de mostrar que o *Framework Scrum* também pode ser usado por equipes inexperientes. O estudo foi realizado em um projeto de desenvolvimento de *software* fora do ambiente acadêmico. Porém, toda a equipe era formada por alunos do curso de Sistemas da Informação, todos sem experiência profissional. O projeto foi dividido em cinco *Sprints* e contou com a participação de um representante do cliente que assumiu o papel de PO. Ao final do estudo, França *et al.* (2010) concluíram que o *Scrum* pode sim ser usado corretamente por equipes inexperientes.

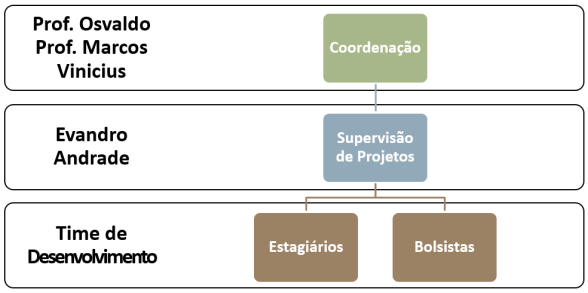
Apesar das várias semelhanças , o autor do presente trabalho não se limitou ao uso do *Scrum*, e sim a elaboração de um processo tomando o *framework* como base juntamente com outras abordagens. Outra diferença mensurável é a condução do estudo de caso, que no presente trabalho foi realizado com oito equipes, e não apenas uma, podendo obter, desta forma, uma resposta melhor sobre o processo proposto.

4 TRABALHO PROPOSTO

4.1 Experiência no N2S como base para definição do processo

O processo proposto foi elaborado enquanto o autor participava de um projeto no Núcleo de Soluções em Software (N2S), que é um centro de estágios da Universidade Federal do Ceará (UFC) Campus Russas, criada no primeiro semestre de 2017. No Núcleo, são desenvolvidos softwares para a comunidade acadêmica da UFC e a comunidade local. No período em que o autor fazia parte do Núcleo, este contava com um coordenador, um vice-coordenador, um supervisor e quatro equipes de projeto, conforme ilustrado na Figura 5.

Figura 5 – Estrutura do N2S



Fonte: Arquivos do N2S

As equipes tinham entre três e cinco membros, sendo estes alunos do campus local, dos cursos de Engenharia de Software e Ciência da Computação. Pelo fato das equipes serem formadas única e exclusivamente por estagiários, falta experiência a essas equipes, principalmente se tratando de gerenciamento e de desenvolvimento profissional de software.

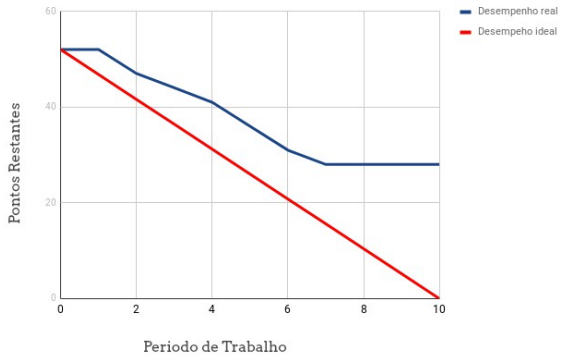
O autor deste trabalho integrou a primeira equipe do N2S, passando assim por todas adversidades que pequenas empresas recém surgidas passam. De acordo com as características e necessidades do Núcleo, foi definido que seria usado o ciclo de vida do *Scrum* para a condução dos projetos. Sua implantação, por sua vez, não é algo trivial, e como processo empírico, leva tempo para correta adequação.

Diante disso, levando em consideração o que deu certo e o que poderia ser melhorado e o *feedback* da equipe do projeto, o autor documentou um passo a passo para auxiliar equipes com pouca ou sem nenhuma experiência a conduzirem com maior facilidade seus projetos. Este trabalho não define o único e melhor processo para gerenciamento de projetos de desenvolvimento de *software*, mas sim uma alternativa levantada pelo autor dentro das variáveis encontradas.

4.1.1 Resultados na elaboração do processo

Observa-se como resultado real da aplicação de práticas ágeis e tradicionais durante a elaboração do processo, um aumento gradativo no desempenho do time ao decorrer do projeto. Isto ocorre devido ao empirismo das práticas ágeis, ou seja, aperfeiçoamento baseado na observação de experimentos. Isso é melhor ilustrado na Figura 6, na Figura 7 e na Figura 8.

Figura 6 – Gráfico *Burndown* da *SPRINT* 1.



Fonte: Elaborado pelo Autor

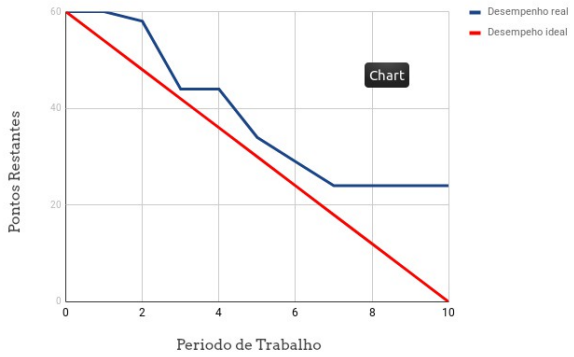
Na Figura 6, o eixo horizontal representa os períodos de trabalho no N2S, que são formados por duas horas cada um. Esta foi a melhor forma de representação de tempo de trabalho encontrada, visto que havia dias da semana onde não ocorria trabalho, devido a regimento interno da Organização onde ocorreu o estudo de caso, e, além do mais, os dias de trabalho não tinham a mesma duração.

O eixo vertical representa as pontos restantes para conclusão da *Sprint* como planejado. A linha vermelha indica o trabalho ideal a ser realizado. Dividindo-se a quantidade de pontos pela quantidade de períodos de trabalhos, pode-se obter o trabalho diário necessário, e apenas o necessário para a conclusão do trabalho destinado à *Sprint*. A linha azul indica o trabalho realizado pelo time de desenvolvimento.

Percebe-se na Figura 6 que em momento algum o time conseguiu atingir a linha de trabalho ideal, estando sempre com o trabalho em atraso, e, em certo momento, ficando estagnada no desenvolver da *Sprint*. Isso ocorreu por dois motivos principais, a alocação demasiada de trabalho e a má pontuação das atividades do *Sprint Backlog*. É comum que equipes inexperientes se superestimem ou subestimem o trabalho a ser realizado, acarretando nos problemas supracitados.

Ao comparar a Figura 6 com a Figura 7 é possível notar uma pequena melhora de

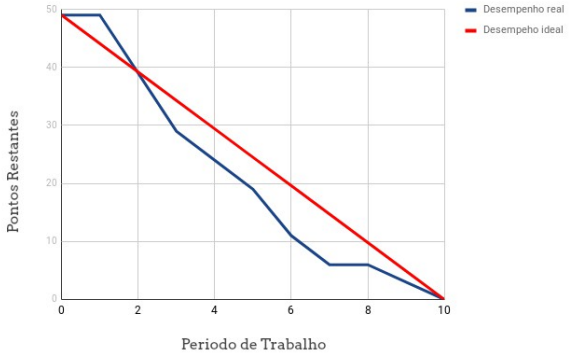
Figura 7 – Gráfico *Burndown* da *SPRINT* 2.



Fonte: Elaborado pelo Autor

desempenho do time de desenvolvimento, havendo uma maior aproximação entre as linhas de trabalho ideal e trabalho realizado, porém, em nenhum dos dois casos o trabalho do *Sprint Backlog* foi totalmente concluído. Essa melhoria gradativa a pequenos passos se repetiu ao decorrer das *Sprints*.

Figura 8 – Gráfico *Burndown* da *SPRINT* 3.



Fonte: Elaborado pelo Autor

A Figura 8 mostra o aprimoramento do time, que manteve o trabalho realizado quase sempre a frente do trabalho ideal. Com a experiência adquirida nas *Sprints* anteriores, a equipe resolveu problemas recorrentes com maior facilidade, pode conhecer melhor suas limitações e todavia estimar com maior exatidão o esforço necessário para concluir as atividades, melhorando seu desempenho em relação aos ciclos de trabalhos anteriores.

Nas Reuniões de Retrospectivas das *Sprints* do projeto trabalhado no N2S, foram levantados pelo time diversos pontos fortes a serem mantidos e fracos a serem repensados. Dos pontos positivos, a estrutura de equipes pequenas e auto-gerenciáveis foi mencionado por todos os membros. Como pontos a serem melhorados, a unanimidade foi a ausência de suporte externo

ao desenvolvimento, sobrecarregando o time com atividades relacionadas a processo e produto. Este cenário se deu devido as restrições do Núcleo, que, pelo pequeno número de pessoas nas equipes, não podia contar com pessoas exclusiva para serem responsáveis por estas atividades. Estes responsáveis poderiam ser os papéis de *Scrum Master* e o PO respectivamente. Deste modo, o processo proposto foi moldado baseado no *framework Scrum*, utilizando seus papéis bem como artefatos e eventos.

Através de formulários, cujo modelo é mostrado no Apêndice C, 75% dos membros da equipe alegaram que seria uma boa prática identificar e documentar riscos no planejamento de cada *Sprint*. Todavia, um subprocesso do Guia PMBOK com esse intuito foi adicionado à proposta do processo. A mesma porcentagem afirmaram haver também a necessidade do estabelecimento de bases antes de se iniciar o projeto. Desta forma, após busca na literatura, verificou-se tal prática pertencente a metodologia DSDM, tendo a mesma, práticas e princípios incorporados ao processo. Outro ponto que fez com que esta metodologia fosse utilizada, foi a fato dela ser um pouco mais prescritiva que o *Scrum*, se tornando assim uma forma menos arriscada de incorporar a agilidade.

Com base na experiência adquirida, no *feedback* da equipe e na revisão da literatura, esta primeira etapa resultou em uma proposta de processo que foi definido e documentado para o estudo de caso posterior.

4.2 O Processo

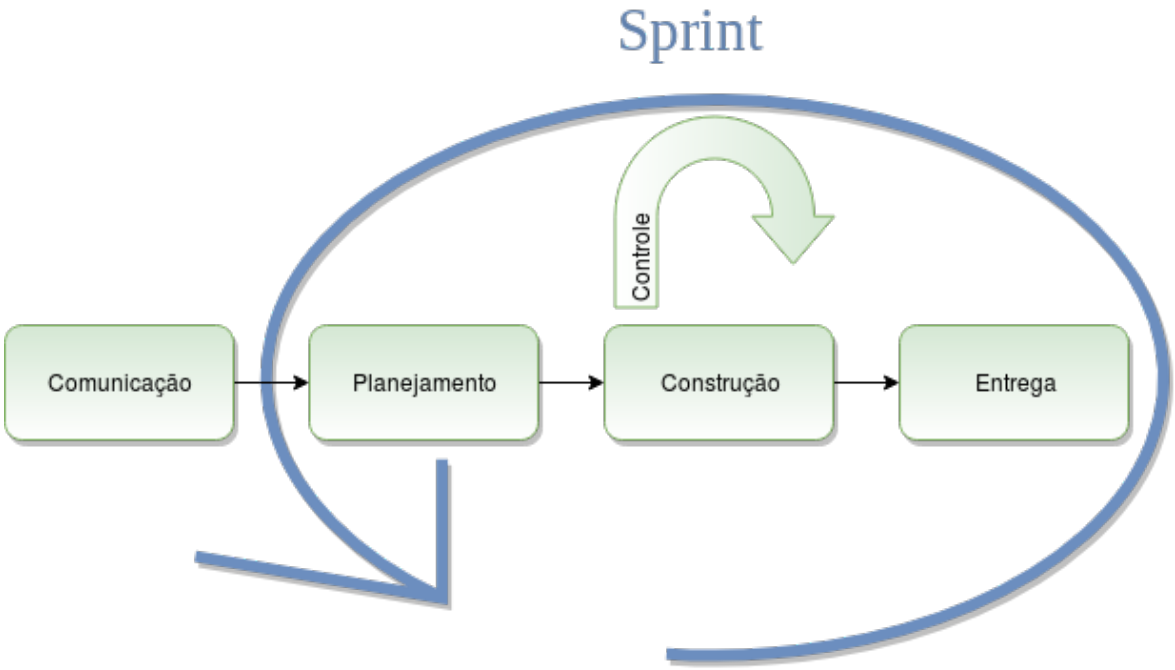
Como visto, existem diversas abordagens para gerenciamento e desenvolvimento, sejam tradicionais ou ágeis, dirigidas a planos ou dirigidas à valor, cada uma com suas características. Portanto, não há uma solução universal que se encaixe em todos os contextos, todos os projetos de todas áreas e complexidades. Assim cabe ao responsável pelo gerenciamento de projeto a escolha das abordagens adotadas e a sua aplicação e adaptação.

Este trabalho define um processo para gerenciamento de projetos de desenvolvimento de *software* tendo como base o *framework Scrum*, ou seja, foi criado usando como esqueleto o ciclo de vida do mesmo, assim como seus eventos, papéis e artefatos e é agregado com práticas de outras abordagens como o Guia PMBOK.

No processo proposto não há o papel do Gerente de Projeto (GP). Cohn (2011) deixa claro que a inexistência deste papel não exclui o trabalho e a responsabilidade do mesmo, apenas há uma transferência para outros papéis.

