UNIVERSIDADE DO EXTREMO SUL CATARINENSE - UNESC

CURSO DE CIÊNCIA DA COMPUTAÇÃO

ANDERSON TORQUATO CARDOSO

ANÁLISE COMPARATIVA ENTRE FERRAMENTAS LIVRES PARA GERENCIAMENTO DE PROJETOS DE SOFTWARE

CRICIÚMA

2014

ANDERSON TORQUATO CARDOSO

ANÁLISE COMPARATIVA ENTRE FERRAMENTAS LIVRES PARA GERENCIAMENTO DE PROJETOS DE SOFTWARE

Trabalho de Conclusão de Curso apresentado para obtenção do grau de Bacharel no curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC.

Orientadora: Prof.ªMSc. Ana Claudia Garcia Barbosa

CRICIÚMA

2014

ANDERSON TORQUATO CARDOSO

ANÁLISE COMPARATIVA ENTRE FERRAMENTAS LIVRES PARA GERENCIAMENTO DE PROJETOS DE SOFTWARE DESENVOLVIDOS COM METODOLOGIAS ÁGEIS

Trabalho de Conclusão de Curso aprovado pela Banca Examinadora para obtenção do Grau de Bacharel, no Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, UNESC, com Linha de Pesquisa em Engenharia da computação

Criciúma, 30 de Junho de 2014

BANCA EXAMINADORA

Orientadora: Profª. Msc. Ana Claudia Garcia Barbosa – UNESC

Membro banca: Profº. Msc. Gustavo Bisognin – UNESC

Membro banca: Profº. Esp. Gilberto Vieira da Silva – UNESC

A minha família.

AGRADECIMENTOS

Agradeço a Jesus por todos os dias que fiquei estudando na Universidade do Extremo Sul Catarinense (UNESC), professores e amigos acadêmicos.

“O planejamento não é uma tentativa de predizer o que vai acontecer. O planejamento é um instrumento para raciocinar agora, sobre que trabalhos e ações serão necessários hoje para merecermos um futuro. O produto final do planejamento não é a informação: é sempre o trabalho.”

Peter Drucker

RESUMO

Este trabalho de conclusão de curso apresenta uma análise comparativa de ferramentas livres para gerenciamento de projetos de software. O gerenciamento de projetos desempenha um papel fundamental na gestão de sistemas e tem por finalidade gerenciar o ciclo de vida de software nas ferramentas. Os processos de gerenciamento de metodologias ágeis ganharam abrangência nos últimos anos, a adoção desses métodos motivou o uso de ferramentas, principalmente as de acesso livre. Essas ferramentas podem auxiliar a gestão dos projetos de software. Com base nessas constatações essa pesquisa compara algumas dessas ferramentas e propõe auxiliar na escolha da mais adequada, aquela que pode auxiliar nas decisões dos desenvolvedores que optam por metodologia ágil. Em primeiro plano foram escolhidas ferramentas livres que tivessem o mínimo de itens necessários aos projetos, no segundo momento os itens usados para essa comparação de ferramentas foram etapas estudadas da metodologia ágil Scrum, além, desses passos, a escolha dos itens a serem analisados foram baseados em constatações de uma vasta pesquisa realizada pela Universidade de São Paulo para levantar o comportamento atual de adoção e adaptação da metodologia ágil nas empresas brasileiras.
Essa análise foi realizada em cinco ferramentas de gerenciamento de projetos ágeis.

Palavras-chaves: Software, Metodologias, Ferramentas, Projetos, Ágeis.

ABSTRACT

This course conclusion work presents a comparative analysis of free tools for managing software projects. Project management plays a key role in managing systems and aims to manage the lifecycle of the software tools. Management processes of agile methodologies have gained coverage in the last years, the adoption of these methods led to the use of tools, mostly free access ones. These tools can assist the management of software projects. Based on these findings this research compares some of these tools and offers help in choosing the most appropriate one, that can aid in decisions of developers who choose to agile methodology. In the foreground were chosen free tools that had the minimal necessary items for the projects, In the second time the items used for this comparison of tools were studied steps of the agile Scrum methodology, beyond these steps, the choice of items to be assessed were based on the findings of an extensive survey conducted by the University of São Paulo to raise the current behavior of adoption and adaptation of agile methodology in Brazilian companies.
This analysis was conducted in five tools of agile project management.

Keywords: Software, Methodologies, Tools, Projects, Agile.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

DSDM	Desenvolvimento de Sistemas Dinâmicos
FDD	Desenvolvimento Guiado por Funcionalidades
PMI	Project Management Institute
SCRUM	Iterativa e Incremental
XP	Programação Extrema

SUMÁRIO

1 INTRODUÇÃO

Na busca por melhores soluções no desenvolvimento de projetos, precisa-se de ferramentas de gerenciamento de software. Visto que o mercado de software está desenvolvendo cada vez mais em metodologias ágeis aplicadas à construção de software (LIMA, 2009).

Na utilização das ferramentas faz-se uso também de engenharia de software, ou seja, organizar uma aplicação, com conceito de gerenciar o projeto no cronograma planejado. Controlar as equipes de forma ágil buscando motivar o desenvolvimento de software (EDUARDO, 2009).

Essa busca de análise consiste em compreender e contextualizar metodologias ágeis, identificar as principais práticas ágeis adotadas nas empresas de desenvolvimento de software, compreender processos de gerência de projetos, definindo qual a melhor prática para metodologias ágeis, analisar recursos e metodologias nas ferramentas de gerenciamento de projetos.

Portanto, essa pesquisa apresenta ferramentas de gerenciamento de projetos específicas para metodologias ágeis. Para realização deste trabalho será feita a identificação das principais práticas ágeis adotadas nas organizações e realizada uma análise comparativa entre ferramentas livres para gerenciamento de projetos de software.

Na engenharia de software o desenvolvimento de novos projetos deve oferecer mecanismos para se planejar e gerenciar o processo de desenvolvimento de software com a finalidade de obter um produto de software de qualidade, sendo uma das principais metas de mercado.

Nesse contexto onde essa pesquisa trata da importância de ferramentas livres de gerenciamento de projetos de software, analisando características e propriedades das ferramentas, comparando qual irá se adequar para o desenvolvimento de metodologias ágeis.

1.1 OBJETIVO GERAL

Realizar uma análise comparativa de ferramentas livres para gerenciamento de projetos em metodologias ágeis.

1.2 OBJETIVOS ESPECÍFICOS

A pesquisa tem os seguintes objetivos específicos:

a) compreender e contextualizar metodologias ágeis;
b) identificar as principais práticas ágeis no desenvolvimento de software;
c) analisar recursos e metodologias nas ferramentas de gerenciamento de projetos.

1.3 JUSTIFICATIVA

Para buscar a qualidade e a produtividade no desenvolvimento de software, é fundamental que sejam empregadas ferramentas que suportem adequadamente os projetos envolvidos. Os produtos comerciais que dominam o mercado de ferramentas de desenvolvimento exigem um investimento financeiro alto e periódico, o que pode colocar em risco sua manutenção em longo prazo.

A adoção de ferramentas de software livre ou de baixo custo, de maturidade comprovada, é uma alternativa viável e que proporciona maior garantia de sustentabilidade à solução (CHAVES, 2010).

Nesse contexto é que essa pesquisa trata da importância de ferramentas livres de gerenciamento de projetos de software, analisando características e propriedades das ferramentas, comparando qual melhor irá se adequar para o desenvolvimento de metodologias ágeis.

1.4 ESTRUTURA DO TRABALHO

Este trabalho de pesquisa está organizado da seguinte forma: No capítulo 2 são citados os conceitos de engenharia de software e como atua no processo de software. No capítulo 3 são mencionados o gerenciamento de projetos, que vem a ser a gestão do software projetados. O capítulo 4 são as metodologias ágeis onde mostra um conceito dos métodos usado no desenvolvimento de software. No capítulo 5 expõe os trabalhos correlatos, que são pesquisas relacionadas a esse trabalho de conclusão de curso. O capítulo 6 apresenta as ferramentas livres utilizadas no desenvolvimento desse trabalho esse capítulo possui a etapa da metodologia ágil nas ferramentas e a verificação das mesmas. E por fim no capítulo

7 está os resultados obtidos da ferramentas de gerenciamento com a escolha do método ágil Scrum, onde mostra os itens da estrutura Scrum sobre os gerenciadores de projetos. Logo após esses capítulos a conclusão das análises, comparação das etapas e verificações das ferramentas, que serve de base de material para futuros trabalhos que venha ser usado no mercado de software, e também pela contribuição de analisar software de gestão para profissionais da área de T.I.

2 ENGENHARIA DE SOFTWARE

Com benefícios e formas de organizações aos desenvolvedores de software, surge à engenharia de software, fazendo com que satisfaça as necessidades da área, de projeto de software. A engenharia de software entra para organizar e modelar esses softwares (JACOBSON, 2005).

Para o desenvolvimento de software segue-se o seguinte:

a) **comunicação:** envolve a comunicação do cliente e colaboradores, onde abrange os requisitos e outras atividades;
b) **planejamento:** abrange um plano, onde segue o trabalho da engenharia de software, entre elas: riscos, técnicas, recurso, cronogramas;
c) **modelagem:** consiste na criação de modelos para atender o cliente, os requisitos e o desenvolvimento do software;
d) **construção**: inclui a codificação e os testes para correção de erros;
e) **implantação**: são entregues ao cliente, onde avalia e fornece feedbacks com a avaliação (PRESSMAN, 2006).

Com esses cinco itens são possíveis desenvolver softwares de pequeno, médio e grande porte.

2.1 DEFINIÇÕES DAS FASES DA ENGENHARIA EM PROJETO DE SOFTWARE

Engenharia de projetos muda constantemente conforme a demanda do mercado de softwares, pois a estudos de novas alternativas e modelos de análises, onde evoluem a cada ano, e mesmo assim não há metodologias flexíveis, e de forma quantitativa que atenda as necessidades até nos dias atuais, onde são associadas a disciplinas clássicas de engenharia de projetos (JACKSON, 2005).

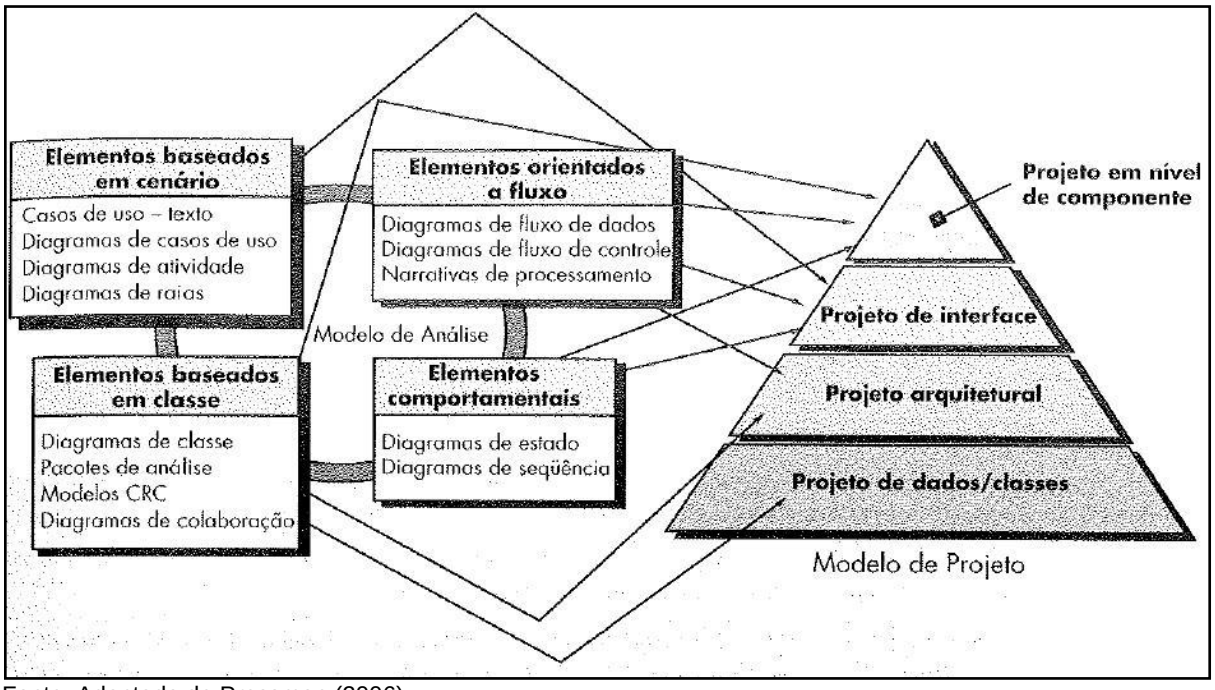
Essas metodologias e modelos de análises são realizadas de forma independente do modelo de software que irá usar, esses métodos são analisados e modelados com os requisitos do software, e por último na engenharia de software une ao projeto do software final. Unir a lógica da engenharia de software em analisar e projetar, para a prática da codificação (DUE, 2009).

O principal objetivo da engenharia de projetos representa um modelo firme, cômodo, os desenvolvedores praticam diversificação.

Segundo Belady (2007) diversificar são alternativas seqüenciais que unem a matéria-prima do projeto, ou seja, componentes e conhecimento, onde está em livros-textos e nas próprias necessidades humanas. Sendo que logo após o desenvolvedor estuda esse modelo e une seqüências alternativas de análises, que atenda os requisitos do software, e ao logo do desenvolvimento do projeto essas alternativas são aceitas ou recusadas, onde o desenvolvedor converge para a configuração de componentes, até o projeto final.

A figura 1 mostra uma representação do modelo de análise em um modelo de projeto.

Figura 1 - Tradução do modelo de análise para um modelo de projeto.



Fonte: Adaptado de Pressman (2006).

Desta forma a análise do software faz uma transição para um projeto de software, com isso o desenvolvimento do projeto se torna mais importante. A importância da análise para o projeto em engenharia de software se define pela qualidade. Com isso pode se representar e avaliar as qualidades do software (LEITE, 2007).

Com as iterações da documentação do projeto, as seqüências de refinamentos que representam o projeto, tornam-se níveis de abstração muito mais baixos. As qualidades do projeto são avaliadas em várias revisões e técnicas, ao decorrer do processo (WESLEY, 1996).

De acordo com Jackson (2009) um engenheiro de software com sua sabedoria, devem distinguir entre um programa que funciona e outro que seja correto tem um diferencial, pois o funcionamento correto são características que o desenvolvedor adquire com suas experiências.

Abaixo seguem algumas definições fundamentais de projetos de softwares que serve de base e para fazê-lo corretamente:

a) **abstração:** são conjuntos definidos de dados que descreve um objeto de dados;
b) **arquitetura:** modo pela qual a estrutura integra os conceitos do sistema, esqueleto integral do software;
c) **padrões:** são modelos de conhecimento definidos, de uma solução do problema recorrente para uma concorrente;
d) **modularidade:** são divisões de módulos de componentes, onde são endereçados e separados para satisfazer a integridade dos requisitos do problema;
e) **ocultamento da informação:** os módulos devem ser projetados e especificados de forma que os dados e os algoritmos de outros módulos não sejam acessados e que não necessita dessa informação;
f) **independência funcional:** são conseqüências diretas de, modularidade, ocultamente de informação e abstração, onde um conceito une-se a outro;
g) **refinamento:** planejar passo-a-passo o projeto até alcançar a linguagem de programação;
h) **refatorar:** procedimento de organizar o código do projeto, sem afetar o seu comportamento, ou seja, mudar o código externo para melhorar a estrutura interna;
i) **classes de projeto:** à medida que o projeto evolui, a equipe do software determina um conjunto de classes, onde são aprimoradas, detalhando-as para ser implementadas, e reuni um conjunto de classes que programa uma infraestrutura para solução de negócios.

Com essas definições e características descritas o desenvolvimento de projetos em engenharia de softwares se torna compreensível ao ser aplicado, onde para alcançar essa compreensão requer desenvolvedores experientes em analisar

um modelo de implementação para atender os requisitos do cliente, e satisfazer aqueles que vão utilizar.

3 GERENCIAMENTO DE PROJETOS

Gestão de projetos são métodos e ferramentas que organizam a sequência de tarefas em execução, e defende recursos e tempo alocados, permiti também rastrear atividades e mede o progresso que foi definido pelo projeto (MOURA, 2007).

Para essa organização funcionar na sequência do projeto, necessita-se a elaboração de um manual do plano do projeto, com ele os desenvolvedores do projeto vivem o progresso do projeto, onde também a manutenção e desenvolvimento se tornam obrigatória. O objetivo desse plano determina as principais atividades necessárias do projeto. Ao longo do desenvolvimento o documento estabelece datas de marcos do progresso, onde começa e termina as atividades associadas, responsáveis e atividades geradas (PMBOK, 2010).

Na figura 2 há uma representação de planejamento de projeto versus a gestão da execução.

Figura 2 - Perspectivas da gestão de projetos de software.

Fonte: Adaptado de Silva (2008).

Segundo Pressman (2006) para desenvolver uma gestão do projeto satisfatória com documentação de um plano do projeto, a gestão focaliza em quatro métodos que são:

a) **pessoal:** a importância de pessoas para o desenvolvimento de projetos se torna tão importante quanto o produto (software). Um modelo de gestão de pessoas foi criado com objetivo de atender e melhorar a gestão de seus produtos, obtendo resultados de atrair, motivar, dispor e talentos no desenvolvimento de softwares;

b) **produto:** antes do planejamento do projeto são definidos objetivos e o escopo do produto, onde compreende o seguinte: restrições técnicas e gerenciais, onde sem essas informações não se pode estimar os custos, avaliações de riscos, cronogramas e relação da realidade de tarefas, essas definições são indicações de progresso. O escopo e os objetivos são feito através de reuniões do desenvolvedor e o cliente;

c) **processo:** o processo de software compreende em uma estrutura, onde pode atender o plano do projeto e desenvolve - ló. Essa estrutura abrange todo o desenvolvimento, independente do tamanho e negocio. Com isso atende as necessidades dos desenvolvedores do projeto;

d) **projeto:** controlar e planejar projetos de software se compreende pelo seguinte objetivo, se torna mais flexível controlar negócios. Na indústria do software nos anos de 1990, indicava que 26% dos projetos de softwares falhavam de início e 46% ultrapassava os prazos e custos.

Com isso a gestão de projetos se define pelos seus manuais e estruturas de atividades na engenharia de softwares.

3.1 A GESTÃO DE PROJETOS

Com planejamento da gestão de projetos documentados e com uma estrutura das atividades na engenharia de software cada vez melhor, os desenvolvedores e gestores do projeto têm a necessidade de tornar as atividades do projeto executadas, de acordo com o software do cliente (ABREU, 2006).

As atividades de gestão se tornam harmonioso ao desenvolvimento do software, ou seja, ao decorrer do projeto a gestão atua como orientação na execução do software. Com isso as informações do projeto devem ser acessadas por todos da equipe, pois ao desenvolver softwares de negócios de médio e grande porte a coordenação fica sendo atividade chave na gestão. Onde o gestor e

coordenador do projeto têm como objetivo coordenar as seguintes atividades (PMBOK, 2004):

a) formação de pessoas em diversas áreas;

b) várias atividades de relação de dependência;

c) uso de vários recursos (ferramentas, laboratórios, entre outros);

d) decidir e aprovar em vários pontos no projeto;

e) alocar recursos humanos e financeiros a atividades.

Sendo assim, as informações do projeto devem ser compartilhadas e viabilizadas, onde as decisões são tomadas de acordo com informações compreendidas de acordo com o desenvolvimento do software.

No inicio do desenvolvimento do projeto são analisadas diversas atividades, para encontrar problemas de erros e inconsistências, para poder trabalhar de forma adequada. Com isso faz com que a gestão de projeto se torne essencial, em qualidade de produto e na execução do projeto (IEEE, 2008).

Para que a gestão do projeto seja desenvolvida com essa essência são observadas duas características fundamentais, entre elas: pessoal e técnica, onde ter pessoas habilidosas para gerir as técnicas de um projeto, também torna essencial para uma boa gestão (PRESSMAN, 2006).

Os gestores cada vez mais procuram estudar e aprimorar seus conhecimentos, pois as empresas reconhecem que motivar esses profissionais se torna fundamental para o plano estratégico da empresa. Sendo que empresa e gestores procuram aprimorar e satisfazer soluções que atenda as necessidades do mercado de softwares (ABREU, 2008).

Esse tipo de plano estratégico se aplica a técnicas e pessoas com intuito de manter os resultados de qualidade e gestores informados de desenvolver um projeto de software. Com isso a gestão de projetos utiliza cinco grupos de processos, entre eles: iniciação, planejamento, execução, controle e encerramento e também suas áreas de conhecimentos que são: gerência de integração, escopo, tempo, custo, qualidade, Rh, comunicação, riscos e aquisição.

A figura 3 representa as áreas de conhecimento de gestão de projetos em engenharia de softwares.

Figura 3 - Áreas de conhecimento da gerência de projetos.

Fonte: Adaptado de Abreu (2006).

3.2 CICLOS DE VIDA DOS PROJETOS

As técnicas e as etapas que são aplicadas em projetos de softwares são interligadas do inicio ao fim de cada desenvolvimento de projeto. A cada novo projeto os desenvolvedores adquirem experiências ao longo dos anos, que ao aplicar um ciclo de vida de projeto não se deve ser estáticas, ou seja, cada software deve ser tratado de forma que atenda especificamente cada cliente, pois cada um exige medidas e procedimentos personalizados. Onde as etapas de cada projeto desses ciclos de vida são sequenciais e se interligam ao desenvolvimento até a entrega do software (PMBOK, 2009).

Esse ciclo de sequência se entende basicamente a forma como o projeto deve ser desenvolvida, onde cada ciclo de projeto aplica em cada novo projeto.

Segundo a PMBOK, não importa o tamanho do projeto, qualquer novo projeto pode ser mapeado da seguinte forma:

a) início do projeto;

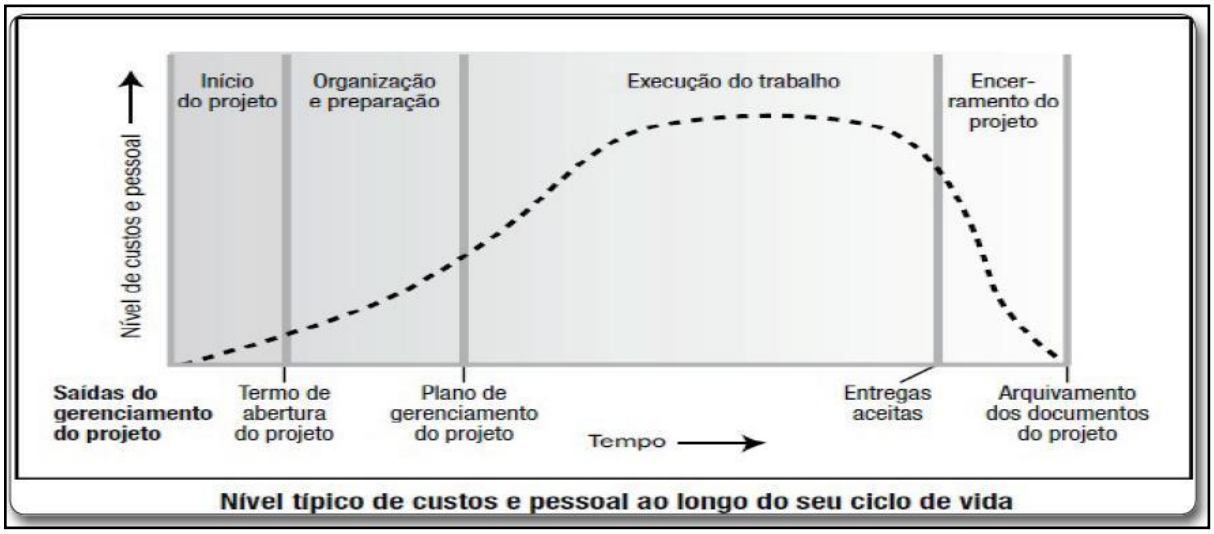
b) organização e preparação;

c) execução do trabalho do projeto;

d) encerramento do projeto.

Um exemplo de um ciclo de vida em projetos de softwares está representado na figura 4.

Figura 4 - Nível típico de custo e pessoal ao longo do seu ciclo de vida.



Fonte: Adaptado de Araújo (2008).

Nessa figura se pode ter uma ideia de como acontece ao longo dos projetos, onde mostra que o ciclo desse projeto faz o gerenciamento com o Project Management Institute (PMI), e mostra a relação do custo e as etapas do projeto (HIGHSMTH, 2004).

Ao longo do desenvolvimento do projeto a equipe tem que estar definida, ou seja, as pessoas que irão participar do início ao final do projeto deve estar com os interessados, os clientes. Os desenvolvedores apresentam as alterações e funcionalidades, onde algumas são aceitas e outras não durante a execução do projeto, pois feito isso reduz custo e tempo, que muitas vezes se torna uma das etapas decisivas (KOPPERNSTEINER, 2003).

As alterações ao decorrer do desenvolvimento dos projetos são de certa forma necessária, onde os clientes possam detalhar melhor as funcionalidades, fazendo de seu software mais ágil. Onde os custos são diminuídos e os clientes interessados podem melhor ser atendidos (DIAS, 2008).

Desta forma, os objetivos do negócio devem estar alinhados as funcionalidades de cada cliente, ou seja, as funções contínuas e repetitivas desses softwares são muito importantes. O ambiente de trabalho ativo para o desenvolvimento do software também deve ser flexível. Com essa estrutura de desenvolvimento e objetivos, o ciclo de vida de um projeto deve englobar os seguintes princípios (PMI, 2004):

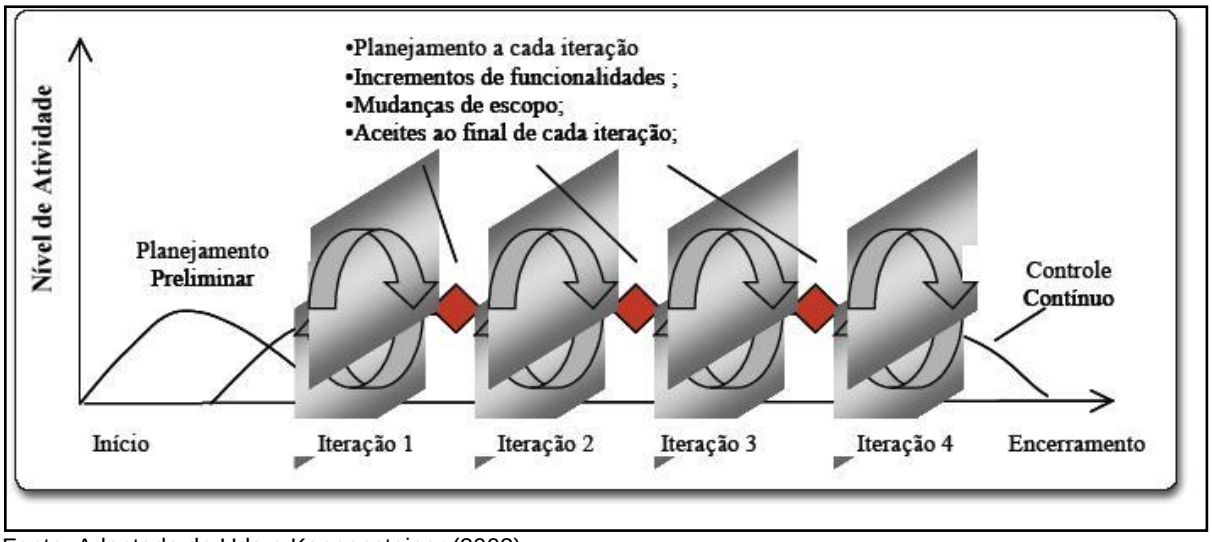
a) explorar a cultura adaptável;

b) fazer organização e disciplina;

c) motivar a consistência e a confiança dos dados;
d) fácil adaptação e maleável;
e) ter visão ao longo do projeto;
f) aprimorar a aprendizagem;
g) compreender melhor cada ciclo ao desenvolver;
h) fornecer tópicos de verificações.

Uma forma de especificar um ciclo de vida de projeto de forma prática e contínua está representada logo a seguir na figura 5.

Figura 5 - Fluxo do gerenciamento ágil de projetos.



Fonte: Adaptado de Udo e Koppensteiner (2003).

Na figura 5 mostra o ciclo de vida do processo Project management Body of Knowledge PMBOK, que representa o grupo de processos de gerenciamento de projetos.

A seguir são especificadas as metodologias ágeis que podem ser uma forma de gerenciar projetos mais rapidamente procurando não perder o foco nos processos de desenvolvimento citados neste capítulo.

4 METODOLOGIAS ÁGEIS

As metodologias ágeis são técnicas de gerenciamento de projetos com papel de agilizar o desenvolvimento de software, ou seja, elas são adaptativas ao invés de serem preditivas. O objetivo das metodologias ágeis é trabalhar com foco em desenvolver software que: funcione e possibilite modificações (LIBARDI; BARBOSA, 2010).

A prática dos métodos ágeis oferece qualidade no software em desenvolvimento aos clientes interessados, onde também visa agregar o prazo de entrega, ou seja, menos tempo ao desenvolver (SOARES, 2009).

As metodologias ágeis apresentam diferenças em relação às metodologias tradicionais, entre elas (OLIVEIRA, 2003):

a) adaptativas ao invés de preditivas;
b) orientadas às pessoas e não a procedimentos;
c) são mais voltadas para o negócio dos interessados(cliente).

4.1 VISÕES GERAIS DA MODELAGEM ÁGIL

Os princípios de modelagens ágeis são (EDER, 2010):

a) **simplicidade:** trabalha com as condições do sistema atuais, ou seja, os requisitos;
b) **mudanças:** ao decorrer do desenvolvimento do projeto, mudanças de requisitos podem ser mudadas pelos interessados do sistema;
c) **objetivo:** deve satisfazer as necessidades dos interessados de forma principal;
d) **trabalhos futuros:** fazer uma boa documentação de todo o sistema torna as próximas fases mais eficiente e rápidas de entendimento;
e) **alterações no desenvolvimento:** aprimorar recursos de forma a melhorar o projeto;
f) **foco na modelagem:** observar se a necessidade de criar novas funcionalidades, para economizar tempo e dinheiro;
g) **qualidade ao desenvolver:** propósito de satisfazer as necessidades do cliente;
h) **modelar:** destacar os modelos de interesse, e descartar o restante;

i) **feedback dinâmico:** o tempo de resposta dos clientes torna a correção do sistema mais rápida.

