RESIGeT On-Line Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica Uni-FACEF CENTRO UNIVERSITÁRIO DE FRANCA
05 Nr. 01 - 2015

SCRUM: UM GUIA PRÁTICO NO GERENCIAMENTO DE PROJETOS

Salmo Roberto Da Silva Junior[1]
salmo@ciandt.com
Daniel Facciolo Pires[2]
daniel@facef.br
Silvio Carvalho Neto[3]
silvio@facef.br

RESUMO

Este trabalho apresenta um guia conceitual de como se utilizar o *framework Scrum* em projetos de desenvolvimento de software que foi realizado com base na experiência do autor e através de pesquisas bibliográficas. Inicialmente são apresentados os conceitos das metodologias ágeis com seus valores e porque elas se diferenciam dos métodos tradicionais. Em seguida é explicado o *framework Scrum*, apresentando seus ritos, personagens com suas funções e mostrando como ele permite flexibilidade no gerenciamento de escopo de projeto. Por fim são apresentados alguns passos para facilitar a implantação do *Scrum* e sua adoção por uma equipe. Ao final do estudo foi possível identificar os benefícios do uso deste *framework*.

Palavras-chave: Metodologia ágil, Scrum, software, flexibilidade, escopo.

ABSTRACT

This paper shows a conceptual guide of how to use Scrum framework on software development projects that was accomplished based on the author's experience and through bibliographical researches. First it shows the agile methodologies concept with its values and the main reasons of why they differ from traditional methodologies. Next it details the Scrum framework, showing its rites, roles and how it allows to gain flexibility on project scope management. Last but not least the paper describes the steps to make the Scrum implementation and its adoption by a team an easy task. At the end of the study was possible to identify the benefits of using this framework.

Keywords: Agile Methodology, Scrum, software, flexibility, scope.

1. Introdução

Os projetos de desenvolvimento de software estão sujeitos a constantes mudanças em seus requisitos durante a execução, o que torna o desenvolvimento de software um desafio. As técnicas de desenvolvimento ágil de aplicações nos auxiliam a lidar com estes desafios.

[1] CI&T Brasil – Discente do curso de Pós-Graduação em Sistemas de Informação do Uni-FACEF
[2] Docente do curso de Pós-Graduação em Gestão e Desenvolvimento de Sistemas Web do Uni-FACEF
[3] Docente do curso de Bacharelado em Sistemas de Informação do Uni-FACEF

As metodologias ágeis de projetos têm por objetivo aumentar a satisfação do cliente, produzindo *software* com alto grau de qualidade e com maior cumprimento dos prazos.

Neste contexto, destaca-se o *Scrum*, criado por Ken Schwaber e Jeff Sutherland que por estarem ligados diretamente ao desenvolvimento de *software* foi a área que o *Scrum* adentrou com maior facilidade, entretanto o *framework* pode ser aplicado em diversas outras área.

O *Scrum.org* apresenta uma maneira para a equipe trabalhar em conjunto no desenvolvimento de *software*. O *Scrum* apoia a necessidade do ser humano no trabalho: pertencer, aprender, fazer, criar, melhorar e interagir com outras pessoas.

Neste sentido, o objetivo deste artigo foi mostrar o funcionamento do *Scrum* em projetos de desenvolvimento de *software*, ressaltando os conceitos e benefícios deste *framework*. O estudo foi realizado por meio de levantamento bibliográfico e estruturado na seguinte ordem: inicialmente foi apresentado o conceito de metodologia ágil e em seguida foi realizado um entendimento do *Scrum* apresentando os atores, artefatos e eventos, por fim, foi apresentado um exemplo conceitual de como implantar o *Scrum* baseado na experiência do autor.

2. Metodologias Ágeis

Apesar de existir a um bom tempo, apenas recentemente a expressão "Desenvolvimento Ágil" vem se tornando mais popular no Brasil por usar uma abordagem simplificada, onde simplicidade significa ter agilidade e fazer a diferença, e ao contrário do que parece, exige muita disciplina e organização.

A abordagem ágil aplicada ao desenvolvimento de projetos ficou mais clara e melhor definida a partir de 2001, quando dezessete especialistas em processo de desenvolvimento de *software* representando várias tendências da época como XP (eXtreme Programming), Scrum, *Dynamic System Development Method* (DSDM), *Crystal, Feature Driven Development* (FDD) e *Lean Development*, se reuniu para discutir e identificar o padrão de desenvolvimento de projetos dentre as técnicas e metodologias existentes. O resultado deste encontro foi a criação dos princípios de um manifesto

para o desenvolvimento ágil, conhecido como “Manifesto Ágil”. Os conceitos chave do “Manifesto Ágil” são: (SOARES, 2004).

- Indivíduos e interações em vez de processos e ferramentas;
- *Software* executável em vez de documentação;
- Colaboração do cliente ao invés de negociação de contratos;
- Respostas rápidas a mudanças em vez de seguir planos;

O que diferencia as metodologias ágeis das metodologias tradicionais são os enfoques e os valores. Onde o enfoque são as pessoas e não os processos ou algoritmos, além de existir a preocupação em gastar menos tempo com documentação e mais com a implementação (COCKBURN, et al., 2001).