O processo proposto define um passo a passo de como o projeto deve ser encarado e é dividido em cinco fases genéricas da Engenharia de *Software*, que Pressman (2011) define sendo as fases de Comunicação, Planejamento, Construção, Controle e Entrega, que estão dispostas conforme Figura 9.

Figura 9 – Ciclo de Vida do Processo



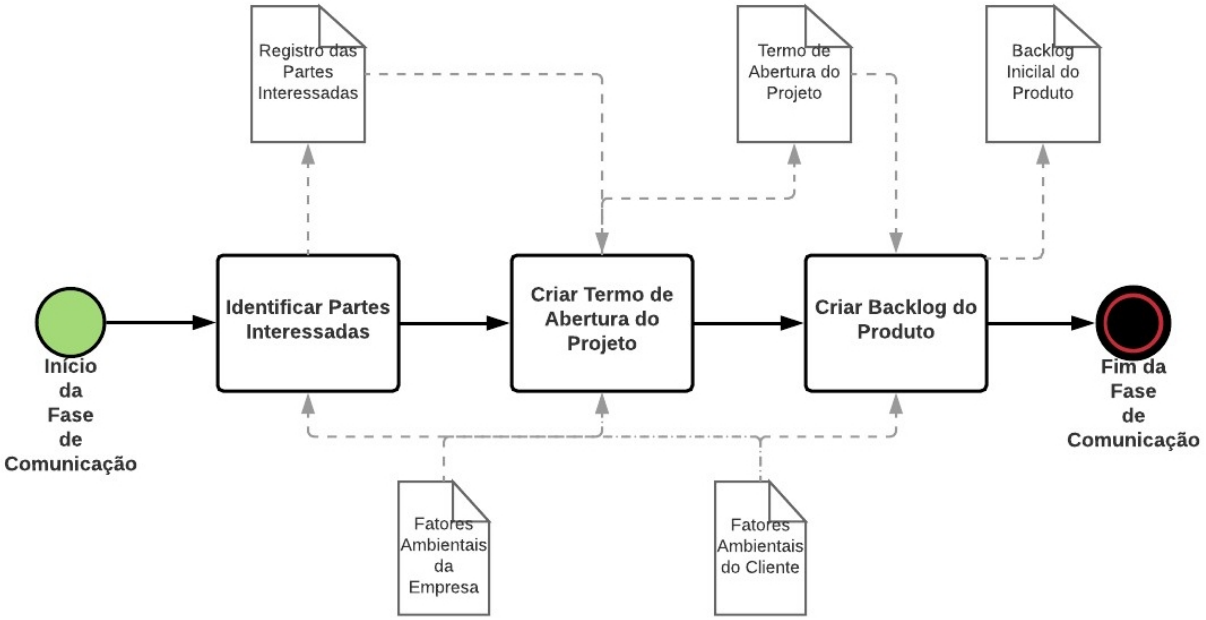
Fonte: Elaborado pelo Autor

As fases são distribuídas linearmente, com exceção das fases de Construção e Controle que ocorrem em paralelo. Uma vez concluídos os processos da fase de comunicação, inicia-se as outras fases na sequencia mostrada na Figura 9. Estas fases tem natureza iterativa, ou seja, se repetem para cada ciclo de trabalho. Cada fase será apresentada juntamente com seus subprocessos a seguir, bem como as entradas, saídas e os meios, que são as ferramentas e técnicas utilizadas para gerar as saídas de cada processo.

4.2.1 Fase de Comunicação

Esta fase contém os processos e atividades relacionados em providenciar o início do projeto, criando uma visão inicial do produto a ser desenvolvido, baseando-se no primeiro processo do FDD, "Desenvolver um modelo abrangente". São identificados as partes interessadas, os requisitos básicos e as restrições impostas, alinhando o projeto com toda a organização. Os processos da fase de Comunicação são mostrados na Figura 10.

Figura 10 – Processos da Fase de Comunicação



Fonte: Elaborado pelo Autor

4.2.1.1 Identificar Partes interessadas

Processo de definir e documentar as pessoas e organizações que possam influenciar ou serem influenciadas pelo projeto. Aqui também deve-se ocorrer a formação do Time *Scrum*. Esse processo é de responsabilidade do PO e conta com a participação do *Scrum Master*, do Time de desenvolvimento e da gerência.

Suas Entradas são os Fatores Ambientais do Cliente (estrutura da organização a qual o produto final se destina, bem como seus usuários finais) e os Fatores Ambientais da Empresa (estrutura da organização onde o projeto será desenvolvido, bem como disponibilidade de recursos físicos e humanos, além de normas internas e legislação vigente). Os meios utilizados neste processo são a Coleta de dados (que pode ser feita através de entrevistas e *brainstorming*), a Análise Dados e Opinião Especializada. Esta ultima trata-se de buscar auxílio a pessoas com conhecimento e habilidades sobre o problema em questão, sendo a técnica mais usada no Guia PMBOK. Como sugeria Steve Jobs: "Concentre-se naquilo que você é bom, delegue todo o resto."

Este processo tem como saída o Registro das Partes Interessadas, que é um documento com a identificação dos envolvidos no projeto, bem como seu poder de decisão e influência. Em muitos casos, é um documento próprio, mas o autor deste trabalho prefere colocá-lo dentro do TAP, que será explicado a seguir.

4.2.1.2 Criar TAP

Processo do Guia PMBOK, da área de integração, que formaliza e inicia oficialmente o projeto, definindo características fundamentais do projeto, como justificativa do projeto, marcos, riscos, restrições entre outros. É de responsabilidade do PO e conta como participantes a gerência, Time de Desenvolvimento e *Scrum Master*. O uso deste processo é importante pois serve de base para as tomadas de decisão do projeto. Como trata-se de um equipe sem experiência, faz-se necessário o estabelecimento de bases solidas. Tal prática é recomendada pela DSDM.

A principal entrada desse processo é a Reunião *Kick-off* do Projeto, que serve para alinhar as expectativas dos envolvidos no projeto, pois mesmo que todos os pontos de dúvida já tenham sido discutidos, pode ser que a equipe, organização e clientes tenham ideias diferentes do que será feito. Outras entradas são o Registro das Partes Interessadas e os Fatores Ambientais da Empresa e do Cliente, incluindo sua missão e visão.

Os meios são Opinião especializada e Reuniões, sempre tomando cuidado para esta última não alongar-se demais ou que venha a se repetir muitas vezes, deixando assim o processo cansativo e improdutivo. E a saída, como o próprio nome do processo já diz, é o TAP, cujo modelo pode ser visto no Apêndice A.

4.2.1.3 Criar Backlog do Produto

Este é o processo de preparação do ambiente para adição de informações sobre o produto a ser desenvolvido. Mesmo não se tendo conhecimento de todos os requisitos inicialmente, já existe uma visão sobre o produto a ser desenvolvido e alguns dos requisitos à alto nível definidos do TAP. Conforme o FDD, estes requisitos são listados no *Backlog* do Produto sem muito detalhamento, algo que será feito na próxima fase, juntamente com informações necessárias e documentos do projeto. É de responsabilidade do PO e conta com a participação do *Scrum Master* e do Time de Desenvolvimento para preparar o *Backlog* do produto. Vale ressaltar que o processo mencionado está na fase de Comunicação, todavia, aqui são colocados apenas os requisitos iniciais definidos no Termo de Abertura. Levantamento e documentação de outros requisitos serão tratados a seguir na fase de Planejamento.

Este processo tem como entradas o TAP e Fatores Ambientais da Empresa. Seus meios são Opinião Especializada, Reuniões e *expertise* do time *Scrum.Expertise* é de suma importância em metodologias ágeis, visto que se as mesmas se baseiam na melhoria constante

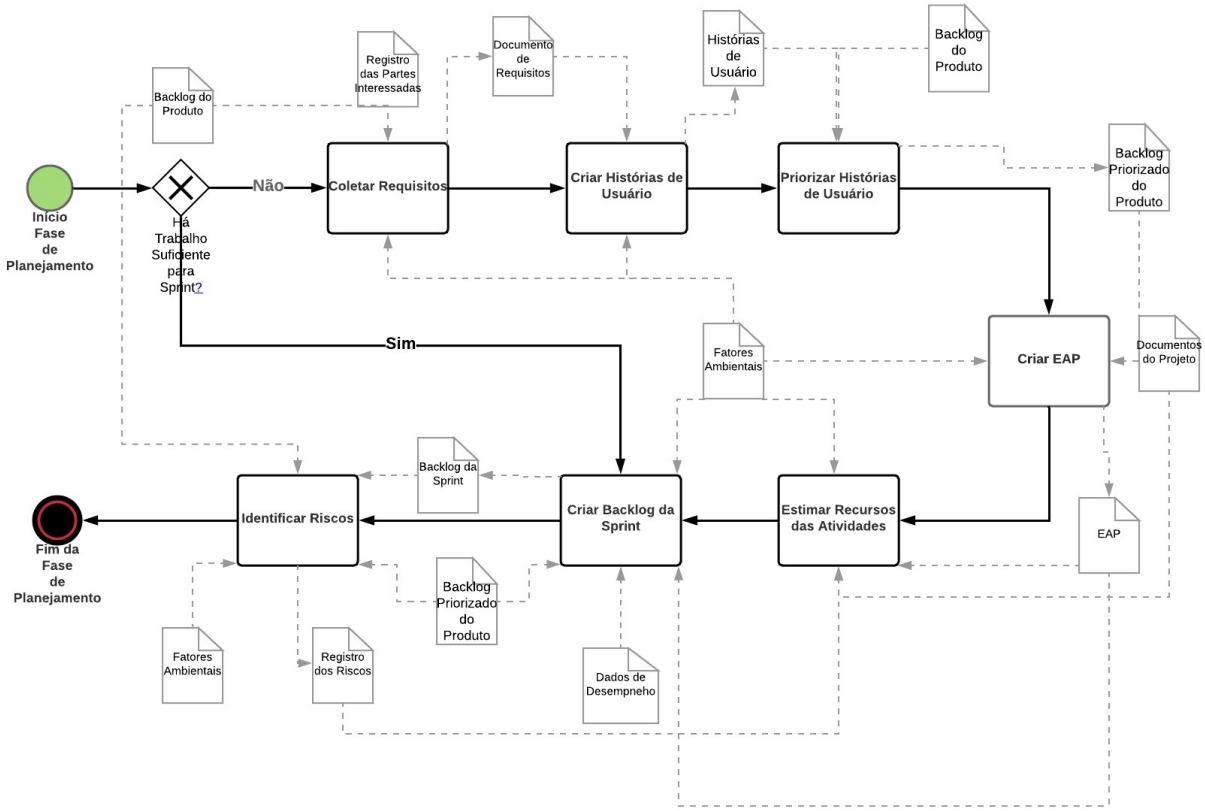
adquirida com a experimentação. Experiências anteriores dos membros do time com atividades similares devem ser exploradas.

A saída deste processo é o *Backlog* Inicial do Produto.

4.2.2 Fase de Planejamento

É a fase que contém os processos e atividades relacionados em definir como o projeto deve ser tratado, para guiar a equipe na abordagem do cronograma, escopo, partes interessadas, riscos, qualidade, recursos e outros aspectos relevantes ao projeto. Os processos da fase de Planejamento são mostrados na Figura 11.

Figura 11 – Processos da Fase de Planejamento



Fonte: Elaborado pelo Autor

4.2.2.1 Coletar Requisitos

Segundo o (PMI, 2017) este processo trata-se da definição e documentação das necessidades das partes interessadas. Mas diferentemente do que ocorre em muitos casos, no estudo em questão, este não é um processo realizado apenas no início do projeto, e sim, em paralelo ao desenvolvimento. Do primeiro ao último dia os requisitos são levantados, refinados e

documentados, pois após uma entrega ou reunião por exemplo, os *stakholders* podem solicitar uma mudança ou inclusão de funcionalidades, acarretando em um novo requisito. Aliás, esse tipo de comportamento é incentivado pelo time, como o pensamento que mudanças são bem vindas.

Este processo é de responsabilidade do PO e conta com a participação de todas as outras partes interessadas identificadas. Em alguns casos, após o fim de um ciclo de trabalho, já há trabalho suficiente para se iniciar um novo, assim, como segue princípio do FDD de detalhar por funcionalidade a ser desenvolvida, durante o planejamento deste ciclo não haverá a execução deste processo. Esta regra serve para os processos "Criar Histórias de Usuário", "Priorizar Histórias de Usuário", "Criar Estrutura Analítica do Projeto (EAP)"e "Estimar Recursos das Atividades."

As entradas necessárias são o *Backlog* do Produto, Fatores Ambientais e Registro das Partes Interessadas. Os meios incluem Opinião Especializada, Coleta (*Brainstorm*, entrevistas, questionários e observação) e Análise de Dados. Através destes meios sem obtém o Documento de Requisitos, sendo este um dos principais artefatos do Projeto. Como trata-se de um processo para ambiente adaptativo, este Documento de Requisitos não se trata apenas de um papel com uma lista imutável de funcionalidades a serem desenvolvidas, e sim de funcionalidades adicionadas ao *Backlog* do Produto tais como quaisquer informações relevantes para o Projeto.