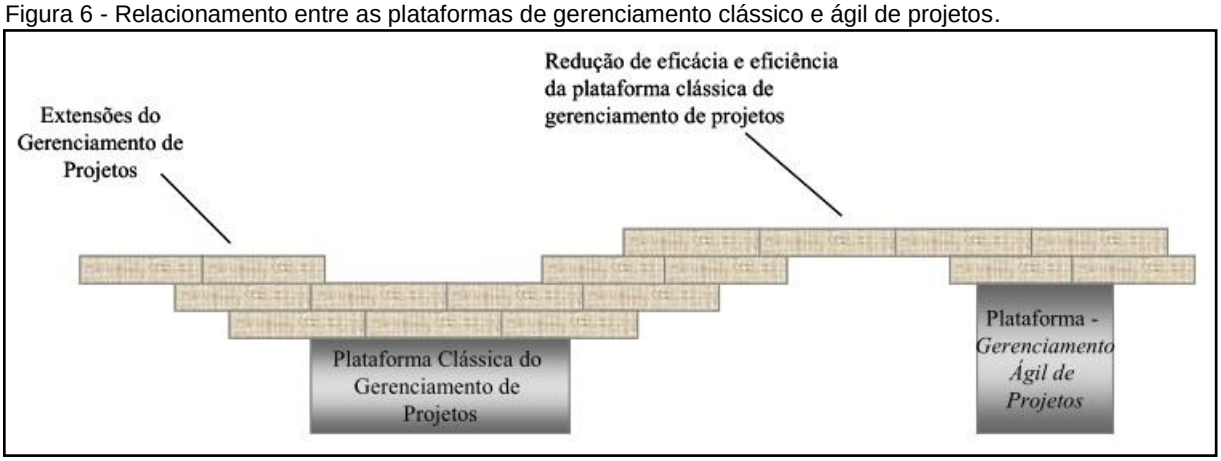
A modelagem ágil proporciona o desenvolvimento de softwares mais eficientes, onde facilita para a equipe e serve de base para escolhas de melhores ferramentas e métodos a ser aplicado nos projetos de software. Pois ao escolher que tipo de modelo de desenvolvimento se deve aplicar no software, torna-se fundamental a escolha dos modelos ágeis, como visto acima, modelos clássicos são mais fixos de mudanças, já em modelos ágeis as mudanças e alterações se torna prático (AMBLER, 2007).

Outra característica da modelagem ágil, ao modelar o software do projeto, se utiliza a participação do usuário do sistema, ou seja, o cliente auxilia no desenvolvimento das etapas, o que faz o modelo ágil flexível a decisões de negócios do sistema (MORAIS, 2006).

O gerenciamento de projetos no mercado de software faz com que as empresas trabalhem de forma dinâmica e com riscos ao ambiente de desenvolvimento de software.

Na figura 6 uma representação da relação entre metodologias clássicas e metodologias ágeis.

Figura 6 - Relacionamento entre as plataformas de gerenciamento clássico e ágil de projetos.



Fonte: Adaptado de Chin (2004).

A figura faz uma comparação das metodologias clássicas e ágeis, ou seja, os tijolos representam a eficiência e a eficácia às plataformas, onde o método clássico necessita de uma maior atenção às atividades teóricas e logo o método ágil requer atividades de controle da própria equipe, fazendo com que cada um faça uma analise visual e técnica ao desenvolvimento de software (ARAÚJO, 2009).

4.2 CICLOS DE VIDA DE PROJETOS ÁGEIS

A base de um ciclo de vida de um projeto se inicia pelo tipo de aplicação de onde irá atuar. Essa fase do projeto estabelece as seguintes características, que podem ser de qualquer tamanho (SCOTT, 2008):

a) ideia e inicio do projeto;

b) ordem e elaboração;

c) desenvolvimento do projeto;

d) conclusão do projeto.

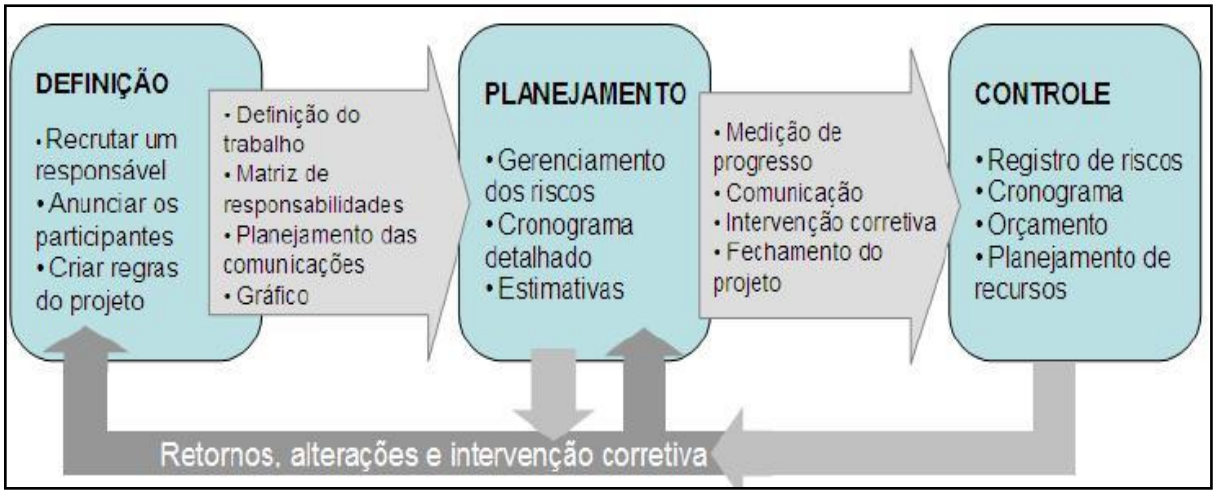
Nas metodologias ágeis esse processo de ciclo de vida começa de forma semelhante a forma clássica, sendo que o foco está na ordem, elaboração e desenvolvimento, também deve estar com a equipe do projeto formada para dar inicio os trabalhos de desenvolvimento (DINSMORE, 2008).

Esse ciclo deve estar bem estruturado para equipes e clientes do projeto analisar e desenvolver funções e mudanças ao decorrer de todo o projeto, pois esse processo do ciclo gera muito tempo e custo (COHEN, 2004).

Com um plano de projeto de software bem objetivo, é possível definir limites de todo o projeto. Conforme esses objetivos não forem tratados e documentados, os resultados esperados pelo projeto ficam comprometidos, ou seja, o software perde qualidade na sua essência (IMPA, 2006).

A figura 7 mostra a definição, planejamento e controle na gestão de projetos ágeis.

Figura 7 - Três funções da gestão de projetos.



Fonte: Adaptado de Viana (2011).

4.3 PRINCIPAIS METODOLOGIAS ÁGEIS

Os métodos ágeis recomendam que se façam as seguintes combinações de processos para desenvolver projetos de software em bom funcionamento:

a) desenvolvedores e especialistas trabalhando de forma mais colaborativa;
b) comunicação fase-a-fase;
c) novas versões do software funcionando;
d) equipe preparada;
e) modo de fazer o código padrão, para evitar confusão em novos requisitos.

Onde dessa forma torna a prática de metodologias ágil importante métrica no processo e progresso de desenvolvimento de software (SILVA, 2009).

Nos seguintes tópicos descritos estão as metodologias ágeis usadas no mercado de trabalho atualmente.

4.4.1 SCRUM

Nos últimos anos empresas de todo o mundo tem se interessado em trabalhar com novas técnicas em desenvolvimento de softwares, pois no mercado de software os avanços acontecem constantemente. Isso se deve as tecnologias e inovações nos negócios das empresas e em seus sistemas desenvolvidos (BOEHM, 2006).

Para agregar técnicas, tecnologias e inovações são necessárias ser simples em ambientes de desenvolvimento de software, onde para alguns a simplicidade não controla e existe desordem, mais isso está se tornando contrário, ser simples às empresas quer dizer que há ordem e muito controle do que está para ser desenvolvido. Essas características de ser ágil a mudanças nos negócios das empresas se dão pelo fato de alcançar lucros nas empresas em seus negócios (HIGHSMITH, 2004).

Em 2001, os novos projetos de software desenvolvido com técnicas ágeis, são de certa forma simples, se tornaram mais claros e melhor definidos, devido ao fato de pessoas interessadas em mudar os conceitos de desenvolver softwares se reuniu para escolher um padrão em usar métodos ágeis, ou seja, criar uma

plataforma simples que serve para auxiliar no processo ágil de desenvolver software (AGILE, 2001).

Na escolha de métodos, práticas e técnicas simples e ágeis no desenvolvimento de software, garante a satisfação dos clientes interessados (BOEHM, 2003). E serve também para garanti a qualidade e rapidez nas datas de entrega dos projetos (SOARES, 2003).

A metodologia iterativa e incremental (SCRUM) vem contribuindo no desenvolvimento de projetos, onde está agregando os seguintes valores: satisfação dos clientes, favorecendo a equipe, colaboração, motivação, integração, onde com isso se obtém resultados de projetos com sucesso (SCHWABER, 2002).

As vantagens de utilizar o método SCRUM se dão pelo fato de ser ágil, onde as equipes ao escolher técnicas e métodos agregam melhorias ao desenvolvimento de projetos de softwares. Pois no ciclo de vida desses projetos ocorrem muitas mudanças de requisitos, o SCRUM resolve isso fazendo com que seja definida também inovação tecnológica e requisitos, e dentre outros tipos de características. Fazendo com que a entrega estabelecida do software seja rápida e não ocorram atrasos, realizando a qualidade esperada de seus clientes (PEREIRA, 2007).

4.4.1.1 Funcionamento da metodologia SCRUM

A prática de funcionamento do Scrum segue fundamentada de iterações bem determinadas, onde essa iteratividade funciona de duas a quatro semanas que se chamam Sprints. Essa iteração de Sprints serve para planejar e reunir-se com o cliente o que dever ser feito durante o período das duas ou quatro semanas (TORREÃO, 2007).

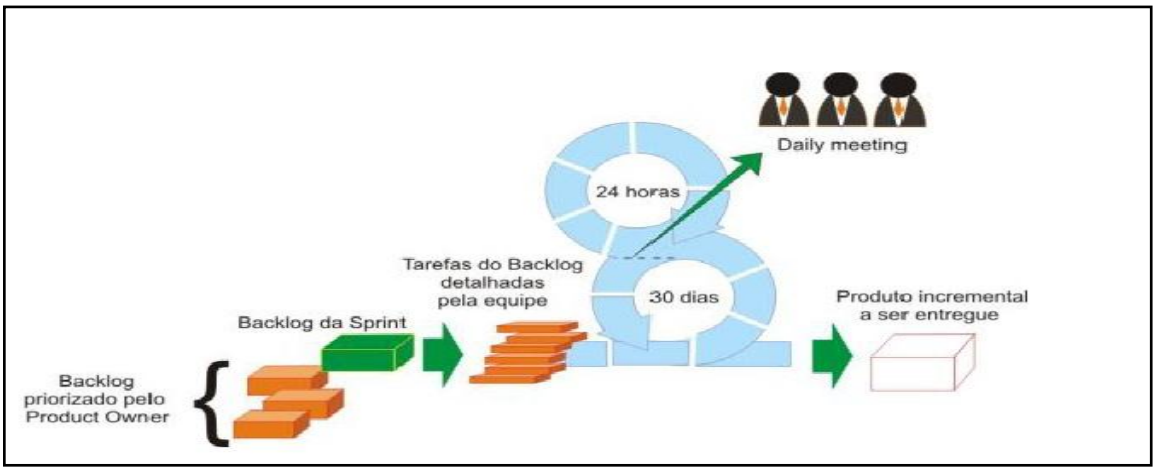
Nessa seqüência de iterações, a próxima será a execução dessas Sprints, onde são realizadas também reuniões rápidas dia-a-dia, que a equipe monitora durante a evolução do desenvolvimento do projeto do software, cada passo evoluído são colocados em um gráfico e no quadro do projeto, onde ao final se verifica se todos os requisitos foram implantados (MARÇAL, 2007).

Seguindo a lógica das iterações o passo seguinte será reuniões de revisão, onde ocorre a validação do que foi desenvolvido e alcançado com os objetivos do negocio desejados. E por fim segue a reunião de retrospectivas, onde

nessa iteração acontece o que foi aprendido, para que nas próximas Sprints sejam melhoradas (PEREIRA, 2007).

Na figura 8 está a representação do funcionamento descrito.

Figura 8 - Ciclo de desenvolvimento do SCRUM de forma simplificada.



Fonte: Adaptado de Vicente (2007).

Os métodos SCRUM são idéias para desenvolvimento de projetos já em funcionamento ou novos, com dinamismo e alterações de requisitos. Para usar esse método com sucesso são necessário obedecer a seus conceitos de cada passo (TORREÃO, 2007).

4.4.2 XP

A metodologia programação extrema (XP) teve inicio em março de 1996 por Kent Beck e Ward Cunningham, onde se tornou o método popular no ambiente de programação de softwares, tem como indicação ser aplicado em empresas de medias para pequeno porte, e composta por uma equipe de dez componentes, e suas principais características se aplica a softwares que seus requisitos são modificados com frequência e não muito definidos (BECK, 1999).

Na prática o método XP se aplica a implementações e um gerenciamento harmônico de ideias de processos. Possui quatro valores entre eles são: comunicação, simplicidade, feedbacke coragem, onde são aplicadas em doze ações práticas que são:

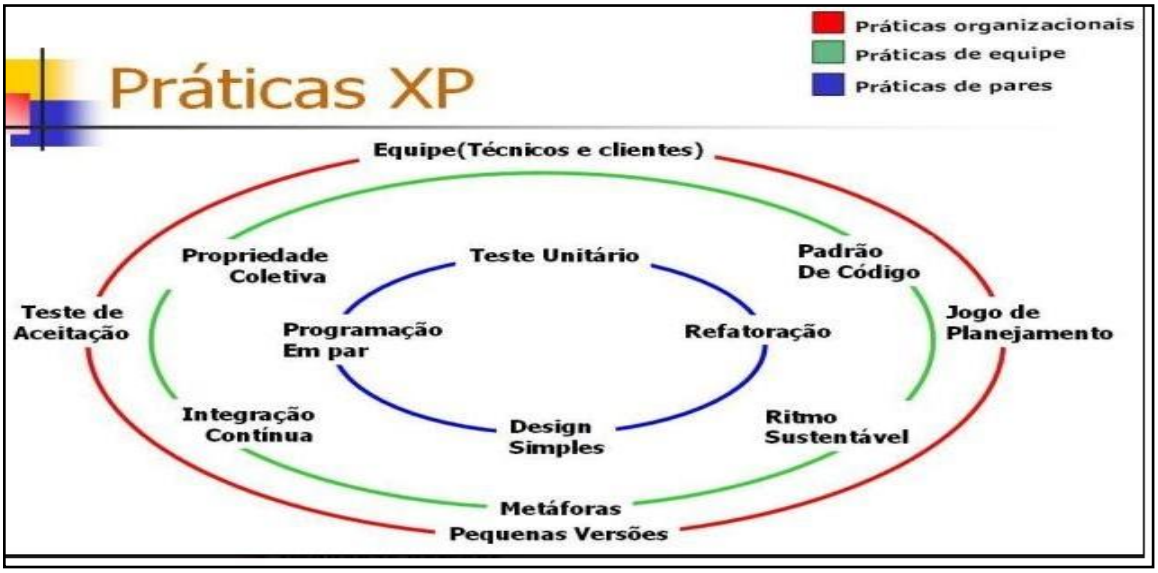
a) jogo do planejamento: define o escopo das novas versões, e estabelecendo características e negócios e técnicas;

b) versões pequenas: implementar um software pequeno, e colocar em funcionamento, e nos próximos dias e semanas entregar novas versões;
c) metáfora: uma descrição simples de como o software funciona, serve de auxilio para o desenvolvimento;
d) projeto simples: o software deve ser implementado de forma clara, não existe complexidade;
e) teste: são testes de unidades, são criados antes de codificar para que a implementação seja executada de forma perfeita e continua. Os clientes também fazem testes e validam as funcionalidades finalizadas;
f) refatoração: consiste em reestruturar o software em todo o seu desenvolvimento, sem alterar seu comportamento externo, serve para flexibilidade e melhorar a codificação do software;
g) programação pareada: são duas pessoas trabalhando a codificação na mesma máquina;
h) propriedade coletiva: em qualquer momento todos da equipe pode alterar a codificação do software;
i) integração continua: o software deve ser desenvolvido em varias partes, e integrado ao fim de cada parte pronta, durante o dia;
j) semana de quarenta horas: consiste em um ritmo de trabalho sem cansar a equipe, fazer horas extras até pode ser normal, mais pode estar acontecendo algo de errado com o projeto;
k) cliente e desenvolvedores: durante o desenvolvimento do projeto os clientes estão trabalhando todo o tempo disponível, ou seja, para responder dúvidas do software;
l) padronização de código: os desenvolvedores codificam o software de forma padronizada, seguindo regras comuns de comunicação por meio de código (PENTEADO, 2010).

Como visto algumas dessas práticas são motivos de criticas, onde para alguns afeta o rendimento do projeto, e a refatoração que acarreta em refazer o que foi feito, essas práticas também não podem ser vistas de forma independente, pois no método XP cada uma tem seu objetivo no processo da metodologia (SIQUEIRA, 2004).

Segue na figura 9 a prática da metodologia ágil XP

Figura 9 - Exemplo de práticas XP a nível organizacional, de equipes e de pares.

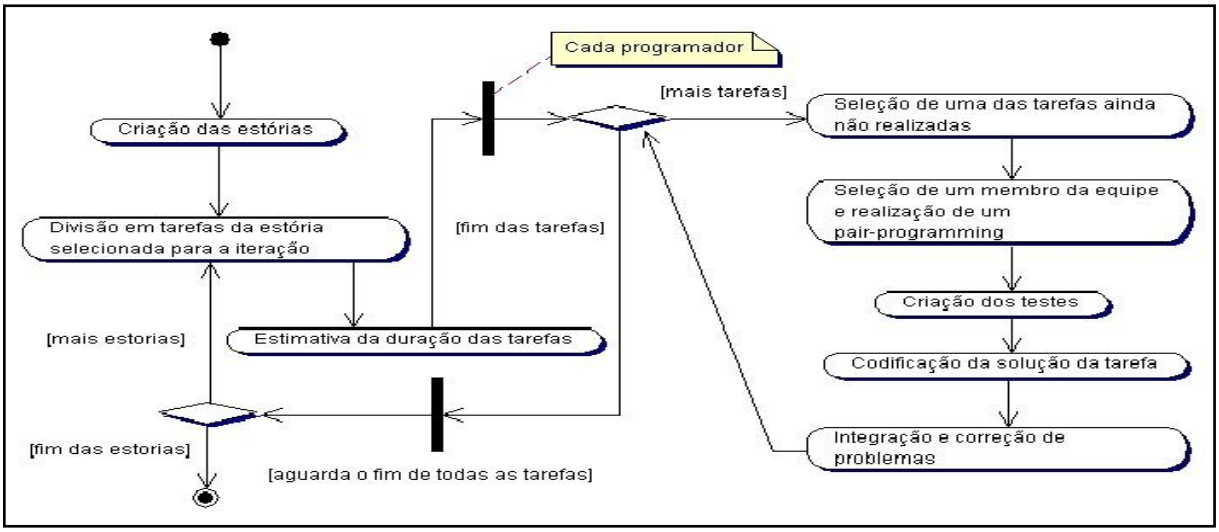


Fonte: Adaptado de Oliveira (2012).

Segundo Highsmith (2004) uma dessas práticas forte, a outra ou a próxima conseqüentemente acontece de maneira simples. Pois alterar uma dessas práticas pode gerar problemas na metodologia XP, entre a equipe de desenvolvimento (SIQUEIRA, 2004).

Na figura 10 representa do ciclo de vida da metodologia XP.

Figura 10 - O processo executado em um projeto normal do XP.



Fonte: Adaptado de Siqueira (2003).

4.4.3 FDD

A metodologia ágil desenvolvimento guiado por funcionalidades, do inglês Feature Driven Development (FDD), que consiste no desenvolvimento guiado por funcionalidades, serve para gerenciar e desenvolver softwares, foi criada em 1997 em Singapura para United Overseas Bank, em um grande projeto em Java (HEPTAGON, 2013).

Esse método de desenvolver por funcionalidades tem como objetivo beneficiar processos rigorosos, entre eles: modelar, prever plano de projeto, controlar o projeto, entre outros. Com isso também possui propriedades dos métodos ágeis dentre eles são: focar na programação, interagir com o cliente e entregar versões do software com mais freqüências. Essa prática para desenvolvimento compreende apenas no software, ou seja, focar na programação que e nas funções de negocio que o software deve realizar, pois o método FDD não se aplica a escolha de ferramentas e tecnologias (SILVA, 2009).

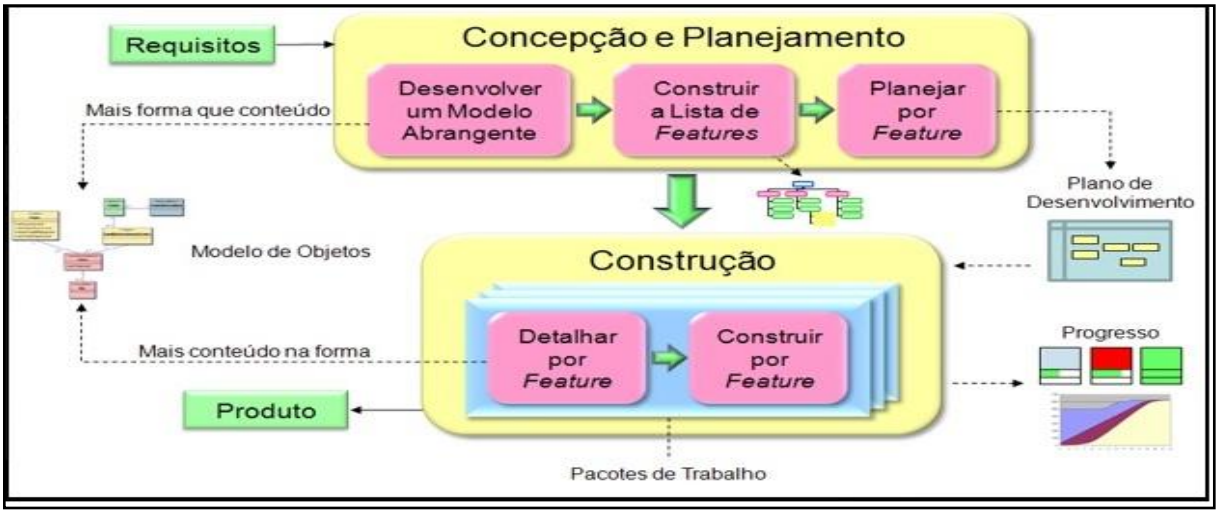
Com essas funcionalidades e propriedades para o desenvolvimento de software, são agregadas e combinadas as melhores práticas de gerenciamento ágil de projetos, fazendo uma abordagem completa em engenharia de software orientado a objetos, e satisfazendo os três tipos de públicos principais em um projeto de software: desenvolvedores, gerentes e clientes (HEPTAGON, 2013).

A metodologia ágil FDD possui cinco processos de desenvolvimento de software, dentre eles são (SILVA, 2009):

a) desenvolver um modelo abrangente: definir o escopo do projeto e estuda um modo detalhado sobre o negocio desenvolvido;
b) construir uma lista de funcionalidades: coletar as funções necessárias para atender as necessidades do cliente;
c) planejar através de funcionalidades: planejar as funções funcionais e não funcionais, onde gera uma lista de classes e são relacionadas aos desenvolvedores do projeto;
d) projetar através de funcionalidades: definir onde cada função da lista terá uma atividade a ser realizada;
e) construir através de funcionalidades: produzir cada iteração e definir cada função, onde agrega valor ao cliente.

Na figura 11 uma apresentação da estrutura da metodologia FDD, esse diagrama mostra cada passo que o método FDD atua em um projeto de software em engenharia de software, onde serve para realizar um gerenciamento de software orientado a objetos (HEPTAGON, 2013).

Figura 11 - Estrutura de funcionamento FDD.



Fonte: Adaptado de Heptagon (2013).

Abaixo uma descrição da lógica da figura 11 onde possui duas fases da metodologia ágil FDD:

a) concepção e planejamento: pensar antes de fazer, onde leva uma a duas semanas;
b) construção: desenvolver de forma iterativa, onde leva duas semanas.

Em seguida um resumo da figura 11 onde seus processos são definidos e integrados:

a) desenvolver um modelo abrangente: análise orientada a objetos;
b) construir a lista de funcionalidades: diluir cada função funcional, analisar as funções;
c) planejar por funcionalidade: planejar de forma incremental;
d) detalhar por funcionalidade: desenho do projeto orientado por objetos;
e) construir por funcionalidade: programar e testar orientado por objetos.

4.4.4 Kanban

A metodologia Kanban originalizou-se no Japão com o objetivo de facilitar a linha de produção em serie, um método criado pela Toyota para controlar e aumentar sua linha de produção automobilística da época. No Brasil foi adotada nos anos oitenta, essa prática se aplica a fluxo de peças e gestão de estoque, onde funciona na prática de cartões, modo visual de gestão de software (SILVEIRA, 2012).

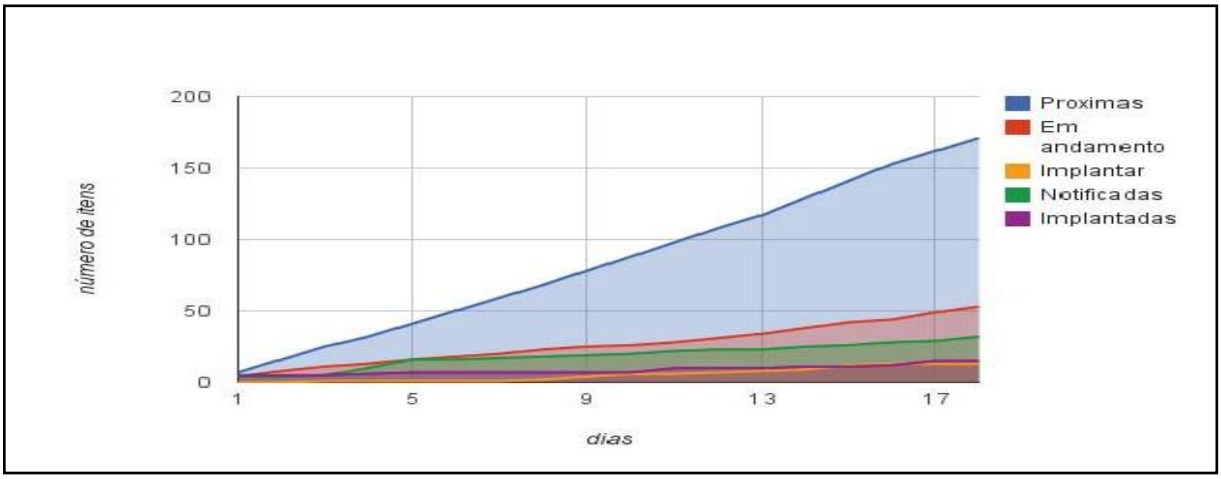
No desenvolvimento de software, o Kanban oferece menos normas e processos para controlar a produção de software, ou seja, com isso cada vez mais as equipes de desenvolvimento utilizam esse método. O método Kanban usa três normas, que são elas:

a) acompanhar o andamento do trabalho atual;
b) limitar o fluxo de trabalho;
c) monitorar e fazer a gestão do trabalho.

Com essas normas, o método acaba se tornando mais satisfatório ao ambiente de desenvolvimento de software, proporcionando a combinação de ferramentas de diversos métodos até obter o processo desejado (SILVA; SANTOS; PEREIRA, 2012).

A figura 12 a seguir representa uma escala do funcionamento da ferramenta visual Kanban entre as normas e procedimentos.

Figura 12 - Diagrama de fluxo cumulativo.



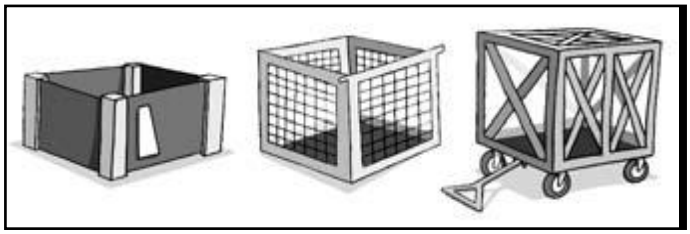
Fonte: Adaptado de SBSI (2012).

A figura 12 mostra a seguinte situação, composta por dois eixos, onde o eixo X representa os dias da iteração e o eixo Y representa o total do numero de itens de cada dia do quadro. Esse tipo de ferramenta visual tem como objetivo mostrar a imagem de todo o processo e comunicar a produtividade da equipe (KNIBERG, 2009).

O funcionamento da metodologia Kanban se caracteriza da seguinte forma, o cartão Kanban serve para a comunicação e o funcionamento do sistema, nele deve estar às informações principais para a linha de produção, se houver a necessidade de mais informação no cartão Kanban, que seja na área especifica onde se pretende implementar a metodologia Kanban (AGUIAR, 2007).

A figura 13 representa o contentor serve para armazenar, por exemplo, em um recipiente padrão, com o numero de peças e um tipo de cartão para essa especificação de peças (PEINADO, 2007).

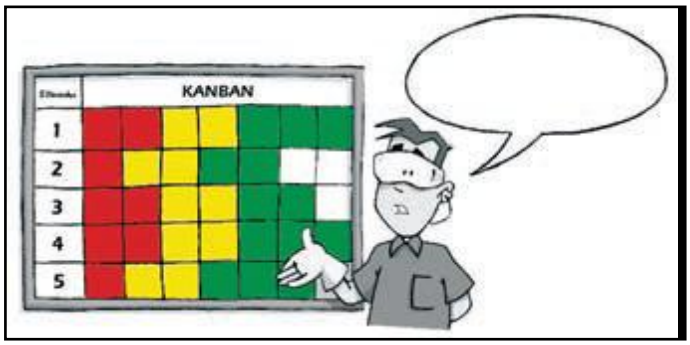
Figura 13 - Ilustração de exemplos de contentores de estoques.



Fonte: Adaptado de Peinado (2007).

A figura 14 a seguir mostra o quadro Kanban, funciona para visualização do estoque que está o setor de peças do estoque, e deve ficar também em um local de acesso a todos (AGUIAR, 2007).

Figura 14 - Ilustração de quadro Kanban.



Fonte: Aguiar e Peinado (2007).