Compreender os valores do Manifesto Ágil traz novas sugestões para a melhoria de métodos, processos e técnicas de desenvolvimento e gestão de projetos, com isto ser ágil traz vantagens como:

- Flexibilidade para mudança de requisitos e prioridades além de maior agilidade na tomada de decisões;
- Entregas frequentes e regulares do produto;
- Melhoria da qualidade final do produto;
- Transparência e visibilidade do status do projeto;
- Maior produtividade e consequentemente maximização do comprometimento;
- Redução de riscos, com antecipação dos problemas e possibilidade de reprioriização do escopo durante o desenvolvimento;
- Valoriza a satisfação do cliente em primeiro lugar;

Pesquisa realizada entre dezembro de 2012 e janeiro de 2013 na Dr. Dobbs por Scott Ambler aponta que 91% das equipes ágeis oferecem valor para as partes interessadas em uma base regular, veja Figura 1. Este calculo foi realizado consideran-

do que toda a equipe alegou que para produzir software trabalharam em todas as *Sprints* ou tiveram uma ou mais interações sem produzir software no início.

3. Introdução ao Scrum

O *Scrum* é bastante objetivo, com papéis definidos, de fácil adaptação e ainda, sua curva de aprendizado é relativamente fácil. É voltada para as pessoas no projeto, e não a tecnologia. Sustentado pelos pilares da transparência, inspeção e adaptação o *framework* ganhou o mundo, desbancou métodos tradicionais e se tornou a forma mais comum de se trabalhar em projetos de desenvolvimento de *software*. *Scrum* ou variantes do *Scrum* foi apontado como a ferramenta ágil mais utilizada pelos participantes de uma pesquisa realizada em 2012 (VERSIONONE, 2012).

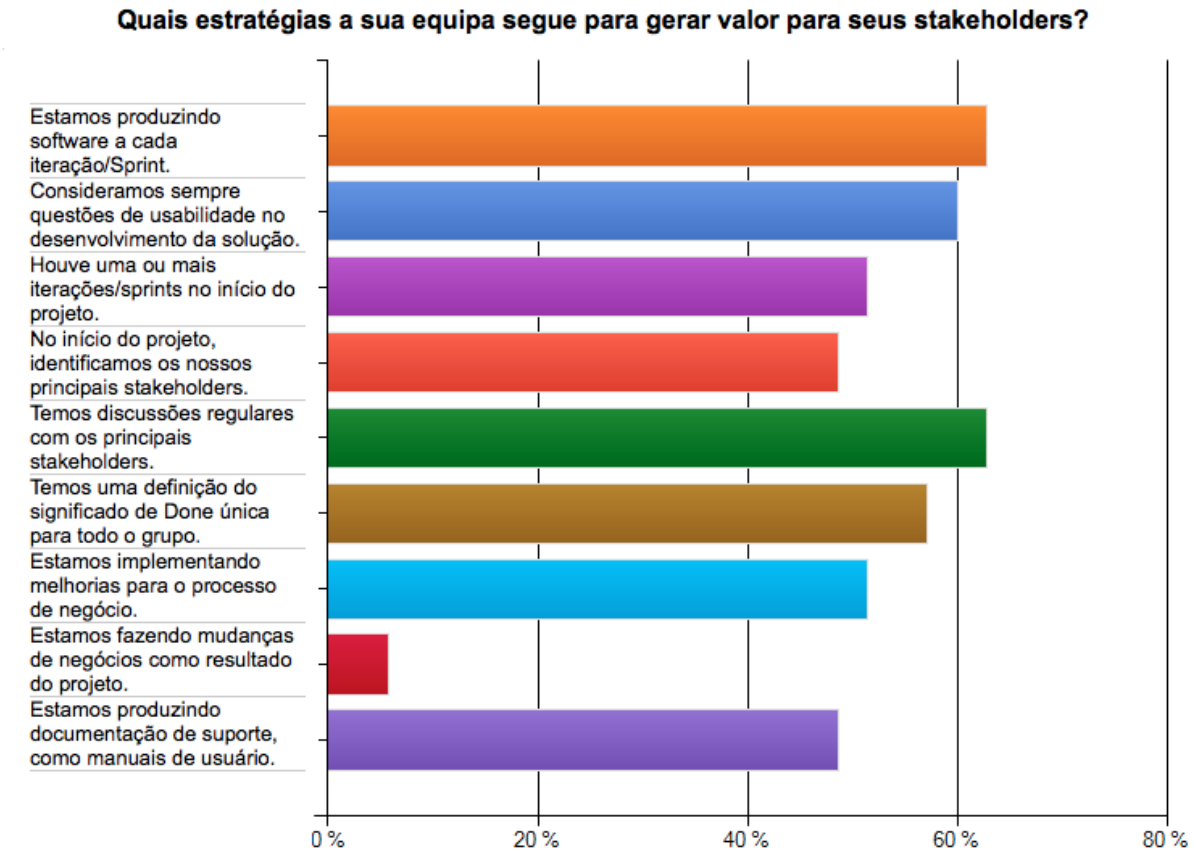

Figura 1– Valores para as partes interessadas
Fonte: Ambler, 2013

Scrum não é um processo previsível, ele não define o que fazer a toda circunstância. É uma ferramenta, um *framework* - um conjunto de práticas que torna tudo visível, o que permite aos participantes saber exatamente o que está acontecendo ao longo

do projeto e com que a equipe tome as decisões ao longo do tempo visando alcançar seus objetivos.