4.2.2.2 Criar histórias do usuário

Este é o processo de transformar os requisitos identificados e adicionados ao *Backlog* do Produto em Histórias de Usuário. É de responsabilidade do PO e conta com a participação dos usuários e clientes. Se faz necessário devido a maior facilidade de entendimento por parte dos desenvolvedores. Para uma boa escrita destas histórias foi utilizada a técnica INVEST, onde cade letra representa uma característica que a história deve ter:

- Independente: histórias devem permitir que sejam implementas em qualquer ordem, sem dependência uma das outras, evitando assim gargalos no desenvolvimento;
- Negociável: o ponto forte das metodologias ágeis é a adaptabilidade, logo, essas histórias devem ser modificáveis;
- Valiosa: devem agregar valor, não adianta criar e dividir histórias se em certo ponto elas não contribuam em nada para o alcance dos objetivos do projeto;
- Estimável: deve ser passível de estimativas de esforço e tempo necessário. Vale salientar

que são estimativas, não há necessidade de rotular um valor exato;

- Pequena: do inglês *small*. É importante pois quanto menor a história mais fácil e precisa será sua estimativa. Necessita de cuidados para não ferir a regra do Valiosa.
- Testável: só se confia em algo que se possa efetuar testes. Se o usuário não encontrar uma testabilidade na história, é possível que ela não esteja clara o suficiente.

As entradas para este processo são o Documento de Requisitos e os Fatores Ambientais da Empresa. Os meios são Opinião Especializada, *expertise* dos membros do time e *Templates* (modelos de história de usuário usadas na literatura). A saída são as Histórias de Usuário, juntamente com seus critérios de aceitação adicionadas ao *Backlog* do Produto.

4.2.2.3 Priorizar histórias de usuário

Após essas histórias criadas, por onde começar o desenvolvimento? Qual funcionalidade deve ser implementada primeiro? Para responder a estas perguntas que várias abordagens adaptativas recomendam a priorização de histórias de usuário. No *Scrum*, essa tarefa de priorização é responsabilidade do PO e conta com a participação dos clientes e usuários. Este processo usa como entrada o *Backlog* do produto com as Histórias de Usuário.

Outro fator que o *Scrum* é que, por ele não ser uma metodologia, ele não diz como priorizar estas histórias. O autor deste trabalho, depois de vasta pesquisa bibliográfica, identificou como melhor e mais aplicável meio de priorização a Técnica *MoSCoW*, onde as letras maiúsculas significam:

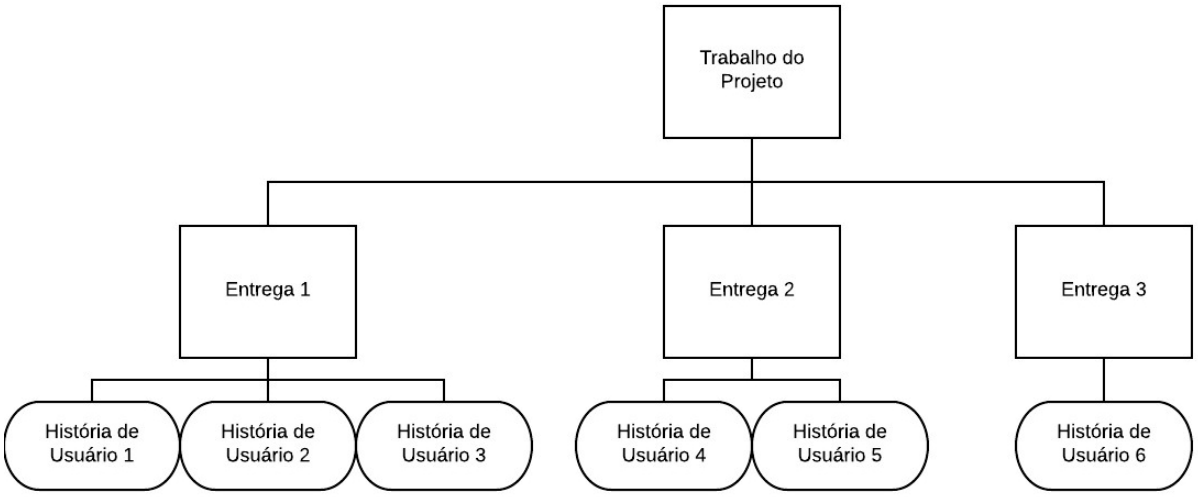
- M: Deve ter, do inglês *Must Have*. São aquelas principais funcionalidades, que sem elas o sistema não funcionaria;
- S: Deveria ter, do inglês *Should Have*. São aquelas funcionalidades importantes, sem elas o sistema não deixa de funcionar, porém perde grande parte da qualidade;
- C: Poderia ter do inglês *Could Have*. Funcionalidades que agregariam algum valor se implementadas, mas que sua ausência não é sentida.
- W: Não terá agora, do inglês *Won't Have for Now*. Funcionalidades que não agregam valor em nada. Se possível, são deixadas para o futura para serem discutidas e implementadas em caso de disponibilidade de recursos.

Este processo tem como saída o *Backlog* Priorizado do Produto. Artefato usado para os processos seguintes.

4.2.2.4 Criar EAP

De acordo com o PMI (2017) este é o processo de decompor o trabalho do projeto em componentes cada vez menores. Neste caso, basicamente, é dividir Histórias em atividades. É de responsabilidade do PO e do Time de desenvolvimento, que dividem o trabalho do projeto em histórias e estas em atividades de nível técnico respectivamente. Este processo ajuda na hora de delimitar escopo e estimar duração, esforço e custo paras as atividades, além de identificar riscos com maior facilidade. A figura 12 ilustra uma forma de representação da EAP. Vale salientar que esta não é a única e nem melhor forma de representação, podendo variar de acordo com cada equipe ou projeto.

Figura 12 – Exemplo de EAP



Fonte: Elaborado pelo Autor

Todo trabalho do projeto é dividido em entregas e posteriormente em histórias de usuário. Os retângulos representam trabalho descomposto e as formas arredondadas pacote mínimo de trabalho a ser executado para gerar resultados. Como dito anteriormente esta representação pode, e deve sofrer variações. Uma delas é a decomposição das histórias em atividades a nível técnico (codificação, prassim pode-se obter melhor detalhamento das atividades, dividindo melhor o trabalho e estimando melhor os recursos necessários. É de responsabilidade do Time de Desenvolvimento, ele a apenas ele decidirá a melhor forma para cumprir com os critérios de aceitação das histórias de usuário. Conta ainda com a participação do *Scrum Master* e do PO.

Este processo utiliza como entradas o *Backlog* do Produto contendo as Histórias de Usuário, Documentos e os Fatores Ambientais da Empresa. Os meio utilizados foram

Decomposição, Opinião Especializada e *Expertise* do time. Esta última tem grande importância, pois os desenvolvedores já sabem em maior parte as atividades necessárias, isso ocorre porque geralmente conhecem a arquitetura do *software* (Padrões Model-View-Controller (MVC), Data Access Object (DAO), etc). As saídas deste processos foram a EAP e o Dicionário da EAP, que fornece informações sobre os componentes da mesma, como detalhes das Histórias e seus critérios de aceitação.

4.2.2.5 Estimar Recursos das Atividades

O Guia PMI (2017) define este como sendo o processo de estimar todos os recursos necessários para a execução de cada atividade determinando a quantidade de material, pessoas e equipamentos. Mas isso pode tornar-se uma tarefa um pouco complicada, principalmente em ambientes adaptativos. Seguindo o que defende algumas metodologias ágeis, a estimativa deste trabalho se dará por esforço necessário para a conclusão das atividades.

Este processo é de responsabilidade do Time de Desenvolvimento e do *Scrum Master* e pode contar com a participação do PO para possíveis esclarecimentos de requisitos e outras informações sobre o produto.

As entradas para este processo são a EAP, Registro dos Riscos Identificados, Documentos do Projeto e Fatores Ambientais da Empresa Desenvolvedora. O principal meio utilizado é o *Planning Poker*, ferramenta do *Scrum* para pontuar atividades, podendo-se utilizar de Ferramentas Computer-Aided Software Engineering (CASE). Outros meios auxiliares são Análise de Dados, Opinião Especializada e Estimativas Análoga, que são aquelas baseadas em situações similares anteriores. A saída gerada por estes meios é a lista de Atividades Pontuadas, podendo assim, dar início a fase de construção do incremento ou produto.

4.2.2.6 Criar Backlog da Sprint

É o processo de alocar parte do *Backlog* do Produto para ser realizado em um ciclo de trabalho. É de responsabilidade do Time de Desenvolvimento, pois só ele mesmo consegue dizer o que realmente pode ser feito dada as limitações existentes. Conta ainda com a participação do *Scrum Master*. As entradas são o *Backlog* Priorizado do Produto, Fatores Ambientais da Empresa desenvolvedora, Lista de Tarefas da EAP, informações sobre duração da *Sprint* e Dados de Desempenho. O principal meio utilizado é a Reunião de Planejamento da *Sprint*, onde se decide pontos relevantes como quanto trabalho realizar e qual trabalho, caso haja empate em

prioridade. As saídas do processo são o *Backlog da Sprint* disposto em um Quadro de Tarefas. De acordo com Sabbagh (2013), com a utilização deste quadro, a informação chega aos olhos de quem é importante chegar sem haver um esforço significativo para buscá-la. É muito mais eficiente do que um programa de computador para gerenciamento de tarefas. Cada *Sprint* deve ter seu próprio Quadro de Tarefas, com as colunas “Por Fazer“, “Fazendo“ e “Feito“, além de outras caso necessário.

4.2.2.7 Identificar Riscos

Este é o processo de levantar os riscos positivos e negativos que podem afetar direta ou indiretamente o projeto bem como os planos para tratamento desses riscos. É de responsabilidade do Time de Desenvolvimento e do *Scrum Master* e conta ainda com a participação do PO, gerência, clientes e usuários.

Tem como entradas o *Backlog* do Produto, Fatores Ambientais da Empresa e Dados de desempenho. Seus meios são Opinião Especializada, Coleta e Análise de Dados, Habilidades Interpessoais e de Equipe, *Expertise* dos participantes e Reuniões, destacando-se a Reunião Diária. As saídas geradas são Registros dos Riscos e atualização no mesmo.

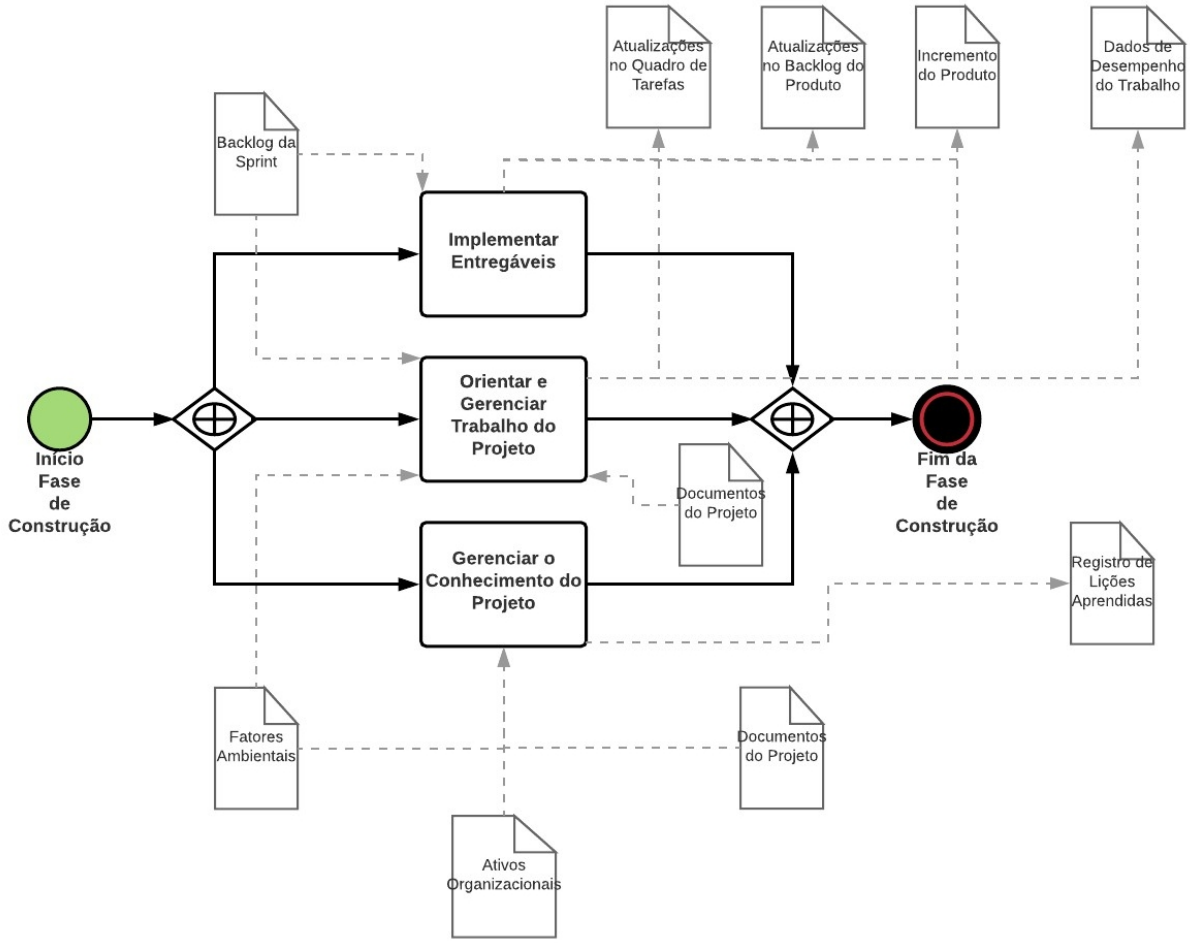
4.2.3 Fase de Construção

Nesta fase estão contidos os processos e atividades relacionados a colocar em prática o que foi planejado. A partir de artefatos oriundos da fase de planejamento gera outros artefatos, principalmente versões funcionais do produto. Seguindo princípios do FDD, nessa fase de construção, a equipe de desenvolvimento pode buscar maior detalhamento, junto ao PO, das funcionalidades alocadas para o *Sprint Backlog*. Os processos da fase de Construção são mostrados na Figura 13.