O Kanban externo fica fora da empresa, que representa o fornecedor, e já o Kanban interno fica dentro da empresa, ou seja, localizado em outro setor, podendo ser uma pintura, pré-montagem (PEINADO, 2007).

E o Kanban controlado por cartão, os contentores cheios, ou seja, em um lugar especifico. Ao retirar algum contentor do estoque de seu fornecedor, observe a posição e o lugar que se posiciona.

Na figura 15 mostra-se a observação do operador durante a posição e local de contentores e sua retirada (AGUIAR, 2007).

Figura 15 - Ilustração de um operário observando.

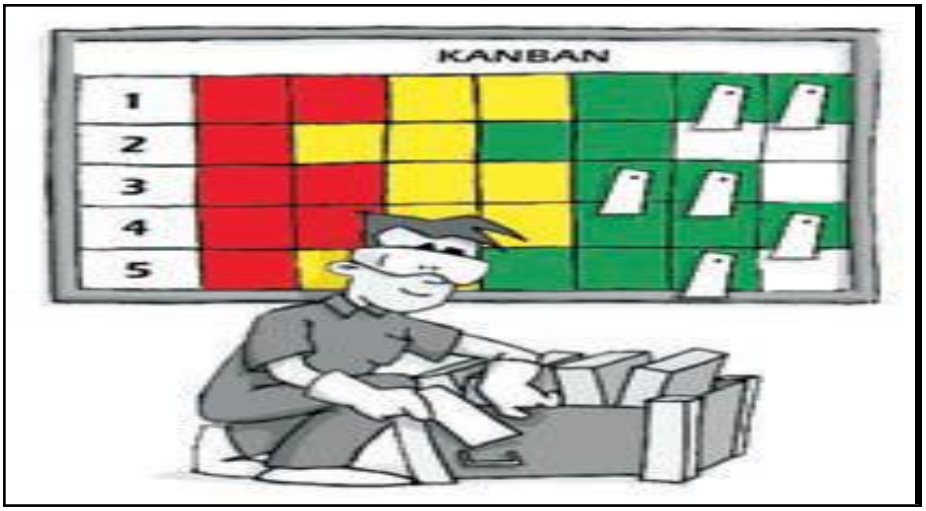
Fonte: Aguiar e Peinado (2007).

Cada cartão representa um contentor, quando o contentor estiver cheio ele, o cartão deve estar fixado ao contentor. Quando o setor de cliente tirar o contentor do estoque, deve também retirar o contentor que estiver usando e colocar no lugar especifico do quadro Kanban. Sempre respeitando a ordem de linhas e colunas, de preferência para os cartões verdes, amarelos e por ultimo os vermelhos (PEINADO, 2007).

Para o fornecedor interno e externo a visualização deve funcionar para saber quantos contentores foram retirados de estoque, e autorizar a produção de mais peças, sempre controlando a quantidade de produção, não passando da quantia necessária, e também controlando o tempo de reposição (AGUIAR, 2007).

A figura 16 Ilustra um operário retirando um cartão de um contentor.

Figura 16 - Ilustração de um operário retirando um cartão de um contentor.



Fonte: Aguiar e Peinado (2007).

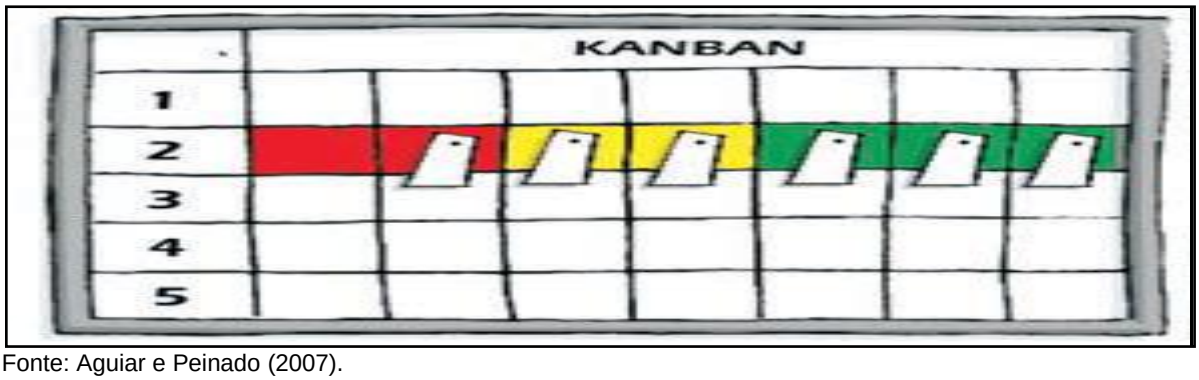
Para um funcionamento controlado de forma satisfatória pela metodologia Kanban, deve ocorrer um balanço perfeito entre a produção e o consumo, pois quanto maior o equilíbrio desses setores melhor o tempo gasto no processo (PEINADO, 2007).

Logo, pois quanto mais contentores for retirados, maior o número de cartões que você terá no quadro do seu fornecedor, quanto o cartão de alguns itens estiver chegando à coluna vermelha, o estoque desse item está no fim, e o seu fornecedor deve produzir mais para repor o estoque necessário (AGUIAR, 2007).

Caso o quadro Kanban estiver vazio, ou seja, sem cartões, significa que os contentores estão completos e não deve ser produzidos (PEINADO, 2007).

Na figura 17 a seguir ilustra um quadro Kanban e um cartão já posicionado na faixa vermelha.

Figura 17 - Ilustração de um quadro Kanban e um cartão já posicionado na faixa vermelha.

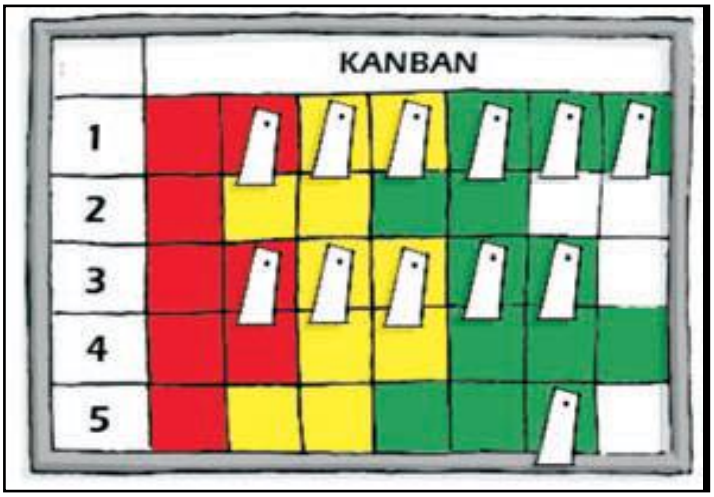


Fonte: Aguiar e Peinado (2007).

As peças que tiverem mais próximas da coluna vermelha são peças que têm prioridade de produção, e também todas que estiverem nessa faixa estão na fase crítica do quadro Kanban (AGUIAR, 2007).

Observe que na figura 18 os componentes das linhas um e três estão cheios de cartões, onde a linha cinco está com apenas um cartão, sendo que esses componentes devem ser produzidos primeiro (PEINADO, 2007).

Figura 18 - Ilustração de um quadro Kanban de prioridades.



Fonte: Aguiar e Peinado (2007).

Se essas peças forem produzidas internamente ocorrendo o Kanban interno, ou for recebido dos fornecedores ocorrendo o Kanban externo, o contentor vai para o lugar de estoque determinado e o cartão deve ser retirado do quadro e colocado fixado ao contentor (AGUIAR, 2007).

4.4.5 DSDM

A metodologia de desenvolvimento de sistemas dinâmicos, do inglês Dynamic Systems Development (DSDM) para desenvolvimento de software faz com que ocorra o envolvimento continuo do usuário, com o objetivo de entregar sistemas no tempo e orçamentos previstos, ou seja, enquanto se ajusta as alterações dos requisitos ao longo do processo de desenvolvimento (TENÓRIO, 2008).

Esse método foi fundado em janeiro de 1994, na Inglaterra, e essa metodologia ágil continua serve de base formal reconhecida pelo governo daquele país. Foi criada com a união de empresários e especialistas da área de

desenvolvimento de software, onde utilizaram experiências em ambas as áreas (WANDERLEY, 2008).

O método desenvolvido consiste em três fases, entre elas: pré-projeto, ciclo de vida do projeto, pós-projeto. E a fase do ciclo de vida está subdividida em cinco práticas onde são: estudo da possibilidade de concentração, estudo de negócio, interação funcional de modelo, interação de design e construção por último implementação. Em alguns casos se aplica outras metodologias integradas à nessas mencionadas (MENEZES, 2008).

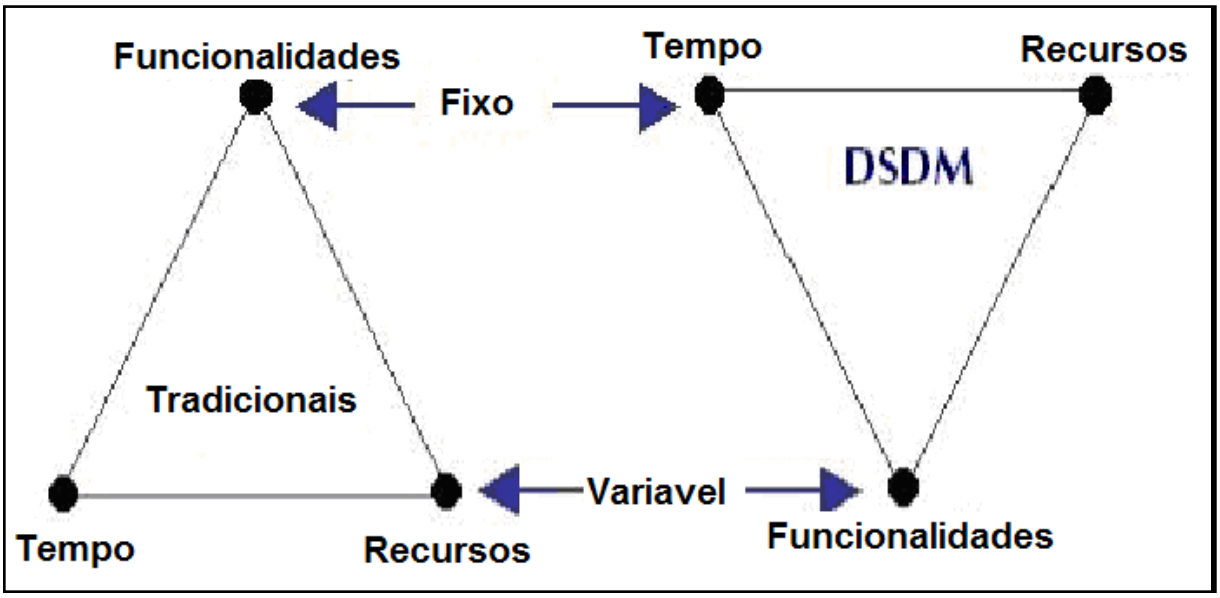
A metodologia DSDM consiste de nove princípios que são:

a) ação ativa dos usuários apresenta características autoritárias;
b) o time envolvido tem que tomar decisões;
c) a entrega de produtos está no foco e mais freqüente;
d) se adéqua aos negócios, um princípio fundamental para a aceitação de produtos entregues;
e) desenvolver de forma iterativa e incremental para concentrar-se na solução de negócios;
f) as mudanças no desenvolvimento podem sofrer reversões;
g) os requisitos são fundamentados em alto nível;
h) os testes são incorporados ao ciclo de vida;
i) as partes interessadas são essenciais em cooperação e colaboração.

A metodologia procura resolver questões de negócios em engenharia de software, onde os princípios citados servem para aproximar clientes, equipes de desenvolvimento, gestores do projeto, e entre outros, a fim de cumprir prazos para entregas, custos (BRODERICK, 2008).

A figura 19 representa princípios fundamentais da metodologia DSDM.

Figura 19 - Princípio fundamental da DSDM.



Fonte: Adaptado de Teixeira (2005).

4.4.5.1 As fases da metodologia DSDM

O método DSDM consiste de três fases em seqüência com mencionado em parágrafos anteriores entre elas: pré-projeto, projeto e pós-projeto. Onde a fase do projeto DSDM consiste na mais detalhada das três fases. Logo abaixo está descrito os cinco níveis detalhado passo-a-passo e iterativo do desenvolvimento de sistemas de informação (TEIXEIRA, 2005).

a) **1ª fase (pré-projeto):** essa fase do projeto consiste na identificação do projeto, ou seja, serve para estabelecer um compromisso a ser realizado, se define um plano para financiar o projeto;

b) **2ª fase (ciclo de vida do projeto):** estabelece um processo que mostrar os cinco níveis do ciclo de vida do projeto que a equipe terá de desenvolver. Onde os dois primeiros que são estudo de viabilidade e o estudo de negócio, são fases que uma complementa a outra de forma seqüencial. Logo que essas fases forem realizadas o sistema pode ser desenvolvido de forma iterativa, e conseqüentemente de forma incremental os níveis Análise Funcional, Desenho e implementação:

- Nível 1 (estudo da viabilidade): este nível do projeto passa por uma análise, os pré-requisitos para a prática do método DSDM são questionados e respondidos, por exemplo: este projeto se adéqua ao

uso do DSDM, quais são os riscos mais importantes envolvidos, e entre outras questões. Neste nível são preparados ao cliente o relatório e o protótipo de viabilidade, onde relata a viabilidade do projeto. E faz-se um registro de risco, onde identifica os riscos mais importantes do projeto,

- **Nível 2 (estudo do negócio):** este nível desenvolve todo o trabalho que o estudo de viabilidade realizou, onde examina o processo de financiar o projeto. Os interessados se reúnem as oficinas e elaboram o sistema proposto. Logo após a reunião faz-se uma lista de requisitos, e são estabelecidas prioridades e entre elas usa-se MoScow[1]. Onde serve para escalonar, e um plano de desenvolvimento construído para ser com guia mestre do resto do projeto. E nesse plano tem uma técnica importante que se chama TimeBoxing[2] que garante a qualidade desejada de orçamento e tempo do projeto. Outra característica importante do sistema está em uma arquitetura do projeto, onde serve de guia para o desenvolvimento do sistema juntamente com o plano de desenvolvimento. E por fim está a lista de prioridades de requisitos, onde são definidos os requisitos do sistema, que são de acordo com o principio do MoScow, e em seguida atualiza-se o registro de riscos,
- **Nível 3 (Análise funcional):** os requisitos anteriores são convertidos em modelo funcional. Nesse nível está uma técnica chamada prototipagem onde faz o envolvimento do utilizador final com o projeto. Esse tipo de protótipo se realiza em diferentes grupos de utilizadores, para garantir a qualidade do projeto, ou seja, esses testes são implementados nas iterações da DSDM. Além do que essa análise funcional está dividida em mais quatro níveis entre eles são: identificação do protótipo funcional, ajustar calendário de

[1]A técnica *MoSCoW* representa um método de definição de prioridades nas tarefas efetuadas. Neste contexto, a técnica *MoSCoW* da DSDM é usada para dar prioridade aos requisitos enunciados. (TEIXEIRA et tal., 2005)

[2]Encapsulamento do tempo, ou seja, realizar o desenvolvimento de um Sistema de Informação no tempo previsto, dentro do orçamento e com a qualidade desejada. (TEIXEIRA et tal,. 2005)

tarefas, criar protótipo funcional e revisar o protótipo. Onde esses tipos de teste são realizados com utilizadores finais,

- **Nível 4 (Desenho):** esse nível realiza-se a integração dos componentes funcionais no sistema, para satisfazer as necessidades do utilizador. O desenho pode ser dividido em quatro sub-níveis: identificar protótipo de desenho, reconhecer requisições funcionais e não funcionais; ajustar calendário de tarefas, determinarem as datas para implementação dos requisitos; criar protótipo de desenho, criar o sistema que possa ser seguro de uso diário aos utilizadores; revisar o protótipo de desenho, checar exaustivamente o sistema desenhado,
- **Nível 5 (implementação):** esse nível serve para começar a serem treinados os utilizadores do sistema, onde são entregue o sistema testado e sua documentação. Onde também esse nível pode ser dividido em quatro sub-níveis entre eles: aprovações do utilizador aprovam o sistema testado, e as linhas mestres os usos do sistema são criados; treinar os utilizadores, para o futuro uso do sistema; Implementação, desenvolver o sistema testado no local de trabalho dos futuros utilizadores finais; ajustar negócio, alcançar o objetivo desejado no inicio do projeto, onde se não obtiver resultados de aprovação voltará para as fases anteriores, ou chegará à fase de pós-vendas;

c) **3ª fase (Pós-projeto):** essa fase serve de atualizações, onde são feitas manutenções de melhoramento conforme são propostas com as características da metodologia DSDM (PINTO; PIRES, 2005).

Com isso o ciclo de vida de um projeto em DSDM constitui-se da seguinte forma:

a) estudo de viabilidade: requisitos básicos e restrições do negócio;
b) estudo do negócio: identifica os requisitos funcionais e define a arquitetura básica da aplicação;
c) iteração do modelo funcional: constrói um conjunto de protótipos incrementais;
d) iteração de projeto e construção: verifica os protótipos construídos garantindo que cada um tenha passado por engenharia;

e) implementação: coloca o último incremento do software no ambiente de produção.

A metodologia ágil DSDM para alcançar resultados de sucesso desejado, deve ter interatividade com a equipe, desenvolvedores e utilizadores. E não apropriado para sistemas de resultados de estado crítico, ou seja, testes e validações afetam as características do método DSDM (SANTOS, 2005).

4.6 COMPARAÇÕES: GERENCIAMENTO CLÁSSICO E GERENCIAMENTO ÁGIL DE PROJETOS

A escolha para desenvolver um projeto de software está submetida ao tamanho, ou seja, para projetos de grande porte sugere que utiliza a metodologia clássica, pois os focos são para importância de planejamento e controle. E na escolha da metodologia ágil funciona no lugar do planejamento a execução, onde apresenta maior número de entrega e de resultados ao cliente, e no lugar do controle a adaptação, oferece alterações de escopo a cada iteração, e com isso atendendo melhor os requisitos do negócio (HIGHSMITH, 2004).

Esses processos são atividades sequenciais onde são desenvolvidos por papéis em equipe ou individual, pequenos softwares ou já o sistema completo. Com isso essas duas metodologias trabalham de forma semelhante, ambas abordam pontos fortes e fracos, onde alguns pontos da metodologia ágil podem ser aplicados na metodologia clássica, e o processo inverso também pode acontecer. Sendo assim forma uma metodologia híbrida entre as duas (DOMINGOS; SANTOS, 2010).

No desenvolvimento de projetos de software, não importa qual metodologia utilizar, o foco principal está na qualidade final e satisfação do cliente. O objetivo do desenvolvimento de software com tudo isso mencionado faz com que reduza possíveis erros que comprometerão a qualidade e satisfação do cliente (DOMINGOS; SANTOS, 2010).

Com os objetivos descritos de cada metodologia acima, pode se chegar a uma comparação entre as principais características dos grupos de processos pelo PMI (2004), do lado clássico: iniciação, planejamento, execução, monitoramento e controle e encerramento, e do lado ágil – visão, especulação, exploração, adaptação e encerramento (KOPPENSTEINER; UDO, 2003).

Na tabela 1 representando a análise comparativa dos gerenciamentos clássicos e ágil de projetos de software.

Tabela 1 - Análise comparativa entre gerenciamento clássico e ágil de projetos.

Grupo de processos do gerenciamento clássico de processos	**Fases do gerenciamento ágil de projetos**
Iniciação: definir escopo inicial do projeto	Visão: definir visão e escopo do produto
Planejamento: planejamento detalhado do projeto.	Especulação: desenvolver plano inicial do projeto, e definir planos individuais de cada iteração.
Execução: executar plano do projeto.	Exploração: entrega das funcionalidades, produtos finalizados a cada ciclo
Monitoramento e controle: controlar progresso dos trabalhos e monitorar mudanças no gerenciamento do projeto.	Adaptação: revisão dos resultados, adaptação do escopo.
Encerramento: realizar processo final do projeto.	Encerramento: aceitação do cliente a cada ciclo e processo final do projeto.

Fonte: Udo e Koppensteiner (2003).

O gerenciamento clássico de projetos enfatiza a importância no planejamento minucioso do projeto e nos processos formais de monitoramento e controle. E no lado do gerenciamento ágil de projetos a prioridade se dá a execução e não planejamento, fazendo uma entrega rápida e resultados em todo o desenvolvimento do projeto, e a troca de controle pela adaptação, onde permite modificações no escopo em cada iteração e atender as mudanças de requisitos de negocio (PMI; THOMSET, 2004).

Uma representação na tabela 2 de comparação em gerenciamento de projetos clássicos e ágeis.

Tabela 2 - Análise comparativa entre gerenciamento clássico e ágil de projetos.

Área de conhecimento	Gerenciamento clássico de projetos.	Gerenciamento ágil de projetos.
Gerenciamento da integridade de projetos.	Coordenação dos elementos do projeto.	Coordenação formal, onde se limita a redução de estrutura e processos.
Gerenciamento do escopo de projetos.	Certificar que tenha somente o trabalho a ser realizado, onde também se define o que está e não está no projeto.	Não há controle formal no andamento do projeto, e o escopo se fixa depois que as iterações estão em andamento.
Gerenciamento do tempo do projeto.	Definição o tempo e atividades, elaborando um cronograma detalhado para garantir a entrega do projeto.	Cronograma baseado em funcionalidades e não em atividades, tempo estabelecido em ciclos e iterações.
Gerenciamento do custo do projeto.	Estabelecer custos, controle e material, partindo dos recursos humanos, para assegurar a entrega do projeto no orçamento confirmado.	Definir o orçamento das funcionalidades do software desejado, onde recursos, funcionalidades, e prazos são balanceados para medir o custo se houver atrasos.
Gerenciamento da qualidade do projeto.	Atender as necessidades estabelecidas e desejadas, a fim de concordar com as adequações especifica e satisfazendo os interesses do cliente.	Focaliza na usabilidade do cliente, onde ao usar o software o usuário apresenta uma resposta de cada iteração, conforme a necessidade que foi estabelecida.
Gerenciamento de recursos humanos do projeto.	Faz-se uso de todas as partes interessadas no projeto.	Focaliza na equipe e não nos indivíduos, incentivação na produtividade da equipe.
Gerenciamento das comunicações do projeto.	Confirmação da coleta, processamento e armazenamento das informações do projeto.	Elimina gastos, padrões, documentos, relatórios, fazendo com todos tenham acesso às informações do projeto.
Gerenciamento das aquisições do projeto.	Aquisição de serviços externos ou produtos, a empresa desenvolvedora, para realizar o projeto.	Aquisições fundamentais para estabelecer bens e serviços, onde oferece melhor e maior colaboração e não a negociação de contratos.

Fonte: Udo e Koppensteiner (2003).

A tabela demonstrou que todas as áreas de conhecimento apontam para o gerenciamento ágil de projetos, onde se relaciona com os clientes e seus valores estabelecidos, nas respostas rápidas às mudanças necessárias e nos méritos pessoais (KOPPENSTEINER; UDO, 2003).

5 TRABALHOS CORRELATOS

5.1 A GERÊNCIA DE PROJETOS NOS PROCESSOS DE DESENVOLVIMENTO DE SOFTWARE: TECNOLOGIAS E FERRAMENTAS PARA EMPRESAS DE PEQUENO PORTE E A SUA INTERAÇÃO COM OS MODELOS ÁGEIS.

Segundo POERCH E CASTIÑEIRA (2010) desenvolveram uma dissertação de trabalho de conclusão de curso em um estudo dos modelos ágeis, ferramentas de software livres disponíveis e de técnicas da gerencia de projetos para oferecer suporte às atividades nas empresas de desenvolvimento de projetos de software. Desenvolvido em 2010 na universidade Unisul em Santa Catarina, onde para a realização foi feita a seleção de uma empresa desenvolvedora de projetos de software, onde aceitou o estudo proposto. Nesse estudo foi aplicado um questionário e uma entrevista, que averiguou como a empresa gerencia e prática a engenharia de software nos softwares desenvolvidos. Com isso pode-se constatar quais ferramentas poderia ser mais usadas e qual o modelo ágil que se adapta a determinados projetos. Mais a empresa não usa uma ferramenta de gerenciamento de software, ela utiliza e-mails devido à quantidade de funcionários. A empresa mostrou-se interessada na adoção do modelo SCRUM, apesar de não ter tempo suficiente para aplicar no estudo proposto.

5.2 UM ESTUDO E UMA FERRAMENTA DE GERÊNCIA DE PROJETOS COM DESENVOLVIMENTO ÁGIL DE SOFTWARE

De acordo com CISCON (2009) desenvolveu uma dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais, realizado em 2009 como requisitos parcial para a obtenção do grau de Mestre em Ciência da Computação, onde foi analisadas características de pessoas e das práticas utilizadas no desenvolvimento de software. Com objetivo de atender o gerenciamento de projetos, qualidade de software e a customização de metodologias de engenharia de software. Não foi realizado um estudo na metodologia clássica de gerenciamento. Visto isso se constatou que gerenciar projetos faz com que ofereça satisfação aos interessados e para sucesso no

desenvolvimento de software de negocio. E com esse estudo contribui-se para escolha da metodologia a ser aplicada e ferramenta a se desenvolvido o projeto. Algumas contribuições apresentadas após o estudo são: definição da metodologia a se adotada, características de qual ferramenta utilizar, elaborar um guia onde serve de orientação para os gerentes de projetos.

5.3 ANÁLISE COMPARATIVA DAS FERRAMENTAS DE GERENCIAMENTO DE PROJETOS DISPONÍVEIS NO MERCADO E SUA APLICABILIDADE EM FÁBRICAS DE SOFTWARE

Segundo SILVA, BARROS, LOPES (2006) desenvolveram um trabalho de conclusão de curso da universidade Fumec/Face em Belo Horizonte Minas Gerais, de iniciação cientifica de 2006 onde trata de uma análise comparativa das ferramentas de gerenciamento de projetos disponíveis no mercado e sua aplicabilidade em fábricas de software. O objetivo foi na área de desenvolvimento de sistemas, onde precisa de uma visão focada em gestão das atividades para atingir sucesso nos prazos, custos e qualidade no software de negocio das empresas. Com isso também se analisou ferramentas de gerenciamento de projetos, para atender os impactos que os recursos humanos podem gerar nos custos de prazos no desenvolvimento de software de negocio, ou seja, profissionais e equipes com poucas experiências.

De qualquer maneira, a analise de comparação de ferramentas livres de gerenciamento de software, mostrou os recursos e conflitos que o ambiente de gerenciamento de projetos ocorre. Com isso também se pode tratar das opções disponíveis de cada ferramenta, em um ambiente com múltiplos projetos.

5.4 GESTÃO DE PROJETOS COM PROCESSOS ÁGEIS

De acordo com PINTO (2010) onde elaborou uma dissertação de mestrado em Engenharia de Informática e de Computadores, realizada em 2010 no Instituto superior técnico da Universidade técnica de Lisboa Portugal onde trata de uma Gestão de Projetos com Processos Ágeis. Com o objetivo de atender as necessidades de clientes e ajustar o conceito de metodologias tradicionais e ágeis ao desenvolvimento de projetos de software. Analisando e comparando ferramentas

com metodologias, pretendendo apoiar a gestão de projetos em ambas as metodologias ágeis ou tradicionais, e promover a comunicação e colaboração das pessoas da equipe de desenvolvimento. Com isto foi elaborado um modelo para comparar e analisar as ferramentas de suporte das metodologias ágeis, entre as ferramentas analisadas: VersionOne, PivotalTracker, RallyDev e TargetProcess. Outra ferramenta foi adicionada para dar suporte aos métodos clássicos que foi a PIT-Enterprise-2007.

Visto esta analise conclui-se que os tempos de trabalho juntamente com os pacotes das tarefas devem ser considerados com um valor em Story Points, bem como uma forma de avaliar o esforço relativo. E outras funções avaliadas como: históricos de pacotes de tarefas e trabalho também podiam ser de fácil acesso. E a ferramenta Eternprise-2007 possibilita só a pratica de métodos clássicos. De certa forma essa comparação e analise promove a sustentabilidade e o alinhamento de projeto independente de cada empresa, juntamente como cada método de desenvolvimento de software deve ser escolhido para atender as dimensões dos projetos.

5.5 ESTUDO COMPARATIVO DA ADERÊNCIA DE FERRAMENTAS LIVRES AO PMBOK (2004)

Segundo FORNARI (2009) desenvolveu uma monografia de graduação ao Departamento de Ciência da Computação da Universidade Federal de Lavras, UFLA, como parte das exigências do curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação, realizada em 2010. Com o objetivo de comparar três gerenciadores de projetos, onde se realiza um estudo comparativo da aderência de ferramentas livres ao PMBOK. Verificaram-se as seguintes funcionalidades e propriedades de cada gerenciador conforme os processos do PMBOK: análise de requisitos; detalhar; projeto; execução e conclusão da avaliação; calendário de gerencia de tempo; quadro de atividades para gerenciar os recursos humanos e e-mails e telefones de contatos de cada membro da equipe que estiver envolvido no projeto na gerencia de comunicação.

Com isso se conclui que os gerenciadores analisados possuem falta de alguns recursos, levando em consideração do projeto a ser desenvolvido, ou seja, apresentaram carência em compatibilidade no guia PMBOK. Mais apesar dessa

ausência de alguns recursos esses softwares de gerencia de projetos analisados, são capazes de fazer o gerenciamento de projetos.

6 FERRAMENTAS DE GERENCIAMENTO DE PROJETOS OPENSOURSE GANTTPROJECT, OPENPROJECT, ARTIA, REDMINE, TRELLO

Neste capitulo serão descritas as ferramentas com suas características importantes do ponto de vista das metodologias ágeis. Foi feito estudo em algumas ferramentas livres sem licença, a escolha foi feita por serem as mais conhecidas do mercado de gerenciamento de software.

A ênfase para descrição das ferramentas foi na metodologia Scrum, pois foi o método escolhido com, mas freqüência no desenvolvimento de projetos nas empresas brasileiras, onde corresponde: planejamento, iteração, reuniões e refatoração, na produção dos projetos de software.

A escolha por descrever as ferramentas deu-se ao fato de melhor conhecê-las para depois analisá-las em seus atributos.

6.1 FERRAMENTA DE GERENCIAMENTO DE PROJETOS ARTIA

Artia possui recursos para administrar projetos de software de forma eficiente, gerencia em uma estrutura cem por cento web, ou seja, o acesso funciona pela internet. Uma solução completa para o mercado de desenvolvimento de aplicativos, e não precisa instalação, pois todas as atualizações, backups e servidores fazem parte de um pacote básico grátis por tempo ilimitado.

