Henrique Mateus Reichert Fernandez

Uma proposta de metodologia ágil para gerenciamento de projetos de software seguindo as boas práticas do PMBOK

Trabalho apresentado ao curso MBA em Gerenciamento de Projetos, Pós-Graduação *lato sensu*, Nível de Especialização, do Programa FGV Management da Fundação Getúlio Vargas, como pré-requisito para a obtenção do Titulo de Especialista.

Edmarson Bacelar Mota

Coordenador Acadêmico Executivo

Denise Margareth O. Basgal

Orientadora

Curitiba – PR

2015

FUNDAÇÃO GETÚLIO VARGAS
PROGRAMA FGV MANAGEMENT
MBA EM GERENCIAMENTO DE PROJETOS

O Trabalho de Conclusão de Curso Uma proposta de metodologia ágil para gerenciamento de projetos de software seguindo as boas práticas do PMBOKelaborado por Henrique Mateus Reichert Fernandeze aprovado pela Coordenação Acadêmica, foi aceito como pré-requisito para a obtenção do certificado do Curso de Pós-Graduação lato sensu MBA em Gerenciamento de Projetos, Nível de Especialização, do Programa FGV Management.

Data da Aprovação: Curitiba - PR, 08 de maio de 2015:

Edmarson Bacelar Mota
Coordenador Acadêmico Executivo

Denise Margareth Oldenburg Basgal
Orientadora

Termo de Compromisso

O aluno Henrique Mateus Reichert Fernandez, abaixo assinado, do curso de MBA em Gerenciamento de Projetos, Turma GP32-Curitiba (1/2013), do Programa FGV Management, realizado nas dependências da instituição conveniada ISAE, no período de 25/02/2013 a 16/01/2015, declara que o conteúdo do Trabalho de Conclusão de Curso intitulado Uma proposta de metodologia ágil para gerenciamento de projetos de software seguindo as boas práticas do PMBOK, é autêntico e original.

Curitiba - PR, 08 de maio de 2015:

Henrique Mateus Reichert Fernandez

Dedico este trabalho aos meus pais Simón Ignácio Fernandez Alvarez e Rosane Teresinha Reichert, pois a minha formação como profissional não poderia ter sido concretizada sem a ajuda dos mesmos, que no decorrer da minha vida, proporcionaram-me, além de extenso carinho e amor, os conhecimentos da integridade, da perseverança e da honestidade. Por essa razão, e por serem minha fortaleza e meu porto seguro, gostaria de dedicar e reconhecer à vocês, minha imensa gratidão e sempre amor.

—

"Faci quod potui, faciant meliora potents."
Aforisma Latino.

Resumo

No mundo do Gerenciamento de Projetos é muito comum se ouvir falar nas boas práticas pregadas pelo Guia PMBOK. Porém na área de Desenvolvimento de *Software* e Tecnologia da Informação, o PMBOK sofre muita rejeição, sendo o mesmo considerado rígido e burocrático, sendo assim inapropriado para uma área tão veloz e sensível a mudanças. Devido a isso, é muito comum a aplicação de Métodos Ágeis de Gerenciamento de Projetos. Neste trabalho serão citadas semelhanças e diferenças entre as abordagens, comparações entre elas e se as mesmas podem ser ou não utilizadas em conjunto. Ao final do trabalho será proposta uma metodologia utilizando as melhores práticas do PMBOK e os princípios ágeis para gerenciamento de projetos de *software*.

Palavras-chave: Gerenciamento de Projetos, Scrum, Métodos Ágeis, Metodologias Ágeis, PMBOK.

Abstract

In the Project Management world is very common to hear about the good practices advocated by the PMBOK Guide. But in Software Development and Information Technology, the PMBOK suffers a lot of rejection , being even considered rigid and bureaucratic , thus inappropriate for an area as fast and sensitive to changes. Because of this, it is very common to apply Agile Methodologies. In this study similarities and differences between the approaches will be mentioned, comparisons between them and they can be used together or not. At the end of the work is proposed a methodology using the best practices of the PMBOK and agile principles for managing software projects.

Key-words: Project Management, Scrum, Agile Software Development, Agile Methodology, PMBOK.

Lista de ilustrações

Sumário

Introdução

Os tempos atuais são conhecidos como a Era da Informação. Isso se deve à velocidade e facilidade acesso a informações. Esta facilidade é consequência do grande avanço tecnológico obtido nas últimas décadas e obviamente dos Softwares desenvolvidos, cada vez mais sofisticados.

O processo de elaboração de um *software*, mesmo sendo um tipo de produto utilizado pela grande maioria da população mundial, ainda é obscuro e desconhecido para muitos. Este fato influencia diretamente no gerenciamento de projetos de *software*, os tornando mais complexos e imprevisíveis.

Por ser um produto diferenciado dos produtos físicos, o processo de implementação de um *software* é bem diferente de uma linha de produção, por exemplo. E para o gerenciamento de projetos não é diferente. Existem metodologias criadas especialmente para estas especificidades. Estas metodologias são conhecidas como Metodologias Ágeis.

As metodologias de gerenciamento conhecidas como "tradicionais" também são utilizadas para a gestão de projetos na área de tecnologia da informação, porém a grande tendência e preferência da maioria dos profissionais da área realmente é pela aplicação de Métodos Ágeis, devido sua grande versatilidade e adaptação rápida a mudanças.

A verdade é que muitos defendem o Ágil como a melhor solução para o Gerenciamento de Projetos, principalmente profissionais ligados à tecnologia, outros acreditam que os métodos de Gerenciamento de Projetos Tradicionais, o PMBOK como grande representante dos mesmos, funcionam para qualquer espécie de projeto.

Os projetos podem ser classificados de acordo sua complexidade e esta variável pode influenciar a metodologia a ser aplicada na gestão do mesmo. Porém, é possível agrupar boas práticas de gerenciamento de projetos de diversas vertentes para formar uma metodologia para um tipo de projeto foco de uma instituição ou organização.

Neste trabalho serão apresentadas as abordagens mais comuns na gestão de projetos de *software*, será demonstrado um pouco de cada visão e ao final será proposta uma metodologia utilizando as melhores práticas das mesmas.

1 Referencial Teórico

1.1 O Gerenciamento de Projetos

A definição mais conhecida de Projeto é a de que um Projeto é: “Um esforço temporário com a finalidade de criar um produto/serviço único” (PMI, 2012). O aspecto temporário indica que o mesmo sempre te um início e um fim, e como o resultado é único, significa que o produto do projeto é diferenciado sob alguma ótica.

Todo projeto possui um ciclo de vida, que determina as fases de desenvolvimento do projeto estabelecendo o que precisa ser feito para a realização do projeto através das Fases do Ciclo de Vida do Projeto. Além disso, oferece pontos de sincronização para o trabalho colaborativo da equipe.

As fases de um projeto auxiliam a saber o que fazer após esta fase, o tempo e o que deve ser gerado ao fim de cada etapa. Além de evidenciar a dependência entre as tarefas e oferecer no seu final uma oportunidade de avaliar os resultados alcançados e decidir sobre a continuação ou não do projeto.

A humanidade planeja e gerencia projetos desde o início da civilização, construindo estradas e mega-estruturas como as pirâmides do Egito ou o Coliseu em Roma. Mesmo sem as ferramentas, técnicas e metodologias que se tem atualmente, ainda havia o gerenciamento para criar prazos de projeto, controlar custos, programar materiais e recursos e avaliar riscos.

Ao longo do tempo, foram criadas técnicas de controle de custos, criação de prazos, aquisição de recursos e gerenciamento de riscos que podiam ser aplicadas a uma ampla série de projetos, seja de aterrissagem na lua ou de exploração para encontrar petróleo ou de desenvolvimento de sistemas de informação. Embora o gerenciamento de projetos como prática já exista há séculos, só foi reconhecido formalmente como profissão após a 2ª. Guerra Mundial.

Atualmente, Gerenciamento de Projeto é a aplicação de conhecimento, habilidades, ferramentas e técnicas em atividades do projeto, a fim de satisfazer ou exceder as necessidades e expectativas dos *stakeholders* (interessados e envolvidos). Satisfazer ou exceder as necessidades e expectativas dos *stakeholders* invariavelmente envolve equilibrar demandas concorrentes em relação a:

- Escopo, prazo, custo e qualidade
- Requisitos identificados (necessidades) e requisitos não identificados (expectativas).
- *Stakeholders* com necessidades e expectativas diferenciadas.

1.2 O Guia PMBOK

O livro “A Guide to the Project Management Body of Knowledge” ou Guia para o Corpo de Conhecimentos de Gerenciamento de Projetos é um marco na história da ciência de Gerenciamento de Projetos. Mais conhecido como "PMBOK Guide", ele é de autoria do Standards Committee (Comitê de Padronização) do Project Management Institute - PMI e procura contemplar os principais aspectos e as melhores práticas que podem ser abordadas no gerenciamento de um projeto genérico. Não se trata de uma metodologia de gerenciamento de projetos e, sim, de uma padronização, identificando e nomeando processos, áreas de conhecimento, técnicas, regras e métodos. Ele foi reconhecido, em 1999, como um padrão de gerenciamento de projetos pelo ANSI - American National Standards Institute. Ele é um material genérico que serve para todas as áreas de conhecimento, ou seja, tanto para construção civil ou processo de fabricação industrial como para a produção de software. Um outro objetivo do PMBOK é a padronização de termos utilizados em Gerenciamento de Projetos.

Mesmo o PMBOK sendo um guia largamente difundido na área de Gerenciamento de Projetos, um grave erro cometido por muitos é acreditar que o PMBOK é uma metodologia para gerenciar projetos, e tentam segui-lo ao pé da letra. Inclusive se vê o grave engano de acreditar que conhecer bem o PMBOK é ser um bom gerente de projetos.

Neste trabalho será utilizado como base a 5ª Edição do PMBOK.

1.2.1 Ciclo de Vida

O Ciclo de Vida genérico de um Projeto ou Fase pregado pelo PMBOK é mostrado na figura 1. Fases do projetos são momentos ou marcos importantes do projeto, como por exemplo conclusão de uma etapa ou entrega. O ciclo de vida não está relacionado com as Fases do Projeto.

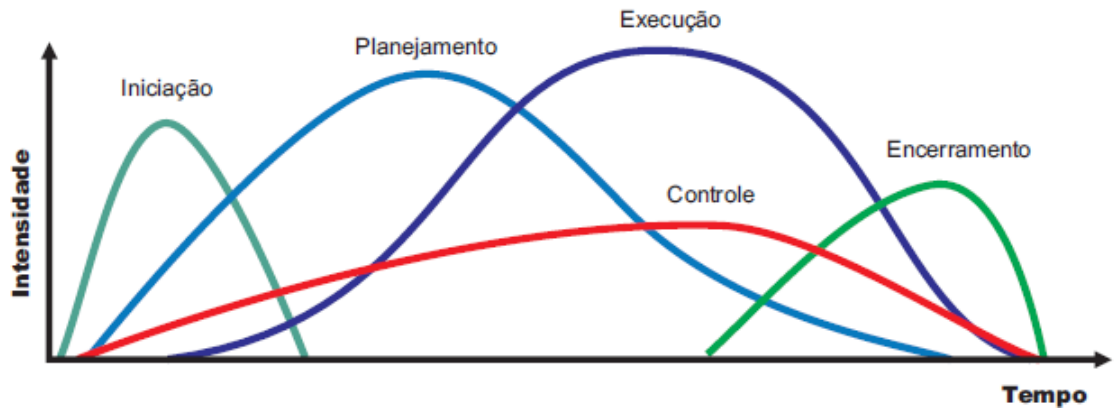

Figura 1 – Ciclo de Vida de um Projeto genérico segundo o PMBOK (ROCHA, 2013)

A diferença entre ciclo de vida e fase do projeto fica mais evidente em projetos iterativos, onde inclusive as processos de iniciação e encerramento podem ser executados novamente, conforme figura 2.