4.2.3.1 Implementar entregáveis

Vale lembrar que este trabalho se orienta ao gerenciamento. Sendo assim, como as atividades de desenvolvimento serão realizadas não faz parte do escopo do mesmo. A nível de gerenciamento cabe dizer que este é o processo de transformar os requisitos em produtos funcionais. É de responsabilidade do Time de Desenvolvedores e conta com a participação do *Scrum Master* e do PO. Tem como entadas o *Backlog* da *Sprint* e o Quadro De Tarefas. Os meios

Figura 13 – Processos da Fase de Construção



Fonte: Elaborado pelo Autor

utilizados são o Time *Scrum* e Ferramentas CASE, gerando como saída Incremento do Produto e atualizações no Quadro de Tarefas e *Backlog* do Produto.

4.2.3.2 Orientar e gerenciar trabalho do projeto

O Guia PMI (2017) descreve este processo como sendo o ato de realizar o trabalho definido no Plano de Gerenciamento de Projeto para atingir os objetivos. Como não há este artefato no processo proposto, deve-se gerenciar e orientar o trabalho definido na Reunião de Planejamento da *Sprint*, para alcançar os objetivos da mesma. O responsável por este processo é o *Scrum Master* e conta com a participação do PO e do Time de Desenvolvimento. As entradas utilizadas são os Documentos do Projeto, o *Backlog da Sprint*, Quadro de Tarefas, Ativos Organizacionais e Fatores Ambientais da Empresa. Os meios utilizados são Opinião Especializada e as Reuniões Diárias, gerando assim, suas saídas, sendo elas Incremento do Produto, Dados de Desempenho do Trabalho (que podem ser mas não se limitam a Gráfico

BurnDown) e atualizações no Quadro de Tarefas e no *Backlog* do Produto.

Dentro deste processo, ainda podem existir tarefas como obter, gerenciar e usar recursos, inclusive materiais, ferramentas, equipamentos e instalações, bem como implementar os padrões e os métodos planejados além de estabelecer e gerenciar os canais de comunicação do projeto, tanto externos como internos à equipe do projeto.

4.2.3.3 Gerenciar o conhecimento do projeto

Este é o processo de extrair o conhecimentos que cada membro possui e propiciar um ambiente para que novos conhecimentos sejam adquiridos, visando alcançar os objetivos do projeto e maximizar a evolução individual, de equipe e organizacional. É mais um oriundo do Guia PMBOK, porém, está muito ligado aos métodos ágeis, tanto que só foi incorporado ao Guia na última versão, cujo práticas ágeis foram inseridas. Em ambientes adaptativos, defende-se a a utilização de times multifuncionais, sendo assim os desenvolvedores tem maior liberdade para trabalhar nos mais diversos pontos do sistema a ser desenvolvido, sempre buscando ajuda daquele membro com maior conhecimento na tarefa a ser realizada, seja banco de dados, testes, camada de apresentação ou qualquer outra. Assim ocorre uma aprendizagem e evolução técnica.

O responsável por este processo é o *Scrum Master* e conta com a participação do Time de Desenvolvimento e PO. As entradas utilizadas são Documentos do Projeto (termo de Abertura, Partes Interessadas, Registros dos riscos dentre outros), Fatores Ambientais da Empresa Desenvolvedora, e, o principal, Ativos Organizacionais. Errar é comum, principalmente em projetos de natureza altamente dinâmica. Basear-se em projetos e atividades anteriores é uma boa solução para solucionar problemas ou mitigar alguns riscos.

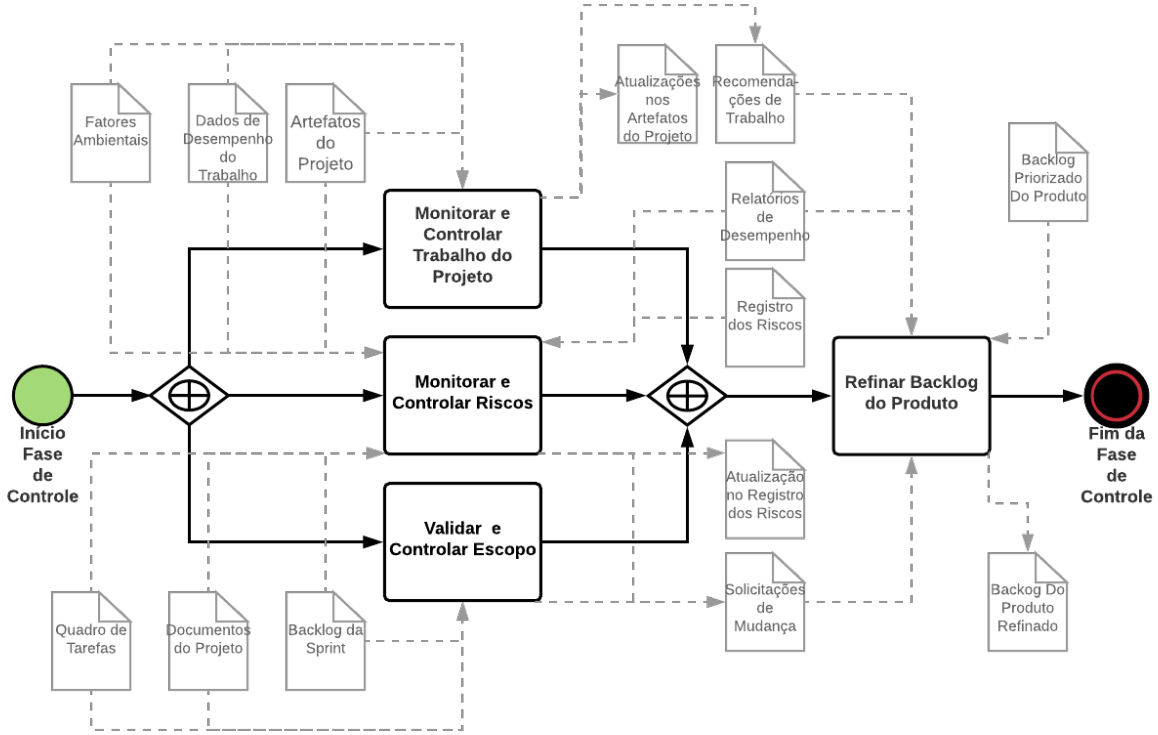
Os meio utilizados são a Opinião Especializada, Habilidades Interpessoais e Reuniões de Retrospectiva da *Sprint*. A saída gerada é o Registro de Lições Aprendidas. Pode-se também fazer uso de uma *Wiki* para registro de Soluções para problemas corriqueiros.

Além destas atividades a nível de gerenciamento, outras de desenvolvimento podem contribuir para discernir conhecimento entre os membros do time, como por exemplo Programação em Par e Propriedade Coletiva do Código da Metodologia Extreme Programming (XP), ficando a cargo da equipe decidir se utiliza ou não tais práticas.

4.2.4 Fase de Controle

Contém os processos e atividades relacionados em monitorar e controlar o andamento do projeto, aferindo desempenho da do time de desenvolvimento e outros indicadores para alinhar o trabalho real com o trabalho planejado, monitorando riscos e possíveis mudanças. Os processos da fase de Controle são mostrados na Figura 14.

Figura 14 – Processos da Fase de Controle



Fonte: Elaborado pelo Autor

4.2.4.1 Monitorar e Controlar o Trabalho do Projeto

De acordo com o PMI (2017), este é o processo de acompanhamento, revisão e ajuste do progresso para atender aos objetivos de desempenho definidos no plano de gerenciamento. No caso desta abordagem, que divide o projeto em ciclos menores mais facilmente gerenciáveis, este acompanhamento é feito em cada *Sprint*, fazendo ajustes em busca dos objetivos do ciclo. Assim, a revisão é feita apenas no trabalho alocado para o mesmo, ou seja, no *Sprint Backlog*. A principal atividade deste processo é comparar o desempenho real versus o planejado para a *Sprint* fornecendo informações para relato de desempenho e incorporação de mudanças. O responsável por tais atividades é o *Scrum Master*, contando com a participação do PO e do Time

de Desenvolvimento.

As entradas deste processo são Fatores Ambientais da Empresa, Artefatos do Projeto e Relatórios de Desempenho de Trabalho. Os meios utilizados são Análise de Dados(documentos ou observação ativa e passiva), Opinião Especializada e Reuniões. As saídas obtidas são Atualizações em Artefatos do Projeto, Relatório de Desempenho de Trabalho e Recomendações no Trabalho do Projeto.

4.2.4.2 Validar e Controlar Escopo

Em métodos tradicionais, geralmente este processo é dividido em dois, o "Validar Escopo" e o "Controlar Escopo". Tratam de aceitação do produto feito em paralelo ao desenvolvimento e alinhamento do trabalho realizado com o planejado para a *Sprint*respectivamente. Em ambientes adaptativos como o desenvolvimento de *software*, os dois devem caminhar juntos. Esta validação é diferente da que ocorre na fase de Entrega pois com o cliente presente, há um *feedback* constante em paralelo a construção do produto, e assim, tornando mais fácil também o controle do trabalho a ser realizado. O responsável por validar e controlar o escopo é o PO e conta com a participação do *Scrum Master*, do Time de Desenvolvimento e clientes.

As entradas para este processo são Documentos do Projeto, Quadro de Tarefas, e *Backlog da Sprint*. As saídas são obtidas através de Opinião do Cliente, *War Room* e Reuniões, sendo estas Solicitações de Mudanças no *Backlog* Priorizado do Produto.

4.2.4.3 Monitorar e Controlar Riscos

Este é o processo de acompanhar os riscos anteriormente identificados e implementar seus planos de resposta. Novos riscos são identificados aqui, logo, o acompanhamento da construção não pode parar para implementação das repostas, e sim, ocorrer em paralelo. Outra função deste processo é a validação das premissas estabelecidas no Termo de Abertura. O responsável por estas atividades é o *Scrum Master* e conta com a participação do Time de Desenvolvimento, do PO, da gerência e dos clientes e usuários.

Para a correta execução deste processo muitas entradas são necessárias, sendo elas Registro dos Riscos, Fatores Ambientais da Empresa, Dados de Desempenho do Trabalho, Relatório de Desempenho, Documentos do Projeto Artefatos do Projeto, Quadro de Tarefas e *Backlog da Sprint*. Os meios utilizados são Reuniões, dando destaque para a Reunião Diária, Análise de Variância (compara o real x planejado), Opinião Especializada e Análise da Dados. As

saídas geradas são Atualizações no Registro dos Riscos e Solicitações de Mudança no *Backlog* do Produto.

4.2.4.4 Refinar Backlog do Produto

Como dito anteriormente, o *Backlog* do Produto é uma lista com a descrição breve das funcionalidades do sistema e informações necessárias para a implementação das mesmas. Esta lista não é estática, muito pelo contrário. Conforme o projeto avança, requisitos são alterados e novos podem, e devem ser descobertos.

Este é o processo de realizar essas mudanças ou inclusões nas funcionalidades do sistema de forma coordenada. É de responsabilidade do PO e apenas ele pode alterar o *Backlog* do Produto, e não simplesmente qualquer um vai e muda o que e quando quiser. Há a participação do Time de Desenvolvimento, do *Scrum Master*, da gerência e dos clientes e usuários. Todos esses envolvidos podem discutir possíveis ajustes (refinamento) no *Backlog*, porém, a mudança deve ser feita por alguém claramente designado pra isso, como o *Product Owner*.

As entradas para este processo são Relatórios de Desempenho, Recomendações de Trabalho (alinhamento do trabalho do time para alcançar os objetivos da *Sprint*) e Solicitação de Mudanças no *Backlog*, ambas saídas dos processos anteriores. Os meios utilizados são Opinião Especializada, *Expertise* do PO e *Feedback* do Cliente e usuário. A saída deste processo é o *Backlog* Priorizado do Produto Refinado. Neste contexto, refinado quer dizer aprimorado, mais próximo do que realmente deve ser. Assim, quanto maior o refinamento, maior as chances de atender as necessidades do usuário.