6.1.1 Etapa de verificação da metodologia SCRUM no gerenciador Artia

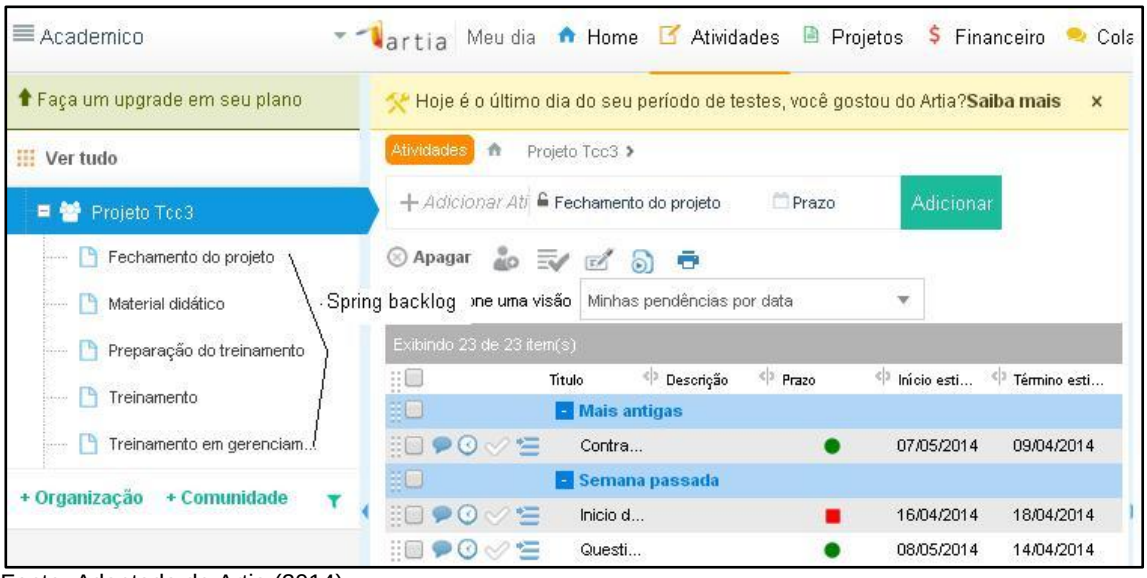
Product backlog: atua na criação de comunidades, projetos e pastas para gerenciar suas necessidades. Uma atividade pode ser vinculada a uma tarefa, onde na figura 20 mostra que o projeto TCC III atua como uma comunidade.

Figura 20 - Projeto e pastas.

Fonte: Adaptado de Artia (2014).

Spring backlog: esse termo funciona com uma reunião pelo método Scrum, onde o produto são o conjunto de atividades e o escopo que deve ser realizado na próxima Sprint.

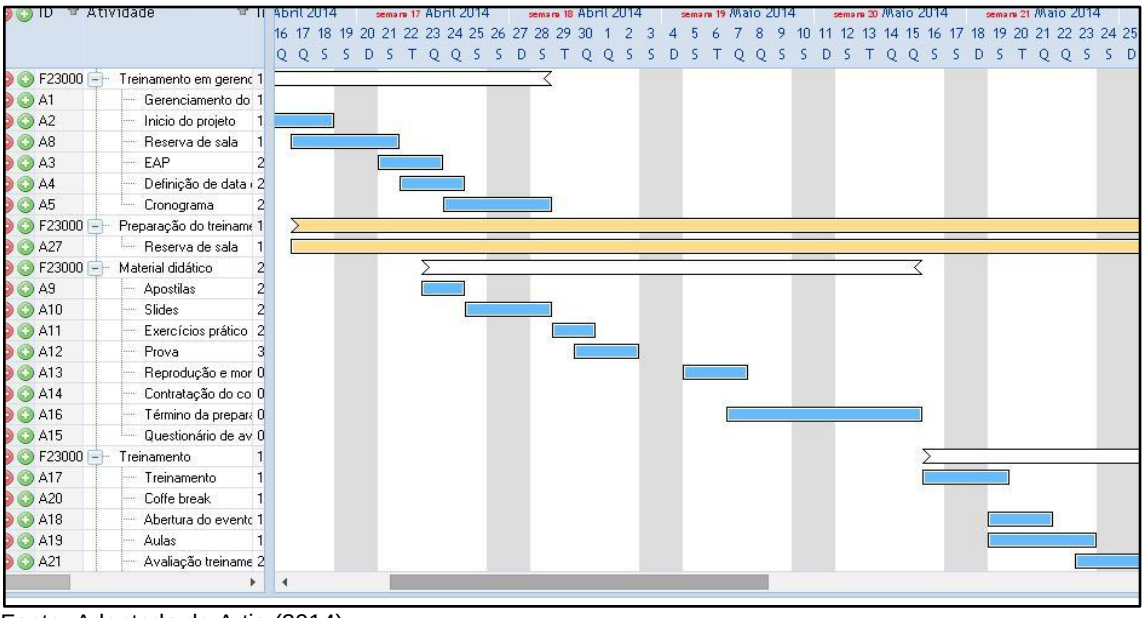
Figura 21 - Escopo de atividades.



Fonte: Adaptado de Artia (2014).

Burndown: o gráfico de gantt na ferramenta Artia possibilita acompanhar as tarefas por atividades, ou seja, selecionar uma tarefa e a figura do projeto aparece de forma clara no projeto em desenvolvimento.

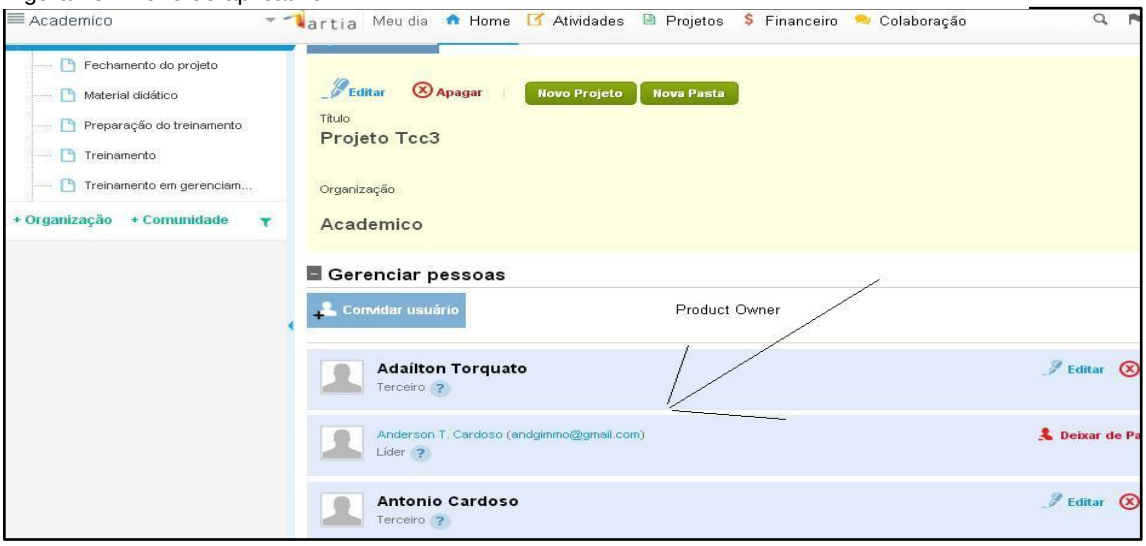
Figura 22 - Gráfico de gantt por atividades.



Fonte: Adaptado de Artia (2014).

Product Owner: na estrutura Artia possui uma tela de cadastrar o dono do projeto, que nesse caso na figura 23 mostra como líder das atividades.

Figura 23 - Dono do aplicativo.



Fonte: Adaptado de Artia (2014).

Equipe: a ferramenta tem uma estrutura de cadastrar e convidar pessoas para fazer parte da equipe, através das comunidades e por pessoas associadas.

Figura 24 - Equipe de desenvolvimento.

Fonte: Adaptado de Artia (2014).

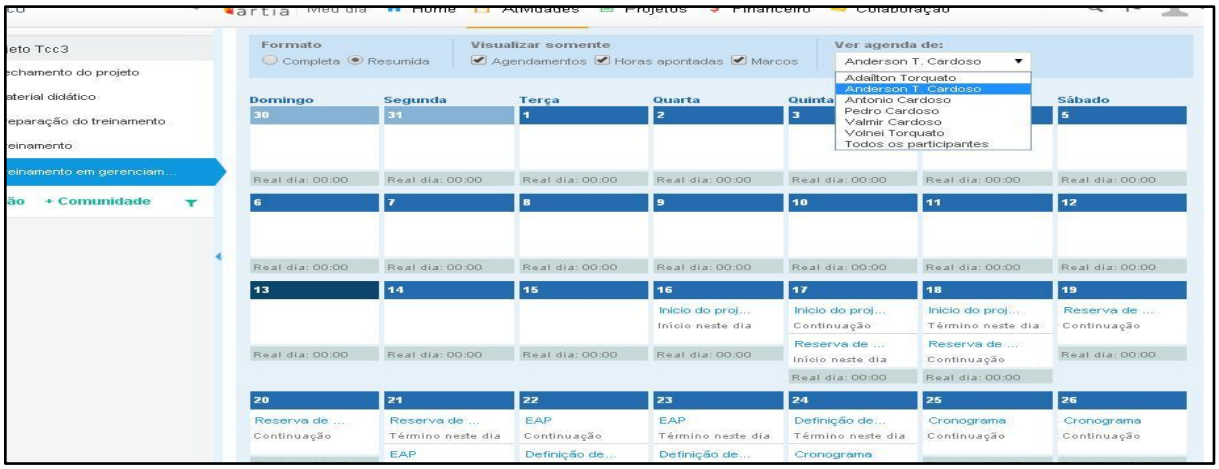
Scrum Master: na tela de gerenciar pessoas existe a opção de editar recursos que pessoas exercem nas tarefas, e como a figura 25 mostra o líder da equipe.

Figura 25 - Líder do projeto.

Fonte: Adaptado de Artia (2014).

Reunião de planejamento sprint: no planejamento de sprints, possibilita visualizar os agendamentos, ou seja, mostra um calendário e nele selecionar os integrantes da equipe, onde mostra as datas das atividades de cada membro da equipe.

Figura 26 - Planejamento por pessoas da equipe.



Fonte: Adaptado de Artia (2014).

Revisão sprint: uma característica da estrutura da ferramenta Artia, acompanhar o andamento do projeto, visualizar as sprints concluídas e as não concluídas.

Figura 27 - Revisão sprint por status.

Fonte: Adaptado de Artia (2014).

Retrospectiva Sprint: nessa sprint possibilita que todos os membros da equipe reflitam sobre os sprints passados, onde podem fazer melhorias e alterar datas, agendamentos, associar pessoas e entre outros recursos.

Figura 28 - Alterar processo na atividade.

Fonte: Adaptado de Artia (2014).

6.2 FERRAMENTA DE GERENCIAMENTO DE PROJETOS REDMINE

Um aplicativo de arquitetura web de gerenciar projetos de software, desenvolvido em Ruby on Rails e roda em vários sistemas operacionais e bancos de dados. Suas características são: controle de acesso por pessoas do projeto, calendário, gráfico de gantt e entre outros.

6.2.1 Verificação da metodologia Scrum no gerenciador Redmine

Product backlog: logo após logar no sistema, na opção tarefas possibilita visualizar uma lista de backlogs, onde consiste selecionar cada uma conforme a necessidade.

Figura 29 - Lista de backlogs.

Fonte: Adaptado de Redmine (2014).

Na figura 30 mostra itens do projeto, ou seja, esses são vindos do cliente que priorizam para o desenvolvimento do projeto do sistema.

Figura 30 - Lista de itens do projeto.

Fonte: Adaptado de Redmine (2014).

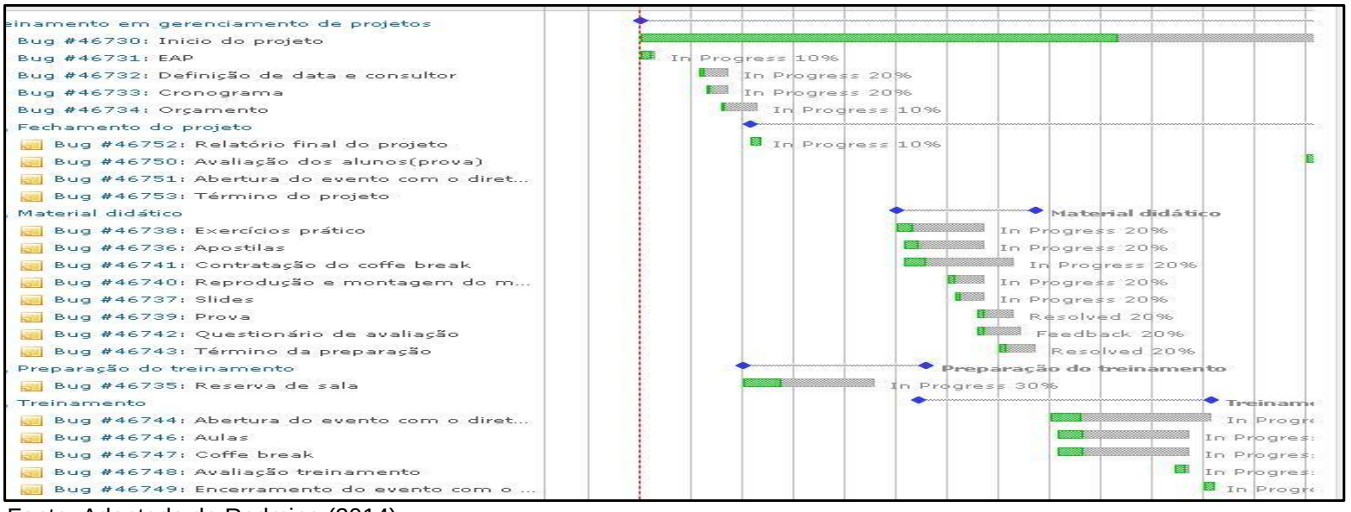
Sprint backlog: ao selecionar um product backlog, e na opção tarefas mostra as atividades que são os sprint backlogs do projeto. A figura 31 mostra com detalhes essa possibilidade.

Figura 31 - Lista de tarefas.

Fonte: Adaptado de Redmine (2014).

Burndown: o gráfico de gantt na estrutura do Redmine mostra a esquerda os product backlogs e logo abaixo os springs backlogs e a direita um gráfico de andamento do projeto. Dessa forma os springs são representados como pastas.

Figura 32 - Gráfico de gantt geral do projeto.



Fonte: Adaptado de Redmine (2014).

Product Owner: na metodologia Scrum o proprietário do produto aparece na ferramenta Redmine na opção de visão geral, onde a visualização na tela está a direita com o nome manager, ao lado das tarefas em aberto a esquerda.

Figura 33 - Proprietário do projeto.

Fonte: Adaptado de Redmine (2014).

Equipe: os membros de um projeto de software no Redmine estão opção configurações e depois em membros, apresenta os usuários e grupos conforme mostra na figura 34, bem como seus respectivos papéis de cada pessoa, possibilita também editar e excluir membros.

Figura 34 - Lista de membros da equipe do projeto.

Fonte: Adaptado de Redmine (2014).

Scrum Master: o líder do projeto também está na opção membros, usuários e grupos, onde possibilita editar e excluir outros lideres ou trocar de maneira que o Product Owner desejar.

Figura 35 - Líder do projeto.

Fonte: Adaptado de Redmine (2014).

Reunião de planejamento sprint: planejar reuniões no Redmine funciona de forma de agendamento no calendário das sprints do projeto, onde mostra o começo na seta verde e o fim na seta vermelha de cada tarefa.

Figura 36 - Planejamento da equipe.

Fonte: Adaptado de Redmine (2014).

Reunião de revisão sprint: ao selecionar uma tarefa do projeto, e logo depois a opção do menu visão geral na figura n, a estrutura Redmine possibilita visualizar as revisões de sprints, ou seja, revisar as tarefas em andamento.

Figura 37 - Revisões por atividades.

Fonte: Adaptado de Redmine (2014).

Retrospectiva sprint: como na opção anterior, também está na mesma estrutura a retrospectiva das sprints (tarefas), ou seja, possibilita fazer melhorias no desenvolvimento que estão em aberto, visualizar características e apoio a suporte.

Figura 38 - Retrospecto sprint por tarefas.

Fonte: Adaptado de Redmine (2014).

6.3 FERRAMENTA DE GERENCIAMENTO DE PROJETOS TRELLO

Uma ferramenta de uso colaborativo que funciona de forma ágil, sua característica de organização seus projetos em quadros, ou seja, que são chamados de boards. Também possibilita a criação de listas de tarefas que são representadas em cards.

6.3.1 Verificação da metodologia Scrum no gerenciador Trello

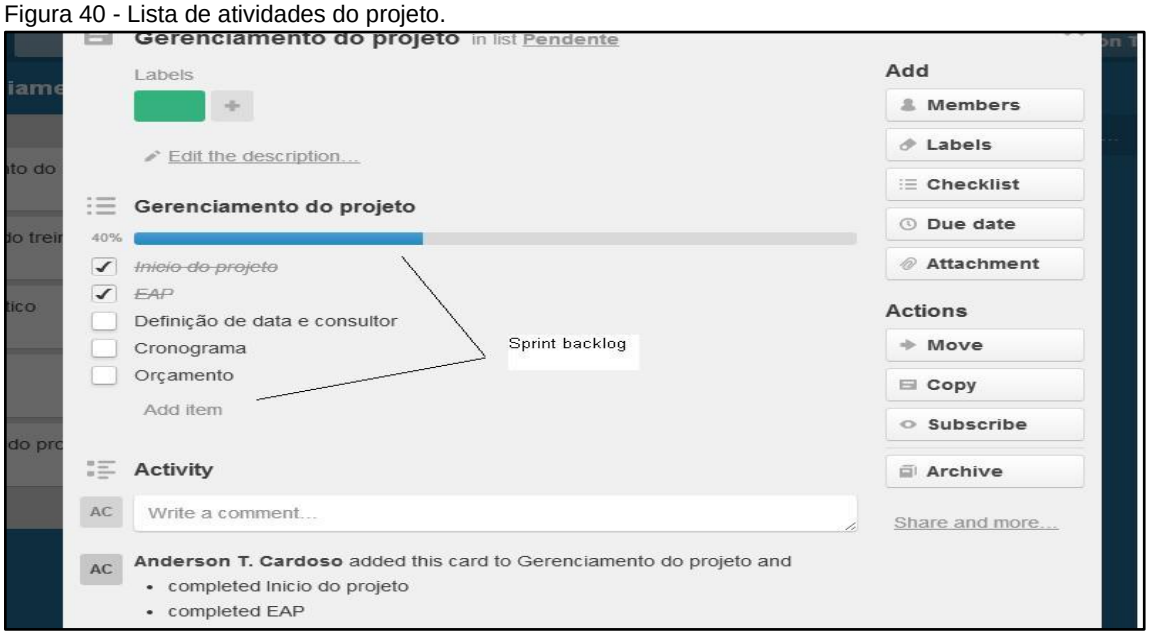
Produto backlog: a forma de representar o cadastro e visualização de um projeto no gerenciador Trello na figura 39, já diferencia de alguns outros gerenciadores, possível de observar cada backlog.

Figura 39 - Lista de tarefas do projeto.

Fonte: Adaptado de Trello (2014).

Spring backlog: Spring backlog: a lista de atividades do projeto pode ser configurada por status de porcentagem, ou seja, após o cadastro de todas as springs backlogs, conforme o desenvolvimento avança o gerente do projeto pode marcar as atividades já concluídas, e entre outras configurações de adicionar datas, membros, mover e anexar arquivos.

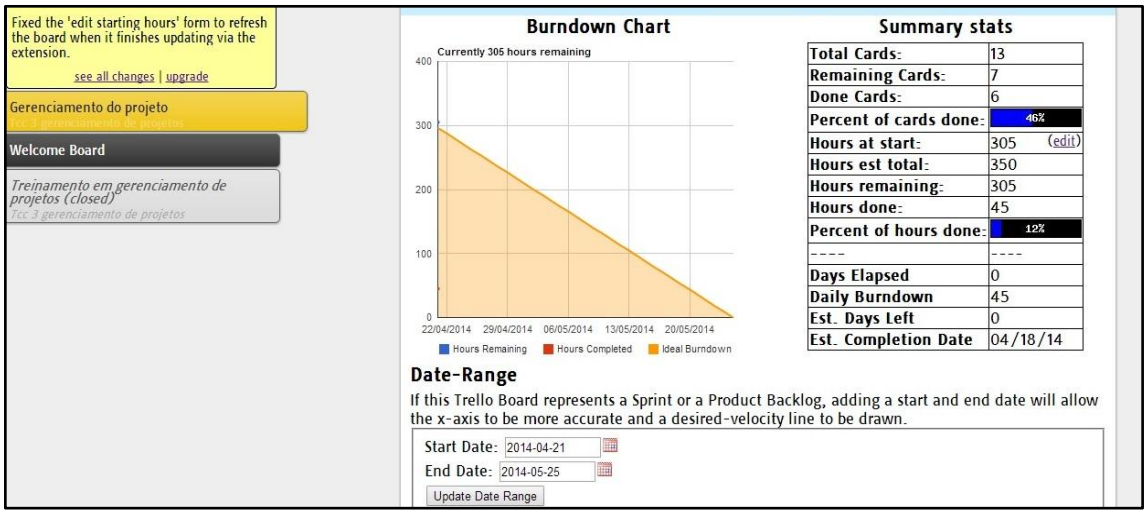
Figura 40 - Lista de atividades do projeto.



Fonte: Adaptado de Trello (2014).

Burndown: exibir o gráfico de gantt no Trello como mostra na figura 41, possibilita configurar as datas gerais do projeto, entre elas o começo e fim do gerenciamento de um projeto, e logo ao lado direito pode-se visualizar a estatistica geral do projeto como: total, restante e cards concluídos.

Figura 41 - Gráfico de gantt.



Fonte: Adaptado de Trello (2014).

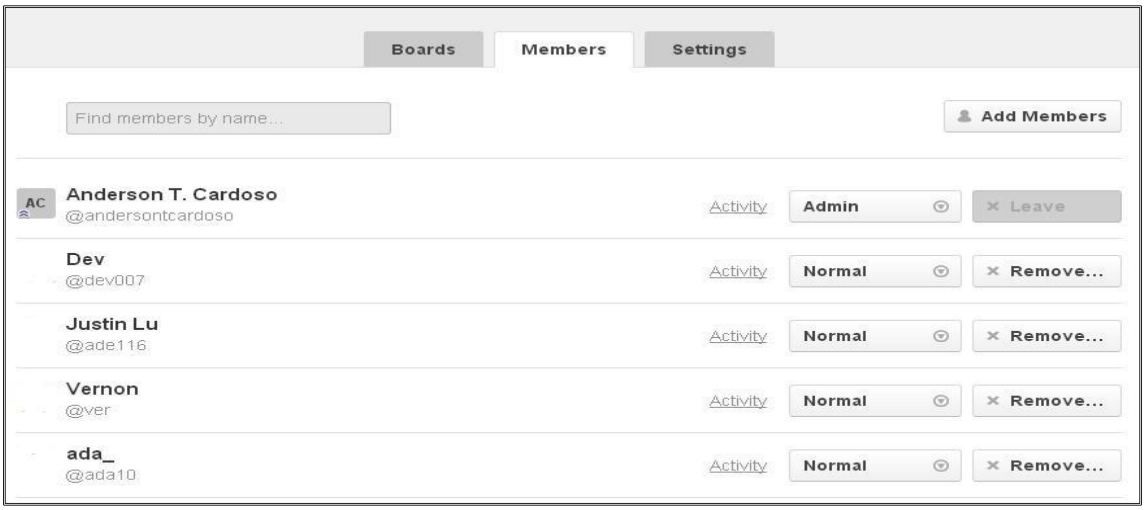
Product Owner: na opção de cadastro de product Owner, pode-se editar perfil e adicionar um endereço de um site pessoal, ou seja, possível de direcionar esse site para o projeto, e também inserir uma foto.

Figura 42 - Product Owner.

Fonte: Adaptado de Trello (2014).

Equipe: na opção membros da equipe que fazem parte do projeto, o mesmo deve estar configurado de forma publica a organização, ou seja, pois ao adicionar pessoas a visualização tem que ser para todos, só depois então organizar as permissões de acesso do projeto.

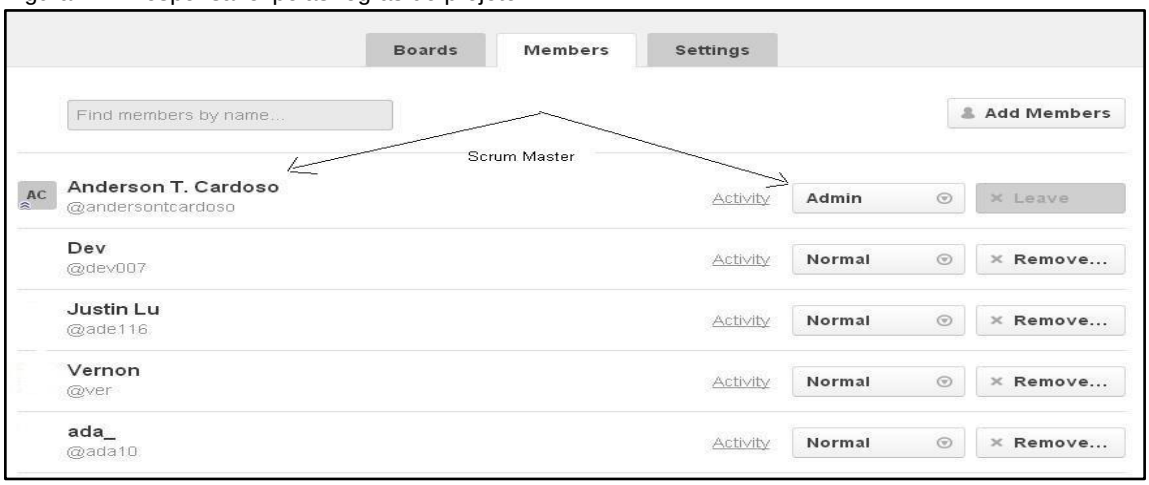
Figura 43 - Responsáveis pelo projeto.



Fonte: Adaptado de Trello (2014).

Scrum Master: o líder da equipe aparece da mesma forma como os outros do projeto, só que está de administrador e responsável geral das permissões.

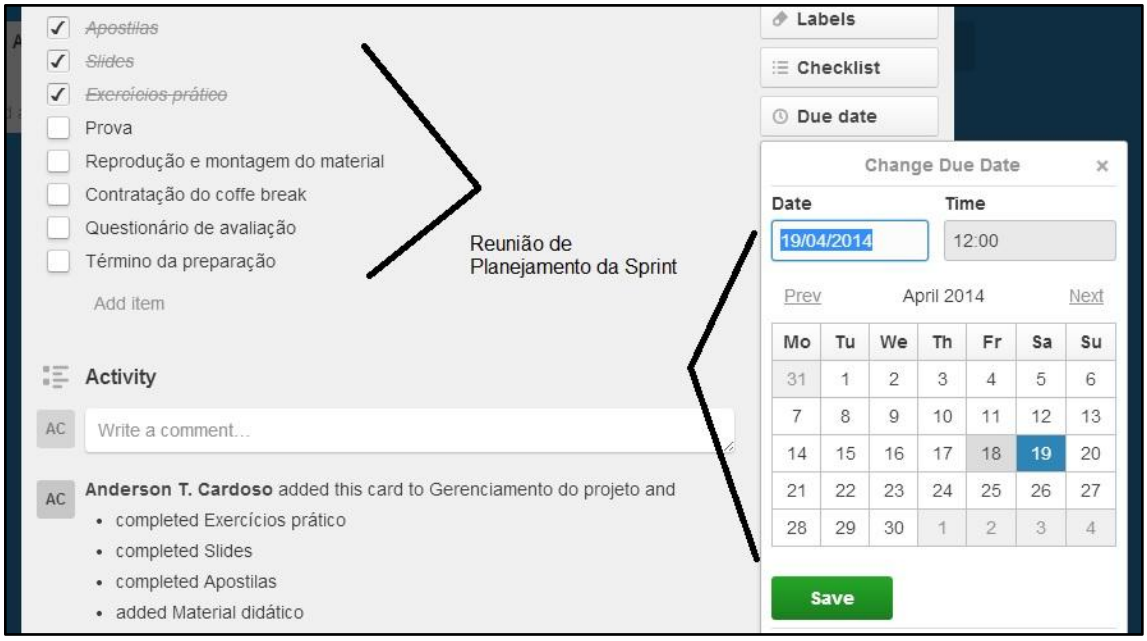
Figura 44 - Responsável pelas regras do projeto.



Fonte: Adaptado de Trello (2014).

Reunião de planejamento sprint: Reunião de planejamento sprint: no ambiente de planejamento sprint do Trello o funcionamento se dá pela inserção das atividades e com um cronograma de datas pelo calendário mensal, conforme figura 45.

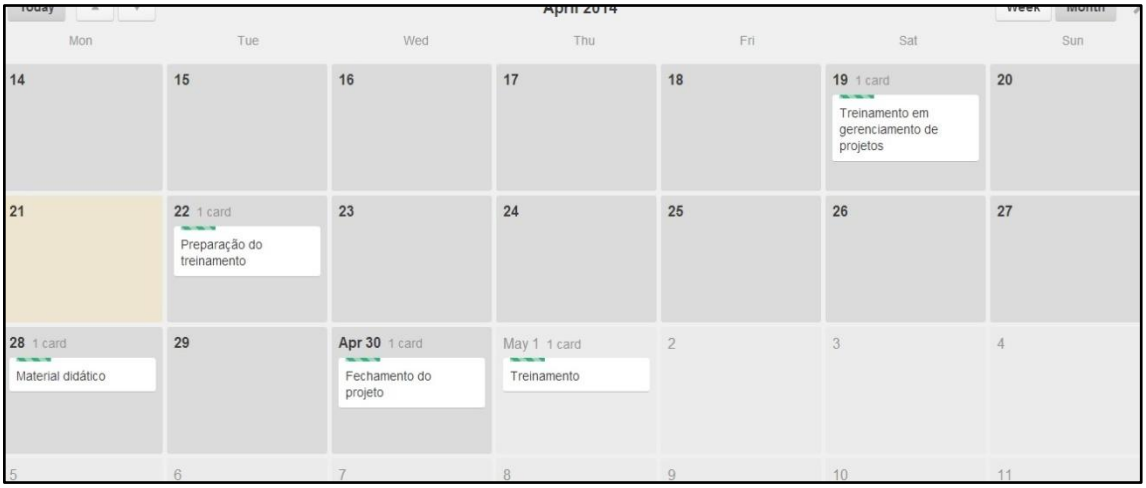
Figura 45 - Planejamento Sprint Trello.