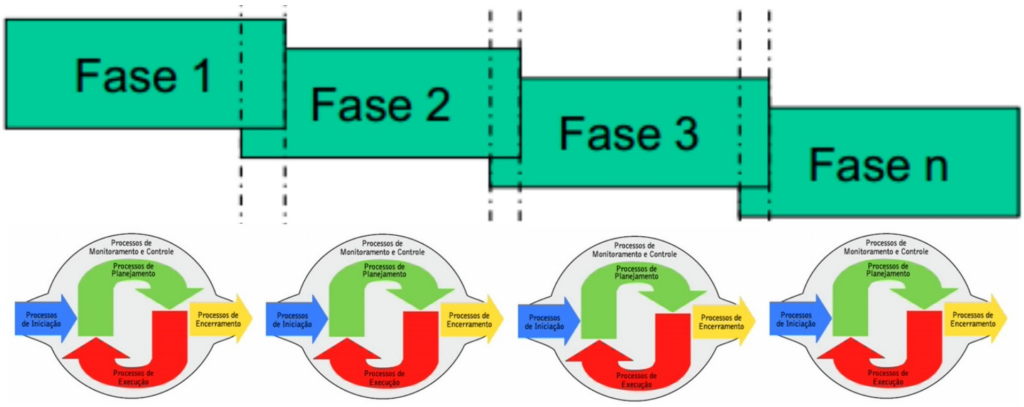

Figura 2 – Ciclo de Vida de Fases - Projetos iterativos
(ROCHA, 2013)

Os grupos de processos não são fases, são somente agrupamento de processos conforme proposta pelo PMBOK. Coincidentemente, o PMBOK utiliza os mesmos nomes para as Fases e para os Grupos de Processos.

1.2.2 As Boas Práticas

O gerenciamento de projetos é realizado através da integração de cinco grupos de processos e dez áreas de conhecimento. Assim, os gerentes de projetos podem padronizar tarefas rotineiras para obter resultados repetitivos e reduzir o número de tarefas que poderiam ser negligenciadas ou esquecidas durante o projeto. (PMI, 2012)

Grupos de Processos de Gerenciamento de Projetos se sobrepõem ou interagem entre si conforme a fase do projeto:

- Iniciação: autorizam o começo do projeto ou a fase.
- Planejamento: definem objetivos e selecionam o melhor esquema de trabalho para cumprir os objetivos propostos do projeto.
- Execução: coordenam pessoas e outros recursos para conduzir o plano.
- Monitoramento e controle: asseguram que os objetivos do projeto são alcançados através do monitoramento e medição do progresso, identificando variações do plano e executando ações corretivas.
- Encerramento: formalizam a aceitação do projeto ou da fase e dirigem o projeto para um fim adequado.

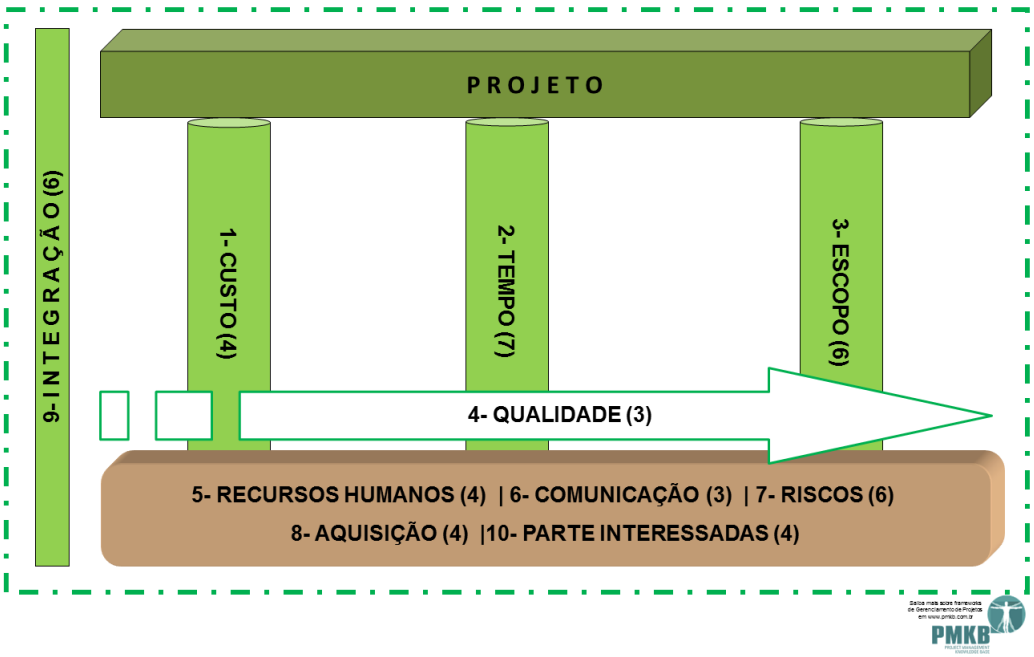

Figura 3 – Áreas do Conhecimento de acordo com o PMBOK
(PMKB, 2013)

Esses processos de gerência de projetos são executados por meio de um ou mais processos organizados em dez áreas de conhecimento da gerência de projetos:

- Integração: apresenta os processos necessários para assegurar que os elementos envolvidos no projeto sejam adequadamente coordenados.
- Escopo: apresenta os processos necessários para assegurar que o projeto contemple o trabalho requerido, e para completar o projeto com sucesso.
- Tempo: apresenta os procedimentos necessários para assegurar que o projeto termine dentro do prazo previsto.
- Custo: apresenta os procedimentos necessários para assegurar que o projeto seja concluído dentro do orçamento previsto.
- Qualidade: apresenta os procedimentos necessários para assegurar que as necessidades que originaram o desenvolvimento do projeto serão satisfeitas.
- Recursos Humanos: apresenta os procedimentos necessários para proporcionar a melhor utilização das pessoas envolvidas no projeto.
- Comunicações: apresenta os procedimentos necessários para assegurar que a geração, captura, distribuição, armazenamento e apresentação das informações do projeto sejam feitos de forma adequada e no tempo certo.

- Riscos: apresenta os procedimentos que dizem respeito à identificação, análise e resposta aos riscos do projeto.

- Aquisições: apresenta os procedimentos necessários para a aquisição de bens e serviços fora da organização que desenvolve o projeto.

- Stakeholders: apresenta os procedimentos necessários para assegurar os interessados do projeto, pode ser pessoas da equipe, grupos de pessoas, organizações ou instituições com algum tipo de interesse ou que poderão ser afetados pelas atividades ou pelos resultados do seu projeto.

1.3 Engenharia de Software

Na Área de Tecnologia de Desenvolvimento de Software, o Gerenciamento de Projetos é estudado em uma área denominada Engenharia de Software. Nela são estudadas os levantamentos de requisitos, análises dos mesmos, implementação. testes e tudo o que é necessário realizar para elaborar um software.

O termo Engenharia de Software tornou-se conhecido após uma conferência em 1968, quando as dificuldades e armadilhas de projetar sistemas complexos foram discutidas francamente. A busca de soluções se concentrou em melhores metodologias e ferramentas. As mais importantes foram as linguagens de programação que refletem os estilos procedimental, modular e, em seguida, orientado a objeto. A Engenharia de Software está intimamente ligada ao aparecimento e aperfeiçoamento desses estilos.

Mais recentemente, o rápido crescimento do poder computacional tornou possível aplicar computação em tarefas cada vez mais complicadas. Esta tendência aumentou drasticamente as demandas por engenheiros de software. Programas e sistemas se tornaram complexos e quase impossíveis de ser completamente compreendidos. A queda dos custos e a abundância de recursos computacionais inevitavelmente reduziram os cuidados para um bom projeto. Hoje as limitações não são dadas por hardwares lentos, mas pela própria capacidade intelectual. Por experiência, pode-se observar que a maioria dos programas pode ser significativamente melhorados, ficando mais confiáveis, econômicos e confortáveis de se utilizar.

1.3.1 Ciclo de Vida

1.3.1.1 Modelo em Cascata (waterfall)

É bem conhecido por ser amplamente empregado à várias décadas, é simples e intuitivo. Todas as atividades do projeto são executadas em série, onde uma atividade só inicia após a conclusão da anterior.

Processos de área de Conhecimento	Grupos de Processos de Gerenciamento de Projetos				Saiba mais sobre frameworks de Gerenciamento de Projetos em www.pmkb.com.br PMKB
	Iniciação	Planejamento	Execução	Monitoramento e Controle	Encerramento
Integração	4.1 Desenvolver o termo de abertura do projeto	4.2 Desenvolver o plano de gerenciamento do projeto	4.3 Orientar e gerenciar a execução do projeto	4.4 Monitorar e controlar o trabalho do projeto 4.5 Realizar o controle integrado de mudanças	4.6 Encerrar o projeto ou fase
Escopo		5.1 Planejar o Gerenciamento do Escopo 5.2 Coletar os Requisitos 5.3 Definir o escopo 5.4 Criar a EAP		5.5 Validar o escopo 5.6 Controlar o escopo	
Tempo		6.1. Planejar o Gerenciamento do Cronograma 6.2 Definir as atividades 6.3 Sequenciar as atividades 6.4 Estimar os recursos das atividades 6.5 Estimar a durações das atividades 6.6 Desenvolver o cronograma		6.7 Controlar o cronograma	
Custos		7.1 Planejar o gerenciamento dos custos 7.2 Estimar os custos 7.3 Determinar o orçamento		7.4 Controlar os custos	
Qualidade		8.1 Planejar o gerenciamento da qualidade	8.2 Realizar a garantia da qualidade	8.3 Controlar a qualidade	
Recursos Humanos		9.1 Planejar o gerenciamento dos recursos humanos	9.2 Mobilizar a equipe do projeto 9.3 Desenvolver a equipe do projeto 9.4 Gerenciar a equipe do projeto		
Comunicações		10.1 Planejar o gerenciamento das comunicacões	10.2 Gerenciar as comunicações	10.3 Controlar as comunicações	
Riscos		11.1 Planejar o gerenciamento dos riscos 11.2 Identificar os riscos 11.3 Realizar a análise qualitativa dos riscos 11.4 Realizar a análise quantitativa dos riscos 11.5 Planejar as respostas aos riscos		11.6 Controlar os riscos	
Aquisições		12.1 Planejar o gerenciamento das aquisições	12.2 Conduzir aquisições	12.3 Controlar as aquisições	12.4 Encerrar as aquisições
Partes Interessadas	13.1 Identificar as Partes Interessadas	13.2 Planejar o gerenciamento das Partes Interessadas	13.3 Gestão do Engajamento das Partes Interessadas	13.4 Controlar o Engajamento das Partes Interessadas	

Figura 4 – Processos propostos pelo PMBOK separados pelos Grupos de Processos (PMKB, 2013)

O modelo de desenvolvimento em cascata pode ser considerado ultrapassado devido a sua principal característica: a necessidade de conclusão de uma atividade para iniciar a próxima. A imagem 5 é a que demonstra mais fielmente o ciclo de vida deste modelo, pois é sequencial.