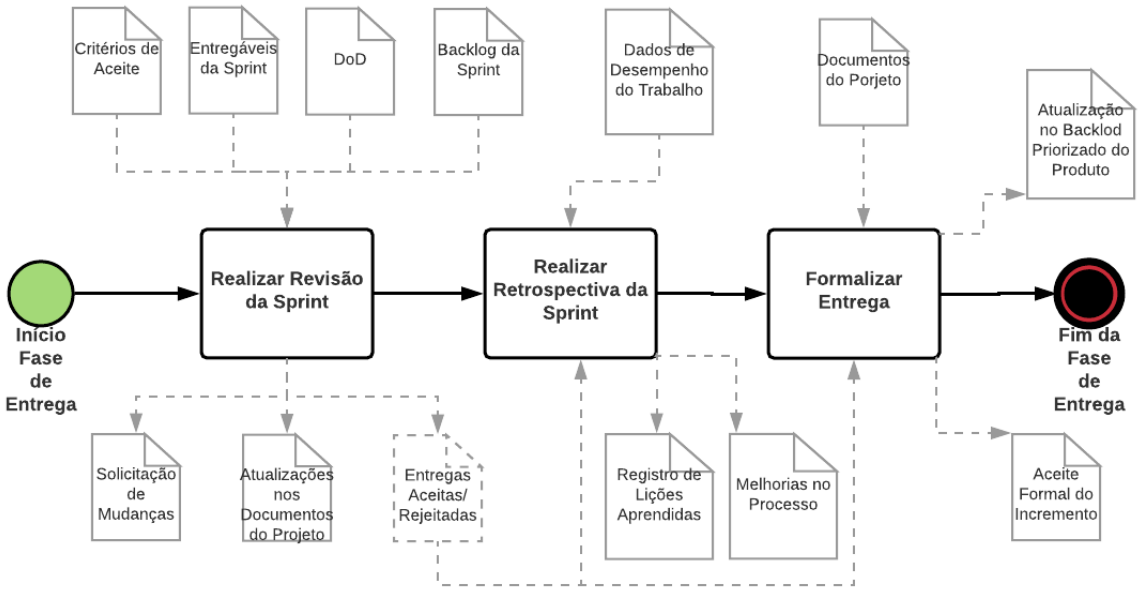
4.2.5 Fase de Entrega

Contém os processos e atividades relacionadas com a transição de artefatos para os usuários finais e clientes e avaliação processual, da equipe e individual. Os processos da fase de Entrega são mostrados na Figura 14.

4.2.5.1 Realizar revisão da Sprint

Este é o processo de demonstração dos resultados obtidos no ciclo de trabalho e sua comparação com o planejado para o mesmo. É responsabilidade do *Scrum Master* e os participantes deste processo são o PO, o Time de Desenvolvimento, gerência, clientes e usuários.

Figura 15 – Processos da Fase de Entrega



Fonte: Elaborado pelo Autor

As entradas deste processo são Entregáveis da *Sprint*, *Backlog* da *Sprint*, DoD e Critérios de Aceite das Histórias de Usuário. Os meios são Reunião de Revisão da *Sprint*, *Expertise* do *Scrum Master* e Opinião Especializada. A principal saída deste processo são entregas aceitas e rejeitadas, sendo esta última passível de adaptação na próxima *Sprint* ou simplesmente descarte. Outras saídas geradas são Atualizações nos documentos do Projeto e Solicitação de Mudanças.

4.2.5.2 Realizar retrospectiva da Sprint

Este é o processo de auto-avaliação do Time levando em consideração o que foi feito ao fim do ciclo de trabalho, levantando pontos fortes a serem mantidos e fracos a serem melhorados. Após o registro destas lições aprendidas, o Time de Desenvolvimento deve definir ações a serem tomadas e prioridade para estas ações, sendo este processo o responsável pela garantia da qualidade, ou seja, foca na melhoria contínua do processo. É responsabilidade do *Scrum Master* conduzir este processo e conta com a participação do Time de Desenvolvimento e do PO.

As entradas para o processo são Dados de Desempenho do Trabalho e Entregas Aceitas e Rejeitadas. Os meio utilizados são Reunião de Retrospectiva da *Sprint*, *Expertise* do *Scrum Master* e Opinião do Time. As saídas geradas por este processo são Melhorias no

Processo e Registro de Lições Aprendidas.

4.2.5.3 Formalizar Entrega

Este é o processo que encerra o ciclo de trabalho e/ou Projeto, documentando os entregáveis aceitos e rejeitados. Após entregas serem aceitas no Processo Realizar revisão da *Sprint*, deve-se fazer um aceite formal, deixando claro quais Histórias de Usuário foram concluídas e se os critérios de aceite foram cumpridos. Este processo é de grande importância, pois as entregas formalizadas são bases estabelecidas dinamicamente ao longo do projeto, que servem de apoio às *Sprints* posteriores. É responsabilidade do PO e conta com a participação da gerência e dos clientes.

As suas entradas são Documentos do Projeto e Entregas aceitas e rejeitadas. Os meios utilizados são Opinião Especializada, *Expertise* do *Product Owner*, Análise de Dados e Reuniões. As suas saídas são Entrega Formal e Atualizações no *Backlog* Priorizado do Produto.

5 ESTUDO DE CASO PARA AVALIAÇÃO DO PROCESSO PROPOSTO

O estudo foi realizado em ambiente acadêmico com oito equipes compostas por cinco ou seis membros, sendo estes alunos da disciplina de processos de *software*. No primeiro dia do estudo de caso, o processo foi apresentado e uma descrição textual do mesmo foi distribuída paras as equipes, juntamente com o termo de participação na pesquisa. Cada equipe se auto-organizou e definiu os papeis de cada um de acordo com as habilidades e necessidades. As mesmas utilizaram o processo proposto para conduzirem projetos com foco em atividades de design de interação e interface, ficando a cargo delas a seleção desses projetos.

O trabalho a ser realizado foi dividido em duas *Sprints* de quinze dias, a primeira tendo foco em projetar a interação e interface e a segunda em avaliar a usabilidade e a User Experience (UX). Por questões de cronograma o escopo deste trabalho se limita as resultados do primeiro ciclo de trabalho.

Algumas atividades foram feitas em sala de aula tendo acompanhamento do autor deste trabalho e da professora orientadora. No entanto, nem todas atividades puderam ter este acompanhamento, sendo avaliadas assim pela inspeção de artefatos gerados e pela observação do quadro de tarefas de cada equipe.

A fase de Comunicação foi inteiramente realizada e acompanhada em sala de aula, havendo explicações sobre a fase e o processo como um todo. A fase de Planejamento foi divida, sendo uma parte, composta pelos subprocessos "Priorizar Histórias de Usuário", "Criar EAP"e "Estimar Recursos das Atividades", feita em sala de aula e outra parte composta pelos demais subprocessos a cargo das equipes. As fases de Construção e a de Controle foram inteiramente executadas apenas com as equipes, e a fase de Entrega com a interação entre equipes, pesquisador e orientadora.

6 RESULTADOS

Já foi mostrado neste trabalho os resultados alcançados ainda na elaboração do processo. Neste capítulo serão apresentados os resultados obtidos com o estudo de caso,que foi a implantação do processo proposto em ambiente acadêmico. Com este estudo, diversos resultados foram obtidos, sendo estes por meio de observação do autor e análise de artefatos gerados e por *feedback* dos participantes da pesquisa.

6.1 Observação e análise de artefatos

Através de observação e análise o autor deste trabalho pode observar que equipes cujos membros tinham maior familiaridade definiram mais rapidamente os papéis, evidenciando assim que o relacionamento interpessoal é peça-chave para a fluidez do projeto.

Na fase de Comunicação todos os processos e atividades foram executados sem grandes problemas. Porém, na fase de Planejamento, mais especificamente na criação da EAP, as equipes tiveram dúvidas, fazendo-se necessárias explicações mais detalhadas e a demonstração de alguns exemplos práticos. De modo geral, os times de desenvolvimento (responsáveis pela atividade) encontraram dificuldades em definir o nível de decomposição das atividades. Uma solução possível para o problema é o compartilhamento da responsabilidade deste subprocesso com o PO, onde este faria a decomposição do trabalho até o nível de Histórias de Usuário, e, a partir daí, o time de desenvolvimento faria a decomposição até o pacote mínimo de trabalho. Porém, mesmo com todas as dificuldades encontradas neste subprocesso, os artefatos gerados foram satisfatórios a maior parte superou as expectativas.

Ainda na fase de Planejamento outro ponto merece ser mencionado: a estimativa de esforço das atividades pela técnica de *Planning Poker* foi muito bem aceita e executada. Todos se envolveram intensamente, inclusive os alunos nos papéis de *Scrum Master* e PO, este último sendo bastante requisitado para esclarecimento dos requisitos. Outro aspecto relevante foi que poucas equipes precisaram de mais de uma rodada para estimativa de alguma atividade.

Na fase de Construção e na de Controle, na maior parte das equipes, o *Scrum Master* teve certa dificuldade em executar algumas atividades relacionadas a seu papel, acarretando em pequenas falhas de controle e demonstração de trabalho, como a não atualização do quadro de tarefas e do gráfico de desempenho. De modo geral, isso ocorreu devido o mesmo estar muito ligado a construção do produto, deixando o processo em segundo plano. É um fato normal e

esperado, visto que este papel necessita de alguma experiência.

A fase de Entrega também ocorreu sem agravantes, visto que é a mais simples juntamente com a fase de Comunicação. O fato de ser simples não a faz menos importante, pelo contrário. Foi através desta fase que as equipes puderam mostrar os produtos construídos durante a *Sprint*, discutir o que poderia ser melhorado e dá um *feedback* em relação ao uso do processo, através do Registro de Lições Aprendidas e de questionários individuais.

6.2 Feedback dos participantes

O *feedback* dos alunos participantes do estudo de caso foi obtido de duas maneiras: através do Registro de Lições Aprendidas, que é um artefato gerado na fase de Entrega, e um formulário respondido ao fim do estudo.

Por meio do Registro das Lições Aprendidas, cujo modelo pode ser visto no Apêndice B, todas as equipes afirmaram ter concluído todas as atividades alocadas para o *Backlog* da *Sprint* e que os produtos gerados foram satisfatórios.

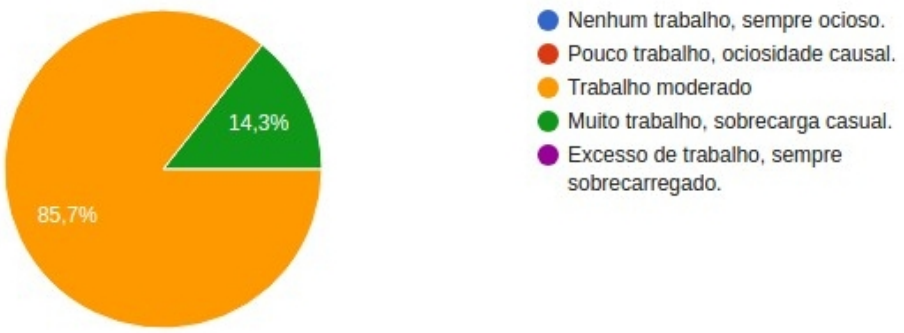
O documento supracitado é coletivo, assim, fez-se necessário uma coleta de dados de cada indivíduo participante da pesquisa por meio de um formulário. Este formulário, que pode ser visto no apêndice E, foi respondido pelos alunos participantes do estudo de caso com o intuito de obter seu *feedback* individual sobre o processo. Através deste formulário algumas informações relevantes puderam ser obtidas, o que não seria possível através apenas de observação. Em uma escala de 1 a 5 sobre a facilidade de uso do processo proposto, a maior parte dos participantes responderam 4. O mesmo ocorreu para o quanto eles tinham seguido o processo.

Outra questão levantada foi a dificuldade em alguns subprocessos específicos de cada papel. 50% dos membros do time de desenvolvimento citaram dificuldade com os subprocessos "Criar EAP"e "Implementar Entregáveis". As dificuldades com este primeiro subprocesso já tinham sido detectadas por observação e as dificuldades com o segundo são esperadas em todo projeto de desenvolvimento de *software*. Dos alunos no papel de *Scrum Master*, 80% disseram terem encontrado dificuldades com o subprocesso "Orientar e Gerenciar Trabalho no Projeto"e 60% levantaram dificuldades em "Monitorar e Controlar o Trabalho do Projeto"e "Monitorar e Controlar os Riscos". Este fato também foi identificado pelo autor através de observação. Das dificuldades encontradas pelos alunos no papel de PO vale ressaltar o subprocesso "Validar e Controlar o Escopo", que foi elencado por 80% dos participantes da pesquisa neste papel.

Diferentemente do que o Autor deste trabalho esperava, o papel em que houve maior

sobrecarga de trabalho foi o PO. A Figura 16, a Figura 17 e a Figura 18 ilustram a carga de trabalho dos papéis envolvidos no estudo de caso.

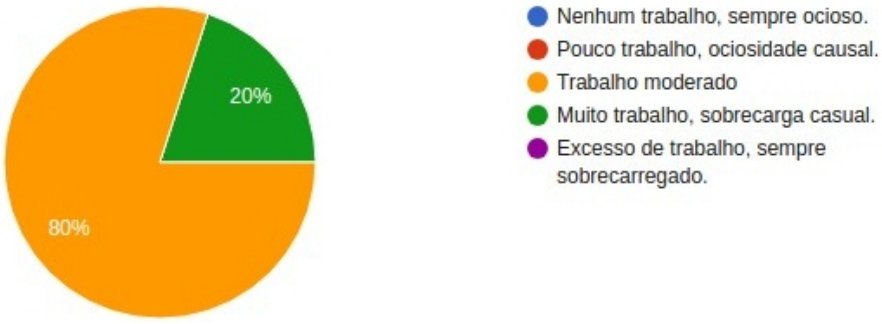
Figura 16 – Carga de Trabalho do Time de Desenvolvimento



Fonte: Elaborado pelo Autor

A Figura 16 mostra que a grande maioria dos integrantes dos times de desenvolvimento alegaram ter trabalho moderado. Para alguns houve sobrecarga casual, que tem como provável causa o esforço considerável que o time faz durante o subprocesso ”Implementar Entregáveis”. *Feedback* muito similar com o dos alunos que atuaram como PO. A Figura 17 mostra como estes alunos enxergaram essa alocação de trabalho.