Fonte: Adaptado de Trello (2014).

Reunião de revisão sprint: pela estrutura de calendário que o Trello tem em sua ferramenta de gerenciamento de projetos, possibilita acompanhar o andamento mensal das tarefas e atividades.

Figura 46 - Revisão Sprint na ferramenta Trello.



Fonte: Adaptado de Trello (2014).

Retrospectiva sprint: na visão geral do projeto onde as tarefas são concluídas, gera um histórico semanal, com isso a equipe pode voltar a ver os trabalhos realizados nas semanas anteriores.

Figura 47 - Visão geral Sprint Trello.

Fonte: Adaptado de Trello (2014).

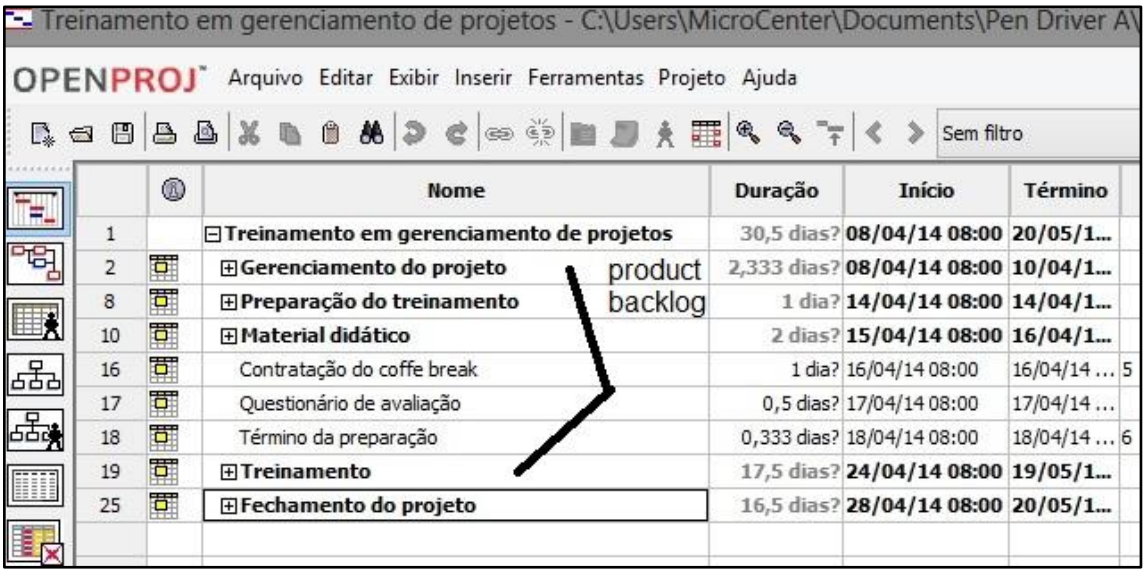
6.4 FERRAMENTA DE GERENCIAMENTO DE PROJETOS OPENPROJECT

A ferramenta OpenProject foi desenvolvida na linguagem Java pela Projity em 2007 e roda em vários sistemas operacionais, atualmente está na versão 1.4 onde este agrega funcionalidades do tipo: gráfico de Gantt, Pert, estrutura analítica de recursos, relatórios de tarefas e estrutura analítica de projetos.

6.4.1 Verificação da metodologia Scrum no gerenciador OpenProject

Produto backlog: de forma acessível às tarefas backlogs da ferramenta OpenProject já aparece a duração e inicio e fim das tarefas e atividades, ao fazer modificações nessas datas o usuário altera apenas a data da tarefa especifica, ou seja, como mostra na figura 48, alterar na faixa de cada tarefa.

Figura 48 - Product Backlog OpenProject.

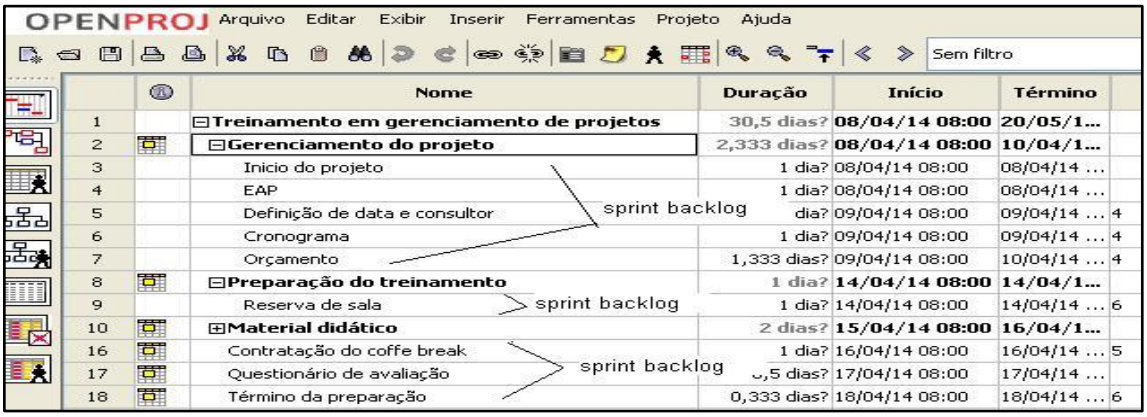


	Nome	Duração	Início	Término
1	Treinamento em gerenciamento de projetos	30,5 dias?	08/04/14 08:00	20/05/1...
2	Gerenciamento do projeto	2,333 dias?	08/04/14 08:00	10/04/1...
8	Preparação do treinamento	1 dia?	14/04/14 08:00	14/04/1...
10	Material didático	2 dias?	15/04/14 08:00	16/04/1...
16	Contratação do coffe break	1 dia?	16/04/14 08:00	16/04/14 ... 5
17	Questionário de avaliação	0,5 dias?	17/04/14 08:00	17/04/14 ...
18	Término da preparação	0,333 dias?	18/04/14 08:00	18/04/14 ... 6
19	Treinamento	17,5 dias?	24/04/14 08:00	19/05/1...
25	Fechamento do projeto	16,5 dias?	28/04/14 08:00	20/05/1...

Fonte: Adaptado de OpenProject (2014).

Spring backlog: a lista de atividades de cada tarefa apresenta uma configuração similar aos product backlogs, ou seja, alterar as datas dentro da faixa de cada atividade.

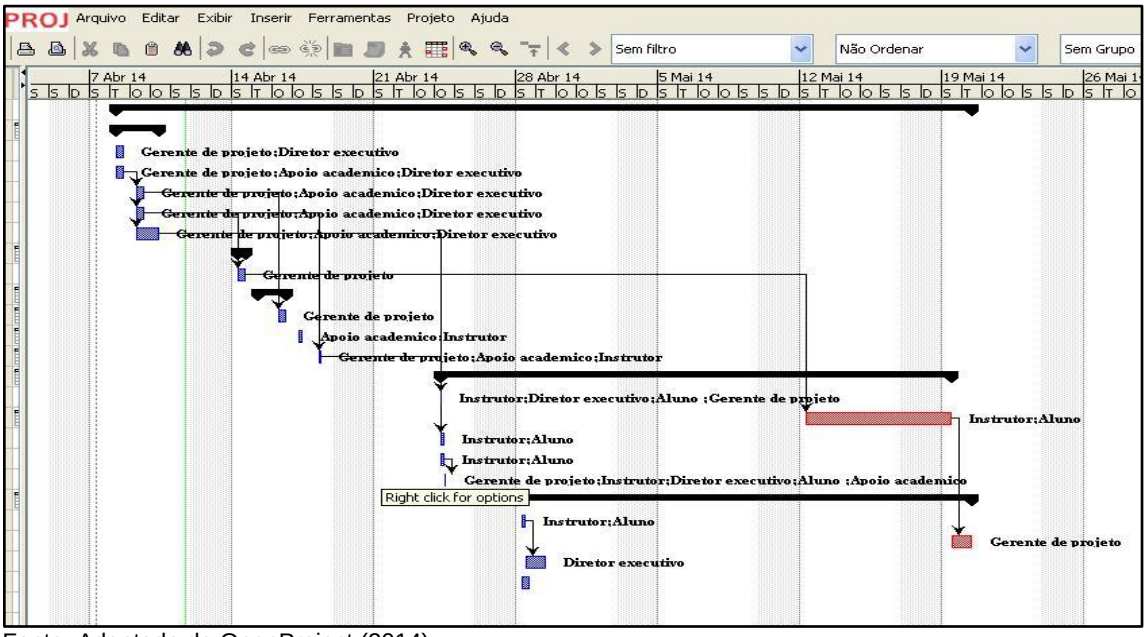
Figura 49 - Lista de atividades.



Fonte: Adaptado de OpenProject (2014).

Burndown: na representação do gráfico de gantt na ferramenta Open Project, pode-se observar todas as tarefas e atividades do projeto, porém com essa visão também pode ser selecionado os filtros de recursos para gerar outros tipos de gráficos, ou seja, através das caixas que as setas apontam.

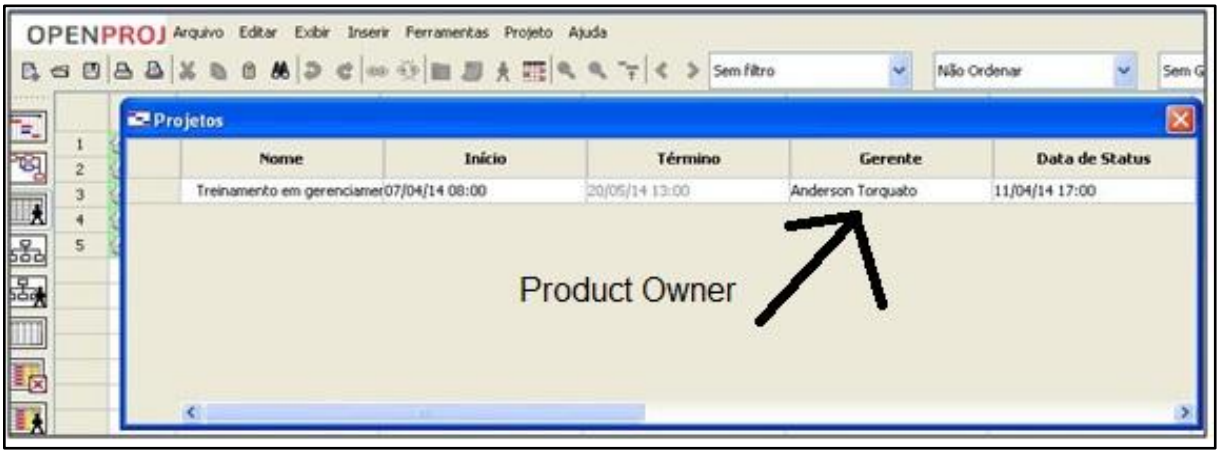
Figura 50 - Gráfico de gantt do projeto geral.



Fonte: Adaptado de OpenProject (2014).

Product Owner: o proprietário do projeto também aparece na lista de cadastro de recursos, onde ao selecionar o tipo de perfil do membro da equipe, a opção oferece como gerente de todo o projeto final.

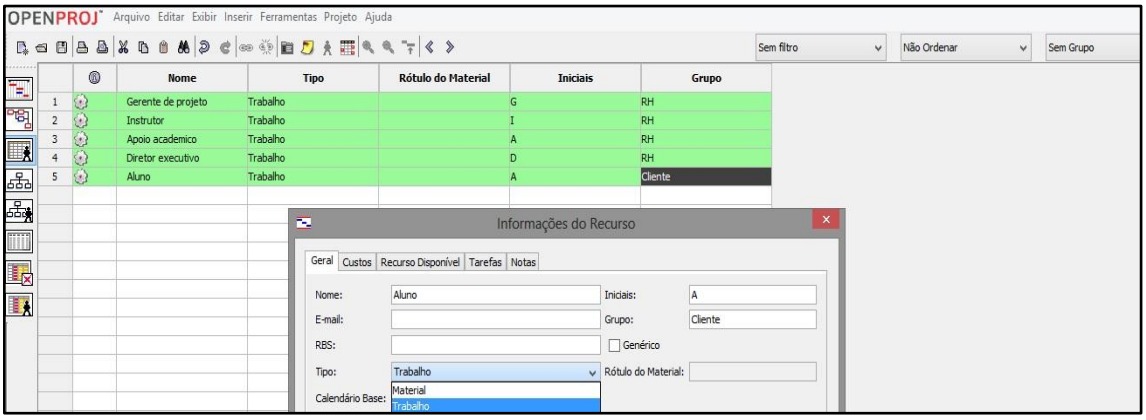
Figura 51 - Tela de cadastro Product Owner.



Fonte: Adaptado de OpenProject (2014).

Equipe: o cadastro de recursos humanos onde são os membros do projeto, ao dar um duplo clique em um dos nomes, aparece à caixa de informações do recurso, ou seja, nela podem-se alterar prioridades de cada membro da equipe.

Figura 52 - Cadastro de recursos humanos.



Fonte: Adaptado de OpenProject (2014).

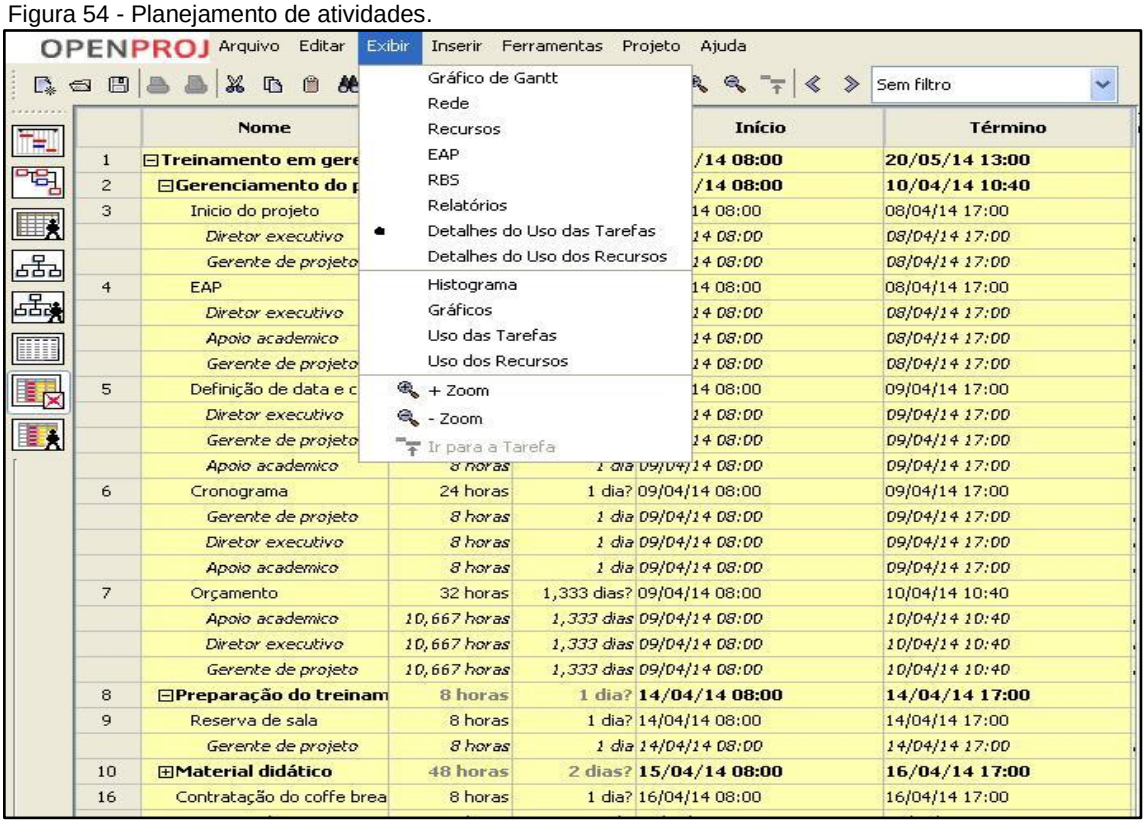
Scrum Master: o cadastro de recurso do líder do projeto se dá pela tela da prioridade de cada membro, onde nesse caso aparece como gerente, esse tipo de opção está vinculada com a configuração de Product Owner.

Figura 53 - Tela do Scrum Master.

Fonte: Adaptado de OpenProject (2014).

Reunião de planejamento sprint: no plano do projeto e na opção detalhes do uso das tarefas, o usuário pode acessar as datas de tarefas e atividades, bem como poder alterar se for a sua necessidade, na figura 54 mostra como fazer.

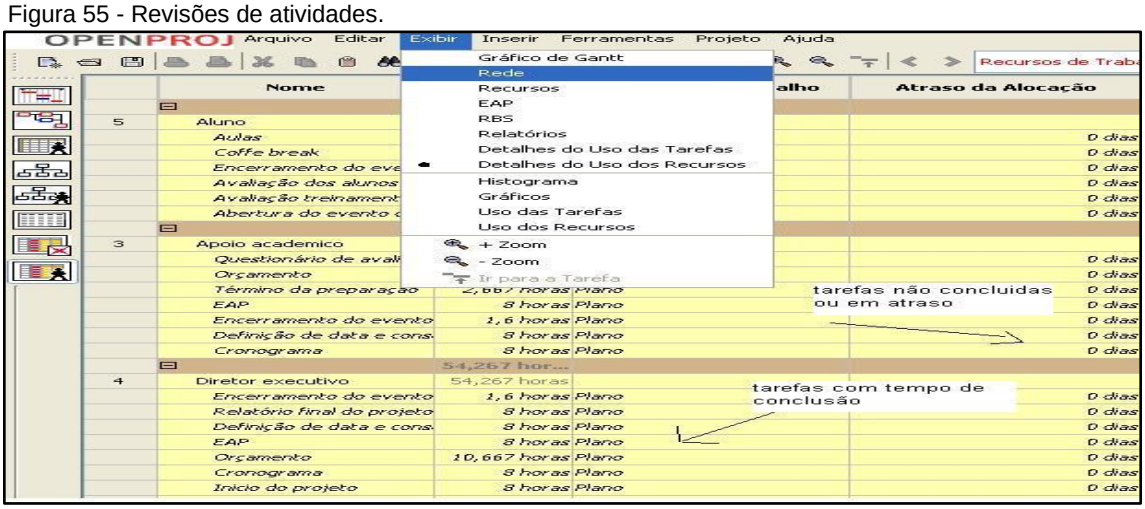
Figura 54 - Planejamento de atividades.



Fonte: Adaptado de OpenProject (2014).

Reunião de revisão sprint: na opção exibir e detalhes do uso dos recursos, o usurário pode acompanhar tarefas em atraso, onde aparece em atraso da alocação, um tipo de ajuda da metodologia ágil Scrum implementada na ferramenta OpenProject.

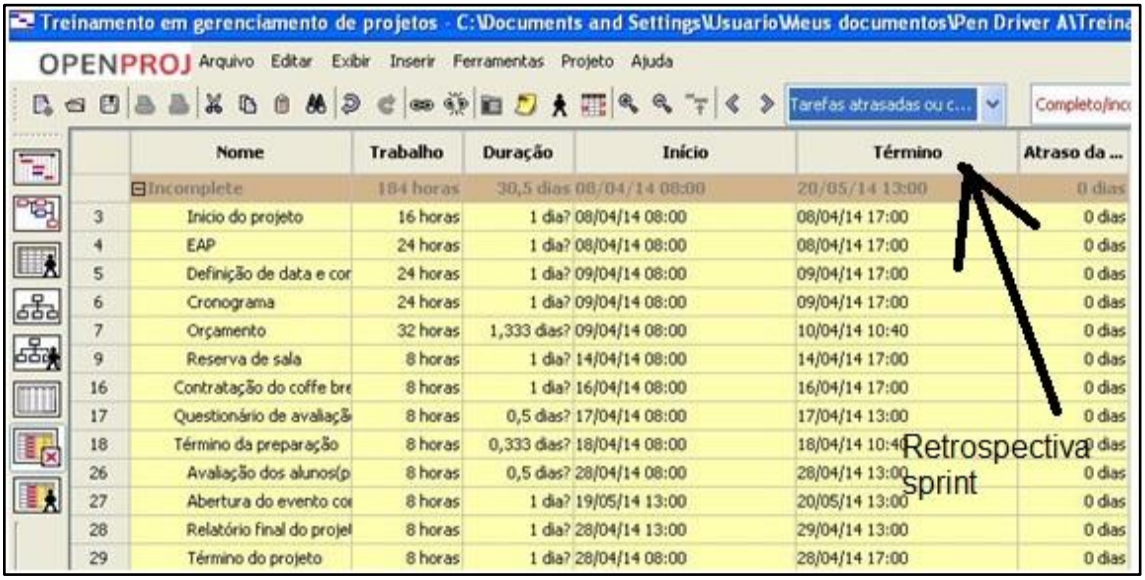
Figura 55 - Revisões de atividades.



Fonte: Adaptado de OpenProject (2014).

Retrospectiva sprint: esse recurso da ferramenta se faz selecionando a opção tarefas atrasadas como mostra na seta de configuração, bem como abaixo nas tarefas e atividades que estiver em atraso aparece os dias.

Figura 56 - Histórico das atividades.



Fonte: Adaptado de OpenProject (2014).

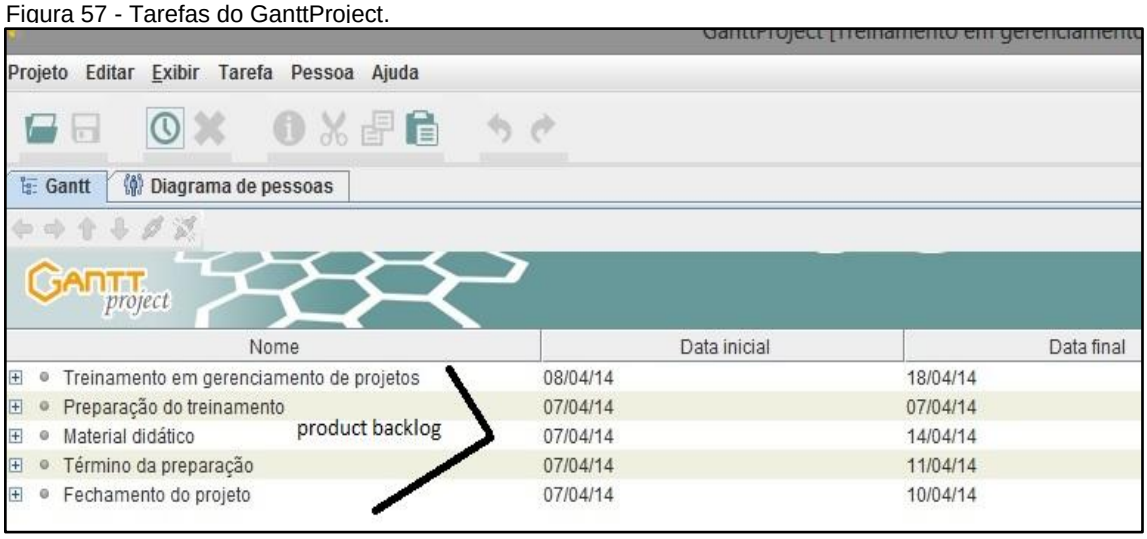
6.5 FERRAMENTA DE GERENCIAMENTO DE PROJETOS GANTTPROJECT

Sistema de gerenciamento de projetos open source, ou seja, de código aberto e também com seu uso sem licença adicional. Construído em linguagem Java e roda em vários sistemas operacionais entre eles: Windows, Mac Os, Linux e entre outros. Com isso seu projeto pode ser desenvolvido e instalado de forma confiável.

6.5.1 Verificação da metodologia Scrum no gerenciador GanttProject

Produto backlog: a lista de tarefas funciona de forma que seu projeto fique organizado, ou seja, nessa lista ao ir inserindo os itens de tarefas, o usuário também configura datas iniciais e finais de cada etapa da tarefa, bem como sendo possível alterar a medida da necessidade de andamento do desenvolvimento do projeto.

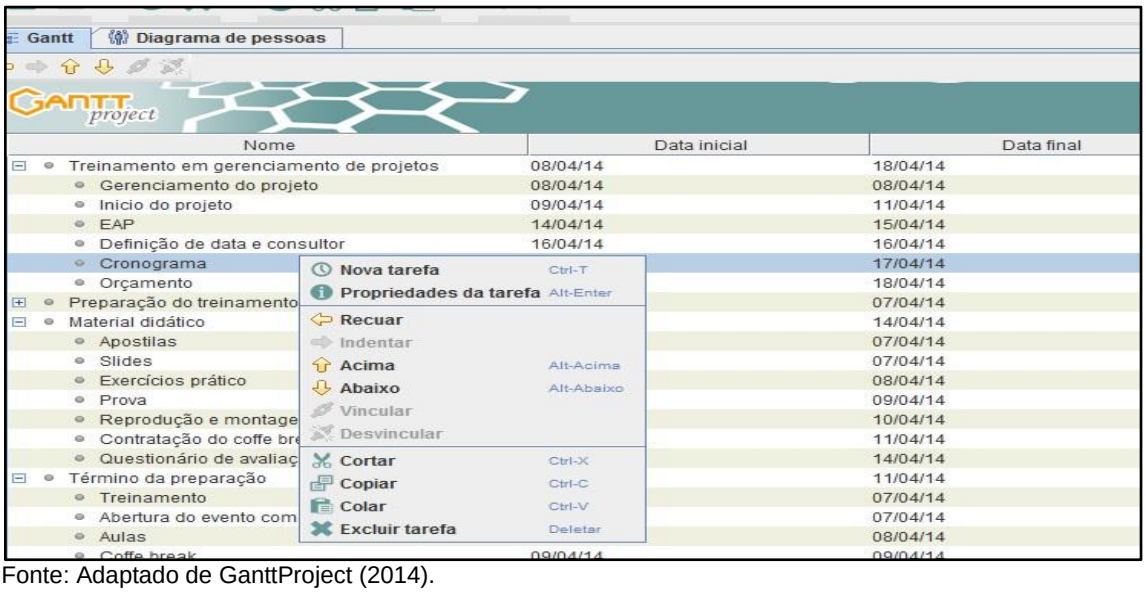
Figura 57 - Tarefas do GanttProject.



Fonte: Adaptado de GanttProject (2014).

Spring backlog: a estrutura de springs de cada product backlog oferece configuração personalizada, ou seja, com botão direito como mostra a figura 58, possibilita fazer ajustes de tarefas e na opção propriedades da tarefa e ligar uma atividade a outra.

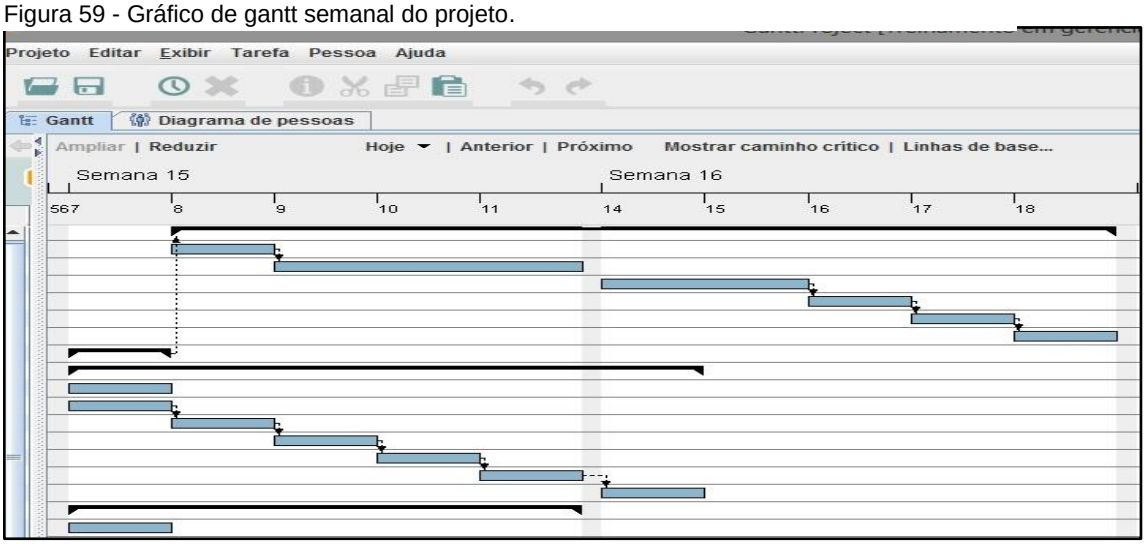
Figura 58 - Lista de atividades do gerenciador GanttProject.



Fonte: Adaptado de GanttProject (2014).

Burndown: representa o gráfico de gantt na ferramenta GanttProject, onde possibilita observar o projeto a nível semanal, onde as tarefas são divididas em cada etapa do product backlog, e uma vinculada (ligada) com a outra, como mostra a figura 59.

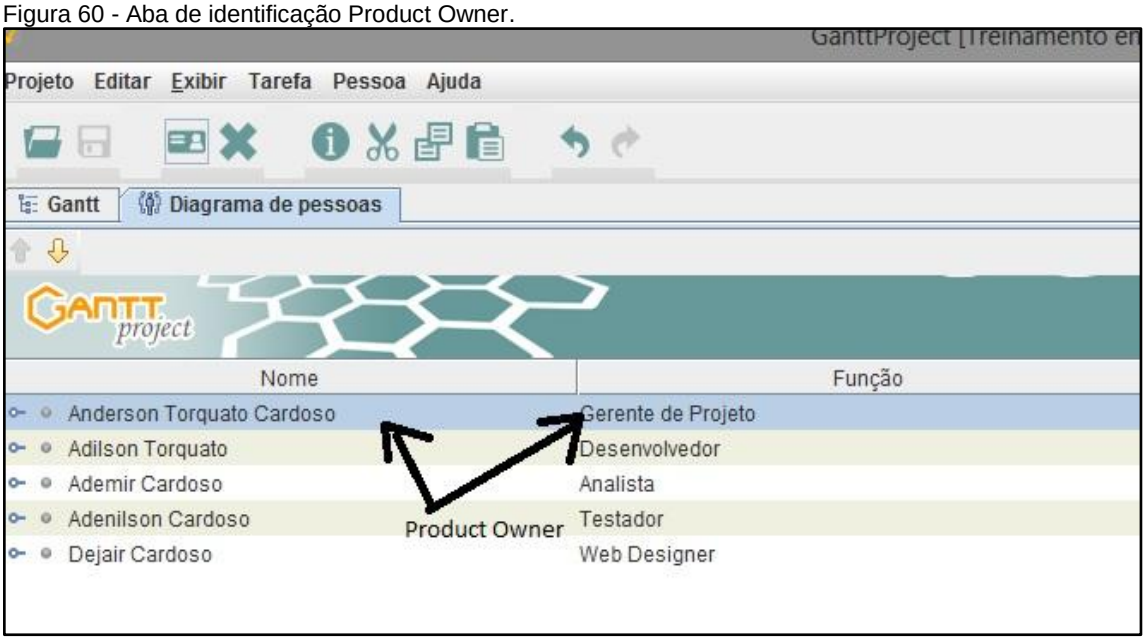
Figura 59 - Gráfico de gantt semanal do projeto.