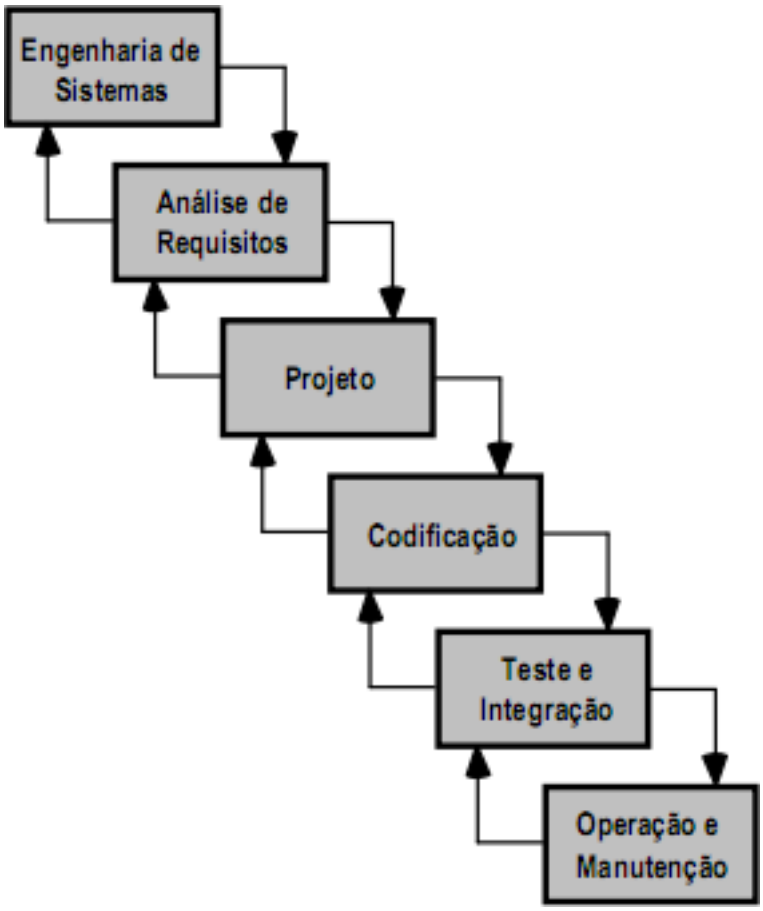

Figura 5 – Modelo em Cascata
(ROCHA, 2013)

1.3.1.2 Modelo Iterativo

Este modelo tem a característica de ter uma fase planejada a cada momento. As fases iniciais da fase seguinte , como por exemplo o planejamento, são realizadas durante a execução da fase atual.

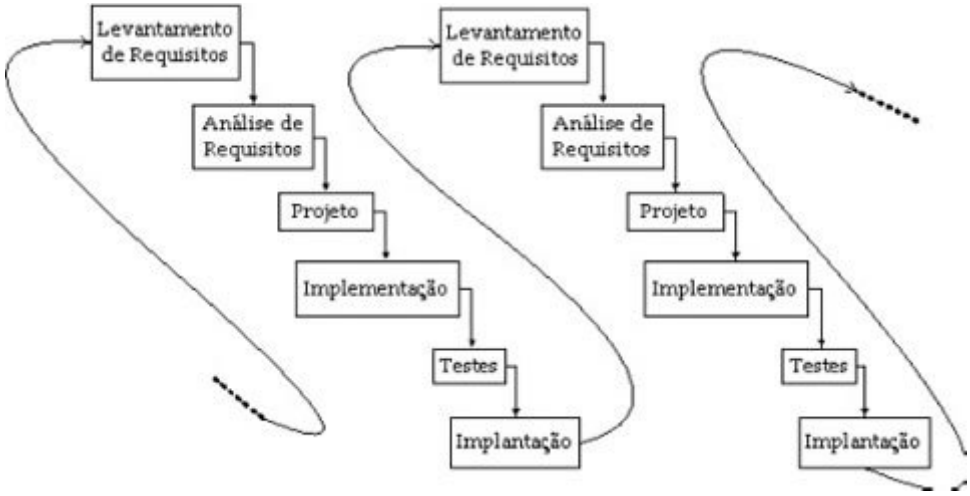

Figura 6 – Modelo Iterativo
(ROCHA, 2013)

No modelo iterativo pode-se visualizar a execução do ciclo de vida também de forma interativa, onde as fases do projeto são executadas novamente a cada iteração. As atividades de cada grupo de processos também podem ser executadas novamente a cada iteração, sendo que os processos de monitoramento e controle podem ser executados de forma única ao longo de todo o projeto sem descaracterizar o modelo (ROCHA, 2013).

1.3.1.3 Modelo em Espiral

Suas principais características são a análise de riscos e a prototipagem. O modelo espiral realiza-os de forma iterativa avaliando o progresso a cada iteração. A cada iteração podem ocorrer atividades diferentes.

A cada iteração, o ciclo as atividades dos grupos de processos repetem-se, tal como o modelo iterativo visto na imagem 6. A diferença é a intensidade que são realizadas a cada iteração. Nas iterações iniciais ocorrem com mais intensidade atividades de iniciação e planejamento, nas seguintes de execução e nas finais de encerramento. Em todas iterações são realizadas atividades da área de conhecimento de Riscos, tando do grupo de processos de planejamento como de monitoramento e controle, que podem determinar a execução novamente de atividades de Iniciação e Planejamento nas próximas interações (ROCHA, 2013).

1.3.1.4 Modelo Iterativo e Incremental

É Um modelo baseado no Modelo em Espiral, visto em 1.3.1.3 e é o modelo mais utilizado atualmente para projetos de software, já que é o modelo que se baseiam as Metodologias Ágeis.

O modelo de ciclo de vida incremental e iterativo foi proposto como uma resposta aos problemas encontrados no modelo em cascata. Um processo de desenvolvimento

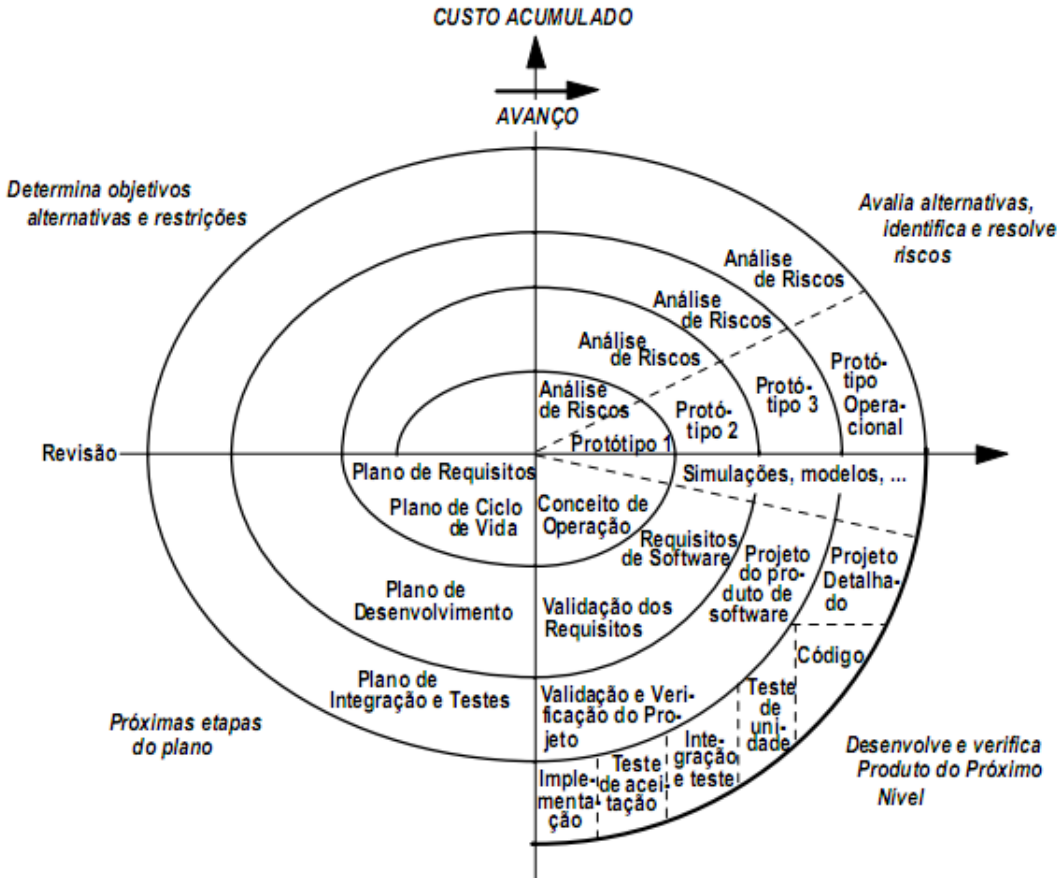

Figura 7 – Modelo Espiral
(ROCHA, 2013)

segundo essa abordagem divide o desenvolvimento de um produto de software em ciclos. Em cada ciclo de desenvolvimento, podem ser identificadas as fases de análise, projeto, implementação e testes. Essa característica contrasta com a abordagem clássica, na qual as fases de análise, projeto, implementação e testes são realizadas uma única vez.

Cada um dos ciclos considera um subconjunto de requisitos. Os requisitos são desenvolvidos uma vez que sejam alocados a um ciclo de desenvolvimento. No próximo ciclo, um outro subconjunto dos requisitos é considerado para ser desenvolvido, o que produz um novo incremento do sistema que contém extensões e refinamentos sobre o incremento anterior.

Assim, o desenvolvimento evolui em versões, através da construção incremental e iterativa de novas funcionalidades até que o sistema completo esteja construído. Apenas uma parte dos requisitos é considerada em cada ciclo de desenvolvimento. Na verdade, um modelo de ciclo de vida iterativo e incremental pode ser visto como uma generalização da abordagem em cascata: o software é desenvolvimento em incrementos e cada incremento é desenvolvido em cascata.

A abordagem incremental e iterativa somente é possível se existir um mecanismo para dividir os requisitos do sistema em partes, para que cada parte seja alocada a um ciclo de desenvolvimento. Essa alocação é realizada em função do grau de importância atribuído a

cada requisito (IFPR SC, 2006).

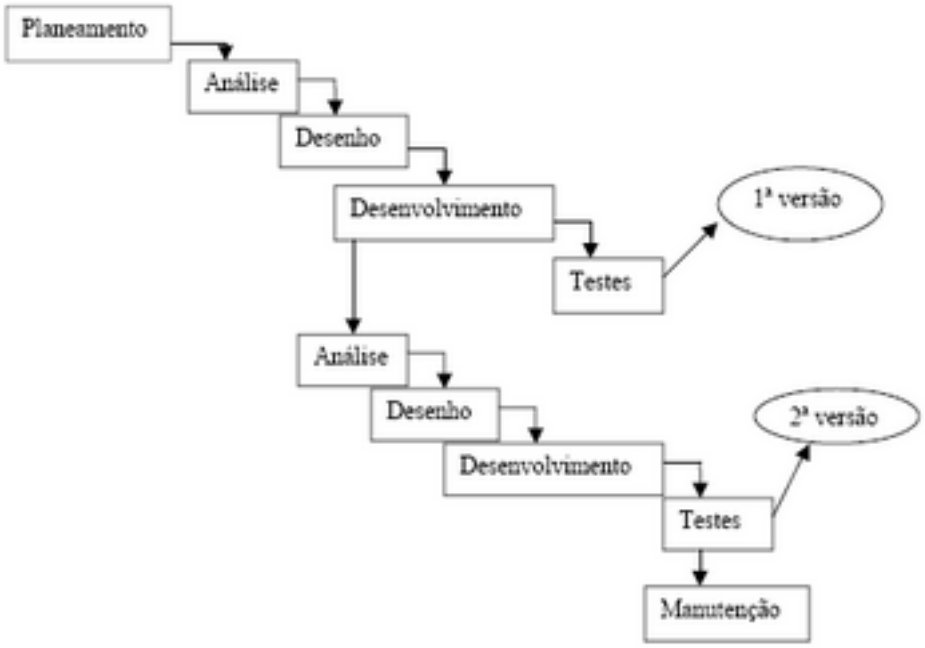

Figura 8 – Modelo Iterativo Incremental
(IFPR SC, 2006)

1.4 Metodologias Ágeis

Por todas as características que diferenciam um projeto de desenvolvimento de software dos projetos tradicionais, métodos tradicionais de Gerenciamento de Projetos costumam ser vistos como burocráticos e antiquados para o desenvolvimento de software. Por estes motivos ouve-se muito falar em Metodologias Ágeis de Desenvolvimento de Software e Gerenciamento de Projetos.