Apesar de ter sido concebido para projetos de *software*, seu uso não se limita a este tipo de projeto, e hoje é utilizado com sucesso por organizações de diversos tamanhos e tipos. A adoção mundial do *Scrum* não significa que todos os problemas estão resolvidos, longe disso. Porém, se bem utilizada pode permitir reduzir os riscos de insucesso, entregar valor mais rápido e desde cedo, e lidar com as inevitáveis mudanças de escopo, transformando-as em vantagens competitivas (SABBAGH, 2013).

O *Scrum é b*aseado em ciclos de 15 a 30 dias chamados de *Sprints*, onde se trabalha para alcançar objetivos bem definidos. Esses objetivos são definidos em uma lista que é constantemente atualizada e priorizada pelo cliente, esta lista é chamada de *Product Backlog*.

A Figura 2 ilustra o ciclo de desenvolvimento do *Scrum* de forma simplificada. No início de cada *Sprint*, faz-se um *Sprint Plannig Meeting*, ou seja, uma reunião de planejamento na qual o *Product Ower* compila todas as mudanças previstas para o produto e prioriza as funcionalidades possíveis, que ficam no chamado Product Backlog. O *Product Backlog* é uma lista de tarefas que está em constante alteração, dentre as tarefas a equipe seleciona as atividades que ela será capaz de desenvolver durante o *Sprint* que se inicia. Antes do inicio de um *Sprint* os objetivos priorizados são transferidos para um *Sprint Backlog*.

A cada dia de uma *Sprint*, são realizas as *Daily Scrum*, uma breve reunião onde é discutido o que foi feito no dia anterior, se existe algum bloqueio que esteja impedindo a equipe de concluir alguma tarefa e quais novos passos de cada membro. O que um programador se comprometer a fazer naquele *Sprint* deve ser cumprido, e na próxima reunião lhe será cobrado, podendo ser ajudado por outro membro da equipe caso necessário, usando técnicas como *Pair Programming*.

Scrum é ideal para projetos suscetíveis a mudanças de requisitos, no entanto é necessário entender seus papéis, responsabilidades, conceitos e artefatos das fases de seu ciclo antes de aplicá-lo.