Fonte: Adaptado de GanttProject (2014).

Product Owner: no cadastro de recursos humanos, onde são os membros da equipe, o dono do projeto aparece na função de gerente de projeto, existe uma pré-definição das funções de cada membro da equipe. Essa estrutura está na aba de diagrama de pessoas, a ferramenta gerencia seus recursos em abas.

Figura 60 - Aba de identificação Product Owner.



Fonte: Adaptado de GanttProject (2014).

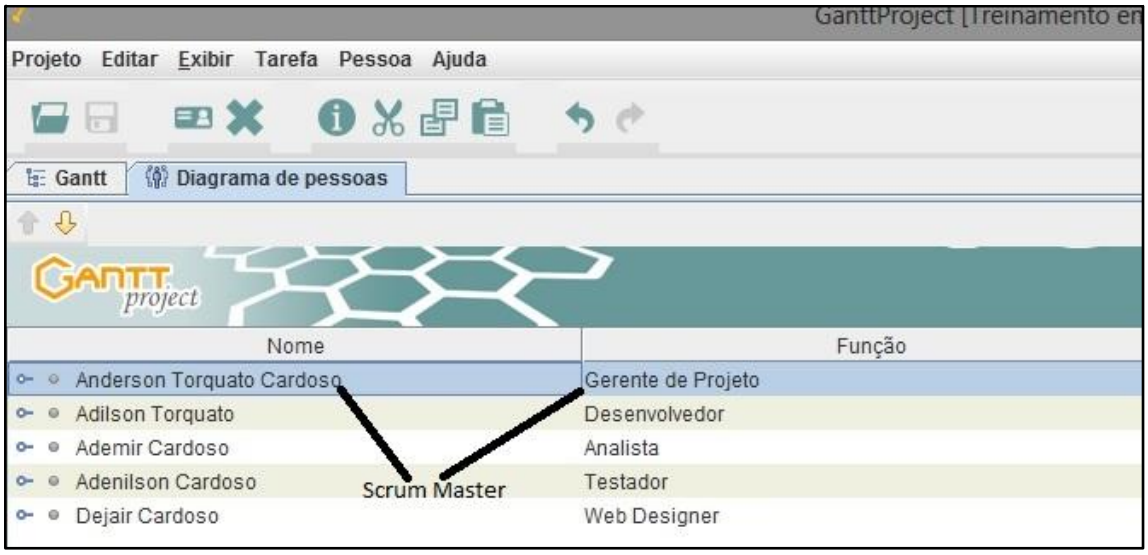
Equipe: os membros da equipe são adicionados ao projeto, cada um com sua devida função e vinculado em uma tarefa e atividades, esse cadastro distribui o desenvolvimento de todo o projeto de forma hierarquia.

Figura 61 - Membros da equipe e atividades.

Fonte: Adaptado de GanttProject (2014).

Scrum Master: na aba de diagrama de pessoas e na estrutura hierarquia de cada membro, se atribui a função do líder do projeto, ou seja, com Scrum Master na ferramenta GanttProject configura-se gerente de projeto.

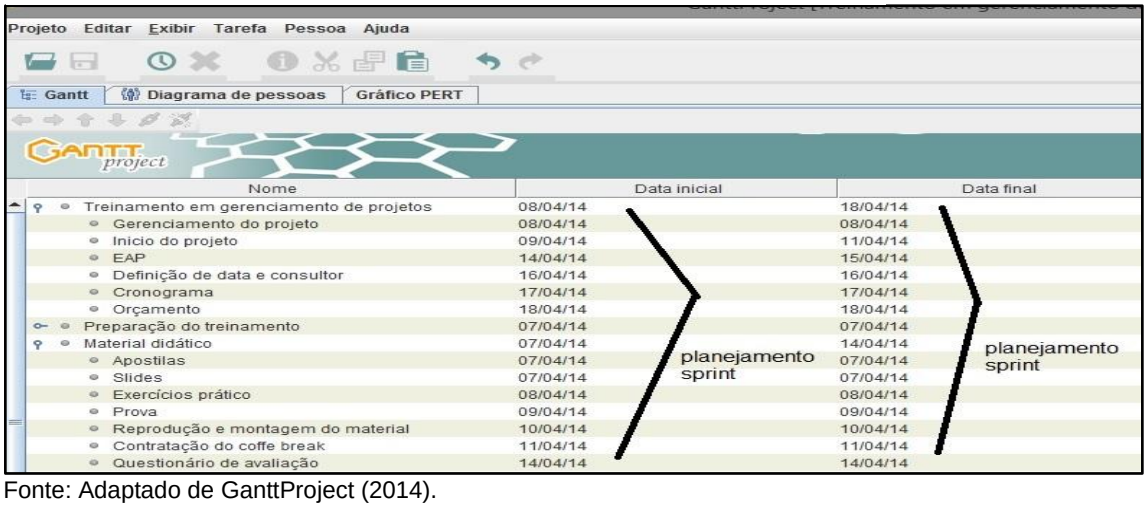
Figura 62 - Scrum Master do GanttProject.



Fonte: Adaptado de GanttProject (2014).

Reunião de planejamento sprint: planejar os sprint começa pela data inicial e final conforme se vê na figura 63, e em cada tarefa vincula-se uma atividade, onde essa organização pode-se identificar o tempo e a comunicação da equipe, e serve de preparação para os sprint backlog.

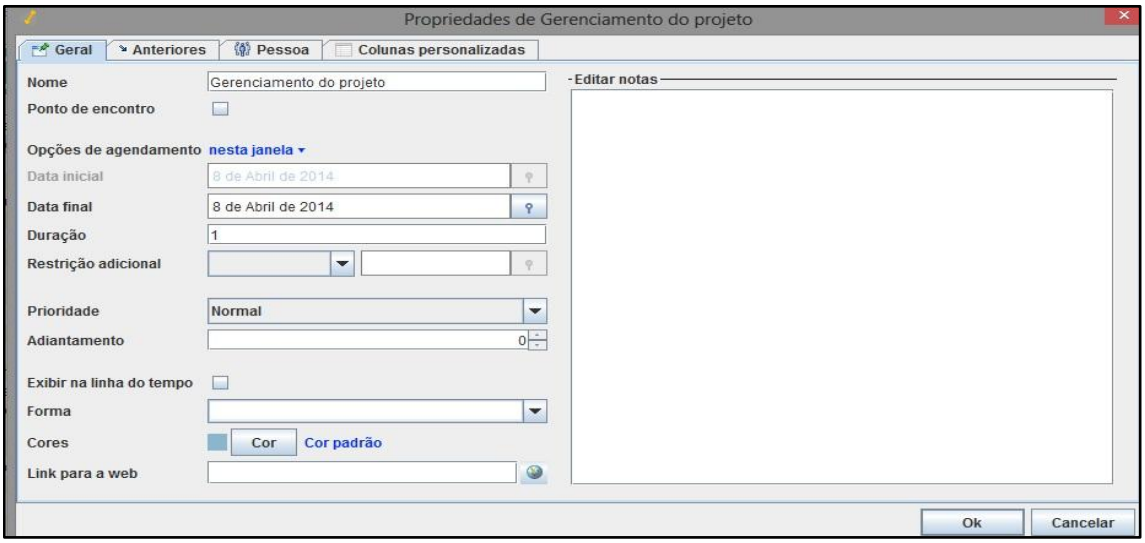
Figura 63 - Planejamento das atividades.



Fonte: Adaptado de GanttProject (2014).

Reunião de revisão sprint: na de propriedades de gerenciamento do projeto, o usuário pode rever os sprints passados, como apresenta na figura 64, preenche os campos e na aba anteriores visualiza-se o que for da necessidade.

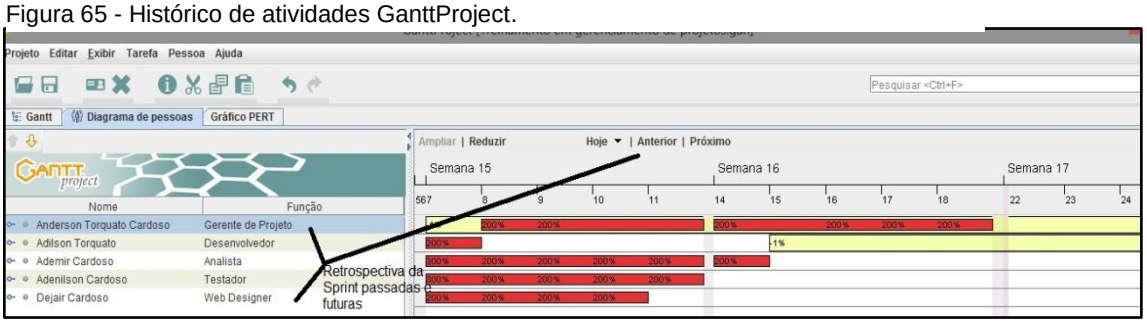
Figura 64 - Revisar atividades no GanttProject.



Fonte: Adaptado de GanttProject (2014).

Retrospectiva sprint: com a opção de diagrama de pessoas, visualizarem as tarefas passadas por datas e membros da equipe, o que foi concluído e as próximas tarefas a ser feitas, o que estiver em vermelho corresponde a sobre carga da pessoa e entre outras características.

Figura 65 - Histórico de atividades GanttProject.



Fonte: Adaptado de GanttProject (2014).

7 RESULTADOS OBTIDOS DAS FERRAMENTAS DE GERENCIAMENTO DE PROJETOS COM MÉTODO ÁGIL SCRUM: ARTIA, REDMINE, TRELLO, OPENPROJECT, GANTTPROJECT

Após as ferramentas serem analisadas pela metodologia ágil SCRUM, esse capítulo descreve os resultados obtidos pela utilização, teste e análise feita nas ferramentas. Cada tópico corresponde a uma pesquisa realizada no mercado de software Brasileiro onde tratam o uso dos métodos ágeis no Brasil, esses requisitos foram analisados nas ferramentas: Artia, Redmine, Trello, OpenProject e GanttProject. A pesquisa foi pelos 10 anos de desenvolvimento de projetos de software com métodos ágeis pelo mundo e no Brasil também, ou seja, um grupo de pesquisadores da Universidade de São Paulo fez uma pesquisa para levantar o comportamento atual de adoção e adaptação da metodologia ágil nas empresas brasileiras.

7.1 PRODUCT BACKLOG

Nesse requisito cada ferramenta de gerenciamento de projetos desenvolve sua interface para cadastrar as funcionalidades de um projeto a ser produzido. Com as particularidades de ambas, o Artia cria uma forma de comunidades e organizam as atividades vinculadas às tarefas, no Redmine precisa selecionar e atribuir em cada tarefa, o Trello funciona diferenciado das duas mencionadas acima, ao inserir tarefas mostra um aspecto em status, onde indica o total das atividades, já na gerencia do OpenProject e GanttProject possibilita marcar as datas iniciais e finais das tarefas e atividades, que nas outras citadas isso não ocorre.

7.2 SPRING BACKLOG

Ao inserir as atividades o Artia mostra ao lado uma descrição do que fazer e opções de edição, exclusão e datas de prazo. Já no Redmine exibe a situação de andamento, prioridades e qual membro estão atribuídos a atividade. Com o Trello esse Spring você marca o que já foi concluído no projeto em forma de porcentagem, com o OpenProject estas opções também são possíveis de fazer porém, o ambiente

limita algumas propriedades entre elas situação e prioridades de andamento como na Redmine. E na ferramenta GanttProject não possui status de acompanhamento como no Trello, mais possibilita personalizar de maneira direta os membros da equipe nesse Spring.

7.3 BURNDOWN (GRÁFICO)

Esse gráfico no Artia aparece ao selecionar as atividades, o que no Redmine isso não ocorre, ou seja, só mostra um gráfico geral de todo o projeto. No gerenciador Trello para gerar um Burndown, o usuário deve instalar e adicionar as extensões do navegador, um plugin do tipo Scrum para Trello, onde permite configurar por data de inicio e fim o status do projeto. Com o OpenProject em ambiente desktop esse gráfico pode ser gerado por meio de filtragem de recursos e não por datas, e atividades como no Artia e Trello, essas filtros podem ser mostrados por tarefas, grupos e ordenação de nomes. Na estrutura do GanttProject esse Burndown gera por atividades semanais, ou seja, exibe de forma limitada pelas tarefas divididas no product backlog e vinculadas umas nas outras.

7.4 PRODUCT OWNER

Nesse item na metodologia ágil Scrum está na ferramenta Artia na forma de gerenciar pessoas, pois no projeto indica ser um líder que também pode convidar usuários. No Redmine pode-se apenas visualizar isso em visão geral que por sua vez configura-se apenas uma vez, ou seja, editar e fazer alterações estão limitados. Com o Trello essa opção possibilita editar, adicionar foto e anexar um link de uma página pessoal desse product Owner, onde esse endereço de site pode ser o da empresa que está solicitando o serviço de produção do software. Essa configuração no OpenProject está em recursos humanos, onde aparece de gerente do projeto, não tem a liberdade de fazer configurações personalizadas como no Artia e Trello. Já o GanttProject funciona semelhante ao OpenProject, onde também está em recursos humanos e cadastra-se como gerente de projeto, uma estrutura limitada para um Product Owner, sendo ele o proprietário do projeto.

7.5 EQUIPE

Com o Artia uma equipe consiste em comunidades ou pessoas associadas, que para cada tarefa do projeto possibilita atribuir uma comunidade e pessoas, com isso o desenvolvimento torna-se flexível no sentido de pessoas participantes ou convidada. No Redmine funciona na opção membros, está limitado em editar, excluir e dividir papéis, apesar de ser aberto a web não permite compartilhar em grupos de convidados ou associar pessoas para participar do projeto. Já no Trello a equipe está semelhante ao artia, onde o projeto consiste de forma publica para depois e fazer as associações de pessoas e estabelecer as permissões de acesso ao projeto. O OpenProject limita-se a opção de recursos humanos, só pode fazer atribuições de pessoas as tarefas aqueles já cadastrados no projeto, pois não permite convidar outros grupos e associar as tarefas com nas ferramentas mencionas como Artia e Trello. Com o GanttProject a equipe está em uma aba de diagrama de pessoas e conforme descrito no OpenProject, depois de adicionar os membros de desenvolvimento, não propicia fazer outro compartilhamento de equipe, como no Trello e Artia, limitando o acesso.

7.6 SCRUM MASTER

Esse requisito da metodologia Scrum no gerenciador Artia não foi implementado de maneira exclusiva, pois está na mesma opção dos membros da equipe que se cadastra como líder e Product Owner do projeto, ou seja, pelo método Scrum não estabelece líder mais pela ferramenta está Artia como líder. O Redmine também está implementado como o Artia, pois não possui uma opção exclusiva de manager, ou seja, configura-se em membros da equipe. No gerenciador Trello consiste no administrador do projeto, no menu members possibilita fazer a permissão (admim), mais não tem também um tipo de implementação para o método Scrum Master. O OpenProject o Scrum Master encontra-se em uma tela de recursos humanos onde representa o gerente, permite vincular a um projeto, mais também só se limita a isso. Já no GanttProject ocorre da mesma forma como as demais ferramentas mencionadas anteriormente, ou seja, gerente de projeto, apesar de ser usada em ambiente desktop, limita-se a recursos de uma ferramenta de

gerenciamento de projetos genérica, onde pode ser implementada diversas metodologias ágeis, e não apenas o método Scrum.

7.7 REUNIÕES DE PLANEJAMENTO SPRINT

Planejar atividades no Artia consiste em uma agenda de calendário semanal, onde cada membro da equipe aparece atribuído a uma atividade do projeto, esse requisito apesar de ser um dos itens do Scrum também funciona em outras metodologias ágeis. O Redmine apresenta em calendário semanal não difere muito do Artia, apenas mostra uma interface visual maior, com setas coloridas indicando o começo e fim das atividades. Com o Trello esse planejamento não está implementado de forma especifica como nas outras ferramentas mencionadas como Artia e Redmine, o gestor configura apenas a data de começo da atividade, ou seja, dentro de cada Product Backlog que são as tarefas mostra um calendário mensal, e atribui só uma data, a do inicio e não do fim, isso tem que ser feito de forma manual para cada tarefa, onde o começo de uma é o final de outra. Com o OpenProject funciona com no Artia e Redmine mais não com no Trello, esse planejamento pode ser filtrado por grupos, tarefas e ordenação, apesar de cada gerenciador possuir uma estrutura de projeto, cabe ao gestor de desenvolvimento optar pelo ambiente das equipes e tarefas a ser implementadas nesse plano de trabalho. E no GanttProject planeja-se de maneira similar ao Artia, Redmine e OpenProject, entretanto ao planejar as tarefas exibe apenas uma comunicação com as demais pessoas da equipe e atividades a ser realiza, limita se por estar em modo desktop e não em web.

7.8 REVISÕES SPRINT

Revisar atividades com o Artia está por status, que são pendências da situação da atividade, em dias de atraso ou responsáveis atribuídos a tarefas e atividades do projeto, exibe em formato de gráfico de pizza. No Redmine funciona como bugs das tarefas, onde esse recurso pode ser compartilhado na ferramenta através de comunidades, mais para isso o projeto deve estar de maneira publica uma opção de resolver e revisar tarefas e atividades passadas. No Trello está implementado como calendário semanal de acordo com o mês planejado das

atividades, essa revisão apesar de estar na web o gestor não pode compartilhar com outros grupos, limitando o uso de pessoas ao desenvolvimento do projeto. Nas ferramentas mencionadas acima tipo Artia, Redmine e Trello não possibilita fazer filtragem de recursos passados, ou seja, no OpenProject a opções de selecionar atraso de alocações anteriores e tarefas em atraso, fazendo com que esse detalhe de uso dos recursos ajude o gestor administrar o projeto e atividades realizadas da equipe. A estrutura de revisar Sprints no GanttProject não possui gráfico ou qualquer outro tipo de status, para isso tem que preencher um formulário em propriedades de gerenciamento do projeto, selecionar campos dos itens desejados e na aba anteriores gera a revisão.

7.9 RETROSPECTIVA SPRINT

No Artia o modo de exibir um breve retrospecto do que ocorreu em cada tarefa, estabelece com que o gestor do projeto selecione a tarefa desejada, onde mostra as datas estimadas e reais das atividades concluídas. Com Redmine acontece de forma que as tarefas passadas fiquem em um link de bugs da visão geral do projeto, limitando fazer quais aspectos o gestor desejar para consultar Sprints passados. Já no Trello não possibilita fazer uma compreensão detalhada de retrospectiva Sprint, na mesma estrutura de planejamento desses Sprints exibe as atividades ocorridas, reduzindo um detalhamento do trabalho realizado. No OpenProject ocorre de maneira diferente do Artia, Redmine e Trello, pois na opção recursos o gestor seleciona a tarefa e atividades desejada através de filtros como: completo/incompleto, havendo alternativas de consultas por Sprints pretendido. Com GanttProject funciona de forma distinta de Artia, Redmine, Trello e OpenProject, ou seja, na aba diagrama de pessoas e no gráfico a direita na opção anterior exibe Sprints passados, ou seja, aparece de maneira limitada mas possibilita fazer um retrospecto de cada membro da equipe na tarefa e atividade desenvolvida.

A tabela 3 analisa a metodologia SCRUM com os itens citados descritos acima.

Tabela 3 - Verificar adequação de metodologias nas ferramentas

Verificar metodologias nas ferramentas						
Requisitos de comparação	Ferramentas / requisitos	Artia	Redmine	Trello	OpenProject	GanttProject
Principais Papéis	Product Owner	A	A	A	A	A
	Development Team	A	A	A	A	A
	Scrum Master	A	A	A	A	A
Artefatos	Product Backlog	A	A	A	A	A
	Sprint Backlog	A	A	A	A	A
	Burndown Chart	A	A	A	A	A
Reuniões	Sprint Planning Meeting	A	A	A	A	A
	Sprint Review	A	A	A	A	A
	Sprint Retrospective	A	A	A	A	A

Legenda.: A – Atende | NA – Não Atende | AP – Atende Parcialmente

Os itens descritos a seguir são relacionados a uma pesquisa realizada na universidade de São Paulo no Brasil, sobre os métodos ágeis nas empresas Brasileiras.

7.10 MELHORAR A VISIBILIDADES DO PROJETO

Com o gerenciamento do Artia a visualização do projeto e recursos para organizar tarefas, faz com que favoreça a comunicação da equipe. No Redmine isso acontece de maneira compartilhada em pastas descrevendo o que fazer e atribuindo aos membros da equipe. O Trello gerencia diferenciado do Artia e Redmine, ou seja, sua interface possui recursos flexíveis de arrastar tarefas cadastradas e visualizar em porcentagem de atividades concluídas. Na gestão do OpenProject funciona limitado a aparência flexível, onde não possibilita acompanhar o projeto na web e acessar o andamento da equipe. E com o GanttProject atende mais com limitações para web, por atuar em modo desktop está como o OpenProject.

7.11 Melhorar a Manutenibilidade/ extensibilidade de software

Fazer manutenção no artia melhora a metodologia do projeto, entre elas: o crescimento das tarefas implementadas. O Redmine não oferece isso de maneira organizada, pois ao compartilhar as tarefas acaba com possíveis ajustes e alterações recentes. Expandir e fazer manutenção no Trello ocorre de forma a

atender alterações por via web, onde o acesso permite acelerar a comunicação da equipe. Com o OpenProject apesar de ser limitado a web, oferece um suporte com que a equipe evolua o projeto sem muitos contratempos. E no GanttProject essa manutenção acontece como Artia, Trello e OpenProject, apesar de também ser desktop como OpenProject esse gerenciador possui recursos suficientes a atender a evolução do desenvolvimento do projeto.

7.12 Melhorar o alinhamento entre TI e negócio

Alinhar negócios no Artia com tecnologia da T.I ajuda a equipe e gestores a traçar decisões relevantes no projeto. Isso com o Redmine também acontece para atribuir recursos decisivos em softwares em desenvolvimentos entre eles: compartilhar tarefas e equipes via internet. O Trello também possibilita gerar relatórios automatizados e enviá-los por e-mail. Já no OpenProject funciona a atender projetos independente de organização e serviços. E com GanttProject ocorre o funcionamento como no OpenProject e também possibilita implementações em sistemas operacionais entre eles: Windows, Linux e Mac OS.

7.13 Simplificar a qualidade de software

Apresentar um resultado de melhor performance nos gerenciadores de projetos, para então desenvolver software de qualidade, ou seja, no artia possui uma estrutura para fragmentar e compartilhar com eficiência recursos ágeis. Com o Redmine essa performance também ocorre trazendo flexibilidade no gerenciamento e no software a ser produzido. No Trello a qualidade atua como no Redmine, flexível a gerenciar tarefas e atividades como arrastar e soltar, simplificando abrir outras telas de redirecionamento do andamento do software em desenvolvimento. O OpenProject possui particularidade diferentes do Artia, Redmine e Trello, mais atende com propriedades desktop, ou seja, recursos limitados na prática de gerenciar e desenvolver software. E no GanttProject funciona como no OpenProject, atendendo com recursos e características limitadas, mais trazendo resultados positivos no softwares desenvolvidos.

7.14 Melhorar a capacidade de gerenciar mudanças e prioridades

No Artia essas mudanças estão na hierarquia de cada Product Backlog, onde possui prioridades a atender o andamento do projeto. Com Redmine funciona de maneira diferenciada do Artia, as modificações e as prioridades são compartilhadas com grupos de equipes, fazendo com isso melhores resultados e agilizando o desenvolvimento do software. O Trello também atua de forma semelhante ao Redmine mais diferente do Artia, pois as alterações prioritárias são compartilhadas e a equipe trabalha via web, com mensagens de e-mail e conversa por meio de chat do gerenciador. No OpenProject atende com características próprias, onde os recursos são desenvolvidos para desenvolver software local e as modificações são realizadas na opção recursos humanos e na propriedade de gerenciamento do projeto. E no gerenciador GanttProject atua também como o OpenProject, onde as alterações são realizadas no cadastro as equipes e na hierarquia dos Sprints Backlogs, mas atende de forma eficiente.

7.15 O método ágil que o mercado segue

Esse requisito da pesquisa que foi realizada pelo departamento de Ciência da computação IME-USP 2012, onde estabelece o método Scrum sendo usado com frequência pelo mercado de desenvolvimento de software.

7.15.1 Scrum

A análise comparativa realizada nesse trabalho de conclusão de curso (TCC) utilizada foi à metodologia Scrum, onde todas as ferramentas de gerenciamento de projeto sendo elas: Artia, Redmine, Trello, OpenProject e GanttProject, atendendo de forma para trazer um desempenho satisfatório em gerenciar e controlar o andamento de projetos e equipes, mais mesmo oferecendo esses recursos ágeis, cada uma trabalha com suas particularidades, algumas sendo mais abertas funcionando na web e outras limitadas a gerenciar projetos de forma desktop.

7.16 Planejamento antecipado

O Artia atende esse requisito com suporte a recursos suficientes no cadastro dos Product Backlog e Sprint Backlog. No Redmine não possui esse recurso de forma definida, ou seja, falta antecipar algumas tarefas, onde essas são gerenciadas apenas na prática de desenvolvimento, pois são compartilhados. Com a utilização do Trello essa antecipação de planejar tarefas e atividades atende de forma eficiente, podendo alterar permissões desejadas conforme a metodologia aplicada. Já no OpenProject atende com as ferramentas mencionadas acima como o Artia e Trello, apenas limitando recursos por ser usada em modo desktop. E na gerencia do GanttProject funciona com uma estrutura apesar de não estar na web, o gerenciamento de equipe possui propriedades de personalizar perfil e help de ajuda conforme for desejado.

7.17 Controle gerencial

As ferramentas como Artia, Redmine, Trello, OpenProject atende de forma satisfatória, onde cada uma fornece recursos de controle particular como agendamentos personalizados e relatórios via mensagens de email, já no GanttProject atende parcial esse requisito, atua limitado pela estrutura do desenvolvimento da ferramenta, pois não atende de forma externa via navegadores, com isso os gestores não possui acesso em ambientes remotos.

7.18 Previsibilidade

No Redmine, Artia e Trello essa falta de prevenção não ocorre, pois cada gerenciador trabalha com agendas e calendários e datas previstas, com isso organizar o projeto de forma preventiva em prazos, faz com que possibilita um auxilio para os membros da equipe em desenvolvimento e Product Owner que utilizaram o projeto logo após seu encerramento. Mais nas ferramentas OpenProject e GanttProject esse requisito está de forma a atender parcialmente, possui alguns filtros de recursos que em algumas vezes ajudam para prever tarefas e atividades completas e incompletas, grupos e entre outros, mais para fins de previsibilidades específicos não está implementado de forma definida.

7.19 Documentação

Documentar o que está realizado e o que realizar no Artia, Redmine e OpenProject atende com uma estrutura organizada e especifica, ou seja, cada uma oferece relatórios em formatos para documentar em .pdf, .xml, .xls, com isso possibilita transportar esses arquivos para lugares distintos. E nas ferramentas Trello e GanttProject esse auxilio não possui atendimento, impedindo fornecer por exemplo um resumo de parte do projeto em forma de arquivos desse tipo.

7.20 Gerenciamento de equipe distribuída

Gerenciar equipes compartilhadas e com uma metodologia ágil para definir o andamento e controle do projeto no Artia, Redmine e Trello possibilita uma produtividade mais avançada em relação ao OpenProject e GanttProject, pois com o mercado de software a cada ano atendendo uma demanda de 31,33%, esse requisito não pode deixar a desejar, por isso esse recurso em uma ferramenta de gerenciamento de software obriga a escolha ser pesquisada e estudada, especificando o projeto em desenvolvimento e a equipe.

7.21 Simplificação do processo de desenvolvimento

Esse processo no mercado de desenvolvimento melhorou em 37%, e na ferramenta Artia, Redmine, Trello, OpenProject e GanttProject, atende de maneira satisfatória para desenvolver com produtividade e organizada sem contratempos, pois a estrutura implementada nos gerenciadores trabalha para reduzir prazos, custos e fornece as necessidades dos Product Owner.

A tabela a seguir faz uma comparação dos itens da pesquisa realizada pela Universidade de São Paulo, pois com esses requisitos foi analisado nessas ferramentas de gerenciamento de projetos.

Tabela 4 - Estudo comparativo das ferramentas de gerenciamento de projetos

Estudo comparativo das ferramentas de gerenciamento de projetos						
Requisitos de comparação	Ferramentas / requisitos	Artia	Redmine	Trello	OpenProject	GanttProject
Razão mais importante para a adoção de métodos ágeis na equipe/ organização	Melhorar a visibilidades do projeto	A	A	A	AP	AP
	Melhorar a Manutenibilidade/ extensibilidade de software	A	AP	A	A	A
	Melhorar o alinhamento entre TI e negócio	A	A	A	A	A
	Simplificar a qualidade de software	AP	AP	A	AP	AP
	Melhorar a capacidade de gerenciar mudanças de prioridades	A	AP	A	A	A
O método ágil que o mercado segue	Scrum	A	A	A	A	A
Maiores preocupações da organização na adoção	Planejamento antecipado	A	AP	A	A	A
	Controle gerencial	A	A	A	A	AP
	Previsibilidade	A	A	A	NA	NA
	Documentação	A	A	AP	A	AP
Principais benefícios obtidos com a adoção de métodos ágeis	Gerenciamento de equipe distribuída	A	A	A	NA	NA
	Simplificação do processo de desenvolvimento	A	A	A	A	A

Legenda.: A – Atende | NA – Não Atende | AP – Atende Parcialmente

O que mais atendeu os requisitos foi o gerenciador Artia onze dos doze itens, com seu uso na web a agilidade prevaleceu. As ferramentas que atendeu parcialmente ficaram empatadas em duas, sendo o Artia e Redmine com um dos doze itens e a que não atendeu foi duas ferramentas, pois obtiveram empate também, sendo dois dos doze itens que foram o OpenProject e GanttProject, apesar dos resultados apresentarem pontos negativos em gerenciadores desktops como: OpenProject e GanttProject, essas gerenciadores possui uma estrutura satisfatória na metodologia ágil SCRUM, mas com características de seu uso ser restrito na web e não possuir uma gestão preventiva, ou seja, antecipar alguns planejamentos. E com relação às ferramentas de atendimento parcial também apresentaram de maneira razoável, pelo seu ambiente de organização e planejamento. E por fim o

Artia, sua estrutura não deixa a desejar possuindo um ambiente de excelente uso e acessível.