O termo "Metodologias Ágeis" tornou-se popular em 2001 quando dezessete especialistas em processos de desenvolvimento de software se reuniram em em uma estação de esqui em Utah, EUA e estabeleceram princípios comuns compartilhados por todos. Foi então criada a Aliança Ágil e o estabelecimento do Manifesto Ágil. Os conceitos chave do Manifesto Ágil são:

- Indivíduos e interação entre eles mais que processos e ferramentas
- Software em funcionamento mais que documentação abrangente
- Colaboração com o cliente mais que negociação de contratos
- Responder a mudanças mais que seguir um plano

"Isto é, ainda que haja valor nos itens à direita, valorizamos mais os itens à esquerda" (BECK et al., 2001).

Além dos conceitos, o Manifesto Ágil ainda se baseia nos doze princípios abaixo:

- Nossa maior prioridade é satisfazer o cliente, através da entrega adiantada e contínua de software de valor.
- Aceitar mudanças de requisitos, mesmo no fim do desenvolvimento. Processos ágeis se adequam a mudanças, para que o cliente possa tirar vantagens competitivas.
- Entregar software funcionando com frequência, na escala de semanas até meses, com preferência aos períodos mais curtos.
- Pessoas relacionadas à negócios e desenvolvedores devem trabalhar em conjunto e diariamente, durante todo o curso do projeto.
- Construir projetos ao redor de indivíduos motivados. Dando a eles o ambiente e suporte necessário, e confiar que farão seu trabalho.
- O Método mais eficiente e eficaz de transmitir informações para, e por dentro de um time de desenvolvimento, é através de uma conversa cara a cara.
- Software funcional é a medida primária de progresso.
- Processos ágeis promovem um ambiente sustentável. Os patrocinadores, desenvolvedores e usuários, devem ser capazes de manter indefinidamente, passos constantes.
- Contínua atenção à excelência técnica e bom design, aumenta a agilidade.
- Simplicidade: a arte de maximizar a quantidade de trabalho que não precisou ser feito.
- As melhores arquiteturas, requisitos e designs emergem de times auto-organizáveis.
- Em intervalos regulares, o time reflete em como ficar mais efetivo, então, se ajustam e otimizam seu comportamento de acordo.

1.4.1 Scrum

Em 1986, Hirotaka Takeuchi e Ikujiro Nonaka descreveram método para aumentar a velocidade e flexibilidade do desenvolvimento de novos produtos no artigo "*The New Product Development Game*". Inicialmente tratava-se de um método de Gerenciamento de Projetos em empresas de fabricação de automóveis e produtos de consumo. Eles compararam este novo método holístico com fases que se sobrepõem e com um time multifuncional ao *Rugby*, onde o time tenta avançar, passando a bola para trás e para frente. Em 1991, DeGrace e Stahl em "*Wicked Problems, Righteous Solutions*" referem-se a este método como Scrum, um termo do próprio *Rugby*.

Em 1990, Ken Schwaber usou a metodologia Scrum em sua empresa, Advanced Development Methods, e ao mesmo tempo, Jeff Sutherland desenvolveu técnicas parecidas na empresa Easel Corporation. Em 1995, Sutherland e Schwaber apresentaram em conjunto um artigo na OOPSLA'95 onde descreviam suas experiências, e no ano seguinte trabalharam juntos na produção do livro "*Agile Software Development with Scrum*" (SCHWABER; BEEDLE, 2001).

1.4.1.1 Definição

Scrum é um *framework* para desenvolver e manter produtos, dentro do qual pessoas podem tratar e resolver problemas complexos e adaptativos, enquanto produtiva e criativamente entregam produtos com o mais alto valor possível. A definição do Scrum consiste em papéis, eventos, artefatos e as regras do Scrum que unem os demais e os mantém integrados.

Scrum não é um processo ou uma técnica para construir produtos; em vez disso, é um *framework* dentro do qual pode-se empregar vários processos ou técnicas. O Scrum deixa claro a eficácia relativa das práticas de gerenciamento e desenvolvimento de produtos, de modo que se possa melhorá-las (SCHWABER; SUTHERLAND, 2014).

1.4.1.2 Teoria

Scrum é fundamentado nas teorias empíricas de controle de processo, ou empirismo. O empirismo afirma que o conhecimento vem da experiência e de tomada de decisões baseadas no que é conhecido. O Scrum emprega uma abordagem iterativa e incremental para aperfeiçoar a previsibilidade e o controle de riscos.

Três pilares apoiam a implementação de controle de processo empírico (SCHWABER; SUTHERLAND, 2014):

- Transparência: Aspectos significativos do processo devem estar visíveis aos responsáveis pelos resultados. Esta transparência requer aspectos definidos por um padrão comum para que os observadores compartilharem um mesmo entendimento do que está sendo visto.

- Inspeção: Os usuários Scrum devem, frequentemente, inspecionar os artefatos Scrum e o progresso em direção a detectar variações. Esta inspeção não deve, no entanto, ser tão frequente que atrapalhe a própria execução das tarefas. As inspeções são mais benéficas quando realizadas de forma diligente por inspetores especializados no trabalho a se verificar.

- Adaptação: Se um inspetor determina que um ou mais aspectos de um processo desviou para fora dos limites aceitáveis, e que o produto resultado será inaceitável, o

processo ou o material sendo produzido deve ser ajustado. O ajuste deve ser realizado o mais breve possível para minimizar mais desvios.

1.4.1.3 Papéis

O Time Scrum é composto pelo *Product Owner*, o Time de Desenvolvimento e o *Scrum Master*. Times Scrum são auto-organizáveis e multifuncionais. Times auto-organizáveis escolhem qual a melhor forma para completarem seu trabalho, em vez de serem dirigidos por outros de fora do Time. Times multifuncionais possuem todas as competências necessárias para completar o trabalho sem depender de outros que não fazem parte da equipe (SCHWABER; SUTHERLAND, 2014).

1.4.1.3.1 Product Owner

O *Product Owner*, ou dono do produto, é o responsável por maximizar o valor do produto e do trabalho do Time de Desenvolvimento. Como isso é feito pode variar amplamente através das organizações, Times Scrum e indivíduos.

1.4.1.3.2 Time de Desenvolvimento

O Time de Desenvolvimento consiste de profissionais que realizam o trabalho de entregar uma versão usável que potencialmente incrementa o produto ao final de cada *Sprint*. Somente integrantes do Time de Desenvolvimento criam incrementos.

1.4.1.3.3 Scrum Master

O *Scrum Master* é responsável por garantir que o Scrum seja entendido e aplicado. O *Scrum Master* faz isso para garantir que o Time Scrum adere à teoria, práticas e regras do Scrum. O *Scrum Master* é um servo-líder para o Time Scrum.

O *Scrum Master* ajuda aqueles que estão fora do Time Scrum a entender quais as suas interações com o Time Scrum são úteis e quais não são. O *Scrum Master* ajuda todos a mudarem estas interações para maximizar o valor criado pelo Time Scrum.

1.4.1.4 Eventos

Eventos prescritos são usados no Scrum para criar uma rotina e minimizar a necessidade de reuniões não definidas no Scrum. Todos os eventos são eventos *time-boxed*, de tal modo que todo evento tem uma duração máxima. Uma vez que a *Sprint* começa, sua duração é fixada e não pode ser reduzida ou aumentada. Os eventos restantes podem terminar sempre que o propósito do evento é alcançado, garantindo que uma quantidade adequada de tempo seja gasta sem permitir perdas no processo (SCHWABER; SUTHERLAND, 2014).

1.4.1.4.1 Sprint

O coração do Scrum é a *Sprint*, um *time-boxed* de um mês ou menos, durante o qual uma versão incremental potencialmente utilizável do produto é criada. *Sprints* tem durações coerentes em todo o esforço de desenvolvimento. Uma nova *Sprint* inicia imediatamente após a conclusão da *Sprint* anterior.

As *Sprints* são compostas por uma reunião de planejamento da *Sprint*, reuniões diárias, o trabalho de desenvolvimento, uma revisão da *Sprint* e a retrospectiva da *Sprint*.

1.4.1.4.2 Reunião de Planejamento da *Sprint*

O trabalho a ser realizado na *Sprint* é planejado na reunião de planejamento da *Sprint*. Este plano é criado com o trabalho colaborativo de todo o Time Scrum.

Reunião de planejamento da *Sprint* possui um time-box com no máximo oito horas para uma *Sprint* de um mês de duração. Para *Sprint* menores, este evento é usualmente menor. O *Scrum Master* garante que o evento ocorra e que os participantes entendam seu propósito. O *Scrum Master* ensina o Time Scrum a manter-se dentro dos limites do time-box.

A reunião de planejamento da *Sprint* responde as seguintes questões:

- O que pode ser entregue como resultado do incremento da próxima *Sprint*?
- Como o trabalho necessário para entregar o incremento será realizado?

1.4.1.4.3 Reunião Diária

A Reunião Diária do Scrum é um evento *time-boxed* de 15 minutos, para que o Time de Desenvolvimento possa sincronizar as atividades e criar um plano para as próximas 24 horas. Esta reunião é feita para inspecionar o trabalho desde a última Reunião Diária, e prever o trabalho que deverá ser feito antes da próxima Reunião Diária.

A Reunião Diária é mantida no mesmo horário e local todo dia para reduzir a complexidade. Durante a reunião os membros do Time de Desenvolvimento esclarecem:

- O que eu fiz ontem que ajudou o Time de Desenvolvimento a atender a meta da *Sprint*?
- O que eu farei hoje para ajudar o Time de Desenvolvimento atender a meta da *Sprint*?
- Eu vejo algum obstáculo que impeça a mim ou o Time de Desenvolvimento no atendimento da meta da *Sprint*?

1.4.1.4.4 Revisão da *Sprint*

A Revisão da *Sprint* é executada no final da *Sprint* para inspecionar o incremento e adaptar o *Backlog* do Produto se necessário. Durante a reunião de Revisão da *Sprint* o Time Scrum e as partes interessadas colaboram sobre o que foi feito na *Sprint*. Com base nisso e em qualquer mudança no *Backlog* do Produto durante a *Sprint*, os participantes colaboram nas próximas coisas que podem ser feitas para otimizar valor. Esta é uma reunião informal, não uma reunião de status, e a apresentação do incremento destina-se a motivar e obter comentários e promover a colaboração.

Esta é uma reunião *time-boxed* de 4 horas de duração para uma *Sprint* de um mês. Para *Sprint* menores, este evento é usualmente menor. O *Scrum Master* garante que o evento ocorra e que os participantes entendam o seu objetivo. O *Scrum Master* ensina a todos a manter a reunião dentro dos limites do *Time-box*.

1.4.1.4.5 Retrospectiva da *Sprint*

A Retrospectiva da *Sprint* é uma oportunidade para o Time Scrum inspecionar a si próprio e criar um plano para melhorias a serem aplicadas na próxima *Sprint*.

A Retrospectiva da *Sprint* ocorre depois da Revisão da *Sprint* e antes da reunião de planejamento da próxima *Sprint*. Esta é uma reunião *time-boxed* de três horas para uma *Sprint* de um mês. Para *Sprint* menores, este evento é usualmente menor. O *Scrum Master* garante que o evento ocorra e que os participantes entendam seu propósito. O *Scrum Master* ensina todos a mantê-lo dentro do *time-box*. O *Scrum Master* participa da reunião como um membro auxiliar do time devido a sua responsabilidade pelo processo Scrum.