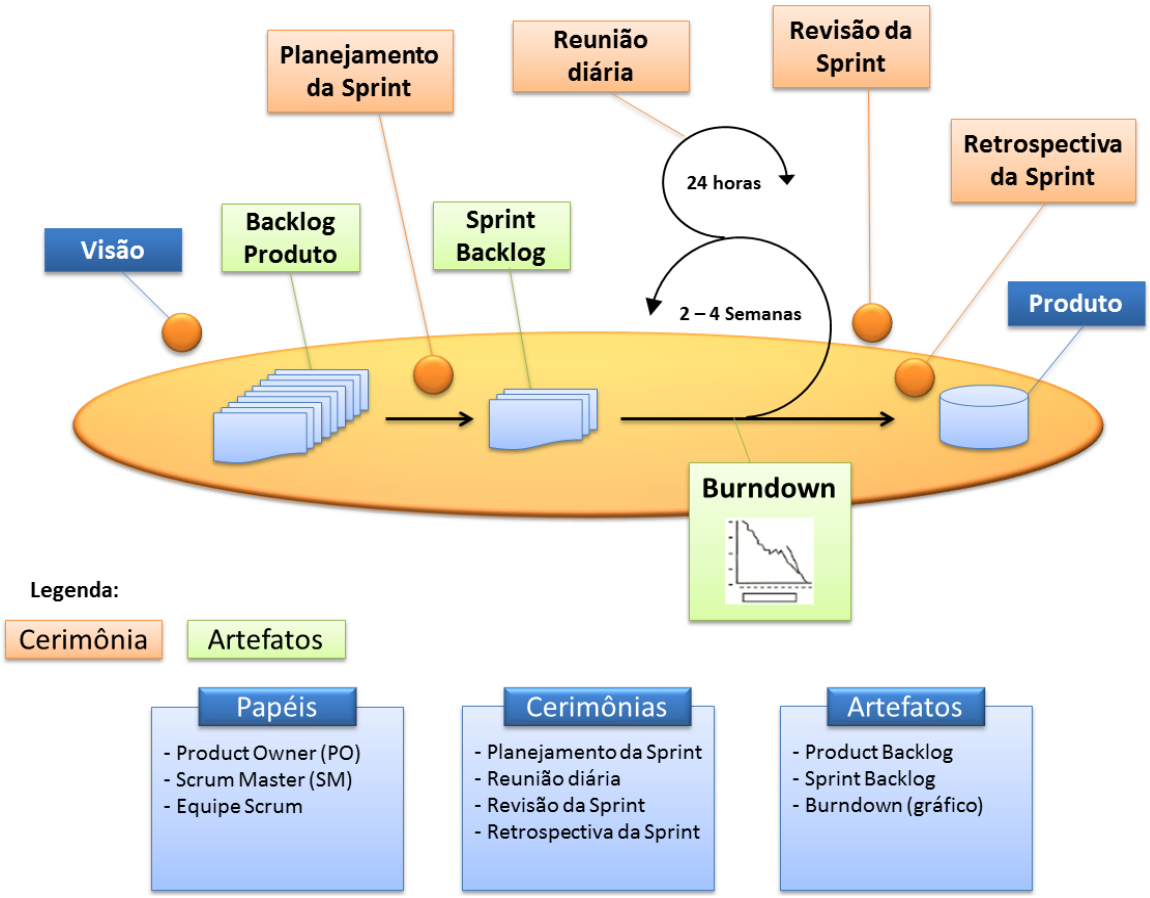

Figura 2 - **Ciclo de Vida Scrum**
Fonte: **Chagas, 2011**

No final do *Sprint* demonstra os resultas para o *Product Ower,* de forma que os itens do *Backlog* sejam considerados prontos pode-se iniciar um novo *Sprint.*

4. Os papéis no *Scrum*

Segundo Sabbagh (2013) há três papéis no *Scrum*: *Product Owner,* o *ScrumMaster* e o *Scrum Team* descritos abaixo:

- *Product Owner*: responsável pelo retorno financeiro (ROI) do projeto, e também por definir, comunicar e manter a Visão do Produto relativamente constante ao longo do projeto. Pode mudar os requisitos e prioridades, aceitar e rejeitar o resultado de cada *Sprint.*

- *ScrumMaster:* garante que o time esteja totalmente funcional e produtivo, atuando quando necessário como um agente de mudança na organização. O *ScrumMaster* está presente e age como um facilitador em todas as reuniões, revisa as *sprints* e planejamento. Facilita a colaboração entre as funções e áreas e elimina os impedimentos do time.

- *Scrum Team:* a equipe deve ser suficientemente pequena, de forma geral se recomenda de 3 a 9 membros. O time é responsável por priorizar os itens que serão executados dentro do *Sprint* de maneira a produzir valor visível no final de cada *Sprint,* para que se possa obter *feedback* dos clientes e demais partes interessadas nas reuniões de *Sprint Review.* Devem ser capazes de se comunicar efetivamente para se auto-organizarem.

Ponto-chave a ser gerenciado	Product Owner	Time de Desenvolvimento	ScrumMaster
Retorno sobre o investimento	X		
Necessidades/objetivos de negócios	X		
Clientes e demais partes interessadas	X		
Visão do Produto	X		
Releases	X		
Tarefas de desenvolvimento do produto		X	
Qualidade interna do produto		X	
Qualidade externa do produto	X	X	
Estimativas ou previsões		X	
Processos (funcionamento do Scrum)			X
Impedimentos no trabalho			X
Relacionamento e motivação do Time		X	X
Riscos	X	X	X
Comunicação	X	X	X

Tabela 1: Pontos-chave a ser gerenciado
Fonte: SABBAGH, 2013

Nos projetos que utilizam a metodologia *Scrum* não há um Gerente de Projetos como é tradicionalmente definido, o trabalho de gestão é distribuído pelos seus três papéis (SABBAGH, 2013).

A tabela 1 ilustra de maneira sintética alguns pontos-chave cujo gerenciamento cabe a diferentes papéis do *Scrum:*

Sabbagh (2013) ainda acrescenta que na prática esta divisão significa que questões das áreas de conhecimento de Projetos como escopo, tempo, custo, riscos, qualidade e recursos humanos não são centralizadas em uma única pessoa. Ao contrário elas são dividias entre o Product Owner, Scrum Master e o Scrum Team onde cada um em sua função trabalha de maneira colaborativa buscando sempre a satisfação dos clientes do projeto.

5. Artefatos do *Scrum*

O *Scrum* define o uso de quatro artefatos: o *Product Backlog,* o *Sprint Backlog,* a Definição de Pronto e o Incremento do Produto.

5.1. *Product Backlog*

O *Product Backlog* é uma lista contendo todas as funcionalidades desejadas para um produto, cujo único responsável é o *Product Owner*. A lista não precisa estar completa no início do projeto, em cada momento ela é atualizada e priorizada de acordo com a importância para os clientes do Projeto.

O *Product Backlog* inclui tudo que possa ser associado com valor de negócio para a finalização do projeto. Portanto, pode conter melhorias a serem realizadas no produto, correções de problemas, questões técnicas, ou seja, tudo que pode vir a ser desenvolvido para se alcançar a Visão do Produto é adicionado à lista.

Sabbagh (2013) afirma que para que o *Product Owner* consiga gerenciar o *Product Backlog* os itens precisam estar identificados e estimados, em tempo ou tamanho, e sua ordem de importância deve ser estabelecida pelo cliente de maneira a orientar o time de desenvolviemtno. A ordenação dos itens deve ser constante, este trabalho não é fácil e exige conhecimento do negócio, das oportunidades e riscos do mercado, e uma série de outros fatores. Seu objetivo é maximizar o retorno ao investimento (ROI) realizado ao longo de todo projeto.