Figura 17 – Carga de Trabalho do Product Owner

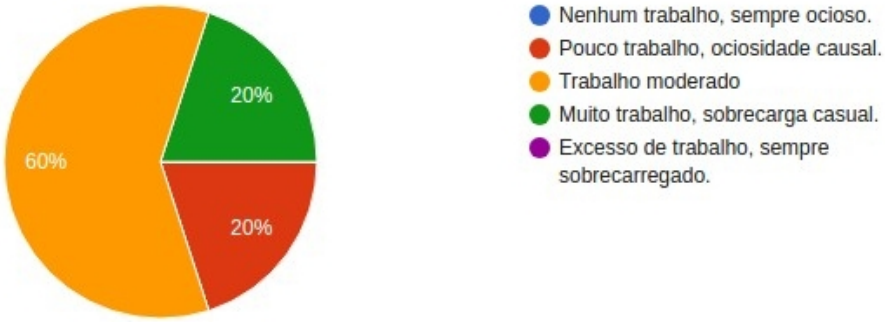


Fonte: Elaborado pelo Autor

A imensa maioria dos envolvidos alegaram trabalho moderado e o restante mencionou uma sobrecarga casual, um pouco maior que a do Time de Desenvolvimento. Este fato pode ter como causa a sazonalidade do trabalho do PO, que é mais frequente em algumas fases do projeto, como Comunicação e Entrega.

Diferentemente dos demais papéis, os alunos que atuaram como *Scrum Master* não tiveram a mesma visão sobre a carga de trabalho alocada para este papel. A Figura 18 mostra a divergência de opiniões.

Figura 18 – Carga de Trabalho do Scrum Master



Fonte: Elaborado pelo Autor

Dos alunos no papel de *Scrum Master*, a maior parte, que totaliza 60% dos envolvidos, viram sua carga de trabalho como moderada e o restante se dividiram igualmente entre pouco ou muito trabalho alocado. Estes resultados não eram esperados, visto que há grande demanda de esforço do *Scrum Master* no processo proposto. Esse fato tem como explicação que alguns dos alunos neste papel não entenderam muito bem a sua função na fase de Construção e na fase de Controle. As questões abertas deste formulário corroboram esta explicação.

Para que os envolvidos na pesquisa tivessem maior liberdade de expressar sua opinião sobre o processo, duas questões abertas foram criadas. Uma pergunta sobre os pontos fortes do processo e a outra sobre os pontos a serem melhorados, e suas respostas podem ser vistas no Apêndice F e no Apêndice G respectivamente.

Nos pontos a serem melhorados no processo apareceu por diversas vezes a sugestão de maior detalhamento do processo suas fases e subprocessos, em especial nas fases de Construção e Controle. A solução para este problema seria o acompanhamento integral das equipes pelo pesquisador, algo que não foi realizado, pois o mesmo acompanhava as equipes apenas durante as aulas, duas vezes por semana.

Ainda em relação aos pontos a serem melhorados, duas respostas merecem destaque: "Acompanhamento do rendimento da equipe"e "[...] cobrar mais os alunos com relação ao cumprimento de prazos, pois o processo se atrasou por este fator."Essas afirmações levantam tarefas que o *Scrum Master* deveria ter feito. É uma questão a ser revista, podendo ter como solução um treinamento inicial para as pessoas que atuaram neste papel.

Outro ponto bastante citado foi em relação à geração e entrega de artefatos. Os envolvidos na pesquisa mencionaram que esta tarefa tomou bastante tempo. Isso se deve pelo fato deles terem trabalhado com o processo uma primeira e única vez. Como as práticas ágeis são empíricas, estes artefatos seriam gerados com maior facilidade de acordo com as sucessivas repetições, se tornando uma rotina simples.

Com relação aos pontos fortes do processo, grande parte dos alunos envolvidos no estudo de caso mencionaram a organização e a divisão das atividades por papéis como um fator importante, evitando uma possível sobrecarga de trabalho. Outro ponto bastante citado foi que o processo melhorou a comunicação das equipes, aumentando a interação entre os membros.

De acordo com as respostas obtidas, o processo foi bem aceito, principalmente pela organização do mesmo. Segundo os alunos, as atividades foram bem selecionadas e distribuídas nas suas respectivas fases. Isso faz com que se tenha uma ideia de linearidade, tendo assim maior controle sobre o andamento das atividades.

As maiores contribuições deste trabalho foram mostrar o ganho de produtividade quando aplicadas práticas e princípios ágeis em projetos de desenvolvimento de *software* e propor um processo simples para estes projetos. Além destas, outras contribuições ocasionadas pelo uso do processo proposto podem ser mencionadas, como:

- O ganho de organização e controle do trabalho quando há um processo bem definido a ser seguido;
- Uma maior proximidade entre o produto real e o produto esperado;
- Melhor divisão do trabalho, evitando sobrecarga em uns membros e ociosidade em outros;
- A prova que há a possibilidade de se adotar práticas mundialmente reconhecidas em um contexto menos grandioso, sendo a aplicação de processos do Guia PMBOK a prova disso;
- Melhoria da comunicação e maior interação entre os membros da equipe.

7 CONCLUSÃO

Este trabalho apresentou um processo para auxiliar equipes com pouca ou nenhuma experiência a conduzirem as atividades relacionadas a projetos de desenvolvimento de *software*, bem como sua elaboração e resultados obtidos. É importante ressaltar que estes resultados foram fortemente influenciados pelo contexto do estudo de caso, que foi em um ambiente acadêmico, e naturalmente algumas equipes e membros estavam mais envolvidos e motivados do que outros.

Apesar da necessidade de algumas melhorias na questão de acompanhamento, o processo foi bem aceito. Os resultados se mostraram positivos tanto na visão do pesquisador como na dos participantes da pesquisa. As equipes planejaram adequadamente o trabalho da *Sprint* e o produto gerado foi satisfatório.

Com base no problema em questão, nos objetivos levantados e nos resultados obtidos, pode-se concluir uma melhoria significativa sobre o planejamento, controle e execução do projeto devido a adoção do processo que estabeleceu um passo a passo a ser seguido, deixando claro quais ações e atividades deveriam ser realizadas. Como corolário, temos que houve uma melhoria da comunicação entre os membros da equipe, promovendo uma maior interação entre eles.

Apesar de que, como afirma França *et al.* (2010), é comum na bibliografia sobre engenharia de *software* a ideia de que práticas ágeis só podem ser usadas por equipes experientes, outra conclusão deste trabalho foi que há a possibilidade de adoção destas práticas por equipes com pouca ou nenhuma experiência.

As considerações finais que podem ser feitas a este estudo, é que mesmo havendo a possibilidade da adoção do processo e de outras abordagens ágeis por equipes inexperientes, os benefícios trazidos por estes seriam maximizados se ao menos um dos membros das equipes, preferencialmente a pessoa no papel de *Scrum Master* tivesse experiência com tais abordagens.

Muito ainda pode ser feito neste estudo. Baseado nos resultados obtidos e no *feedback* dos participantes da pesquisa, melhorias e adaptações podem ser feitas no processo proposto. Outro ponto de grande importância para ser estudado posteriormente é a adoção deste processo por uma organização, saindo assim do ambiente acadêmico. Deste modo, um leque de possibilidades seria aberto, como por exemplo a avaliação do nível de maturidade deste processo pelo Melhoria de Processos do Software Brasileiro (MPS.BR). Logo, o processo proposto pode ser melhorado ainda mais para atingir níveis de maturidade cada vez melhores segundo o modelo supracitado, com o intuito de acompanhar a crescente demanda por qualidade nos processos das empresas desenvolvedoras de *software*.

REFERÊNCIAS

ABES. **ABES SOFTWARE Mercado Brasileiro de Software - Panorama e Tendências**. 2017. Disponível em: <http://central.abessoftware.com.br/Content/UploadedFiles/Arquivos/Dados\%202011/ABES-Publicacao-Mercado-2017.pdf>. Acesso em: 2017-10-23.

ABRAHAMSSON, P.; WARSTA, J.; SIPONEN, M. T.; RONKAINEN, J. New directions on agile methods: a comparative analysis. In: IEEE. **Software Engineering, 2003. Proceedings. 25th International Conference on**. Portland, OR, USA, 2003. p. 244–254.

ALFONSO, M. I.; BOTIA, A. An iterative and agile process model for teaching software engineering. In: **18th Conference on Software Engineering Education Training (CSEET'05)**. Ottawa, Canada: [s.n.], 2005. p. 9–16. ISSN 1093-0175.

ALMEIDA, G. A. M. de. **Fatores de escolha entre metodologias de desenvolvimento de software tradicionais e ágeis**. 2017. Acesso em: 2017-11-17. Disponível em: <http://www.teses.usp.br/teses/disponiveis/3/3136/tde-11042017-143311/>.

BERMEJO, P. H. de S.; ZAMBALDE, A. L.; TONELLI, A. O.; SOUZA, S. A.; ZUPPO, L. A.; ROSA, P. L. Agile principles and achievement of success in software development: A quantitative study in brazilian organizations. **Procedia Technology**, v. 16, n. Supplement C, p. 718 – 727, 2014. ISSN 2212-0173. CENTERIS 2014 - Conference on ENTERprise Information Systems / ProjMAN 2014 - International Conference on Project MANagement / HCIST 2014 - International Conference on Health and Social Care Information Systems and Technologies. Disponível em: <http://www.sciencedirect.com/science/article/pii/S2212017314002485>.

BERNI, J. C. A. **Gestão para o processo de desenvolvimento de software científico, utilizando uma abordagem ágil e adaptativa na microempresa**. Santa Maria, Rio Grande do Sul: [s.n.], 2010. Disponível em: <http://repositorio.ufsm.br/bitstream/handle/1/8132/BERNI,\%20JEAN\%20CARLO\%20ALBIERO.pdf>.

BONACIN, R.; BARANAUSKAS, M. C. C.; RODRIGUES, M. A. An agile process model for inclusive software development. In: FILIPE, J.; CORDEIRO, J. (Ed.). **Enterprise Information Systems**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. p. 807–818. ISBN 978-3-642-01347-8.

COHN, M. **Desenvolvimento de Software com Scrum. Aplicando Métodos Ágeis com Sucesso (Em Portuguese do Brasil)**. Porto Alegre, Rio Grande do Sul: Bookman, 2011. ISBN 8577808076.

CONFORTO, E. C. Gerenciamento ágil de projetos: proposta e avaliação de método para gestão de escopo e tempo. **Escola de Engenharia de São Carlos, University of São Paulo, São Carlos**, 2009. Acesso em: 2017-11-17. Disponível em: <http://www.teses.usp.br/teses/disponiveis/18/18140/tde-28072009-090239/en.php>.

COSTA, W. N. F. **Modelagem de um Novo Processo de Desenvolvimento de Software com Base em Metodologias Ágeis**. Presidente Epitácio, São Paulo: [s.n.], 2010. Disponível em: <https://pep.ifsp.edu.br/moodledata/filedir/fa/89/fa89e7dc1ea88c0c2292de2f571959b941f39a3e>.

CRUZ, F. **Scrum e Pmbok: Unidos no Gerenciamento de Projetos**. Rio de Janeiro: Brasport, 2013. ISBN 8574525944.

DSDM CONSORTIUM . **O Framework de Projetos àgeis DSDM**. 2014. Disponível em: <https://www.agilebusiness.org/content/choosing-dsdm-your-agile-approach-0>. Acesso em: 2017-10-30.

FRANÇA, C. T. P. L.; SERRA, A. de B.; PATRICIO, R. G.; DONATO, I. A. **SCRUM com Equipes Inexperientes**. Juazeiro do Norte, Ceará: [s.n.], 2010. Disponível em: <https://www.infobrasil.inf.br/userfiles/SCRUM\%20com\%20Equipes\%20Inexperientes.pdf>.

KERZNER, H. **Gestão de Projetos. As Melhores Praticas (Em Portuguese do Brasil)**. Porto Alegre, Rio Grande do Sul: Bookman, 2005. ISBN 8536306181.

LAANTI, M.; SALO, O.; ABRAHAMSSON, P. Agile methods rapidly replacing traditional methods at nokia: A survey of opinions on agile transformation. **Inf. Softw. Technol.**, Butterworth-Heinemann, Newton, MA, USA, v. 53, n. 3, p. 276–290, mar. 2011. ISSN 0950-5849. Disponível em: <http://dx.doi.org/10.1016/j.infsof.2010.11.010>.

MARTINS, J. **Gerenciando projetos de desenvolvimento de software com PMI,RUP E UML**. Rio de Janeiro: Brasport, 2010. ISBN 9788574524511.

OFFICE OF GOVERNMENT COMMERCE. **Gereciando Projetos De Sucesso Com PRINCE2: [Brazilian Portuguese Print Version of Managing Successful Projects with PRINCE2] (Portuguese Edition)**. United Kingdom: The Stationery Office, 2011. ISBN 0113313470.