1.4.1.5 Artefatos

1.4.1.5.1 *Backlog* do Produto

O *Backlog* do Produto é uma lista ordenada de tudo que deve ser necessário no produto, e é uma origem única dos requisitos para qualquer mudança a ser feita no produto. O *Product Owner* é responsável pelo *Backlog* do Produto, incluindo seu conteúdo, disponibilidade e ordenação (SCHWABER; SUTHERLAND, 2014).

Um *Backlog* do Produto nunca está completo. Os primeiros desenvolvimentos apenas estabelecem os requisitos inicialmente conhecidos e melhor entendidos. O *Backlog* do Produto evolui tanto quanto o produto e o ambiente no qual ele será utilizado evoluem. O *Backlog* do Produto é dinâmico; mudando constantemente para identificar o que o produto necessita para ser mais apropriado, competitivo e útil. O *Backlog* do Produto existirá enquanto o produto também existir.

1.4.1.5.2 *Backlog* da *Sprint*

O *Backlog* da *Sprint* é um conjunto de itens do *Backlog* do Produto selecionados para a *Sprint*, juntamente com o plano para entregar o incremento do produto e atingir o objetivo da *Sprint*. O *Backlog* da *Sprint* é a previsão do Time de Desenvolvimento sobre qual funcionalidade estará no próximo incremento e sobre o trabalho necessário para entregar essa funcionalidade em um incremento "Pronto".

1.4.1.5.3 Incremento

O incremento é a soma de todos os itens do *Backlog* do Produto completados durante a *Sprint* e o valor dos incrementos de todas as *Sprints* anteriores. Ao final da *Sprint* um novo incremento deve estar "Pronto", o que significa que deve estar na condição utilizável e atender a definição de "Pronto" do Time Scrum. Este deve estar na condição utilizável independente do *Product Owner* decidir por liberá-lo realmente ou não.

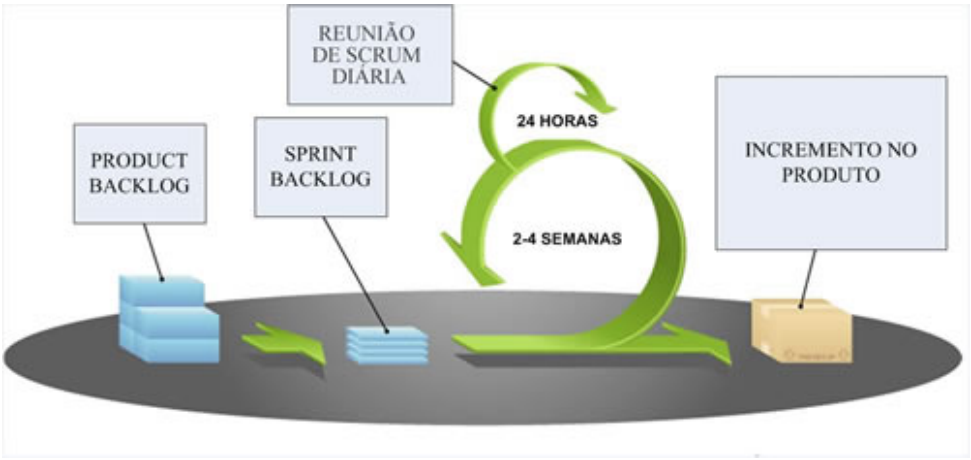

Figura 9 – Ciclo Scrum
(B2ML, 2014)

2 A Quebra do Paradigma Tradicional

Muitos conceitos utilizados e pregados pelas Métodos Ágeis representam uma quebra de paradigmas de um modelo rígido e tradicional, comumente utilizado. Por exemplo, a documentação do Projeto. O modelo tradicional, baseado nas práticas do PMBOK requer uma série de documentos formais e padronizados para o gerenciamento do projeto. Já as Metodologias Ágeis focam no produto e na documentação estritamente necessária, ou seja funcional, de arquitetura e para facilitar as manutenções.

Talvez o maior impactante em Métodos Ágeis, como o Scrum, é o fato de não haver uma figura como o Gerente do Projeto, não há um líder para o Projeto, alguém que imponha autoridade no processo. Existe uma responsabilidade compartilhada, pois o Time de Desenvolvimento é auto-organizado e tem autonomia para decidir como ira realizar as tarefas e realizar as entregas. O *Product Owner*, que seria a figura mais próxima a um Gerente de Projetos não pode interferir no Time de Desenvolvimento. Cabe a ele apenas priorizar as entregas que agregam maior valor ao produto. Isso gera a forte premissa de que a equipe que utiliza este tipo de metodologia deve ser formada por profissionais mais experientes, já que não existe a figura de um gestor, mas sim uma equipe autogerenciável.

Outro fator bastante diferente é o fato de poder disponibilizar entregas para o cliente de forma constante. No modelo utilizado tradicionalmente, o cliente participa no levantamento de escopo, pois é o mesmo que possui os requisitos a serem entregues. Após isso, é firmado um contrato entre o gerente do projeto e o cliente, com o escopo acordado, tempo, custo e demais informações importante, e o cliente receberá o produto apenas ao final do projeto. Os Métodos Ágeis primam justamente pelo contrário, entregas constantes para diminuir riscos e facilitar adaptações.

O Scrum, por exemplo, não possui processos de gerenciamento para áreas como custos ou aquisições. A aplicação é focada para a etapa de execução, controle e mudanças do projeto, onde a equipe precisa produzir e entregar os itens do *backlog* para poder formar um produto entregável. Já nas metodologias tradicionais de gerenciamento de projetos, a preocupação e gerenciamento dos custos é um dos itens mais importantes do projeto e a metodologia é focada no planejamento do mesmo.

As entregas parciais durante o projeto é um fator chave dos Métodos Ágeis e acarreta na grande diferença entre eles e os modelos tradicionais, a aceitação e acolhimento de mudanças. O fato do cliente ver o produto enquanto ainda em fase de elaboração estimula o pedido de mudanças e adaptações por parte do mesmo, mitigando grandes riscos e retrabalhos. Este fator é fundamental para o mundo de Desenvolvimento de Software, pois no início do projeto o cliente não sabe ao certo o que quer, pois pouquíssimas pessoas entendem realmente o que é um software e o que ele pode fazer, diferentemente do processo de construção civil, por exemplo, que é físico e do conhecimento de todos. Não evitar

mudanças é um aspecto intrínseco de um software, elas são bem-vindas e necessárias, pois este se adéqua às necessidades do negócio, ou seja, ele nunca está pronto enquanto o negócio existir.

Ainda fazendo uma comparação com construção civil, é muito difícil mudar uma parede de lugar depois que a mesma já está construída, pode ser até impossível dependendo da estrutura. Software é flexível, pois é uma entidade abstrata e por isso tem a mudança como um fator intrínseco. Por este motivo é que Engenharias tradicionais projetam e planejam todo o projeto antes de iniciar a execução física, utilizando o Modelo Cascata, visto em 1.3.1.1. Em Engenharia de Software normalmente se utiliza o Modelo Iterativo e Incremental, visto em 1.3.1.4, para planejar apena o imediato, já que ao elevado número de mudanças não faz sentido um planejamento a longo prazo.

Devido ainda ao fato de mudanças constante, outro fator, já citado, é bastante diferente entre as abordagens, que é a presença do cliente. No modelo tradicional o cliente participa do levantamento de escopo e após isso, normalmente o próximo contato será na entrega do produto. Nos Métodos Ágeis, o relacionamento com o cliente é constante, pois as entregas são constantes e mudanças e aperfeiçoamentos são sempre bem-vindos, já que o foco é sempre a satisfação do cliente.

3 As Afinidades

Apesar de à primeira vista parecerem totalmente opostos, o PMBOK, como representante da abordagem tradicional e o Scrum, representando os Métodos Ágeis têm mais em comum do que parecem, a começar pelo objetivo em comum. Independente do sistema utilizado, o objetivo de ambas abordagens é trilhar um caminho para que um projeto possa obter sucesso.

Ambas abordagens são iterativas sob a ótica de planejamento. Sim, o PMBOK é iterativo. Quando o mesmo é citado neste trabalho como um Método Cascata é devida a uma visão de projeto mais ampla, do seu início à conclusão do projeto, então a forma com que ele é utilizado é primeiro planejar tudo para depois executar, ao contrário dos Métodos Ágeis. Porém dentro da Fase de Planejamento o processo é iterativo e incremental, assim como os Métodos Ágeis.

Nenhuma das abordagens citadas é uma metodologia. o PMBOK possui boas práticas para se gerenciar um projeto, ou seja, é um modelo que contém os processos adequados e mais utilizados para o gerenciamento de projetos no modelo tradicional. O Scrum é por definição um *framework*, isto é, foi feito para ser base e receber novos processos para a gestão dos projetos.

Ao natural, sem nenhuma inserção no *framework* Scrum, o mesmo possui um foco bastante claro na implementação de software e não no gerenciamento formal como um todo. Já o PMBOK é muito forte neste aspecto. Possui documentos e informações bem estruturadas que por vezes fazem falta aos Métodos Ágeis.

Os Métodos Ágeis não se preocupam com o gerenciamento formal de tempo, custo, recursos humanos, comunicações, entre outras áreas. Estas são informações muito importantes do ponto de vista da organização onde os Métodos Ágeis estão inseridos e é uma lacuna que os processos do PMBOK podem suprir. Já o gerenciamento de mudanças do Ágil é mais veloz e menos burocrático que o PMBOK.

Enfim, além das semelhanças entre as duas abordagens, as mesmas se complementam. Onde uma possui defeitos a outra possui virtudes que podem ser utilizadas para sanar estes defeitos. Na figura 10 podem ser visualizados os processos que compões a 5ª Edição do PMBOK. Esta visão deixa claro que o PMBOK também pode ser visto como um *framework*.

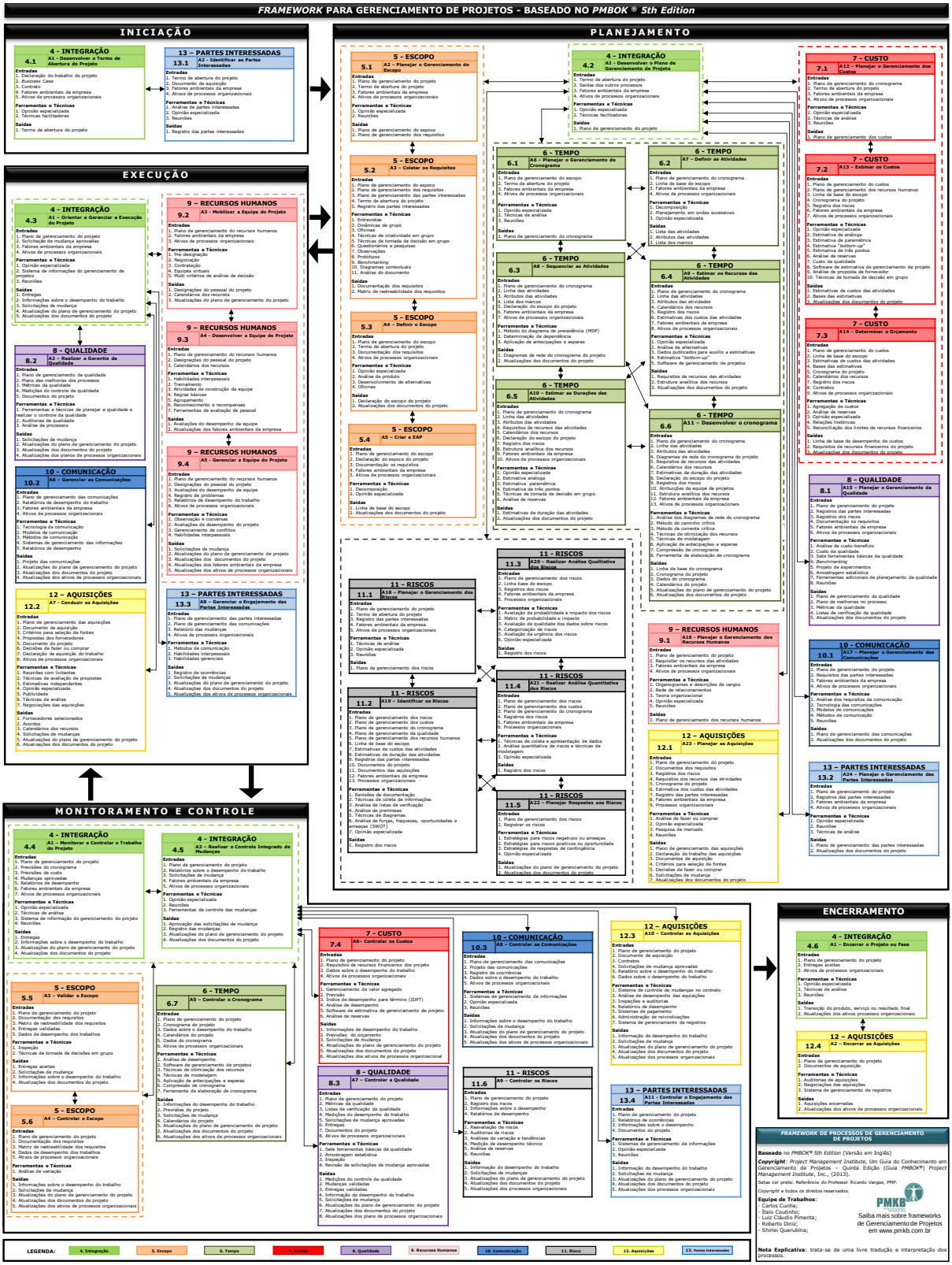

Figura 10 – Processos propostos pelo PMBOK (PMKB, 2013)