> A melhor e menos complexa forma de do *Product Backlog* maximizar o ROI dos clientes do projeto é manter o foco em entregar para esses clientes funcionalidades que eles mais necessitam em cada momento do projeto e obter o *feedback* o mais cedo possível sobre essas funcionalidades para, assim, realizar as mudanças necessárias e reduzir os riscos do projeto. (SABBAGH, 2013, p. 115).

Uma das técnicas mais conhecidas e utilizadas por times Ágeis para realizar a estimativa dos itens do *Product Backlog* é a *Planning Poker,* criada por James Grenning.

Segundo Grenning (2002) a mecânica do *Planning Poker* é bem simples e se resume nos seguintes passos:

- Cliente lê um item da lista;
- Os membros do Time de Desenvolvimento discutem brevemente para esclarecer possíveis dúvidas;
- Cada membro escreve sua estimativa para o trabalho necessário em uma carta, sem mostra a carta imediatamente;
- Todos mostram as cartas ao mesmo tempo;
- Se houver divergência na estimativa os membros conversam entre si até chegarem a um consenso;

5.2. *Sprint Backlog*

O *Sprint Backlog* é uma lista de itens selecionados no alto do *Product Backlog* que o *Scrum Team* se compromete a fazer em um *Sprint.* A lista de itens é escolhida pelo time e negociada com o *Product Owner* na reunião de *Sprint Planning.*

O *Product Backlog* deve ser bem ordenado, onde terá em seu topo itens que juntos levam a alcançar a necessidade do negócio. Esta necessidade alinhada com a capacidade da equipe se transformará na Meta da *Sprint,* cujo resultado é a soma dos itens completos no *Sprint*, ou seja, o Incremento do Produto.

Cada item do *Sprint Backlog* será dividido em conjuntos de tarefas que representam pequenos passos que são necessários para que o item esteja pronto. A estimativa de horas para cada tarefa pode ser adicionada para que o time possa acompanhar

seu progresso em direção ao final do *Sprint,* porém mesmo que não haja o detalhamento de horas por tarefa a *Sprint* tem uma duração estabelecida, e não se deve alterar sua data final, pois todo o projeto é guiado por essa duração.

O time possui um quadro de trabalho onde organiza as atividades dos itens do *Sprint Backlog* separando-as em quatro fases: para fazer, em andamento, para verificar e concluído. Este quadro é do time, portanto são eles os responsáveis pela manutenção do mesmo.

5.3. Definição de Pronto

A Definição de Pronto é um acordo formal entre o time e o *Product Owner* onde todos possam entender o que "Pronto" significa. Todos os integrantes do projeto devem ter o mesmo entendimento do que significa o trabalho estar completo no incremento do produto.

A mesma definição de pronto deve ser utilizada em cada item da *Sprint* e orienta o time no conhecimento de quantos itens do *Product Backlog* podem selecionar para a mesma *Sprint*.

De acordo com Sabbagh (2013) uma definição de pronto ideal estabelece que o resultado de um *Sprint* seja um entregável, ou seja, nenhum desenvolvimento, teste ou qualquer tarefa adicional para que o Incremento do Produto produzido possa ser entregue ao cliente sem restrições.

A definição de pronto é criada antes do início do desenvolvimento e pode ser modificada ao longo do projeto e/ou desenvolvimento de acordo com as necessidades.

5.4. Incremento do Produto

O incremento do produto é a soma de todos os itens completos do *Sprint,* ou seja, todos os itens que foram selecionados para o *Sprint Backlog*, do mais importante para o menos importante visando atingir a Meta do *Sprint.*

Espera-se que o incremento do produto seja um entregável de acordo com as regras definidas na Definição de Pronto, de forma que o *Product Owner* possa decidir realizar um *Release* para o cliente ao final da *Sprint.*

6. Eventos do *Scrum*

Os eventos do *Scrum* são o próprio ciclo de desenvolvimento, chamado de *Sprint* assim como as reuniões e cerimônias realizadas durante o ciclo.

Os eventos do *Scrum* possuem uma duração definida, chamada de *Time Box* onde se define o tempo máximo ou exato em que um evento deve ocorrer.

6.1. *Sprint*

No *Scrum,* os projetos são divididos em ciclos baseados em uma série de interações, chamados de *Sprint*. O *Sprint* representa um *Time Box* dentro do qual um conjunto de atividades deve ser executado, e o qual se recomenda durar entre 2 a 4 semanas. Uma nova *Sprint* inicia imediatamente após a conclusão da *Sprint* anterior.

Os *Sprints* são compostos por diversas reuniões além do desenvolvimento propriamente dito, como *Sprint Planning, Daily Sprint, Sprint Review* e *Sprint Retrospective.* Todos estes eventos estão dentro do *timebox* do *Sprint.*

Cada *Sprint* pode ser considerada como um projeto que não deve durar mais que um mês, cujo objetivo deve ser bem definido e negociado entre o time e o *Product Owner.* É através do *Sprint* que é possível inspecionar o progresso em direção à meta pelo menos a cada mês ocorrido.