OLIVEIRA, M. F. de. METODOLOGIA CIENTÍFICA: um manual para a realização de pesquisas em administração. **Manual (pós-graduação) – Universidade Federal de Goiás**, Catalão, GO, 2011. Disponível em: <https://adm.catalao.ufg.br/up/567/o/Manual_de_metodologia_cientifica_-_Prof_Maxwell.pdf>.

O'CONNOR, R. V.; COLEMAN, G. An investigation of barriers to the adoption of software process best practice models. **Proceedings of the Australasian Conference on Information Systems**, Toowoomba, Australian, n. 18, p. 780 – 789, December 2007. Disponível em: <https://www.researchgate.net/publication/228641056_An_Investigation_of_Barriers_to_the_Adoption_of_Software_Process_Best_Practice_Models>.

PMI, P. M. I. **A Guide to the Project Management Body of Knowledge (PMBOK® Guide)–Sixth Edition (BRAZILIAN PORTUGUESE) (Portuguese Edition)**. Newtown Square, Pensilvânia, EUA: Project Management Institute, 2017. ISBN 1628251921.

PRESSMAN, R. S. **Engenharia de Software - uma Abordagem Profissional**. Porto Alegre, Rio Grande Sul: Mc Graw Hill, 2011. ISBN 8563308335.

PRIKLADNICKI, R. **Métodos Ágeis Para Desenvolvimento de Software (Em Portuguese do Brasil)**. Porto Alegre, Rio Grande do Sul: Bookman, 2014. ISBN 8582602073.

RASNACIS, A.; BERZISA, S. Method for adaptation and implementation of agile project management methodology. **Procedia Computer Science**, v. 104, n. Supplement C, p. 43 – 50, 2017. ISSN 1877-0509. ICTE 2016, Riga Technical University, Latvia. Disponível em: <http://www.sciencedirect.com/science/article/pii/S187705091730056X>.

RIBEIRO, R. D.; RIBEIRO, H. da Cunha e S. **Gerenciamento de projetos com métodos ágeis**. Rio de Janeiro: s.n, 2015. ISBN 9788591910212.

RIBEIRO, R. D.; RIBEIRO, H. da Cunha e S. **Gerenciamento de Projetos Orientados a Planos**. Rio de Janeiro: SPIN, 2015. ISBN 978-85-919102-2-9.

SABBAGH, R. **Scrum**: Gestão Ágil para projetos de sucesso. São Paulo: CASA DO CODIGO, 2013. ISBN 8566250109.

SBROCCO, P. C. d. M. José Henrique Teixeira de C. **Metodologias ágeis: engenharia de software sob medida**. 1. ed. São Paulo: Erica, 2011. ISBN 978-85-365-0979-2.

SCHWABER, K.; SUTHERLAND, J. **Guia do Scrum Um guia definitivo para o Scrum: As regras do jogo**. 2016. Disponível em: <http://www.scrumguides.org/docs/scrumguide/v2016/2016-Scrum-Guide-US.pdf#zoom=100>. Acesso em: 2017-10-23.

SCRUM STUDY . **A Guide to the Scrum Body of Knowledge (SBOK Guide)**. Phoenix, Arizona: VMEdu Inc., 2013. ISBN 098992520X.

SOMMERVILLE, I. **Engenharia de Software (Em Portuguese do Brasil)**. 9. ed. São Paulo: Pearson, 2011. ISBN 8579361087.

SUTHERLAND, J. **Scrum**: A arte de fazer o dobro do trabalho na metade do tempo. São Paulo: LeYa, 2016. ISBN 8544104517.

TORRES, L. F. **Fundamentos do Gerenciamento de Projetos**. Rio de Janeiro: Elsevier, 2013. ISBN 8535271724.

TSUI, F.; KARAM, O.; BERNAL, B. **Essentials Of Software Engineering**. Burlington,MA: Jones & Bartlett Learning, 2013. ISBN 1449691994.

VARGAS, R. **Gerenciamento de Projetos. Estabelecendo Diferenciais Competitivos (Em Portuguese do Brasil)**. [S.l.]: Brasport, 2017. ISBN 8574528366.

ŠPUNDAK, M. Mixed agile/traditional project management methodology – reality or illusion? **Procedia - Social and Behavioral Sciences**, v. 119, n. Supplement C, p. 939 – 948, 2014. ISSN 1877-0428. Selected papers from the 27th IPMA (International Project Management Association), World Congress, Dubrovnik, Croatia, 2013. Disponível em: <http://www.sciencedirect.com/science/article/pii/S187704281402196X>.

ANEXO A – PROCESSOS DO GUIA PMBOK

Figura 19 – Mapeamento dos Processos, Grupo de Processos e Áreas de Conhecimento do Guia PMBOK

Áreas de Conhecimento	Grupos de Processos				
	Processos de Iniciação	**Processos de Planejamento**	**Processos de Execução**	**Processos de Monitoramento e Controle**	**Processos de Encerramento**
1.Gerenciamento da Integração	1.1 Desenvolver Termo de Abertura do Projeto	1.2 Desenvolver Plano de Gerenciamento do Projeto	1.3 Orientar e gerenciar o Trabalho do Projeto 1.4 Gerenciar o conhecimento do Projeto	1.5 Monitorar e Controlar o Trabalho do Projeto 1.6 Realizar o Controle Integrado de Mudanças	1.7 Encerrar Projeto ou Fase
2.Gerenciamento do Escopo		2.1 Planejar o Gerenciamento do Escopo 2.2 Coletar os Requisitos 2.3 Definir o Escopo 2.4 Criar a EAP		2.5 Validar o Escopo 2.6 Controlar o Escopo	
3.Gerenciamento do Cronograma		3.1 Planejar o Gerenciamento do Cronograma 3.2 Definir as Atividades 3.3 Sequenciar as Atividades 3.4 Estimar as Durações das Atividades 3.5 Desenvolver o Cronograma		3.6 Controlar o Cronograma	
4.Gerenciamento dos Custos		4.1 Planejar o Gerenciamento dos Custos 4.2 Estimar os Custos 4.3 Determinar o Orçamento		4.4 Controlar os Custos	
5.Gerenciamento da Qualidade		5.1 Planejar o Gerenciamento da Qualidade	5.2 Gerenciar a Qualidade	5.3 Controlar a Qualidade	
6.Gerenciamento dos Recursos		6.1 Planejar o Gerenciamento dos Recursos 6.2 Estimar os Recursos das Atividades	6.3 Adquirir Recursos 6.4 Desenvolver a Equipe 6.5 Gerenciar a Equipe	6.6 Controlar os Recursos	
7.Gerenciamento da Comunicação		7.1 Planejar o Gerenciamento das Comunicações	7.2 Gerenciar as Comunicações	7.3 Monitorar as Comunicações	
8.Gerenciamento dos Riscos		8.1Planejar o Gerenciamento dos Riscos 8.2 Identificar os Riscos 8.3 Realizar a Análise Qualitativa dos Riscos 8.4 Realizar a Análise Quantitativa dos Riscos 8.5 Planejar as Respostas aos Riscos	8.6 Implementar Respostas aos Riscos	8.7 Monitorar os Riscos	
9.Gerenciamento das Aquisições		9.1 Planejar o Gerenciamento das Aquisições	9.2 Conduzir as Aquisições	9.3 Controlar as Aquisições	
10.Gerenciamento das Partes Interessadas	10.1 Identificar as Partes Interessadas	10.2 Planejar o Engajamento das Partes Interessadas	10.3 Gerenciar o Engajamento das Partes Interessadas	10.4 Monitorar o Engajamento das Partes Interessadas	

Fonte: Adaptado de Guia PMBOK 6a edição

APÊNDICE A – TERMO DE ABERTURA DO PROJETO

TERMO DE ABERTURA DO PROJETO

NOME DO PROJETO

Controle de Versões:

Versão	Data	Autor	Notas da Revisão
1.0	00/00/0000	Xxxxx Xxxxxx	Foi alterado xxx

Sumário

1 Objetivos deste documento

Este documento tem como objetivo autorizar formalmente o início do projeto, atribuir principais responsáveis e documentar requisitos iniciais, principais entregas, premissas e restrições e critérios de sucesso.

2 Situação atual e justificativa do projeto

Como é o atual contexto e o problema a ser resolvido, além de como o sistema atuará em relação a isso. Destacar o porquê do desenvolvimento do projeto.

3 Objetivos SMART e critérios de sucesso do projeto

O projeto será considerado um sucesso se atender a todos os critérios de aceitação das entregas que podem ser (respeitar as restrições e cumprir o cronograma de execução por exemplo) e principalmente atender os objetivos abaixo:

• Objetivo 1: citar um objetivo descrito com a sigla SMART;

• Objetivo 2: citar um objetivo descrito com a sigla SMART;

• Objetivo 3: citar um objetivo descrito com a sigla SMART.

4 Principais entregas

Descrever aqui as principais entregas, geralmente incrementos funcionais do sistema ao usuário ou cliente. Insumos para desenvolvimento não devem ser documentados aqui, como por exemplo Documento de Requisitos, Diagrama etc.

Entrega	Descrição
Entrega 1	Controle de Login e cadastro de Usuário
Entrega 2	Busca por livros pelo título/autor

5 Principais requisitos das principais entregas/produtos

Documentar aqui os principais requisitos tidos como base para a elaboração do projeto, aqueles que justificam o desenvolvimento do sistema, definidos na reunião Kick-Off, antes mesmo da coleta propriamente dita.

6 Definição de Pronto

Definir aqui os critérios para considerar uma atividade concluída. Podemos citar como exemplos de critérios: Implementada, aprovada em testes unitários, aprovado no teste de integração, não necessita de alteração, aprovado pelo Product Owner, etc.

7 Marcos

Documentar aqui os pontos significativos do projeto, aqueles cuja conclusão deve ser informada para as partes interessadas.

Marco	Data Prevista	Responsável
Projeto aprovado		PO
Requisitos Levantados		PO
Primeira Sprint Iniciada		
Primeira entrega		Todo Time

8 Partes Interessadas

Demonstrar de forma clara quem são os envolvidos, mostrando seu poder de decisão no projeto e influência de suas atitudes. Pode-se também fazer relatório do nível de engajamento para com o projeto, assim podemos procurar meios de incentivar pessoas com engajamento baixo e saber seus motivos. A tabela abaixo relaciona as principais partes interessadas do projeto:

Nome	Papel	Poder	Influência	Engajamento
Ana	Cliente	Alto	Média	Alto
Carlos	Desenvolvedor	Médio	Média	Alto
João	Scrum Master	Médio	Alta	Baixo

9 Restrições

Aqui são documentados as limitações imutáveis do projeto. Segue exemplos:

Restrição 01: o projeto deve ser encerrado obrigatoriamente no dia 15/12/2017.

Restrição 02: Não haverá dispêndio monetário para aquisição de softwares, hardwares, cursos e etc.

10 Premissas

São condições tomadas como verdade para embasar o início do projeto. Segue exemplos:

Premissa 01: Os desenvolvedores têm experiência com a linguagem a ser usada.

Premissa 02: Caso necessário, membros da equipe disponibilizarão horas extras para trabalho e estudo.

Premissa 03: Clientes estão disponíveis para esclarecer quaisquer dúvidas relacionadas ao projeto dentro do horário de funcionamento da empresa.

11 Principais Riscos

Aqui são registrados os principais riscos inicialmente levantados, de forma macro, bem como suas respostas. Riscos específicos posteriormente identificados não são documentados aqui.

ID Risco	Descrição	ID Resposta	Descrição
RC001	Atraso em relação aos marcos	RP001	Equipe adicionar mais tempo de

			trabalho semanalmente, para alinhamento com o planejado.
		RP002	Solicitar ajuda de membros de outras equipes desde que não atrapalhe o andamento destas outras equipes.

Russas, ____/____/_____

_____ _____

Responsável pelo Projeto Cliente

APÊNDICE B – REGISTRO DE LIÇÕES APRENDIDAS

REGISTRO DE LIÇÕES APRENDIDAS

NOME DO PROJETO			
EQUIPE		**REFERENTE À**	Sprint 1

1. Objetivos

Este documento tem o objetivo de documentar os acontecimentos que impactaram o projeto positiva ou negativamente, buscando maximizar e minimizar respectivamente.

2. Orientações

Cada lição aprendida deve conter os seguintes aspectos:

- O evento (O que ocorreu?);
- A causa (Por que ocorreu?);
- O impacto (Quais as consequências?);
- Ações a serem tomadas(Quais as sugestões para próximas Sprints e/ou Projetos?).

3. Registro

Evento	Causa	Impacto	Ações

4. Questões

PERGUNTA	SIM	NÃO	COMENTÁRIOS
Todas as atividades do Sprint Backlog foram concluídas?			
Foi colocado trabalho demais ou pouco trabalho para a Sprint?			
As atividades foram estimadas da melhor forma?			
Algum papel teve elevada dificuldade?			
O produto gerado foi satisfatório?			

5. Pontos de destaque

5.1 Os principais pontos que correram bem e devem ser mantidos para a próxima Sprint.

5.2 Os principais itens que podem ser ainda melhorados, e serem ainda mais positivo na próxima Sprint.

5.3 Os principais itens que devem ser descartados e retirados da próxima Sprint.

APÊNDICE C – FORMULÁRIO USADO NO N2S

Questionário 1 - Práticas no N2S

Avaliação das práticas e técnicas utilizadas no N2S por parte da equipe, visando a eficiência dos projetos do núcleo.