4 A Escolha da Abordagem

Quando se questiona se Scrum é melhor que PMBOK para um projeto, ou vice e versa, a questão a ser verificada é a complexidade do projeto. Essa classificação de complexidade do projeto é feita em função da tecnologia e dos requisitos envolvidos. Os requisitos do projeto, por exemplo, podem ser classificados em uma escala de certo a incerto, e a tecnologia ou métodos e técnicas envolvidas na execução do projeto podem ser classificada em uma escala de conhecida a desconhecida. A classificação da complexidade de um projeto é subjetiva, e depende de cada equipe e organização que irá realizá-lo (BUCK, 2012).

A figura 11 ilustra o gráfico de complexidade do projeto, as zonas onde ele pode está localizado e as técnicas mais apropriadas para a gestão do projeto, dependendo de sua classificação.

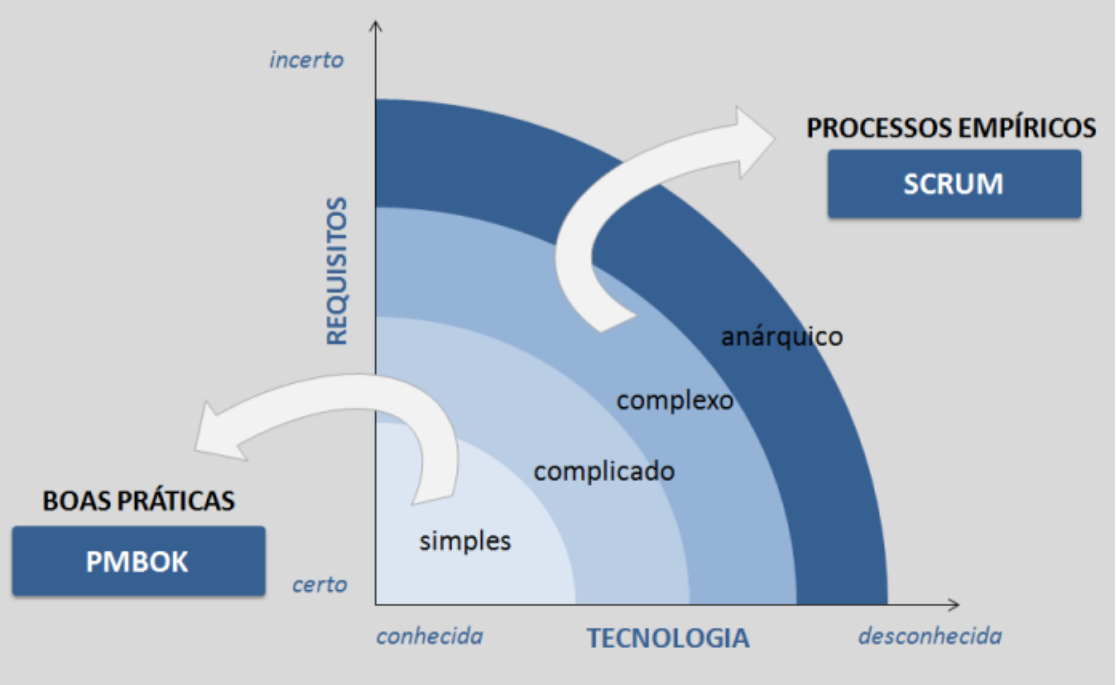

Figura 11 – Classificação de um Projeto
(BUCK, 2012)

Por ser um processo empírico, o Scrum é mais apropriado para projetos complexos. Isso porque ele não define 100% os processos que devem ser seguidos para que a equipe atinja os objetivos do projeto. O Scrum define apenas processos base para que outros processos (que explorem mais os requisitos e a tecnologia) sejam executados, de forma que o projeto gere valor de negócio desde o início.

PMBOK é mais apropriado para projetos simples. Isso porque o PMBOK tem coletado uma série de processos que tem funcionado para projetos onde os requisitos e a tecnologia são bem conhecidos no início da empreitada. São processos que lidam com a gestão do

escopo, do tempo, entre outros, e fornecem uma série de indicadores excelentes para projetos dessa natureza, como construção civil. Para projetos simples, as boas práticas praticamente indicam os processos que devem ser seguidos para o projeto ser um sucesso.

Para projetos complicados, tanto o Scrum quanto o PMBOK podem ser utilizados. Contudo, lembrando-se que a classificação é subjetiva e essa fronteira entre projetos simples e complexos nem sempre é bem definida. É razoavelmente trivial saber se um projeto tende mais para complexo ou mais para simples, mas não é imediato saber se ele está exatamente na zona do complicado. Pode-se usar o aspecto de certeza ou incerteza sobre os requisitos para decidir entre Scrum ou PMBOK.

Sobre projetos anárquicos não há muito o que se fazer. São aqueles que, geralmente, o risco de realizá-lo praticamente inviabiliza sua execução.

4.1 Uma Nova Metodologia

Mesmo existindo meios de classificar projetos e, através disso, enquadrá-los em uma melhor abordagem para o gerenciamento dos mesmos, é possível mesclar os benefícios de várias abordagens e criar uma metodologia que sirva para várias organizações e equipes que lidem com projetos de determinadas características. Já que independente da classificação do projeto, alguns elementos ou aspectos podem ser mais ou menos complexos que o projeto.

O foco deste trabalho são projetos complexos, pelos critérios descritos no capítulo 4, por ser a classificação da maioria absoluta dos projetos na área de Desenvolvimento de Software. Ainda que o projeto seja enquadrado como complexo e a escolha apropriada seja a utilização de Métodos Ágeis, como o Scrum, também é possível utilizar as boas práticas do PMBOK para complementar o Scrum.

É importante ressaltar que nem Scrum nem PMBOK são Metodologias de Gerenciamento de Projetos. O PMBOK é um guia de boas práticas de projetos e vem servindo para base na Gerenciamento de Projetos de sucesso durante muitos anos e foi gerado através de muito conhecimento já adquirido. Ou seja, ele serve para orientar os processos a serem realizados para o gerenciamento de projetos de forma genérica. Já o Scrum, é um *framework* para gerenciamento projetos e desenvolvimento de produtos. Isso significa que o mesmo já foi criado com o intuito de agregar processos para serem executados através da "engrenagem" Scrum. Prova disso é que a grande maioria das implementações das técnicas de gerenciamento do Scrum se utilizam de técnicas e artefatos de outras Metodologias Ágeis, como o uso de *User Histories*, *Pair Programming* e *Stand Up Meeting*, todas estas originárias do Extreme Programming.

Um casamento entre as duas abordagens parece realmente ser algo muito interessante, gerando assim uma Metodologia a ser seguida. Inclusive pode se trazer as práticas de outras Metodologias Ágeis para esta, como citado anteriormente sobre as práticas de

Extreme Programming normalmente utilizadas juntamente com o Scrum.

5 Metodologia Proposta

5.1 A União Teórica

É importante reforçar que o PMBOK possui diversos processos que abrangem todas as fases de um projeto do início ao fim, porém não descreve como estes processos deverão ser utilizados e quais os momentos ideais. Já o Scrum, metodologia ágil que foi escolhida pelo seu grande uso e versatilidade para ser o foco como representante de métodos ágeis neste trabalho, possui os eventos claramente posicionados no tempo conforme o andamento do projeto e possuem uma definição clara sobre papéis, artefatos, eventos, regras e como implementá-los.

Ao pensar em gerenciamento ágil, o foco volta-se automaticamente para a execução de um projeto, o objetivo desta proposta é trazer valores de gerenciamento tradicional sem burocratizar o processo, porém fortalecendo o foco em planejamento e controle de áreas como custos, riscos, aquisições e etc.

Muitos clientes fazem questão também de certos documentos formais sobre o projeto como termo de abertura, por exemplo, até mesmo por garantia legal e por este motivo têm certo receio dos Métodos Ágeis. Este é mais um ponto a favor da união dos processos do PMBOK ao ciclo de vida Scrum.

A metodologia proposta tem como base a metodologia elaborada em (CRUZ, 2013) e partirá do princípio que o Scrum, como é um *framework* suportará os processos do PMBOK dentro de suas engrenagens e eventos temporais. Como o Scrum foca em projetos que podem ser desenvolvidos de forma ágil, cabe aos processos do PMBOK abrangerem as demais especificidades de outros projetos e fazê-los funcionar de forma ágil. Por isso mesmo após da união das técnicas, os princípios dos métodos ágeis deverão permanecer, como:

- Não burocratização;
- Sem documentações extensas;
- Processos enxutos;
- Entregas constantes;
- Manter Time de Desenvolvimento focado.

5.2 O Time

Assim como em (CRUZ, 2013) o time do projeto será composto por:

- **Time de desenvolvimento:** Assim como no Scrum, utiliza-se o termo Time de Desenvolvimento para referir-se à equipe de execução, a equipe responsável por realizar a entrega do trabalho planejado;
- **Product Owner:** Mesmo papel que no Scrum, representa os interesses do cliente frente ao projeto. Podendo ser o próprio cliente;
- **Scrum Master:** Também com o mesmo papel do Scrum, possui a responsabilidade de garantir que a metodologia está sendo seguida e deve retirar empecilhos que afetem o Time de Desenvolvimento;
- **Gerente do Projeto:** Figura do Gerente de Projetos tradicional, podendo também ser uma equipe de projetos, com vários gerentes de projetos.

É importante frisar que para esta metodologia que está sendo proposta, é aceitável que uma mesma pessoa ocupe a posição de *Scrum Master* e Gerente do Projeto, dependendo do tamanho do projeto e da expertise da pessoa em questão.

5.3 A União dos Processos

Parte-se do princípio que o leitor deste trabalho possui conhecimento vasto tanto nas boas práticas do PMBOK e seus processos, quanto no *framework* Scrum. Por este motivo, não será detalhado nenhum dos processos envolvidos na metodologia proposta.

Neste trabalho sempre que um processo do PMBOK for citado, o mesmo será acompanhado do seu respectivo número de uso na 5ª Edição do PMBOK. Também apresentados na figura 10.