6.1.1. Cancelamento do *Sprint*

Caso julgue que a meta do *Sprint* se tornou obsoleto seja por condições do mercado e/ou tecnologias, ou mesmo a direção da organização mudar o *Product Owner* pode cancelar um *Sprint.*

Quando um *Sprint* é cancelado e caso haja incremento do produto, este deve ser revisado, antecipando a reunião de *Sprint Review*, assim como a *Sprint Retrospective*. Em seguida, um novo *Sprint* será iniciado.

6.2. *Sprint Planning Meeting*

O *Sprint Planning Meeting* é uma reunião entre o time e o *Product Owner,* no qual o *Scrum Master* atua como um facilitador. A reunião tem por objetivo planejar o ciclo de desenvolvimento (*Sprint*) que se inicia.

Durante a reunião o *Product Owner* descreve os itens de maior prioridade para a equipe, e determinam em conjunto o que será desenvolvido e qual a meta deste *Sprint*. Posteriormente, o time define como os itens serão desenvolvidos, construindo então o *Sprint Backlog*.

6.3. *Daily Scrum*

Durante a *Sprint,* o time, controla como as tarefas devem ser executadas e não deve haver interferência externa durante este período. Este é um dos principais papéis do *Scrum Master*, blindar o time de qualquer desvio do objetivo traçado.

O acompanhamento do progresso da *Sprint* é realizado através do *Daily Meeting* ou *Daily Scrum,* que são reuniões curtas e realizadas diariamente onde participam os membros do time de desenvolvimento e o *Scrum Master.* Membros externos ao projeto podem participar, mas só poderão escutar.

Durante o *Daily Scrum* cada membro da equipe responde cada uma das perguntas abaixo:

- O que você fez ontem?
- O que você fará hoje?
- Há algum impedimento no seu caminho?

O objetivo do *Daily Scrum* é disseminar conhecimento sobre o que foi feito no dia anterior, identificar impedimentos e priorizar o trabalho a ser realizado no dia que se inicia, portanto não deve ser utilizada para resolução de problemas. Todo e qualquer problema encontrado durante o *Daily Scrum* deve ser tratado em outra reunião, geralmente com grupo menor.

.A alocação de recursos para cada tarefa da *Sprint* é determinada pela própria equipe, cada membro seleciona a tarefa que pode executar e o time estabelece a ordem de dependência entre elas, o que torna a equipe auto-gerenciável.

Após alocado os recursos cada membro deve reportar as horas que serão gastas para execução de sua tarefa, para que o valor de horas restantes seja calculado corretamente e o time possa verificar seu progresso, que é acompanhado pelo Gráfico de Sprint Burndown.

O *Burndown* não é parte integrante do *Scrum*, mas é a maneira mais prática e rápida de visualizar o andamento da *Sprint*, por isto é tão utilizado. Ele mostra o consumo de horas diário, onde o eixo X indica a escala de horas totalizando o valor de horas estimado para a *Sprint,* e o eixo Y o trabalho restante estimado para um *Sprint.* A figura 3 mostra um exemplo de *Sprint Burndown* ao final de um *Sprint* bem sucedido.

Se houver oscilações na soma das horas para cima indica que algumas estimativas foram erradas ou novas tarefas foram adicionadas. Estas oscilações são perigosas e devem ser monitoradas diariamente.