8 CONCLUSÃO

Ao analisar as cinco ferramentas de gerenciamento de projeto, para metodologia ágil, pode-se avaliar que o método Scrum aponta ser mais adequado ao mercado de software e atende a demanda de negócios automatizados. O método SCRUM escolhido pelo mercado após aplicado nas ferramentas atendeu de forma satisfatória nessa análise comparativa entre ferramentas livres para gerenciamento de projetos de software desenvolvidos com metodologias ágeis. Apesar das particularidades de implementação de cada gerenciador. Considerando ao final um resultado satisfatório para a escolha de uma organização ou de um gerente de projetos.

A pesquisa citada auxiliou na identificação das práticas ágeis visto que foram aplicadas na prática das empresas, logo as ferramentas analisadas atenderam os requisitos da metodologia ágil SCRUM, onde vários requisitos do método foram atendidos satisfatoriamente pelas ferramentas de gerenciamento de projetos dentre elas: Artia Redmine, Trello, OpenProject e GanttProject.

Logo após essa análise das ferramentas sobre o método ágil Scrum foi também analisado a pesquisa de métodos ágeis no Brasil, onde por sua vez teve bons resultados nas ferramentas escolhidas, ou seja, a tabela de comparação apresentou os seguintes resultados, o Artia atendeu os requisitos, onde apenas no item simplificar a qualidade de software ficou como atende parcialmente, que significa a gestão em alguns itens do sistema Artia ser burocrático, possuir dependências. No Redmine ocorre de forma como o Artia no quesito anterior e em outra como manutenção, mudanças em prioridades e planejamento antecipado, onde o sistema atende parcialmente, são subordinados a outros. O Trello atende a todos, só a documentação encontra-se fraca e atendendo parcialmente. Já no OpenProject ocorre como no Artia no requisito qualidade de software e em visão de projeto atendendo parcialmente, também em previsibilidade e equipes distribuídas ocorre o não atendimento, fazendo um ponto negativo na falta desses itens. E por ultimo GanttProject que após analisado aparece como no OpenProject nos mesmos itens, e no quesito documentação não atende, com a falta desse item trás um ponto negativo a ferramenta. Mas a pesquisa também aponta outros números interessantes como à região do Brasil que aplica métodos ágeis e pessoas que utilizam.

Um dos fatores predominantes na adoção dos métodos ágeis na pesquisa deste trabalho de conclusão de curso (TCC) foi o fato de oferecer material de estudo para profissionais da área de Gestão de projetos.

Este trabalho de pesquisa contribuiu para obter um conhecimento em ferramentas de gerenciamento de projetos com métodos ágeis, ou seja, fazer com que o gerente de projetos autônomo ou uma organização desenvolvedora de software, possam estudar e analisar esse material e atingir seus objetivos entre eles o nível de suporte que cada gerenciador de projeto pode se adequar ao projeto a realizar, ajustando conforme a necessidade do Product Owner.

Para trabalhos futuros podemos analisar:

a) Uma analise comparativa de outras metodologias ágeis como: XP, Kanban e outros;
b) Especificar uma ferramenta em gerenciamento de projeto com método ágil XP para uso web;
c) Estudo de aderência para comparação de ferramentas pagas;
d) Testar essa pesquisa em uma organização de pequeno porte um estudo de caso;

REFERÊNCIAS

ABREU, Antônio Cesar. **Enfoque no gerenciamento de projetos de software.** 2006. 187 f. Dissertação (Mestrado) - Departamento de Administração da Faculdade de Economia, Universidade de São Paulo, São Paulo, 2006.

ABREU, Roberto. **Sugestão e modelagem de prática de desenvolvimento ágil para aplicação em desenvolvimento distribuído:** OnshoreInsourcing. 2008. 92 f. Tese (Mestrado) - Departamento de Instituto de Física e Matemática Departamento de Informática, Universidade Federal De Pelotas, Pelotas, 2008.

AGILE MANIFESTO (Org.). **Princípios por trás do Manifesto Ágil.** Disponível em: <http://agilemanifesto.org/principles.html>. Acesso em: 12 out. 2013.

AGUIAR, Giancarlo De França. **COMPREENDENDO O KANBAN:** UM ENSINO INTERATIVO ILUSTRADO. Curitiba - PR: Unicenp/centro Universitário Positivo, 2007. 14 p.

ALBERT, Alan. **A importância do software.** São Paulo: Makron Books do Brasil, 2009. 530 p.

AMBLER, Brian. **Modelos ágeis de processos.** 6.ed.Brasil: Mcgraw-hill Interamericana, 2007. 752 p.

ANDERSON, Cloves. **Determinação de prazos.** São Paulo: Makron Books, 2003. 165 p.

ARAÚJO, Eduardo de Miranda. **Gerenciando Projetos de Software com RUP e PMBOK.** 2008. 110 f. Dissertação (Mestrado) - Departamento de Pós-graduação Lato Sensu em Tecnologia da Informação, Unibratec - União Dos Institutos Brasileiros de Tecnologia, Recife – PE, 2008.

ARAÚJO, Vitor de Barros. **Status do uso de metodologias ágeis na indústria de software Brasileira.** 2009. 60 f. Tese (Trabalho de Conclusão de Curso) - Departamento de Graduação Em Ciência Da Computação, Universidade Federal De Pernambuco, Pernambuco - PE, 2009.

AVELINO, Luciana Perretti; ALMEIDA, Marcos Ozorio de. Método de seleção de Ferramentas livres de Cronograma e EAP para uso Comparativo: Infraestrutura Tecnológica. In: CONSERPRO, 05,2009.Passos Fundo. **Anais... .** PUC - RS: Universidade de Passo Fundo, 2009. p. 01 - 21.

BARBOSA, Libardi. **Administração de projetos:** Estimativas. São Paulo: Makron Book do Brasil, 2010.

BARBOSA, Vitor de Barros Araújo. **Metodologias ágeis em software.**2010. 80 f. Tcc (Trabalho de Conclusão de Curso) - Departamento de Graduação em Ciência Da Computação, Universidade Federal De São Paulo, São Paulo, 2010.

BARROS, Otávio Rodolfo. **RUP– Rational Unified Process.** 2007. 90 f.

Dissertação (Mestrado) - Departamento de Sistemas de Informação, Universidade do Contestado - Unc, Mafra, 2007.

BASSI, Benjamim. **Software.** 6.ed.Brasil:Mcgraw-hillInteramericana, 2010. 752 p.

BECK, Tom. **Software:** o processo e seu gerenciamento. São Paulo: Makron Book do Brasil, 1999. 530 p.

BECK, Toman. **Modelo de processo pessoal e de equipe.** 6ª Edição Brasil:Mcgraw-hill Interamericana, 2001. 752 p.

BOEHM, Andy. **O papel evolutivo do software.** 6ª Edição Brasil: Mcgraw-hill Interamericana, 2003. 752 p.

BOEHM, Madame. **Software legado.** 6ª Edição Brasil:Mcgraw-hill Interamericana, 2006. 752 p.

BRASIL (Org.). **Qualidade para Desenvolvedores de Software no SPB.** Disponível em: <http://www.softwarepublico.gov.br/5cqualibr/xowiki/Desenvolvedor>. Acesso em: 12 out. 2013.

BRODERICK, Kerley. **Analise de riscos.** São Paulo: Makron Books do Brasil, 2008. 530 p.

CARDIAS JUNIOR, Alexandre Brito et al. Uma Análise Avaliativa de Ferramentas de Software Livre no Contexto da Implementação do Processo de Gerência de Requisitos do MPS.BR. Para, PA, 28 abr. 2012.

CHRISTOPHER, Verzuh. **Gerência de projetos:** Métricas de software. São Paulo: Makron Book do Brasil, 2008. 530 p.

COCKBURN, Lucie. **Modelos incrementais de processos.** 6ª Edição Brasil: Mcgraw-hill Interamericana, 2002. 752 p.

COHEN, George. **Evolução de software.** 6ª Edição Brasil: Mcgraw-hill Interamericana, 2004. 752 p.

COHN, Fred. **Engenharia de software.** 6ª Edição Brasil: Mcgraw-hill Interamericana, 2006. 752 p.

COSTA, Lídio Mauro Lima. **Gerenciando Projetos de Software.** 2008. 94 f. Tese (Doutorado) - Curso de Teleinformática, Departamento de Engenharia de Teleinformática, Universidade Federal do Pará (ufpa) – Campus Universitário de Castanhal – PA – Brasil, Pará - PA, 2008.

CAVALHEIRO, Bruno Bailuni; ANTONIO, Daniel Soldati. **Proposta de Modelo de Gerenciamento de riscos em Projetos:** Estudo de caso para projetos de avaliação patrimonial. Niterói: Universidade Federal Fluminense, 2008.

CESAR FILHO. **Gerenciamento de projetos de software.** 2007. 140 f. Tese (Mestrado) - Departamento de Administração da Faculdade de Economia, Universidade de São Paulo, São Paulo, 2007.

CHAVES, André Abe. **ATMM uma ferramenta para gerenciamento de métricas de teste no contexto de métodos ágeis.** 2010. 112 f. Dissertação (Mestrado) - Instituto de Ciências Matemáticas e de Computação Universidade de São Paulo (icmc), São Paulo, 2010.

CHAVES, Gustavo Leite de Mendonça. Desenvolvimento de software com ferramentas livres e de baixo custo: metodologia e estudo de caso. **Desenvolvimento de Software**, Campinas, SP, 20 jan. 2010.

COCKBURN, Aldous. **Modelos prescritivos de processo.** 6ª Edição Brasil: Mcgraw-hill Interamericana, 2007. 752 p.

CRISPIM, José Joaquim Rosado (Org.). **Microsoft Project:** Manual de formação. São Paulo: Laboratório de Controle e Microinformática, 2004. Manual de revisão.

DIAS, Aline Antunes. Ferramentas Gratuitas para Gerência de Requisitos. Graduanda em Sistemas de Informação UFLA. Disponível em: <http://pt.scribd.com/doc/55804055/Ferramentas-Gratuitas-para-Gerencia-de-Requisitos>. Acesso em: 28 abr. 2012.

DIAS, André Barcaui. **Porque gerenciar Projetos.** 2008. 85 f. Tcc (Trabalho de Conclusão de Curso) - Departamento de Engenharia Informática, Universidade Federal De São Paulo, São Paulo - SP, 2008.

DIAS, Ciro Oliveira. **Desenvolvimento de software:** Processos Ágeis ou Tradicionais uma visão critica. 2005. 105 f. Tese (Mestrado) - Departamento de Sistemas e Computação, Faculdade Presidente Antônio Carlos de Teófilo Otoni, Teófilo Otoni MG, 2005.

DIAS, Eduardo. **Integração das metodologias RUP e PMI no desenvolvimento de projetos de software:** A Experiência do Escritório de Projetos da Academia Nacional da Polícia Federal. 2007. 105 f. Dissertação (Doutorado) - Departamento de Engenharia da Computação - Campus Curitiba, Escola Politécnica, Curitiba - PR, 2007.

DINSMORE, Dave. **Tecnologia de processo.** 6ª Edição Brasil: Mcgraw-hill Interamericana, 2008. 752 p.

DOTPROJECT. **Introdução a dotProject.** Brasília, 2006. 45 p.

DUE, Rowe Richard. **Engenharia de Requisitos:** Gestão de Requisitos. 6ª Edição Brasil: Mcgraw-hill Interamericana, 2009. 752 p.

EDER, Taylor. **Prática de Engenharia de Software:** A essência da prática. 6ª Edição Brasil: Mcgraw-hill Interamericana, 2010. 752 p.

EDUARDO NETO. **Testes no contexto de métodos ágeis:** técnicas, padrões e apoio automatizado. 2009. 156 f. Dissertação (Mestrado) – PUC - RS, Porto Alegre, 2009.

FABIO CAMARA. Gerência de projetos: Scrum. Uma Metodologia ágil. **Imasters**, Campinas, SP, n. , p.02-03, 17 nov. 2008.

FIGUEIREDO, Edes Garcia da Costa. **Métodos Ágeis e Padrões Organizacionais e de Processo.** 2006. 85 f. Tcc (Trabalho de Conclusão de Curso) - Departamento de Computação, Universidade Federal de São Carlos, São Paulo, 2006.

FRAGA, Joni da Silva; FARINES Jean Marie (Org.). **Ambiente de produção de software para sistemas distribuídos em tempo real.** Londrina: Laboratório de Controle e Microinformática, 2009. Depto. de Engenharia Elétrica - UFSC.

GANTTPROJECT: Especificação de Ferramenta. Versão 1.0. Londrina: Universidade de Londrina, 2009. Gerenciamento de serviços de T.I no Pólo Moveleiro de Arapongas.

GOLDMAN, Alfredo. **Workshop Brasileiro de Métodos Ágeis:** Conferência Brasileira sobre Métodos Ágeis de Desenvolvimento de Software Agile Brasil. São Paulo: PUC - RS, 2010.

GOLDMAN, Daisy. **Medidas e Métricas.** 2007. 90 f. Tcc (Trabalho de Conclusão de Curso) - Universidade de Federal de Santa Maria, São Paulo, 2007.

GUSTAFSON, Addison. **Gestão de Qualidade:** Qualidade. 6ª Edição Brasil: Mcgraw-hill Interamericana, 2002. 752 p.

HEPTAGON. **As Cinco Doenças do Gerenciamento de Projetos.** Disponível em: <http://www.heptagon.com.br/>. Acesso em: 18 out. 2013.

HETZEL, Pierre Jules; IEEE. **Indústria e conselho divergem sobre falta de engenheiros.** Disponível em: <http://www.ieee.org.br/news-item/industria-e-conselho-divergem-sobre-falta-de-engenheiros/>. Acesso em: 18 out. 2013.

HIGHSMITH, Roger. **Monitoração e controle.** São Paulo: Makron Books, 2004. 115 p.

HIGHSMITH, Tennyson. **Gerenciamento de projetos:** Planejamento. São Paulo: Makron Book do Brasil, 2004. 530 p.

HIGHSMTH, Willian. **Avaliação de processo.** 6ª Edição Brasil: Mcgraw-hill Interamericana, 2004. 752 p.

HIGHSMTH, Willian. **Princípios de Modelagem de Projetos:** Princípios. 6ª Edição Brasil: Mcgraw-hill Interamericana, 2009. 752 p.

HILSON, Alexandre Lazaretti; VILAIN, Patrícia. Universidade de Passo Fundo - UPF ICEG - Ciência da Computação. **Uma Análise do Método Ágil Scrum Conforme Abordagem Nas Áreas de Processo Gerenciamento e Desenvolvimento de Requisitos do Cmmi**, Passo Fundo, n., p.5-6, 25 out. 2005.

IEEE: Revista IEEE América Latina. São Paulo: Institute OfElectricalAndElectronic Engineers, 12 nov. 2013. Mensal.

IEEE: Metodologias ágeis. São Paulo: Institute OfElectricalAndElectronic Engineers, 12 nov. 2008.

IMAP (Org.). **Software.** Disponível em: <http://www.imapaudits.com/software/ >. Acesso em: 12 out. 2013.

JACKSON, Willian. **Modelagem de Análise:** Modelagem de Dados. 6.ed. Brasil: Mcgraw-hill Interamericana, 2005. 752 p.

JACOBSON, Ivar. **Desenvolvimento Ágil:** O que é agilidade. 6ª Edição Brasil: Mcgraw-hill Interamericana, 1999. 752 p.

KITCHENHAM, Ronald. **Análise dos riscos.** São Paulo: Makron Books, 2001.

KNIBERG, Henrik. **Kanban e Scrum:** obtendo o melhor de ambos. Disponível em: <http://www.infoq.com/br/minibooks/kanban-scrum-minibook>. Acesso em: 20 out. 2013.

KONSULTE. **Análise.** Disponível em: <http://www.techgig.com/projects>. Acesso em: 12 out. 2013.

KOPPERNSTEINER, Jones. **Software e engenharia de software.** 2003. 530 f. Tese (Mestrado) - Universidade de São Paulo, São Paulo, 2003.

KOREN, Barbara. **A importância do software.** São Paulo: Makron Book, 2011.

KRIESER, Steves. **Uma visão genérica da engenharia de software.** São Paulo: Makron Books, 2013.120 p.

LACY, Mathew. **Garantia de qualidade de software.** São Paulo: Makron Book, 2011. 540 p.

LEITE, Marisa. **Gerenciamento de projetos de Desenvolvimento de software.** 2007. 212 f. Dissertação (Mestrado) - Departamento de Administração e Contabilidade, Universidade de São Paulo, São Paulo, 2007.

LIBARDI, Paula L.o.; BARBOSA, Vladimir. **Métodos Ágeis.** Limeira: Universidade Estadual de Campinas, 2010.

LIMA, Alfredo. **Workshop Brasileiro de Métodos Ágeis.** 2009. 156 f. Dissertação (Mestrado) - Puc - Rs, Porto Alegre, 2009.

LIMA, Aurélio Camara. Análise Comparativa De Ferramentas Livres Para Adequação Às Áreas De Processo Do Nível 2 Do Modelo Cmmi. In: Universidade De Passo Fundo, 2009, Passo Fundo. Passo Fundo: Universidade De Passo Fundo, 2009.

LIVRE, Espirito. **Princípios de análise.** Disponível em: <http://www.revista.espiritolivre.org/>. Acesso em: 12 out. 2013.

MANUAL de Implementação de Processo do MPS. BR com Apoio Ferramental: Gerência de Configuração. 1.2 Para, PA: Universidade do Para, 2009. Versão 1.0.

MARÇAL, Carlos Dos Santos. **Ferramentas de análise dinâmicas.** 2007. 85 f. Tcc (Trabalho de Conclusão de Curso) - Universidade de Brasília, São Paulo, 2007.

MARSHALL, Levesson. **Gestão de Riscos:** Riscos de Software. 6ª Edição Brasil: Mcgraw-hill Interamericana, 2006. 752 p.

Melo Claudia de O. ; Viviane A. Santos; Hugo Corbucci; Eduardo Katayama; Alfredo Goldman; Fabio Kon. Métodos ágeis no Brasil: estado da prática em times e organizações. Relatório Técnico RT-- MAC- 2012-- 03. Departamento de Ciência da Computação. IME -- USP. Maio, 2012.

MELO, Fábio. **ClassMock:** A Testing Tool for Reflective Classes Which Consume Code Annotations. 2012. 150 f. Dissertação (Mestrado) - Universidade Federal de São Paulo (unifesp), São Paulo, 2012.

MENDES, Marcos Alves. **Gerenciador de Projetos Web (JEE):** Desenvolvimento Orientado a Objetos usando metodologia RUP um estudo de caso. 2010. 115 f.

Dissertação (Doutorado) - Departamento de Faculdade De Computação E Informática Bacharelado Em Sistemas De Informação, Universidade Presbiteriana Mackenzie, São Paulo - SP, 2010.

MENEZES, Frederico. **Gestão Ágil de Projetos.** Pernambuco - Pe: Lcte, 2008. 31 p.

MORAIS, Daniel Dinis. **DSDM:** Dynamic Systems Development Methodology. 2006. 114 f. Tese (Mestrado) - Departamento de Engenharia Informática, Faculdade de Engenharia da Universidade do Porto, Portugal, 2006.

MORAIS, Rodrigo De Oliveira. **Gerenciando Projetos de Software com RUP e PMBOK.** 2008. 95 f. Dissertação (Mestrado) - Departamento de Faculdade De Computação E Informática Bacharelado Em Sistemas De Informação, Universidade Presbiteriana Mackenzie, São Paulo - SP, 2008.

MOURA, André. **Gerenciar Projetos.** 2009. 92 f. Tcc (Trabalho de Conclusão de Curso) - Departamento de Informática, Universidade Estadual de Londrina, Londrina - PR, 2009.

MÜLLER NETO, Gert Uchôa. **Métodos Tradicionais versus Ágeis:** Um estudo Comparativo através do Trainingcad. Caruaru: Universidade de Pernambuco, 2009.

OLIVEIRA, Cristine Gusmão. **Gestão ágil de Projetos.** 2003. 98 f. Tese (Mestrado) - Departamento de Sistemas e Computação, Universidade de Pernambuco, Pernambuco - PE, 2003.

PARDELINHA, Albert. **Administração de projetos.** São Paulo: Makron Books, 2011. 141 p.

PEDROSO, Lealís Dos Santos. **Ferramentas e Técnicas em Softwares Livres para o gerenciamento de projetos:** Estudo de caso para projetos de avaliação patrimonial. Curitiba: Universidade Federal do Paraná, 2008.

PEINADO, Jurandir. **COMPREENDENDO O KANBAN:** UM ENSINO INTERATIVO ILUSTRADO. Curitiba - PR: Unicenp/centro Universitário Positivo, 2007. 14 p.

PENTEADO, Jurandir. **Funcionamento do kanban:** O quadro kanban. 2010. 110 f. Tcc (Trabalho de Conclusão de Curso) - Universidade Federal do Paraná, Curitiba - PR, 2010.

PEREIRA, Danilo. **Métricas de Acompanhamento para Metodologias Ágeis:** Metodologias ágeis. 2007. 94 f. Tcc (Trabalho de Conclusão de Curso) - Departamento de Centro De Ciências Exatas E Tecnologia, Universidade Federal De São Paulo, São Paulo - SP, 2007.

PEREIRA, Luciano Malaquias. **Métricas ágil XP:** Extreme Programming. 2007. 97 f. Tcc (Trabalho de Conclusão de Curso) - Departamento de Centro De Ciências Exatas E Tecnologia, Universidade Federal Do Estado Do Rio De Janeiro, Rio de Janeiro - Rj, 2007.

PFLEEGER, Maldonado. **Medidas e Métricas.** 2007. 85 f. Monografia (Trabalho de Conclusão de Curso) - Universidade de Passo Fundo, São Paulo, 2007.
PIRES, Anderson; PINTO, Nathália. **Ciclo de vida do XP.** Pernambuco - Pe: Lcte, 2005. 25 p.

PMBOK (Org.). **Guia PMBOK ® e Normas.** Disponível em: <http://www.pmi.org/PMBOK-Guide-and-Standards.aspx >. Acesso em: 12 out. 2013.

PMBOK. **PMI padrões atuais de Projetos:** Padrões de atividades de desenvolvimento. Guias e Normas. Disponível em: <http://www.pmi.org/PMBOK-Guide-and-Standards.aspx>. Acesso em: 06 out. 2013.

PMI (Org.). **Desenvolvimento e Educação.** Disponível em: <http://brasil.pmi.org/>. Acesso em: 12 out. 2013.
PMI. **O que são os Padrões de Projeto.** Guias e Normas. Disponível em: <http://brasil.pmi.org/>. Acesso em: 06 out. 2013.

PMP. **Gestão.** Disponível em: <http://www.pmi.org/Business-Solutions.aspx>. Acesso em: 12 out. 2013.

PRESSMAN, Roger S.. **Prática de Engenharia de Software:** Princípios da modelagem de análise. 6ª Edição Brasil: Mcgraw-hill Interamericana, 2006. 752 p.

PROJECT, Open. **Gestão de Projetos.** Disponível em: <http://sourceforge.net/projects/openproj/>. Acesso em: 12 out. 2013.

PUSTNUKE. **DotProject:** ferramenta de Gerenciamento de Projetos Open Source. Disponível em: <http://www.dotproject.net/index. php>. Acesso em: 28 set. 2013.

PUSTNUKE. **Flexível Sistema de Gerenciamento de Conteúdo.** Disponível em: <http://www.postnuke.com/>. Acesso em: 19 out. 2013.

SANTOS, Carlos Eduardo. **Uma proposta de aplicação do Kanban do controle de estoque de uma empresa comercial de pequeno porte.** 2006. 85 f. Dissertação (Mestrado) - Universidade Federal de Juiz de Fora, Juiz de Fora - MG, 2006.

SANTOS, Rafael Dos; DOMINGOS, André Luiz. **Métodos Ágeis de Desenvolvimento de Software.** Porto Alegre - RS: Lcte, 2010. 45 p.

SATO, Danilo Toshiaki. **Uso eficaz de métricas em métodos ágeis de desenvolvimento de software.** 2007. 155 f. Dissertação (Mestrado) - Instituto de Matemática e Estatísticas - IME - USP, São Paulo, 2007.

SCHWABER, Richard. **Processo:** uma visão genérica. 6ª Edição Brasil: Mcgraw-hill Interamericana, 2002. 752 p.

SCOTT, Allan. **A política de desenvolvimento ágil.** 6ª Edição Brasil: Mcgraw-hill Interamericana, 2008. 752 p.

SCRUM (METODOLOGIA PARA O DESENVOLVIMENTO ÁGIL DE SOFTWARE). S.l: Rio de Janeiro, R.j, 20 nov. 2010.
SILVA, Anderson. **Métodos ágeis nas organizações.** 2009. 94 f. Tcc (Trabalho de Conclusão de Curso) - Departamento de Informática, Universidade Federal De São Paulo, São Paulo - Sp, 2009.

SILVA, José Paulo. **Metodologias Ágeis e Extreme Programming (XP):** Uma Introdução. 2009. 85 f. Tese (Mestrado) - Universidade de São Paulo, São Paulo, 2009.

SILVA, Júlio Eduardo da. **Integração das metodologias RUP e PMI no desenvolvimento de projetos de software:** A Experiência do Escritório de Projetos da Academia Nacional da Polícia Federal. 2009. 110 f. Dissertação (Mestrado) - Departamento de Administração Estratégica de Sistemas de Informação, Fundação Getúlio Vargas – Fgv, Núcleo de Brasília., Brasília - DF, 2009.

SILVA, Luiz da; RICARDO NETO,; SANTOS, Mateus. **Modelos evolucionários de processo de software.** São Paulo: Makson, 2012. 350 p.

SILVEIRA, Luciano Pereira. **Métricas de Acompanhamento para Metodologias Ágeis:** Metodologias ágeis. 2012. 91 f. Tcc (Trabalho de Conclusão de Curso) - Departamento de Centro De Ciências Exatas E Tecnologia, Universidade Federal De São Paulo, São Paulo - SP, 2012.

SIMONSEN, Capers. **Gestão de Riscos:** Riscos de Software. 6ª Edição Brasil: Mcgraw-hill Interamericana, 2004. 752 p.

SIQUEIRA, Fábio Levy. **Projetos de software.** 2005. 95 f. Tese (Mestrado) - Universidade de São Paulo, São Paulo, 2005.

SIQUEIRA, Mauro. **Gerenciamento de projetos na engenharia de software.** 2004. 94 f. Tese (Mestrado) - Departamento de Engenharia da Computação - Campus Curitiba, Escola Politécnica, Pará - PA, 2004.

SOARES, Anderson. **Aplicação de métodos ágeis em um processo de desenvolvimento de software.** 2009. 96 f. Tese (Mestrado) - Departamento de Bacharelado em Sistemas de Informação – Campus Canoas, Universidade Luterana do Brasil (ULBRA), Canoas - RS, 2009.

SOARES, Michel Dos Santos. Comparação entre Metodologias Ágeis e Tradicionais para o Desenvolvimento de Software. **Comparação entre Metodologias Ágeis e Tradicionais**, Campinas, SP, n., p.02-02, 20 nov. 2011.

SOFTEX (Org.). **Guia de Implementação – Parte 1:** Fundamentação para Implementação do Nível G do MR-MPS. Campinas, SP, 2009. 31 p.
SOMMERVILLE, Ian. **Uma introdução à computação em nuvem.** Disponível em: <http://www.software-engin.com/ >. Acesso em: 12 out. 2013.

TAGUCHI, Frederick. **Escopo de projetos.** São Paulo: Makron Books, 2009. 149 p.
TEIXEIRA, Betânia Oliveira. **Utilização de métricas nos projetos de desenvolvimento de sistemas de informação:** Um Survey com Gerentes de Projetos. 2005. 88 f. Tcc (Trabalho de Conclusão de Curso) - Departamento de Centro De Ciências Exatas E Tecnologia, Universidade Federal Do Estado Do Rio De Janeiro, Rio de Janeiro - Rj, 2005.

TENÓRIO, Ludmila Varela Arruda. DESENVOLVIMENTO ÁGIL DE SOFTWARE: UMA ANÁLISE SINTÉTICA A PARTIR DA METODOLOGIA KANBAN. **Desenvolvimento Ágil**, Palmas - To, n., p.4-5, 10 maio 2008.

THOMSET, Steve; PMI. **Gerencia de projeto.** São Paulo: Makron Books, 2004. 150 p.

TOFFER, Morrow. **O Processo de Software:** Engenharia de Software - Uma tecnologia em Camadas. 6ª Edição Brasil: Mcgraw-hill Interamericana, 1980. 752 p.
TORREÃO, Mary. **Mitos do software.** 6ª Edição Brasil: Mcgraw-hill Interamericana, 2007. 752 p.

UDO, Randell; KOPPENSTEINER, Evans. **O plano de projeto de software.** São Paulo: Makron Books, 2003. 105 p.

VIÉGAS, Michael. **Métricas de qualidade de software.** São Paulo: Makron Book, 2005. 545 p.

VINÍCIUS, Diogo. **Os benefícios do uso de Kanban na gerência de projetos de manutenção de software.** 2012. 80 f. Dissertação (Mestrado) - Infoway Tecnologia e Gestão em Saúde Ltda, Teresina – PI, 2012.

WANDERLEY, Danillo. **Gerência de Projetos.**Pernambuco - PE: Lcte, 2008. 31 p.
WESLEY, Howard. **Engenharia de Sistemas:** A hierarquia da engenharia de sistemas. 6ª Edição Brasil: Mcgraw-hill Interamericana, 2009. 752 p.