5.3.1 Processos de Iniciação

No Scrum não existe nenhuma preocupação com a formalização de início ou encerramento de um projeto, ao contrário do que prega as boas práticas do PMBOK. Porém, pode-se perceber que esta prática é realmente muito interessante e bem apreciada durante os projetos, principalmente pelo alinhamento de expectativas com o cliente/*sponsor* e por melhorar a sinergia entre equipe de projeto e cliente, além de alinhamentos sobre o escopo e restrições iniciais do mesmo.

Outra prática fundamental é o levantamento dos *stakeholders*, muitas vezes não tratados com a devida importância e que podem ser tornar grandes empecilhos ou até mesmo aliados ao projeto. Não existe o termo partes interessadas ou *stakeholders* nos Métodos Ágeis, porém esta é uma figura muito presente em todo processo. Logo, a única modificação é o registro dos mesmos, como propõe o PMBOK. O registro e classificação dos *stakeholders* irá influenciar na frequência e na participação de determinados *stakeholders* nas reuniões de Sprint Review.

Por estes motivos serão mantidos os Processos de Iniciação pregados pelo PMBOK:

- Desenvolver o Termo de Abertura do projeto [4.1];
- Identificar as Partes Interessadas [13.1].

Ambos os processos são de responsabilidade do Gerente do Projeto e do *Product Owner* e deverão ser realizados no início do projeto, antes mesmo de qualquer iteração Scrum.

5.3.2 Processos de Planejamento

Na fase de Planejamento do PMBOK existem os seguintes processos:

- Desenvolver o Plano de Projeto [4.2];
- Planejar o Gerenciamento do Escopo [5.1];
- Coletar os Requisitos [5.2];
- Definir o Escopo [5.3];
- Criar a EAP [5.4];
- Planejar o Gerenciamento do Cronograma [6.1];
- Definir as Atividades [6.2];
- Sequenciar as Atividades [6.3];
- Estimar os Recursos das Atividades [6.4];
- Estimar as Durações das Atividades [6.5];
- Desenvolver o Cronograma [6.6];
- Planejar o Gerenciamento dos Custos [7.1];
- Estimar os Custos [7.2];
- Determinar o Orçamento [7.3];
- Planejar o Gerenciamento da Qualidade [8.1];
- Planejar o Gerenciamento dos Recursos Humanos [9.1];
- Planejar o Gerenciamento das Comunicações [10.1];
- Planejar o Gerenciamento dos Riscos [11.1];
- Identificar os Riscos [11.2];

- Realizar Análise Qualitativa dos Riscos [11.3];
- Realizar Análise Quantitativa dos Riscos [11.4];
- Planejar Respostas aos Riscos [11.5];
- Planejar o Gerenciamento das Aquisições [12.1];
- Planejar o Gerenciamento das Partes Interessadas [13.2].

Muitos destes processos são relacionados a documentar e formalizar como o gerenciamento de determinadas áreas do conhecimento ocorrerá. Esta formalização não é utilizada pelos métodos ágeis, porém não podem ser consideradas burocráticas ou desnecessárias. Utilizar estes processos citados a seguir é opcional para o gerenciamento utilizando esta metodologia, mas é bastante aconselhável segui-los. Esta decisão irá variar conforme o tamanho e complexidade do projeto e principalmente da cultura organizacional onde o mesmo está inserido, se esta presa por documentações formalizadas ou não.

- Desenvolver o Plano de Projeto [4.2];
- Planejar o Gerenciamento do Escopo [5.1];
- Planejar o Gerenciamento do Cronograma [6.1];
- Planejar o Gerenciamento dos Custos [7.1];
- Planejar o Gerenciamento da Qualidade [8.1];
- Planejar o Gerenciamento dos Recursos Humanos [9.1];
- Planejar o Gerenciamento das Comunicações [10.1];
- Planejar o Gerenciamento dos Riscos [11.1];
- Planejar o Gerenciamento das Aquisições [12.1];
- Planejar o Gerenciamento das Partes Interessadas [13.2].

Todos estes processos são de responsabilidade do Gerente do Projeto e devem descrever como ocorrerá o gerenciamento do projeto e das áreas de conhecimento. Neles devem ser descritos inclusive os rituais do Scrum e das Metodologias Ágeis que serão utilizadas no gerenciamento do projeto. Por exemplo, o gerenciamento de escopo incluirá o *backlog* do produto, o gerenciamento das partes interessadas incluirá as *Sprint Reviews* e assim por diante. Esses processos e métodos que serão utilizados ficarão mais evidentes no decorrer do trabalho, quando forem apresentados os demais processos das áreas de conhecimento.

Ao utilizar o Scrum, é muito comum realizar a denominada Sprint Zero. Uma *sprint* sem um prazo definido, ou seja, não é um evento *time-boxed* como as demais *sprints*, que

serve para preparar os artefatos que serão utilizados durante o projeto, como o *backlog* inicial, reunir o Time de Desenvolvimento, apresentar o projeto, iniciar a mensuração de velocidade do time, entre outros. Ou seja, a Sprint Zero nada mais é que um período de planejamento para a Sprint 1. Assim sendo, muito dos processos de planejamento sugeridos pelo PMBOK são realizados naturalmente em uma Sprint Zero.

Para criar um *backlog* inicial do produto pode-se utilizar os processos de Gerenciamento de Tempo:

- Coletar os Requisitos [5.2];
- Definir o Escopo [5.3];
- Criar a EAP [5.4].

Na verdade os dois primeiros processos já são utilizados empiricamente na criação do *backlog* no Scrum, e não apenas no início do projeto, mas em todo início de *sprint*, pois o *backlog* é vivo e é revisto, detalhado e priorizado a cada nova iteração. Como a participação das áreas interessas é uma premissa muito forte nas Metodologias Ágeis, requisitos são coletados o tempo todo para definir o escopo.

O mais interessante é a utilização do processo "Criar a EAP [5.4]" para auxiliar na elaboração das *user stories* que formarão o *backlog* inicial e formalizá-las como o escopo inicial. Outro conceito que surge neste momento são as histórias de usuários, ou *user stories*. Uma forma padrão de descrever requisitos muito utilizada no Scrum, apesar de ter sido popularizada por outra Metodologia Ágil, o Extreme Programming, e não fazer parte dos artefatos oficiais do Scrum. Nesta metodologia mista, pode-se criar a EAP utilizando *user stories* no lugar de pacotes de entregas, já que o *backlog* será formado por estas histórias.

A responsabilidade por estes processos que irão gerar o *backlog* é do Gerente do Projeto e principalmente do *Product Owner*. Os demais membros do projeto podem sugerir requisitos e alterações de escopo, porém a decisão final fica a cargo do *Product Owner*.

Gerenciamento de Tempo é fundamental para qualquer projeto, mas neste ponto PMBOK e Scrum possuem abordagens um pouco distintas. O *backlog* do produto deverá ser estimado, através do *Planning Poker* e priorizado. Isto se assemelha muito aos seguintes processos do PMBOK:

- Sequenciar as Atividades [6.3];
- Estimar as Durações das Atividades [6.5].

A grande diferença é que no método tradicional as entregas são quebradas em atividades para serem sequenciadas e estimadas, já no Scrum e consequentemente nesta metodologia proposta os requisitos, em forma de *user stories* é que são estimados e priorizados. Estes requisitos serão quebrados em atividades apenas após a definição da *sprint backlog* na

segunda parte da reunião de planejamento da *sprint*, assim executando o seguinte processo do PMBOK:

- Definir as Atividades [6.2].

A priorização do *backlog*, ou seja, do processo "Sequenciar as Atividades [6.3]", é de única responsabilidade do *Product Owner*. Já as estimativas das *user stories* e sua decomposição em atividades, respectivamente os processos "Estimar as Durações das Atividades [6.5]" e "Definir as Atividades [6.2]", são de única e exclusiva responsabilidade do Time de Desenvolvimento. Estes processos não ocorrem apenas na Sprint Zero, mas no planejamento de todas as *sprints*. No caso da priorização, o ritual de *grooming* é realizado como requisito para realizar o planejamento das *sprints*.

Como o Scrum parte do princípio que o Time de Desenvolvimento será único durante o projeto e este time irá desenvolver todas as atividades necessárias para a conclusão do mesmo, não existe nenhum processo de alocação de recursos, até porque o Scrum prega um time auto-gerenciável onde o time seleciona e distribui suas atividades. Nesta metodologia será respeitado o mesmo princípio, porém será tomado um cuidado que talvez não seja muito claro no Scrum, a estimativa dos recursos materiais a serem utilizados. No Scrum é dito que o *Scrum Master* é responsável por retirar os empecilhos que possam afetar o Time de Desenvolvimento. Nesta metodologia, será deixado a cargo do Gerente de Projetos e do *Scrum Master* o levantamento de recursos a serem utilizados durante a execução das tarefas no projeto, assim contemplando o seguinte processo, também de Gerenciamento de Tempo:

- Estimar os Recursos das Atividades [6.4].

Um passo fundamental a ser tratado na Sprint Zero é o Planejamento das *Releases*, ou seja, planejar quando e quais serão os pontos de entregas de um potencial produto ao cliente, e o que cada *release* deverá conter. Podemos considerar este plano de *releases* como o cronograma a ser cumprido, logo temos o último processo de planejamento de tempo:

- Desenvolver o Cronograma [6.6].

A responsabilidade da definição deste cronograma é do Gerente de Projetos, juntamente com o *Product Owner*.

Um ponto que o Scrum não trata, ou sequer cita, é o Gerenciamento de Custos, passo fundamental para muitos projetos. Isso corrobora ainda mais com esta proposta de metodologia mista, pois assim esta fraqueza será suprida e demonstra também a importância da figura do Gerente de Projetos. Como este não é um ponto coberto pelo Scrum, seu gerenciamento será totalmente realizado através dos processos do PMBOK, que são:

- Estimar os Custos [7.2];

- Determinar o Orçamento [7.3].

Estes processos serão executados em paralelo ao ciclo do Scrum e assim como no PMBOK, são de responsabilidade do Gerente do Projeto.

Muitas definições são tomadas durante a Sprint Zero, como a duração das *sprints*, a definição de *done*, a velocidade do Time de Desenvolvimentos, etc. Estas decisões são tomadas por todo o Time Scrum em reuniões. Nesta metodologia, será aproveitado estes momentos em que toda a equipe esta reunida para, agora juntamente com o Gerente de Projetos tomar outras decisões, como decisões de aquisições por exemplo, executando os critérios de *Make or Buy* já definidos. Mas, além das definições necessárias ao Scrum, o principal objetivo nestas reuniões é o início do Gerenciamento de Riscos.

O Gerenciamento de Riscos é possivelmente o principal gerenciamento durante um projeto, pois todos os outros existem devido a este. Se não houvessem riscos ao projeto, não era necessário planejá-lo, pois todas as ações dariam certo e os projetos seriam sempre um sucesso. Pensando nisso, todo o time trabalhará nos processos de Gerenciamento de Riscos sugeridos pelo PMBOK durante esta fase. São eles:

- Identificar os Riscos [11.2];
- Realizar Análise Qualitativa dos Riscos [11.3];
- Realizar Análise Quantitativa dos Riscos [11.4];
- Planejar Respostas aos Riscos [11.5].

A filosofia do Scrum tem como um dos principais preceitos a mitigação dos riscos. O simples fato de não planejar todo projeto antes de começar a executá-lo ou de ter ciclos curtos, são exatamente para minimizar os riscos e seus impactos. A diferença desta vez é a formalização e análise já feita antes da primeira *sprint*. Dentro do ciclo Scrum existem vários momentos para a realizar o controle e até mesmo a identificação e análise de novos riscos, as reuniões de *review*, retrospectiva, reuniões diárias do Time de Desenvolvimento, também conhecida como *Daily Meeting*, porém o principal deles é a reunião de planejamento.