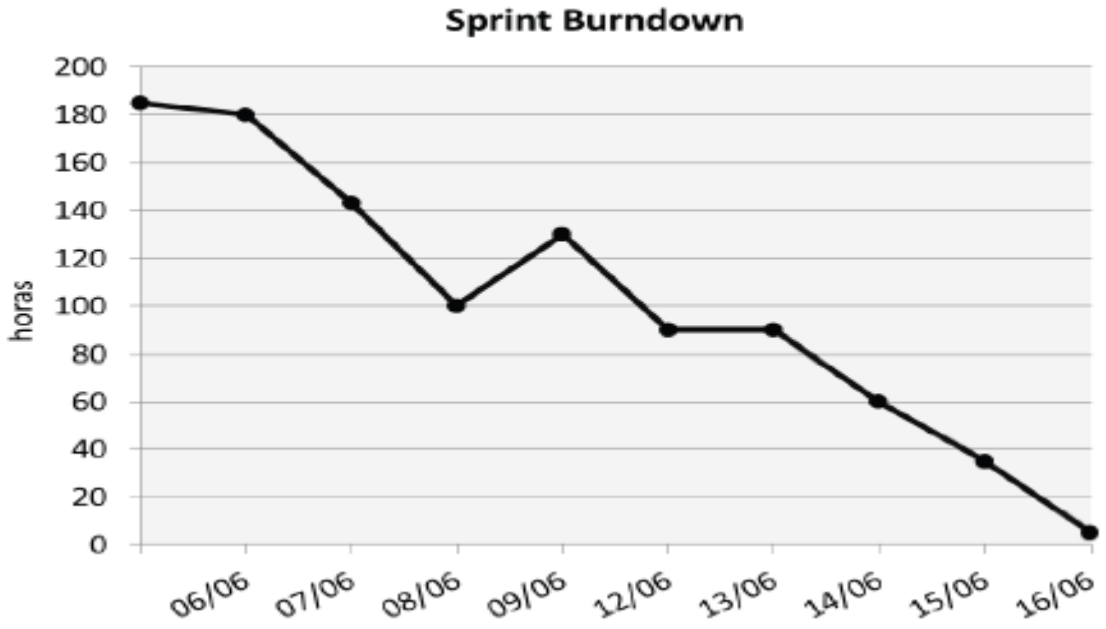

Figura 3: Burndown ao final de um *Sprint* bem sucedido
Fonte: SABBAGH, 2013

6.4. *Sprint Review* e *Sprint Retrospective*

Ao final do *Sprint* o time deverá ter gerado um Incremento do Produto entregável, que representa valor visível ao cliente. No último dia do *Sprint* o time se reúne para duas reuniões consecutivas de inspeção e adaptação, ambas facilitadas pelo *Scrum Master.* A primeira reunião é chamada de *Sprint Review*, onde é realizada a inspeção e adaptação do produto para verificar se o mesmo está como foi pensando e conta com a presença do *Product Owner*, enquanto a reunião de *Sprint Retrospective*, podendo ser chamada também de "lições aprendidas" onde o time levanta tudo de bom e ruim que aconteceu durante a execução do *Sprint* e procura estabelecer pontos de melhoria. Essas ações são levadas para o próximo *Sprint* a fim de melhorar o processo e/ou produto.

O ciclo do *Scrum* é repetido até que se tenha finalizado todos os itens do *Product Backlog*.

7. Como implantar o *Scrum*

Por ser um *framework* o *Scrum* não possui padrões, processos a serem seguidos para sua implantação. O *Scrum*.org prega a seguinte definição:

> "*More specifically, Scrum is a simple framework for effective team collaboration on complex projects. Scrum provides a small set of rules that create just enough structure for teams to be able to focus their innovation on solving what might otherwise be an insurmountable challenge. However, Scrum is much more than a simple framework. Scrum supports our need to be human at work: to belong, to learn, to do, to create and be creative, to grow, to improve, and to interact with other people. In other words, Scrum leverages the innate traits and characteristics in people to allow them to do great things together*"

Um bom começo para a implantação do Scrum é apresentar o framework a toda equipe, para que todos saibam como funciona e o que eles terão de acrescentar no seu dia a dia de trabalho.

É importante que a equipe seja comprometida e saiba receber *feedback*, pois as falhas aparecem durante o projeto, e não é possível esconde-las. O papel do *Scrum Master* é crucial nesta fase de implantação e o mais difícil é conseguir disseminar o processo e se preocupar com as pessoas ao mesmo tempo.

Para o sucesso do projeto é muito importante à escolha do *Product Owner*, ele deve ser uma pessoa que entenda perfeitamente as regras e necessidades do cliente e é muito importante que ele mantenha o *Product Backlog* sempre priorizado.

Após o treinamento da equipe é preciso reuni-los para montar o *Product Backlog,* criar as histórias e ordena-las de acordo com as prioridades para então criar o primeiro *Sprint.* Fechada a primeira reunião de *Scrum Planning*, deve ser implantado o sistema de acompanhamento do *Sprint*, uma sugestão é a adoção do Quadro de [4]*Kanban*. A Figura 4 mostra um exemplo do *Kanban* que deve estar em lugar visível para todo o time.

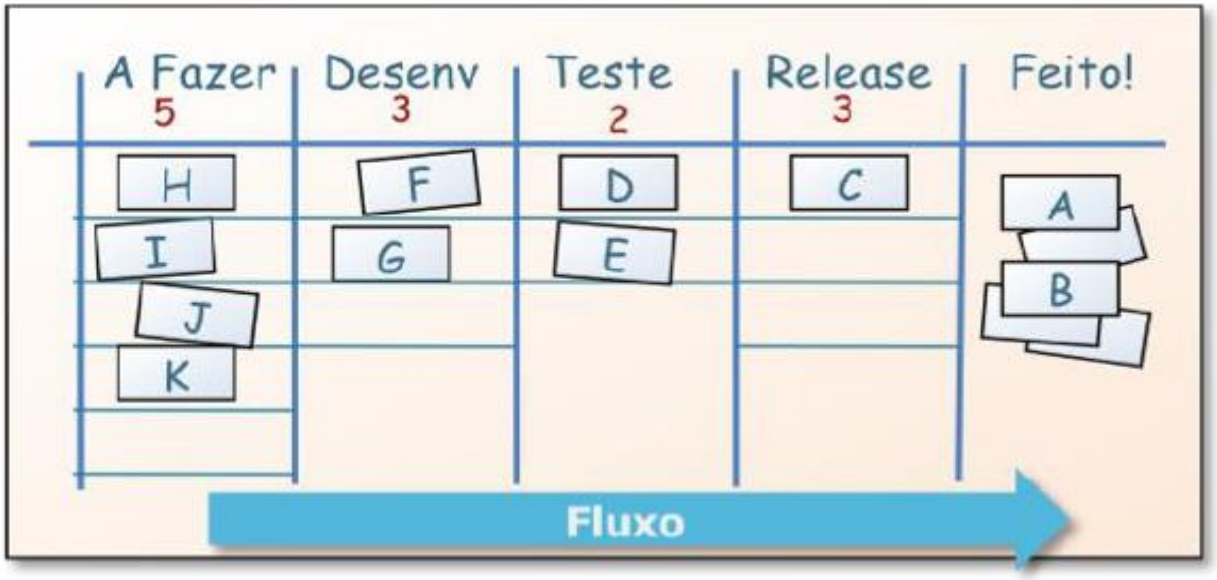

Figura 4: Quadro de Kanban

Fonte: Kniberg et al. (2009)