APÊNDICE A

APÊNDICE A - ANÁLISE COMPARATIVA ENTRE FERRAMENTAS LIVRES PARA GERENCIAMENTO DE PROJETOS DE SOFTWARE DESENVOLVIDOS COM METODOLOGIAS ÁGEIS

Análise Comparativa Entre Ferramentas Livres para Gerenciamento de Projetos de Software Desenvolvidos com Metodologias Ágeis

Anderson Torquato Cardoso[1], Ana Claudia Garcia Barbosa[2]

[1]Universidade do Extremo Sul Catarinense (UNESC)
Criciúma – Santa Catarina – SC – Brasil

[2] Departamento de Ciência da Computação

Universidade do Extremo Sul Catarinense Criciúma (UNESC) – Criciúma, SC – Brasil

andgimmo@hotmail.com, agb@unesc.net

Abstract. *This course conclusion work presents a comparative analysis for free project management software tools. This study compares some of these tools and offers help in choosing the most appropriate, in the decisions of developers who choose to agile methodology. In the foreground were chosen free tools that have minimal items necessary for the project, the second time the items used for this comparison tools have been studied stages of the agile Scrum methodology, beyond these steps, the choice of items to be assessed were based in an extensive survey conducted by the University of São Paulo to raise the current behavior of adoption and adaptation of agile methodology in Brazilian companies. This analysis was conducted in five tools for agile project management.*

Resumo. *Este trabalho de conclusão de curso apresenta uma análise comparativa de ferramentas livres para gerenciamento de projetos de software. Essa pesquisa compara algumas dessas ferramentas e propõe auxiliar na escolha da mais adequada, nas decisões dos desenvolvedores que optam por metodologia ágil. Em primeiro plano foram escolhidas ferramentas livres que tivessem o mínimo de itens necessários aos projetos, no segundo momento os itens usados para essa comparação de ferramentas foram etapas estudadas da metodologia ágil Scrum, além, desses passos, a escolha dos itens a serem analisados foram baseados em uma vasta pesquisa realizada pela Universidade de São Paulo para levantar o comportamento atual de adoção e adaptação da metodologia ágil nas empresas brasileiras. Essa análise foi realizada em cinco ferramentas de gerenciamento de projetos ágeis.*

1. Introdução

Com a demanda de software no mercado de desenvolvimento a busca por equipes que aplicam metodologias ágeis no gerenciamento de projetos se torna cada vez constante, visto que essa prática pode auxiliar com a agilidade a entrega de software (LIMA, 2009).

Com isso agilizar o desenvolvimento de software com praticas ageis e gerenciadores de projetos, faz com que os projetos e as equipes fiquem de forma planejada e no cronograma programado (EDUARDO, 2009).

Portanto, para realizar esse estudo apresenta ferramentas de gerenciamento de projetos livres especificas para a prática de aplicação das metodologias ágeis.

2. Engenharia de software

A engenharia de software aplica organização e modelagem aos projetos, trazendo beneficios aos desenvolvedores nas práticas aplicadas para o desenvolvimento de software entre elas: comunicação, planejamento, modelagem, construção e implatação (PRESSMAN, 2006).

3. Gerenciamento de Projetos

A gestão de projetos nos software desenvolvida é a utilização de metodologias e ferramentas de gerenciamento, onde permite organizar e controlar as tarefas e atividades rastreando o andamento do desenvolvimento do projeto (MOURA, 2007).

Para obter organização sobre projetos necessita-se elaborar um plano de todo o projeto, com objetivo de determinar as principais atividades necessarias do projeto. Onde ao longo do desenvolvimento esse plano estabelace as datas de marcos do processo, bem como começa e termino das atividades associadas aos responsáveis da equipe e atividades geradas (PMBOK, 2010).

4. Metodologias Ágeis

A metodologias ágeis são conceitos de métodos aplicados aos projetos de software, onde vem trazer agilidade nos desenvolvimento de soluções projetos entre elas, possibilita fazer modificações de requisitos (LIBARDI; BARBOSA, 2010).

Com esses conceitos sobre métodos ágeis, se dá uma visão geral sobre a modelagem agil, onde são: simplicidade, mudanças, objetivos, trabalhos futuros, modificações no desenvolvimento, modelagem, qualidade e feedback.

4.1. SCRUM

A Metodologia ágil SCRUM funciona de forma iterativa e incremental, contribuindo no desenvolvimento de projetos, e agregando os seguintes valores: satisfação, colaboração, motivação e integração de clientes e equipe, onde com isso obtém-se resultados de sucesso em projetos de software (SCHWABER, 2002).

Essas iterações são bem definidas que funciona de duas a quatro semanas que são os Sprints (atividades), a próxima iteração são reúniões rápidas dia-a-dia com as equipes monitorando a evolução do projeto. Na sequencia são gerados os graficos do projeto, bem como a concepção das tarefas e das atividades agregadas com a equipe de desenvolvimento, e por fim são gerados os relatórios que são os retrospectiva e planejamento das atividades já realizadas.

4.1. XP

O método ágil XP trabalha com equipes médias e pequenas, onde por sua vez irão desenvolver o projeto com requisitos de constantes mudanças. Com isso, são adotadas também estratégias de constante acompanhamento e realiza-se ajustes pequenos ao desenvolvimento de software.

Com isso o seu funcionamento se dá pelas seguintes práticas: planejamento, metáfora, fases pequenas, desing simples, time coeso, testes de aceitação, semana de quarenta horas, reuniões em pé, propriedade coletiva, programação em duplas, padrão no código, desenvolver orientado a testes, refatoração e integração continua (PENTEADO, 2010).

4.1. FDD

Essa metodologia consiste no desenvolvimento guiado por funcionalidades, onde serve de base para gerenciar e desenvolver projetos de software. Foi criada em 1997 em Singapura para United Overseas Bank, em um grande projeto Java (HEPTAGON, 2013).

Esse método de desenvolver com funcionalidades tem por objetivo beneficiar processos rigorosos, entre eles: modelar, prever plano de projeto, controlar o projeto, entre outros. Com isso também possui propriedades dos métodos ágeis dentre eles são: focar na programação, interagir com o cliente e entregar versões do software com mais frequências.

Essas práticas para desenvolvimento compreendem apenas no software, ou seja, focar na programação e nas funções de negócio que o software deve realizar, pois o método FDD não se aplica a escolha de ferramentas e tecnologias (SILVA, 2009).

4.1. KANBAN

O método Kanban originou-se no Japão com objetivo de facilitar a linha de produção em série, criado pela Toyota para controlar e aumentar sua linha de produção automobilística da época. No Brasil foi adotada nos anos oitenta, essa prática se aplica a fluxo de peças e gestão de estoque, onde funciona na prática de cartões, modo visual de gestão de software (SILVEIRA, 2012).

O funcionamento da metodologia Kanban se caracteriza da seguinte forma, o cartão Kanban serve para a comunicação e o funcionamento do sistema, nele deve estar às informações principais para a linha de produção, se houver a necessidade de mais informação no cartão Kanban, que seja na área especifica onde se pretende implementar a metodologia Kanban (AGUIAR, 2007).

4.1. DSDM

A metodologia de desenvolvimento de sistemas dinâmicos, do inglês Dynamic Systems Development (DSDM) para desenvolvimento de software faz com que ocorra o envolvimento continuo do usuário, com o objetivo de entregar sistemas no tempo e orçamentos previstos, ou seja, enquanto se ajusta as alterações dos requisitos ao longo do processo de desenvolvimento (TENÓRIO, 2008).

A metodologia procura resolver questões de negócios em engenharia de software, onde os princípios citados servem para aproximar clientes, equipes de desenvolvimento, gestores do projeto, e entre outros, a fim de cumprir prazos para entregas, custos (BRODERICK, 2008).

5. Ferramentas de Gerenciamento de Projetos livres com verificação da metodologia ágil Scrum nos gerenciadores Artia, Redmine, Trello, OpenProject e GanttProject

Neste capítulo serão descritas as ferramentas com suas características importantes do ponto de vista das metodologias ágeis. Foi feito estudo em algumas ferramentas livres de licença, a escolha foi feita por serem as mais conhecidas do mercado de gerenciamento de software.

A escolha por descrever as ferramentas deu-se ao fato de melhor conhecê-las para depois analisá-las em seus atributos. Nos itens a seguir uma descrição dos processos da metodologia agil Scrum com seus grupos praticos e seus principais papéis.

Product backlog: consiste em uma lista de tarefas a ser determinada pelo cliente, onde logo após se mantém como requisitos do projeto.

Sprint backlog: compreende em uma lista de atividades de cada tarefa descrita acima, onde representa em tempo real o andamento do projeto.

Burndown: são os graficos das atividades desenvolvidos durante o dia.

Product Owner: representa o dono do projeto, ou seja, a voz do cliente, onde agrega valor ao negócio durante o desenvolvimento.

Team: são as pessoas da equipe, onde fazem o trabalho real e entregam o produto final.

Scrum Master: garante o desenvolvimento do processo do método Scrum. Responsável pela aplicação das regras, proteger a equipe e manter o foco nas tarefas na prática.

Sprint planning meeting: indica o trabalho a ser feito, prepara o tempo das atividades em toda a equipe.

Sprint Review: rever o trabalho que foi concluído e o que não foi concluído.

Sprint Retrospective: todos as pessoas da equipe refltem sobre a atividade passada, e fazem melhorias contínuas de processo.

6. Resultados Obtidos das Ferramentas de Gerenciamento de Projetos com método ágil Scrum: Artia, Redmine, Trello, OpenProject e GanttProject

Após as ferramentas serem analisadas pela metodologia ágil SCRUM, esse capítulo descreve os resultados obtidos pela utilização, teste e análise feita nessas ferramentas. Cada tópico corresponde a uma pesquisa realizada no mercado de software Brasileiro onde tratam dos métodos ágeis no Brasil, esses requisitos foram analisados nas ferramentas: Artia, Redmine, Trello, OpenProject e GanttProject.

Essa pesquisa foi pelos 10 anos de desenvolvimento de projetos de software com métodos ágeis no Brasil também, um grupo de pesquisadores da Universidade de São Paulo fez uma pesquisa para levantar o comportamento atual de adoção e adaptação da metodologia ágil nas empresas brasileiras. Logo a seguir no itens abaixo, uma descrição dos resultados obtidos pelas ferramentas analisadas.

Product backlog: Artia cria uma forma de comunidades e organizam as atividades vinculadas às tarefas. No Redmine precisa selecionar e atribuir em cada tarefa, o Trello funciona diferenciado das duas mencionadas acima, ao inserir tarefas mostra um aspecto em status, onde indica o total das atividades, já na gerencia do OpenProject e GanttProject possibilita marcar as datas iniciais e finais das tarefas e atividades, que nas outras citadas isso não ocorre.

Sprint backlog: No Artia mostra ao lado uma descrição do que fazer e opções de edição, exclusão e datas de prazo. Já no Redmine exibe a situação de andamento, prioridades e qual membro estão atribuídos a atividade. Com o Trello esse Spring você marca o que já foi concluído no projeto em forma de porcentagem, com o OpenProject, o ambiente limita algumas propriedades entre elas situação e prioridades de andamento como na Redmine. E na ferramenta GanttProject não possui status de acompanhamento como no Trello, mais possibilita personalizar de maneira direta os membros da equipe nesse Spring.

Burndown: O gráfico no Artia aparece ao selecionar as atividades, o que no Redmine isso não ocorre, ou seja, só mostra um gráfico geral de todo o projeto. No gerenciador Trello para gerar um Burndown, o usuário deve instalar e adicionar as extensões do navegador, um plugin do tipo Scrum para Trello, onde permite configurar por data de inicio e fim o status do projeto. Com o OpenProject em ambiente desktop esse gráfico pode ser gerado por meio de filtragem de recursos e não por datas, e atividades como no Artia e Trello, essas filtros podem ser mostrados por tarefas, grupos e ordenação de nomes. Na estrutura do GanttProject esse Burndown gera por atividades semanais, ou seja, exibe de forma limitada pelas tarefas divididas no product backlog e vinculadas umas nas outras.

Product Owner: na ferramenta Artia na forma de gerenciar pessoas, pois no projeto indica ser um líder que também pode convidar usuários. No Redmine pode-se apenas visualizar isso em visão geral que por sua vez configura-se apenas uma vez. Com o Trello essa opção possibilita editar, adicionar foto e anexar um link de uma página pessoal desse product Owner. No OpenProject está em recursos humanos, onde aparece de gerente do projeto, não tem a liberdade de fazer configurações personalizadas. Já o GanttProject funciona semelhante ao OpenProject, onde também está em recursos humanos e cadastra-se como gerente de projeto, uma estrutura limitada para um Product Owner, sendo ele o proprietário do projeto.

Team: com o Artia uma equipe consiste em comunidades ou pessoas associadas, que para cada tarefa do projeto possibilita atribuir uma comunidade e pessoas. No Redmine funciona na opção membros, está limitado em editar, excluir e dividir papéis. Já no Trello o projeto consiste de forma publica para depois e fazer as associações de pessoas e estabelecer as permissões de acesso ao projeto. O OpenProject, só pode fazer atribuições de pessoas as tarefas aqueles já cadastrados no projeto. Com o GanttProject a equipe está em uma aba de diagrama de pessoas, depois de adicionar os membros de desenvolvimento, não propicia fazer outro compartilhamento de equipe.

Scrum Master: no gerenciador Artia não foi implementado de maneira exclusiva, pois está na mesma opção dos membros da equipe que se cadastra como líder e Product Owner do projeto. O Redmine não possui uma opção exclusiva de manager, ou seja, configura-se em membros da equipe. No gerenciador Trello consiste no administrador do projeto, no menu members possibilita fazer a permissão (admim), mais não tem também um tipo de implementação para o método Scrum Master. O OpenProject o Scrum Master encontra-se em uma tela de recursos humanos onde representa o gerente, permite vincular a um projeto. Já no

GanttProject ocorre da mesma forma como as demais ferramentas mencionadas anteriormente, ou seja, limita-se a recursos de uma ferramenta de gerenciamento de projetos genérica, onde pode ser implementada diversas metodologias ágeis, e não apenas o método Scrum.

Sprint planning meeting: Artia consiste em uma agenda de calendário semanal, onde cada membro da equipe aparece atribuído a uma atividade do projeto. O Redmine apresenta em calendário semanal, com setas coloridas indicando o começo e fim das atividades. Com o Trello esse planejamento não está implementado de forma especifica o gestor configura apenas a data de começo da atividade. Com o OpenProject esse planejamento pode ser filtrado por grupos, tarefas e ordenação, apesar de cada gerenciador possuir uma estrutura de projeto, cabe ao gestor de desenvolvimento optar pelo ambiente das equipes e tarefas a ser implementadas nesse plano de trabalho. E no GanttProject ao planejar as tarefas exibe apenas uma comunicação com as demais pessoas da equipe e atividades a ser realiza, limita se por estar em modo desktop e não em web.

Sprint Review: revisar atividades com o Artia está por status, que são pendências da situação da atividade, em dias de atraso. No Redmine funciona como bugs das tarefas, onde esse recurso pode ser compartilhado na ferramenta através de comunidades. No Trello está implementado como calendário semanal de acordo com o mês planejado das atividades. No OpenProject a opções de selecionar atraso de alocações anteriores e tarefas em atraso. A estrutura de revisar Sprints no GanttProject não possui gráfico ou qualquer outro tipo de status, para isso tem que preencher um formulário em propriedades de gerenciamento do projeto, selecionar campos dos itens desejados e na aba anteriores gera a revisão.

Sprint Retrospective: no Artia o modo de exibir um breve retrospecto do que ocorreu em cada tarefa. Com Redmine acontece de forma que as tarefas passadas fiquem em um link de bugs da visão geral do projeto. Já no Trello não possibilita fazer uma compreensão detalhada de retrospectiva Sprint, na mesma estrutura de planejamento desses Sprints exibe as atividades ocorridas. No OpenProject ocorre de maneira diferente do Artia, Redmine e Trello, pois na opção recursos o gestor seleciona a tarefa e atividades desejada através de filtros. Com GanttProject funciona de forma distinta de Artia, Redmine, Trello e OpenProject, ou seja, na aba diagrama de pessoas e no gráfico a direita na opção anterior exibe Sprints passados.

Melhorar a visibilidade do projeto: com o gerenciamento do Artia a visualização do projeto e recursos para organizar tarefas. No Redmine isso acontece de maneira compartilhada em pastas descrevendo o que fazer e atribuindo aos membros da equipe. O Trello sua interface possui recursos flexíveis de arrastar tarefas cadastradas e visualizar em porcentagem de atividades concluídas. Na gestão do OpenProject funciona limitado a aparência flexível, onde não possibilita acompanhar o projeto na web e acessar o andamento da equipe. E com o GanttProject atende mais com limitações para web, por atuar em modo desktop está como o OpenProject.

Melhorar Manutenibilidade / extensibilidade de software: melhora a metodologia do projeto, entre elas: o crescimento das tarefas implementadas. O Redmine não oferece isso de maneira organizada, pois ao compartilhar as tarefas acaba com possíveis ajustes e alterações recentes. Expandir e fazer manutenção no Trello ocorre de forma a atender alterações por via web, onde o acesso permite acelerar a comunicação da equipe. Com o OpenProject apesar de ser limitado a web, oferece um suporte com que a equipe evolua o projeto sem muitos contratempos. E no GanttProject essa manutenção acontece como Artia, Trello e OpenProject,

apesar de também ser desktop como OpenProject esse gerenciador possui recursos suficientes a atender a evolução do desenvolvimento do projeto.

Melhorar o alinhamento entre TI e negócio: alinhar negócios no Artia com tecnologia da T.I ajuda a equipe e gestores a traçar decisões relevantes no projeto. Isso com o Redmine também acontece para atribuir recursos decisivos em softwares em desenvolvimentos entre eles: compartilhar tarefas e equipes via internet. O Trello também possibilita gerar relatórios automatizados e enviá-los por e-mail. Já no OpenProject funciona a atender projetos independente de organização e serviços. E com GanttProject ocorre o funcionamento como no OpenProject e também possibilita implementações em sistemas operacionais entre eles: Windows, Linux e Mac OS.

Simplificar qualidade de software: no artia possui uma estrutura para fragmentar e compartilhar com eficiência recursos ágeis. Com o Redmine essa performance também ocorre trazendo flexibilidade no gerenciamento e no software a ser produzido. No Trello a qualidade atua como no Redmine, flexível a gerenciar tarefas e atividades como arrastar e soltar, simplificando abrir outras telas de redirecionamento do andamento do software em desenvolvimento. O OpenProject atende com propriedades desktop, ou seja, recursos limitados na prática de gerenciar e desenvolver software. E no GanttProject funciona como no OpenProject, atendendo com recursos e características limitadas, mais trazendo resultados positivos no softwares desenvolvidos.

Melhorar a capacidade de gerenciar mudanças e prioridades: no Artia essas mudanças estão na hierarquia de cada Product Backlog, onde possui prioridades a atender o andamento do projeto. Com Redmine as modificações e as prioridades são compartilhadas com grupos de equipes. O Trello, pois as alterações prioritárias são compartilhadas e a equipe trabalha via web, com mensagens de e-mail e conversa por meio de chat do gerenciador. No OpenProject atende com características próprias, onde os recursos são desenvolvidos para desenvolver software local e as modificações são realizadas na opção recursos humanos e na propriedade de gerenciamento do projeto. E no gerenciador GanttProject atua também como o OpenProject, onde as alterações são realizadas no cadastro as equipes e na hierarquia dos Sprints Backlogs, mas atende de forma eficiente.

O método ágil Scrum que o mercado segue: a análise comparativa realizada nesse trabalho de conclusão de curso (TCC) utilizada foi à metodologia Scrum, onde todas as ferramentas de gerenciamento de projeto sendo elas: Artia, Redmine, Trello, OpenProject e GanttProject, atendendo de forma para trazer um desempenho satisfatório em gerenciar e controlar o andamento de projetos e equipes, mais mesmo oferecendo esses recursos ágeis, cada uma trabalha com suas particularidades, algumas sendo mais abertas funcionando na web e outras limitadas a gerenciar projetos de forma desktop.

Planejamento antecipado: o Artia atende esse requisito com suporte a recursos suficientes no cadastro dos Product Backlog e Sprint Backlog. No Redmine falta antecipar algumas tarefas, onde essas são gerenciadas apenas na prática de desenvolvimento, pois são compartilhados. Com a utilização do Trello essa antecipação de planejar tarefas e atividades atende de forma eficiente, podendo alterar permissões desejadas conforme a metodologia aplicada. Já no OpenProject atende apenas limitando recursos por ser usada em modo desktop. GanttProject funciona com uma estrutura apesar de não estar na web, o gerenciamento de equipe possui propriedades de personalizar perfil e help de ajuda conforme for desejado.

Controle gerencial: as ferramentas como Artia, Redmine, Trello, OpenProject atende de forma satisfatória, onde cada uma fornece recursos de controle particular como agendamentos personalizados e relatórios via mensagens de email, já no GanttProject atende parcial esse requisito, atua limitado pela estrutura do desenvolvimento da ferramenta, pois não atende de forma externa via navegadores, com isso os gestores não possui acesso em ambientes remotos.

Previsibilidade: no Redmine, Artia e Trello essa falta de prevenção não ocorre, pois cada gerenciador trabalha com agendas e calendários e datas previstas, com isso organizar o projeto de forma preventiva em prazos, faz com que possibilita um auxilio para os membros da equipe em desenvolvimento e Product Owner que utilizaram o projeto logo após seu encerramento. OpenProject e GanttProject esse requisito está de forma a atender parcialmente, possui alguns filtros de recursos que em algumas vezes ajudam para prever tarefas e atividades completas e incompletas, grupos e entre outros, mais para fins de previsibilidades específicos não está implementado de forma definida.

Documentação: documentar no Artia, Redmine e OpenProject atende com uma estrutura organizada e especifica, ou seja, cada uma oferece relatórios em formatos para documentar em .pdf, .xml, .xls, com isso possibilita transportar esses arquivos para lugares distintos. E nas ferramentas Trello e GanttProject esse auxilio não possui atendimento, impedindo fornecer por exemplo um resumo de parte do projeto em forma de arquivos desse tipo.

Gerenciamento de equipe distribuída: no Artia, Redmine e Trello possibilita uma produtividade mais avançada em relação ao OpenProject e GanttProject, pois com o mercado de software a cada ano atendendo uma demanda de 31,33%, esse requisito não pode deixar a desejar, por isso esse recurso em uma ferramenta de gerenciamento de software obriga a escolha ser pesquisada e estudada, especificando o projeto em desenvolvimento e a equipe.

Simplificação do processo de desenvolvimento: esse processo no mercado de desenvolvimento melhorou em 37%, e na ferramenta Artia, Redmine, Trello, OpenProject e GanttProject, atende de maneira satisfatória para desenvolver com produtividade e organizada sem contratempos, pois a estrutura implementada nos gerenciadores trabalha para reduzir prazos, custos e fornece as necessidades dos Product Owner.

A tabela a seguir faz uma comparação dos itens da pesquisa realizada pela Universidade de São Paulo, pois com esses requisitos foi analisado nessas ferramentas de gerenciamento de projetos.

Estudo comparativo das ferramentas de gerenciamento de projetos						
Requisitos de comparação	Ferramentas / requisitos	Artia	Redmine	Trello	OpenProject	GanttProject
Razão mais importante para a adoção de métodos ágeis na equipe/ organização	Melhorar a visibilidades do projeto	A	A	A	AP	AP
	Melhorar a Manutenibilidade/ extensibilidade de software	A	AP	A	A	A
	Melhorar o alinhamento entre TI e negócio	A	A	A	A	A
	Simplificar a qualidade de software	AP	AP	A	AP	AP
	Melhorar a capacidade de gerenciar mudanças de prioridades	A	AP	A	A	A
O método ágil que o mercado segue	Scrum	A	A	A	A	A
Maiores preocupações da organização na adoção	Planejamento antecipado	A	AP	A	A	A
	Controle gerencial	A	A	A	A	AP
	Previsibilidade	A	A	A	NA	NA
	Documentação	A	A	AP	A	AP
Principais benefícios obtidos com a adoção de métodos ágeis	Gerenciamento de equipe distribuída	A	A	A	NA	NA
	Simplificação do processo de desenvolvimento	A	A	A	A	A

Figura 1. Estudo comparativo das ferramentas de gerenciamento de projetos

O que mais atendeu os requisitos foi o gerenciador Artia onze dos doze itens, com seu uso na web a agilidade prevaleceu. As ferramentas que atendeu parcialmente ficaram empatadas em duas, sendo o Artia e Redmine com um dos doze itens e a que não atendeu foi duas ferramentas, pois obtiveram empate também, sendo dois dos doze itens que foram o OpenProject e GanttProject, apesar dos resultados apresentarem pontos negativos em gerenciadores desktops como: OpenProject e GanttProject, essas gerenciadores possui uma estrutura satisfatória na metodologia ágil SCRUM, mas com características de seu uso ser restrito na web e não possuir uma gestão preventiva, ou seja, antecipar alguns planejamentos. E com relação às ferramentas de atendimento parcial também apresentaram de maneira razoável, pelo seu ambiente de organização e planejamento. E por fim o Artia, sua estrutura não deixa a desejar possuindo um ambiente de excelente uso e acessível.

7. Conclusão

Ao analisar as cinco ferramentas de gerenciamento de projeto, para metodologia ágil, pode-se avaliar que o método Scrum aponta ser mais adequado ao mercado de software e atende a demanda de negócios automatizados. Visto que tanto o método SCRUM escolhido pelo mercado como essa pesquisa utilizada nesse projeto foram ambos atendidos pelos gerenciadores nessa análise comparativa entre ferramentas livres para gerenciamento de projetos de software desenvolvidos com metodologias ágeis. Apesar das particularidades de implementação de cada gerenciador. Considerando ao final um resultado satisfatório para a escolha de uma organização ou de um gerente de projetos.

Logo após essa análise das ferramentas sobre o método ágil Scrum foi também analisado a pesquisa de métodos ágeis no Brasil, onde por sua vez teve bons resultados nas ferramentas escolhidas, ou seja, a tabela de comparação apresentou os seguintes resultados, o Artia atendeu todos os requisitos, onde apenas no item simplificar a qualidade de software ficou como atende parcialmente, que significa a gestão em alguns itens do sistema Artia ser burocrático, possuir dependências a outros. No Redmine ocorre de forma como o Artia no quesito anterior e em outra como manutenção, mudanças em prioridades e planejamento antecipado, onde o sistema atende parcialmente, são subordinados a outros. O Trello atende a todos, só a documentação encontra-se fraca e atendendo parcialmente. Já no OpenProject ocorre como no Artia no requisito qualidade de software e em visão de projeto atendendo

parcialmente, também em previsibilidade e equipes distribuídas ocorre o não atendimento, fazendo um ponto negativo na falta desses itens. E por ultimo GanttProject que após analisado aparece como no OpenProject nos mesmos itens, e no quesito documentação não atende, com a falta desse item trás um ponto negativo a ferramenta. Mas a pesquisa também aponta outros números interessantes como à região do Brasil que aplica métodos ágeis e pessoas que utilizam.

8. Referências

AGUIAR, Giancarlo De França. COMPREENDENDO O KANBAN: UM ENSINO INTERATIVO ILUSTRADO. Curitiba - PR: Unicenp/centro Universitário Positivo, 2007. 14 p.

BRODERICK, Kerley. Analise de riscos. São Paulo: Makron Books do Brasil, 2008. 530 p.

EDUARDO NETO. Testes no contexto de métodos ágeis: técnicas, padrões e apoio automatizado. 2009. 156 f. Dissertação (Mestrado) – PUC - RS, Porto Alegre, 2009.

HEPTAGON. As Cinco Doenças do Gerenciamento de Projetos. Disponível em: <http://www.heptagon.com.br/>. Acesso em: 18 out. 2013.

LIBARDI, Paula L.o.; BARBOSA, Vladimir. Métodos Ágeis. Limeira: Universidade Estadual de Campinas, 2010.

LIMA, Alfredo. Workshop Brasileiro de Métodos Ágeis. 2009. 156 f. Dissertação (Mestrado) - Puc - Rs, Porto Alegre, 2009.

LIMA, Aurélio Camara. Análise Comparativa De Ferramentas Livres Para Adequação Às Áreas De Processo Do Nível 2 Do Modelo Cmmi. In: Universidade De Passo Fundo, 2009, Passo Fundo. Passo Fundo: Universidade De Passo Fundo, 2009.

MOURA, André. Gerenciar Projetos. 2009. 92 f. Tcc (Trabalho de Conclusão de Curso) - Departamento de Informática, Universidade Estadual de Londrina, Londrina - PR, 2007.

PENTEADO, Jurandir. Funcionamento do kanban: O quadro kanban. 2010. 110 f. Tcc (Trabalho de Conclusão de Curso) - Universidade Federal do Paraná, Curitiba - PR, 2010.

PRESSMAN, Roger S.. Prática de Engenharia de Software: Princípios da modelagem de análise. 6ª Edição Brasil: Mcgraw-hill Interamericana, 2006. 752p.

SCHWABER, Richard. Processo: uma visão genérica. 6ª Edição Brasil: Mcgraw-hill Interamericana, 2002. 752 p.

SILVA, Luiz da; RICARDO NETO,; SANTOS, Mateus. Modelos evolucionários de processo de software. São Paulo: Makson, 2012. 350 p.

SILVEIRA, Luciano Pereira. Métricas de Acompanhamento para Metodologias Ágeis: Metodologias ágeis. 2012. 91 f. Tcc (Trabalho de Conclusão de Curso) - Departamento de Centro De Ciências Exatas E Tecnologia, Universidade Federal De São Paulo, São Paulo - SP, 2012.

TENÓRIO, Ludmila Varela Arruda. DESENVOLVIMENTO ÁGIL DE SOFTWARE: UMA ANÁLISE SINTÉTICA A PARTIR DA METODOLOGIA KANBAN. Desenvolvimento Ágil, Palmas - To, n., p.4-5, 10 maio 2008.

ABREU, Antônio Cesar. Enfoque no gerenciamento de projetos de software. 2006. 187 f. Dissertação (Mestrado) - Departamento de Administração da Faculdade de Economia, Universidade de São Paulo, São Paulo, 2006.

ABREU, Roberto. Sugestão e modelagem de prática de desenvolvimento ágil para aplicação em desenvolvimento distribuído: OnshoreInsourcing. 2008. 92 f. Tese (Mestrado) - Departamento de Instituto de Física e Matemática Departamento de Informática, Universidade Federal De Pelotas, Pelotas, 2008.

AGILE MANIFESTO (Org.). Princípios por trás do Manifesto Ágil. Disponível em: <http://agilemanifesto.org/principles.html>. Acesso em: 12 out. 2013.