Após o fim da Sprint Zero, finalmente é iniciado a primeira *sprint*, tendo como marco inicial a Reunião de Planejamento da Sprint. Nesta reunião alguns dos processos já vistos serão revisitados, como já descritos acima:

- Coletar os Requisitos [5.2];
- Definir o Escopo [5.3];
- Sequenciar as Atividades [6.3];
- Estimar as Durações das Atividades [6.5];
- Definir as Atividades [6.2];

- Estimar os Recursos das Atividades [6.4];
- Identificar os Riscos [11.2];
- Realizar Análise Qualitativa dos Riscos [11.3];
- Realizar Análise Quantitativa dos Riscos [11.4];
- Planejar Respostas aos Riscos [11.5].

5.3.3 Processos de Execução

A fase de execução num projeto Scrum é muito intuitiva, e nesta metodologia não será diferente. Ao final da *Sprint Planning*, é iniciada a fase de execução do trabalho combinado para a *sprint*. Neste ponto é fundamental diferenciar macrogerenciamento e microgerenciamento.

Macrogerenciamento é responsabilidade do Gerente de Projetos e do *Product Owner*, que devem se preocupar com o gerenciamento do prazo, do escopo, do custo, qualidade, etc. Além de trabalhar intensamente na comunicação do projeto. Cabe nesta fase ao Gerente do Projeto, além de conduzir as aquisições necessárias, gerenciar as expectativas dos *stakeholders*, aproveitando-se dos eventos de *Sprint Planning* e principalmente da *Sprint Review*, onde as principais partes interessadas podem ver o produto se formando. O que abrange os seguintes processos do PMBOK:

- Gerenciar as Comunicações [10.2];
- Conduzir as Aquisições [12.2];
- Gerenciar o Engajamento das Partes Interessadas [13.3].

Neste caso, focou-se na comunicação para fora do projeto, porém a comunicação entre o Time do Projeto é ainda mais importante. Para isso existem artefatos muito utilizados com o Scrum e que resultam em excelentes resultados. Devido a isso, o papel do *Scrum Master* é muito importante neste momento, pois é ele quem deve assegurar que os rituais e melhores práticas do Scrum estão sendo compridos, a fim de garantir os três pilares de um processo empírico: Inspeção, Transparência e Adaptação. Estes pilares por si só já garantem uma boa comunicação no projeto.

O microgerenciamento é de total responsabilidade do Time de Desenvolvimento, pois é premissa do Scrum que o time seja autogerenciável. O Time de Desenvolvimento não deve sofrer influências externas. É ele quem se organiza, divide e realiza as atividades. O Time de Desenvolvimento deve ser blindado para exercer sua liberdade e autoridade sobre seus métodos e processos. Ao Gerente do Projeto não importa o microgerenciamento realizado pelo time, e este é mais um ponto forte desta metodologia que une os paradigmas

tradicionais aos ágeis. Cabe ao Time de Desenvolvimento com o apoio do *Scrum Master*, gerir-se e comunicar-se, além de realizar os *reports* necessários ao Gerente do Projeto.

Uma das ferramentas de comunicação e controle mais importantes é o *burndown*. São utilizados 2 *burndowns*, um para controle do projeto, mostrando o andamento das entregas, *sprints* e *releases*, denominado *Burndown* do Projeto. E um *burndown* para controle da *sprint*, mostrando as horas trabalhadas e faltantes, além dos pontos já finalizados durante a *burndown*. Os *burndowns* podem ser adaptados para mostras mais informações de acordo com a vontade do Time de Desenvolvimento, *Scrum Master* ou Gerente do Projeto.

Apesar de não estar no Guia Oficial do Scrum, o quadro de tarefas ou *taskboard* é uma ferramenta utilizada na grande maioria dos projetos que utilizem a metodologia, e nesta proposta não será diferente. Também conhecido como quadro Kanban, esta é uma importante ferramenta para comunicação entre o Time de Desenvolvimento, pois é possível verificar visualmente a situação atual das tarefas da *sprint*.

Mesmo as duas ferramentas citadas anteriormente apresentando excelentes resultados, o grande responsável pelo sucesso do Time de Desenvolvimento pode ser atribuído à *Daily Meeting*, que nesta metodologia proposta será chamada de *Stand Up Meeting*, puxando mais um dos conceitos do Extreme Programming. *Stand Up Meeting* é basicamente o mesmo evento que o *Daily Meeting* do Scrum, porém este ocorre com os participantes em pé, para se assegurar que a reunião realmente será breve.

Através dos conceitos e ferramentas apresentadas acima, pode-se assegurar que o Time do Projeto, juntamente com o *Scrum Master* é responsável pelos seguintes processos PMBOK:

- Orientar e Gerenciar a Execução do Projeto [4.3];
- Realizar a Garantia da Qualidade [8.2];
- Mobilizar a Equipe do Projeto [9.2];
- Desenvolver a Equipe do Projeto [9.3];
- Gerenciar a Equipe do Projeto [9.4].

No caso do processo “Realizar a Garantia da Qualidade [8.2]”, ocorre também o envolvimento do *Product Owner* no momento de elaborar a *user storie* a ser entregue, pois a mesma possui os critérios de aceitação necessários.

5.3.4 Processos de Monitoramento e Controle

Como já citado em 5.3.3, existem duas formas de gerenciamento envolvidas nesta metodologia, o macrogerenciamento e o microgerenciamento. A parte de macrogerenciamento, de responsabilidade do Gerente do Projeto, implica que o mesmo é responsável pelos seguintes processos do PMBOK:

- Realizar o Controle Integrado de Mudança [4.5];
- Controlar o Cronograma [6.7];
- Controlar os Custos [7.4];
- Controlar as Comunicações [10.3];
- Controlar os Riscos [11.6];
- Controlar as Aquisições [12.3];
- Controlar o Engajamento das Partes Interessadas [13.4].

Estes processos devem ser administrados conforme as boas práticas do PMBOK e paralelamente ao Scrum.

É de responsabilidade do Time de Desenvolvimento, responsável pelo microgerenciamento, além de auxiliar no processo "Controlar a Qualidade [8.3]", o processo do PMBOK:

- Monitorar e Controlar o Trabalho do Projeto [4.4].

Ao final da *sprint*, existe um evento chamado Revisão da Sprint, ou *Sprint Review*. Neste momento, todo o time de projeto é convocado, no caso do Gerente de Projetos apenas para ciência e para capturar possíveis informações relevantes a ele, para apresentar ao *Product Owner*, o entregável gerado na *sprint*. Outros *stakeholders* podem ser convidados para este evento, se for de interesse. O Time de Desenvolvimento apresenta o produto potencialmente entregável para o *Product Owner* e o mesmo inspeciona as entregas, validando-as se cumprem com o escopo e a qualidade esperada e acordada. Caso alguma entrega não seja aceita, a mesma volta para o *backlog* do produto. Este é um momento comum em que o *Product Owner* percebe o que pode ser feito e solicita algumas mudanças, ou seja, adiciona novos itens ao *backlog*. Por este motivo, considera-se que através da textitSprint Review, o Time de Desenvolvimento, juntamente com o *Product Owner* são responsáveis pelos seguintes processos:

- Controlar o Escopo [5.6];
- Validar o Escopo [5.5];
- Controlar a Qualidade [8.3].

5.3.5 Processos de Encerramento

Após a *Sprint Review* ocorre a retrospectiva da *sprint*, ou *Sprint Retrospective*, onde são apontadas a lições aprendidas na *sprint* e pontos de melhorias a serem levados para as próximas iterações. Assim todo o Time do Projeto é responsável por:

- Encerrar o Projeto ou Fase [4.6].

E o Gerente do Projeto é o único responsável pelo processo:

- Encerrar Aquisições [12.4].

6 Conclusão

Neste trabalho foram apresentadas e houve foco em duas abordagens para Gerenciamento de Projetos: Uma abordagem mais tradicional, baseada no PMBOK; e outra conhecida por utilizar-se de princípios ágeis, utilizados principalmente na Área de Desenvolvimento de Software, o Scrum.

Após conhecer um pouco do mundo de Desenvolvimento de Software, seus aspectos e especificidades, foi estudado uma forma de classificar os projetos e que este tipo de projeto normalmente possuem requisitos incertos e/ou tecnologias não dominadas, classificando-os como Projetos Complexos.

Concluiu-se que apesar da escolha simples normalmente ser por optar por uma Metodologia Ágil, é sim possível utilizar as práticas da abordagem tradicional juntamente com a abordagem ágil, a fim de gerar uma metodologia para este tipo de projetos, assim maximizando o aproveitamento das boas práticas dos dois mundos e suavizando os pontos fracos de cada um deles.

Com isso, foi proposta uma metodologia que mescla os dois paradigmas, tradicional e ágil, demostrando que é sim possível combinar as duas, e não apenas isso, pode-se observar que ambas abordagens se misturam quase que naturalmente, resultando em uma nova metodologia de alta sinergia entre os processos.

6.1 Trabalhos Futuros

Propõe-se para um trabalho futuro, o detalhamento dos processos e *templates* a serem utilizados nesta metodologia, bem como a aplicação prática da mesma. Este trabalho deverá demostrar os resultados obtidos com esta abordagem mista, demostrar seus pontos fortes e fracos e também possíveis melhorias ao processo.

Esta metodologia, apesar de generalista, foi pensada para o universo de Desenvolvimento de Software, propõe-se também um trabalho de aplicação de tal metodologia em outras áreas de conhecimento, onde os métodos ágeis não sejam amplamente utilizados.

Referências

B2ML. *Metodologia ágil*. [S.l.], 2014. Disponível em: ⟨http://www.b2ml.com.br/b2ml/solucoes-sob-demanda/metodologia-agil⟩. Acesso em: 30 set. 2014. Citado na página 34.

BECK, K. et al. *Manifesto for Agile Software Development*. 2001. Disponível em: ⟨http://www.agilemanifesto.org/⟩. Acesso em: 30 set. 2014. Citado na página 28.

BUCK, E. M. F. P. R. *Scrum vs PMBOK - Como defini em qual seu projeto se encaixa melhor*. [S.l.], 2012. Disponível em: ⟨http://www.inovagp.com/2012/01/scrum-vs-pmbok-como-definir-em-qual-seu-projeto-se-encaixa-melhor/⟩. Acesso em: 30 set. 2014. Citado na página 39.

CRUZ, F. *Scrum e PMBOK unidos no Gerenciamento de Projetos*. [S.l.]: Brasport, 2013. ISBN 9788574525945. Citado na página 43.

IFPR SC. *Ciclo de Vida Iterativo e Incremental*. [S.l.], 2006. Disponível em: ⟨http://wiki.sj.ifsc.edu.br/wiki/index.php/Ciclo_de_Vida_Iterativo_e_Incremental⟩. Acesso em: 30 set. 2014. Citado na página 28.

PMI. *Um Guia do Conhecimento em Gerenciamento de Projetos (Guia PMBOK)*. 5. ed. Newtown Square, PA: Project Management Institute, 2012. Citado 2 vezes nas páginas 19 e 21.

PMKB. *PMBOK (PMI)*. [S.l.], 2013. Disponível em: ⟨http://pmkb.com.br/sig/padroes-frameworks/pmbok-pmi/⟩. Acesso em: 30 set. 2014. Citado 3 vezes nas páginas 22, 24 e 38.

ROCHA, F. L. *Ciclo de Vida de Projeto*. [S.l.], 2013. Disponível em: ⟨http://felipelirarocha.wordpress.com/2013/07/13/ciclo-de-vida-de-projeto/⟩. Acesso em: 30 set. 2014. Citado 5 vezes nas páginas 20, 21, 25, 26 e 27.

SCHWABER, K.; BEEDLE, M. *Agile Software Development with Scrum*. 1st. ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2001. ISBN 0130676349. Citado na página 30.

SCHWABER, K.; SUTHERLAND, J. The scrum guide. 2014. Citado 3 vezes nas páginas 30, 31 e 33.