*Obrigatório

1. **Ciclo vida do Scrum foi eficiente?** *
   Marcar apenas uma oval.
   - ◯ Não foi eficiente
   - ◯ Pouco eficiente
   - ◯ Eficiente na medida do possível
   - ◯ Muito eficiente

2. **Qual a melhor forma de coleta de requisitos?** *
   Marcar apenas uma oval.
   - ◯ Totalmente no ínicio do projeto
   - ◯ Grande parte no ínicio do projeto
   - ◯ No planejamento de cada ciclo de trabalho
   - ◯ Em paralelo ao desenvolvimento

3. **Como riscos devem ser identificados?** *
   Marcar apenas uma oval.
   - ◯ Não deve ter preocupação
   - ◯ Devem ser identificados informalmente pelos desenvolvedores
   - ◯ Devem ser identificados e tratados em intervalos constantes
   - ◯ Devem ser identificados e documentados em paralelo ao desenvolvimento
   - ◯ Devem ser identificados e documentados no planejamento de cada Sprint

4. **Como a qualidade deve ser trabalhada?** *
   Marcar apenas uma oval.
   - ◯ Não há necessidade de ser trabalhada
   - ◯ Deve ocorrer de modo informal
   - ◯ Deve ter uma pessoa específica para este fim
   - ◯ Deve ser trabalhada em paralelo ao desenvolvimento
   - ◯ Atributos e requisitos de qualidade devem ser definidos e controlados

5. **O que deve ocorrer antes do ínicio do planejamento** *
Marcar apenas uma oval.

- ◯ Nada, o projeto começa no planejamento.
- ◯ Acordos verbais entre as partes interessadas
- ◯ Estabelecimento de bases a serem obedecidas, como restrições de tempo recursos e etc
- ◯ Estabelecimento de linhas de base e cronogrma de artefatos a serem gerados
- ◯ Estabelecimento prévio de um cronograma detalhado para as atividades do projeto

6. **Descreva pontos fortes(que deve ser mantido) e fracos(que devem ser repensados) no projeto na qual participou.** *

_____

_____

_____

_____

_____

Powered by
Google Forms

APÊNDICE D – TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO

TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO

Pesquisa: "Proposta de um Processo Ágil Para Projetos de Desenvolvimento de Software"

Prezado Senhor (a),

Vocês estão sendo convidados (as) para participar de uma pesquisa que estudará a implantação de um processo para projetos de desenvolvimento de software.

Sua participação na pesquisa **não** é obrigatória.

1) Procedimento

Este estudo será feito da seguinte forma: (1) Você receberá este Termo de Consentimento Livre e Esclarecido (TCLE); (2) Você irá adotar este processo para trabalho acadêmico; (3) Você irá gerar os artefatos e relatará suas opiniões e experiências.

2) Tratamento de possíveis riscos e desconfortos

Serão tomadas todas as providências durante a coleta de dados de forma a garantir a sua privacidade e seu anonimato. Os dados coletados durante o estudo destinam-se estritamente a atividades de pesquisa, não sendo utilizados em qualquer forma de avaliação profissional ou pessoal.

3) Benefícios e Custos

Este estudo contribuirá com resultados importantes para a pesquisa de um modo geral nas áreas de Engenharia de Software (ES) e Processos da mesma.

Você não terá nenhum gasto ou ônus com a sua participação no estudo e também não receberá qualquer espécie de reembolso ou gratificação devido à participação **na pesquisa**.

4) Confidencialidade da Pesquisa

Toda informação coletada neste estudo é confidencial e seu nome e o da sua equipe não serão identificados de modo algum, a não ser em caso de autorização explícita para esse fim.

5) Participação

Sua participação neste estudo é muito importante e voluntária. Você tem o direito de não querer participar ou de sair deste estudo a qualquer momento, sem penalidades. Em caso de você decidir se retirar do estudo, favor notificar um pesquisador responsável. Os pesquisadores responsáveis pelo estudo poderão fornecer qualquer esclarecimento sobre o mesmo, assim como tirar dúvidas, bastando entrar em contato pelos seguintes emails:

Pesquisador: Luan Dharlin Lemos – luandharlinls@gmail.com e Pesquisadora Orientadora: Anna Beatriz dos Santos Marques - annaa.marques@gmail.com

6) Declaração de Consentimento

Li ou alguém leu para mim as informações contidas neste documento antes de assinar este termo de consentimento. Declaro que toda a linguagem técnica utilizada na descrição deste estudo de pesquisa foi explicada satisfatoriamente e que recebi respostas para todas as minhas dúvidas. Confirmo também que recebi uma cópia deste Termo de Consentimento Livre e Esclarecido. Compreendo que sou livre para me retirar do estudo em qualquer momento, sem qualquer penalidade. Declaro ter mais de 18 anos e dou meu consentimento de livre e espontânea vontade para participar deste estudo.

Russas, 03 de maio de 2018.

_____	_____
Participante: Exemplo de Nome de Participante	Pesquisador: Luan Dharlin Lemos da Silva

APÊNDICE E – FORMULÁRIO USADO NO ESTUDO DE CASO

Feedback sobre o Processo

Este formulário tem como propósito a avaliação de uma proposta de processo para projetos de software. Todavia, não tem nenhuma influência sobre a avaliação acadêmica do aluno.

*Obrigatório

1. **Qual seu papel na equipe?** *
   Marcar apenas uma oval.
   - ◯ Time de Desenvolvimento *Ir para a pergunta 6.*
   - ◯ Scrum Master *Ir para a pergunta 11.*
   - ◯ Product Owner *Ir para a pergunta 13.*

2. **O quanto você e sua equipe seguiram o processo?** *
   Marcar apenas uma oval.

	1	2	3	4	5	
Processo não foi seguido	◯	◯	◯	◯	◯	Processo foi seguido plenamente

3. **Qual nota você dá a facilidade de uso do processo proposto?** *
   Marcar apenas uma oval.

	1	2	3	4	5	
Muito difícil	◯	◯	◯	◯	◯	Muito fácil

4. **O quanto o processo ajudou a distribuir as atividades?** *
   Marcar apenas uma oval.

	1	2	3	4	5	
Não ajudou	◯	◯	◯	◯	◯	Ajudou muito

5. **Digite sua matrícula.** *

Seção para o Time de Desenvolvimento

6. **Classifique a alocação de trabalho para o seu papel.** *
   Marcar apenas uma oval.

   - ◯ Nenhum trabalho, sempre ocioso.
   - ◯ Pouco trabalho, ociosidade causal.
   - ◯ Trabalho moderado
   - ◯ Muito trabalho, sobrecarga casual.
   - ◯ Excesso de trabalho, sempre sobrecarregado.

7. **Qual a relevância do P.O. da sua equipe para a conclusão do trabalho?** *
   Marcar apenas uma oval.

	1	2	3	4	5	
Sem relevância	◯	◯	◯	◯	◯	Muito relevante

8. **Qual a relevância do Scrum Master da sua equipe para a conclusão do trabalho?** *
   Marcar apenas uma oval.

	1	2	3	4	5	
Sem relevância	◯	◯	◯	◯	◯	Muito relevante

9. **O quanto o processo ajudou o time a construir os produtos?** *
   Marcar apenas uma oval.

   - ◯ Não ajudou
   - ◯ Ajudou pouco
   - ◯ Ajudou moderadamente
   - ◯ Ajudou muito
   - ◯ Foi essencial

10. **Marque os subprocessos que geraram mais dificuldade.** *
    Marque todas que se aplicam.

    - ☐ Criar EAP
    - ☐ Estimar Recursos das Atividades
    - ☐ Criar Backlog da Sprint
    - ☐ Identificar Riscos
    - ☐ Implementar Entregáveis
    - ☐ Realizar Retrospetiva da Sprint
    - ☐ Nenhum

Ir para a pergunta 16.

Seção para Scrum Master

11. **Classifique a alocação de trabalho para o seu papel.** *
Marcar apenas uma oval.

- ◯ Nenhum trabalho, sempre ocioso.
- ◯ Pouco trabalho, ociosidade causal.
- ◯ Trabalho moderado
- ◯ Muito trabalho, sobrecarga casual.
- ◯ Excesso de trabalho, sempre sobrecarregado.

12. **Marque os subprocessos que geraram mais dificuldade.** *
Marque todas que se aplicam.

- ☐ Estimar Recursos das Atividades
- ☐ Identificar Riscos
- ☐ Orientar e Gerenciar Trabalho do Projeto
- ☐ Gerenciar o Conhecimento do Projeto
- ☐ Monitorar e Controlar o Trabalho do Projeto
- ☐ Monitorar e Controlar os Riscos
- ☐ Realizar Revisão da Sprint
- ☐ Realizar Retrospectiva da Sprint
- ☐ Nenhum

Ir para a pergunta 16.

Seção para P.O.

13. **Marque os subprocessos que geraram mais dificuldade.** *
Marque todas que se aplicam.

- ☐ Criar Termo de Abertura do Projeto
- ☐ Definir Partes Interessadas
- ☐ Criar Backlog do Produto
- ☐ Criar Histórias de Usuário
- ☐ Priorizar Histórias de Usuário
- ☐ Validar e Controlar Escopo
- ☐ Refinar Backog do Produto
- ☐ Formalizar Entrega
- ☐ Nenhum

14. **Qual foi seu grau de envolvimento com o validador dos requisitos (professor/cliente/usuário)?** *
Marcar apenas uma oval.

- ◯ Nenhum envolvimento.
- ◯ Pouco envolvido;
- ◯ Envolvido moderadadamente;
- ◯ Muito envolvido;
- ◯ Presente no dia-a-dia.

15. **Classifique a alocação de trabalho para o seu papel.** *
Marcar apenas uma oval.

- ◯ Nenhum trabalho, sempre ocioso.
- ◯ Pouco trabalho, ociosidade causal.
- ◯ Trabalho moderado
- ◯ Muito trabalho, sobrecarga casual.
- ◯ Excesso de trabalho, sempre sobrecarregado.

Ir para a pergunta 16.

Seção perguntas abertas

16. **Descreva pontos fortes do processo proposto** *

17. **Descreva pontos a serem melhorados no processo proposto** *

Powered by
Google Forms

APÊNDICE F – RESPOSTAS PARA A QUESTÃO 16: QUAIS OS PONTOS FORTES DO PROCESSO?

- Comunicação;
- Escolha das atividades (2);
- Melhor organização, maior envolvimento da equipe;
- Nada a declarar sobre isso;
- Ser apoiado no *Scrum*;
- Ficou muito bem dividido as funções para os membros, não ocasionando sobrecarga de trabalho de alguma das partes;
- Não sei (2);
- O sequenciamento das atividades que produzem artefatos para outras atividades posteriores é muito eficiente e faz com que o desenvolvimento do projeto seja de forma linear e concisa;
- Utilização da EAP, divisão da equipe;
- Ajuda a ter mais organização;
- Bom sequenciamento lógico;
- Facilita a divisão das atividades para cada papel;
- Dá uma visão de como um processo de software é aplicado na vida real, separando a equipe em papeis e atribuindo diferentes responsabilidades;
- A fase do planejamento foi bem elaborada;
- A organização das atividades a ser realizadas em fases, assim se tem uma melhor visualização do que está se fazendo no momento;
- *Plaining poker*;
- O baseamento no *Scrum* é um ótimo ponto;
- É fácil entender a ideia proposta;
- Atividades bem definidas; Ordem das atividades correspondem com a necessidade do trabalho para aquele momento;
- O processo é bem estruturado; trabalha com uma "ordem cronológica"ótima, ou seja, tarefas anteriores ajudam muito nas que vêm em seguida;
- Organizado, fácil e confiável;
- Protótipos bem elaborados;
- Permite uma maior interação com a equipe;
- Permitiu boa comunicação entre os membros da equipe.

APÊNDICE G – RESPOSTAS PARA A QUESTÃO 17: QUAIS OS PONTOS A SEREM MELHORADOS NO PROCESSO?

- Melhor distribuição de atividades;
- Escolha de ferramentas (2);
- Melhorar o entendimento de cada fase do processo;
- Nada a melhorar;
- Nada a declarar sobre isso;
- Acredito que o processo deveria ter características próprias, não misturando com gerencia de projetos;
- Enxugar documentação;
- O que gerou um pouco de problema foi a correria em entregar os artefatos, talvez com mais alguns dias na *sprint* isso poderia ser melhorada. De resto o processo é ótimo;
- Acompanhamento do rendimento da equipe;
- Agilidade e atribuições de responsabilidade para os membros da equipe;
- Explicar mais detalhadamente cada processo de forma prática, mostrar mais exemplos de artefatos, detalhar melhor como as fases de controle ocorrem paralelamente à de execução;
- Alguns pontos não necessitam serem realizados de forma tão abrangente
- Melhorar a organização de como o processo é aplicado, quero dizer, cobrar mais os alunos com relação ao cumprimento de prazos, pois o processo se atrasou por este fator;
- Poderia diminuir algumas coisas, pra ser um processo ágil ele está com muita burocracia;
- Uma descrição melhor de cada artefato e atividade para melhor; entendimento, pois são muitos e acaba confundindo o que deve ser feito;
- Descrever melhor as atividades, pois algumas se tornam meio complexas para entender;
- A quantidade de artefatos. Para um processo ágil, o processo proposto apresenta uma quantidade bastante elevada de artefatos;
- O processo pode ser trabalhado mais detalhadamente, como mais tempo para acompanhamento;
- EAP;
- Clareza das etapas do processo.