Outro passo a ser seguido é a adaptação do time à rotina de *Daily Scrum*, no inicio o time pode relatar suas atividades voltado ao *Scrum Master* e não ao time parecendo uma reunião de *Status Report*, isto deve ser evitado. Também deve ser observado a restrição de tempo e diminuir as interferências externas a equipe.

Após o encerramento de cada *Sprint* devem ser realizadas as reuniões de *Sprint* Review e *Sprint Retrospective* sucessivamente.

[4] Quadro de Kanban: quadro de sinalização que controla os fluxos de produção.

Existem indicações de tempo para cada rito, os chamados *time box*, porem cada equipe pode adapta-los, com tanto que não seja perdido o foco e o propósito de cada rito. Os *time box* devem ser testados nas primeiras *Sprints*, mas após definidos devem ser fixos para todas as outras *Sprints* seguintes. É essencial que todos os ritos sejam seguidos e que todos se comprometam participando ativamente, afinal uns dos pontos forte do *Scrum* é a colaboração que ele proporciona entre todos da equipe, servindo não só para o bom desenvolvimento do projeto, mas também como fator motivacional, pôs faz com que todos se sintam importantes para a conclusão do projeto.

Como os resultados vão aparecendo e sendo aprimorados no decorrer das primeiras *Sprints*, é importante que se tenha paciência na fase de implantação e que não se desista nos primeiros erros.

8. Conclusão

Neste trabalho foram abordados os objetivo das metodologias ágeis e foi apresentado também um modelo de como se implantar uma metodologia em um ambiente de desenvolvimento de *software*. Dentre as várias metodologias existentes e utilizadas hoje no mercado, foi escolhido para este trabalho o *Scrum*, por sua grande mudança de paradigma de como se desenvolver *software* e pela grande aceitação que vem ganhando entre as empresas de desenvolvimento no mercado do mundo todo.

O Scrum possuí hoje duas organizações principais, a *Scrum.org e a Scrum Alliance*, que padronizam e mantém ele sempre atualizado, fornecendo material de estudo, certificações e fazendo reuniões entre os membros mais participativos para mater o framework em constante aprimoramento.

Mesmo com toda organização, documentação e sendo o *Scrum* formado de conceitos básicos e de não ser necessário muito tempo para compreende-lo por completo, o Scrum é difícil de ser aplicado por completo, porque depende de mudanças na forma de pensar e de trabalhar de toda a equipe, sem essa aceitação e mudança é praticamente impossível sua implantação.

Afim de ajudar neste processo, este trabalho apresentou um manual com os passos para a implantação do *Scrum*. Foram detalhados os ciclos de interações, os atores com seus papéis e como cada membro deve agir para que tudo corra como o esperado. O Scrum pode levar algumas *sprints* para que a equipe comece a perceber os resultados e a conhecer o seu real nível de rendimento, se tornando assim mais auto-gerenciável e estando apta a trabalhar de forma agíl.

Referências

AMBLER, Scott. How Agile Are You? 2013 Survey Results. Disponível em: <http://www.ambysoft.com/surveys/howAgileAreYou2013.html>. Acesso em 19 out. 2013.

CHAGAS. J. D. E. Gestão de Projetos Ágeis. Disponível em http://danielettinger.files.wordpress.com/2011/04/engrenagem-do-scrum1.png. Acesso em 03 nov 2013.

COCKBURN, A. e HIGHSMITH, J. "Agile Software Development: The Business of Innovation", IEEE Computer, Set., pp. 120-122, 2001.

GRENNING, J. Planning Poker or how to avoid analysis paralysis while release planning. 2002. Disponível em <http://renaissancesoftware.net/files/articles/Planning Poker-v1.1.pdf> . Acesso em 03 nov. 2013.

KNIBERG, H. e SKARIN, M. **Kanban e scrum obtendo o melhor de ambos**. C4Media. Estados Unidos, 2009

SABBAGH, Rafael. **Scrum gestão ágil para projetos de sucesso**. Caso do Código. São Paulo, 2013

SOARES, Michel Comparação entre Metodologias Ágeis e Tradicionais para o Desenvolvimento de Software. Disponível em: <http://www.dcc.ufla.br/infocomp artigos/v3.2/art02.pdf> Acesso em: 20 out. 2013.

SCRUM org Disponível em <https://www.scrum.org/> Acesso em: 05 nov. 2013

DESENVOLVIMENTO ágil Disponível em:< http://danielettinger.files.wordpress.com/2011/04/engrenagem-do-scrum1.png >Acesso em: 19 out. 2013

VERSION one Disponível em:< http://www.versionone.com/pdf/7th-Annual-State-of-Agile-Development-Survey.pdf > Acesso em: 19 out. 2013