

# Blog do Abu - Apostila de Apoio

**Abuzitos** <http://www.abuzitos.com.br/>

**Blog do Abu** <http://blogdoabu.blogspot.com/>

Versão 2.1 – Em fase de correção ortográfica e técnica.

Prof. Nelson Abu Samra Rahal Junior – abuzitos@gmail.com  
Certified SCRUM Master

## Contents

|  |    |
|--|----|
| Autores .....  | 4  |
| Nelson Abu Samra Rahal Junior.....   | 4  |
| Neilza Andrea Abu Samra Rahal.....   | 4  |
| Scrum .....  | 5  |
| Você sabe o que é Scrum?.....  | 5  |
| Framework Scrum.....   | 7  |
| Papéis no Scrum.....   | 8  |
| Product Owner (PO).....  | 8  |
| Scrum Máster (SM).....   | 8  |
| Equipe (EQ) .....  | 9  |
| Visão do Produto .....   | 11 |
| Executando a Visão do Produto.....   | 11 |
| Exemplo de Visão do Produto .....  | 12 |
| Product Backlog.....   | 13 |
| Exemplo de Product Backlog .....   | 13 |
| Estimativas, Velocidade da Equipe, Quantidade de Sprint's e Product Burndown Chart.....                        | 15 |
| Contando os Pontos.....  | 16 |
| Exemplo de Pontos Contados .....   | 17 |
| Determinando o Tempo do Projeto.....   | 17 |
| Product Burndown Chart.....  | 18 |
| Priorização e Distribuição de Requisitos por Sprint's .....  | 20 |
| Executando a Priorização .....   | 20 |
| Sprint, Planejamento da Sprint #1, Planejamento da Sprint #2, Sprint Burndown Chart e Execução da Sprint ..... | 22 |
| Planejamento da Sprint #1 .....  | 23 |
| Planejamento da Sprint #2.....   | 24 |
| Sprint Burndown Chart .....  | 24 |
| Calculo de Velocidade da Sprint.....   | 25 |
| Exemplo de Tarefas na Sprint.....  | 25 |
| Exemplo de Ciclo de Vida de Uma Sprint .....   | 26 |
| Executando a Sprint .....  | 28 |
| Desenvolvimento e Reuniões Diárias .....   | 30 |
| Sete Fundamentos para reuniões diárias eficazes .....  | 30 |
| Impedimentos.....  | 31 |
| Dia a Dia da Reunião Diária.....   | 31 |
| Review.....  | 32 |
| Retrospectiva .....  | 33 |
| Linha do Tempo (Timeline).....   | 33 |
| Retrospectiva Rápida.....  | 37 |
| Dia a Dia Após a Retrospectiva .....   | 38 |
| Release (Versões).....   | 39 |
| Scrum de Scrum.....  | 41 |
| Potencializando Scrum.....   | 43 |
| Épicos, Temas e Historias .....  | 45 |
| Cartão de Historia .....   | 45 |
| Exemplo de Cartão de Historia .....  | 45 |
| Template de Cartão de Historia .....   | 45 |
| Organizando Cartões de Historia .....  | 46 |

|   |    |
|---|----|
| Exemplos de Cartão de Historia .....                      | 48 |
| Escopo do Projeto .....                                   | 49 |
| Épico, Tema e Historia.....                               | 50 |
| Trabalhando com Temas .....                               | 51 |
| Exemplos de Testes criados para Cartões de História ..... | 52 |
| Templates de Documentos .....                             | 55 |
| Visão do Produto .....                                    | 55 |
| Planejamento da Sprint #1 .....                           | 57 |
| Planejamento da Sprint #2.....                            | 58 |
| Review.....   | 59 |
| Retrospectiva .....                                       | 59 |
| Softwares Para Gerenciamento Ágil de Projetos.....        | 61 |

## **Autores**

### **Nelson Abu Samra Rahal Junior**

- Paulista de Assis
- Professor universitário desde 1996
- Atuo na área de desenvolvimento de software desde 1991
- Especialista em gerenciamento de projetos ágeis
- Graduado em Processamento de Dados
- Pós-graduação em Didática e Metodologia de Ensino
- Pós-graduação em Gerência de Projetos de TI (PMI)
- Mestre em Ciência da Computação
- Certificado ScrumMaster
- Consultor da Innovit Gestão de Projetos e Processos

### **Neilza Andrea Abu Samra Rahal**

- Paranaense de Araçongas
- Professora universitária desde 1998
- Especialista em gerenciamento de projetos na área de Teste de Software
- Graduado em Processamento de Dados
- Pós-graduação em Didática e Metodologia de Ensino
- Pós-graduação em Gerência de Projetos de TI (PMI)
- Mestre em Ciência da Computação
- Certificada em Teste de Software

## **Scrum**

### **Você sabe o que é Scrum?**

\* Escrito em parceria com Marcelo dos Santos

Scrum é um processo de gerenciamento de projetos ágeis, adaptado para a área de desenvolvimento de software, pelo especialista Ken Schwaber. Ken define Scrum, em um de seus livros, como: “um processo Ágil ou ainda um framework para gerenciamento de projetos Ágeis. É um processo de gerência de projetos, certamente não é uma metodologia, pois isto seria pesado demais”.

A primeira experiência com o Scrum ocorreu em uma fábrica de automóveis, onde se constatou que a utilização de equipes pequenas e multidisciplinares produzia melhores resultados. Em analogia a essas equipes, associou-se a formação do Scrum a de um jogo de Rugby. A partir de 1995, Ken Schwaber formalizou a definição de Scrum e ajudou a implantá-lo em desenvolvimento de software em todo o mundo.



**Ilustração 1 - Formação de Scrum em jogo de Rugby**

Martin Fowler, um dos maiores estudiosos em desenvolvimento de software, comenta em seu artigo A Nova Metodologia que: “Nos últimos anos, vem crescendo rapidamente o interesse em metodologias ágeis (ou “leves”). Também caracterizadas como um antídoto contra a burocracia ou uma licença para “hackear”. Estas metodologias despertaram os interesses em toda a extensão da indústria do software”.

Dentre as técnicas de utilização do Scrum, há a entrega de produtos em períodos de tempo pré-estabelecidos, nunca inferiores a uma semana ou superiores a trinta dias.

Para estimular o contato entre empresa e cliente, os projetos são interrompidos em períodos regulares de tempo. A essas ações dá-se o nome de Sprint. Ao término de cada Sprint, o cliente recebe um conjunto de funcionalidades desenvolvidas e prontas para serem utilizadas. A melhor maneira de comprovar se o software atende às necessidades é fazer com que o cliente o utilize, apontando as qualidades e o que falta ser aperfeiçoado.

Importante destacar que a participação ativa do cliente no processo de desenvolvimento de software faz com que sejam atribuídas a ele algumas responsabilidades como definição das funcionalidades do produto, decisão quanto às datas de lançamento de conteúdo e ajuste de funcionalidades.

## Framework Scrum

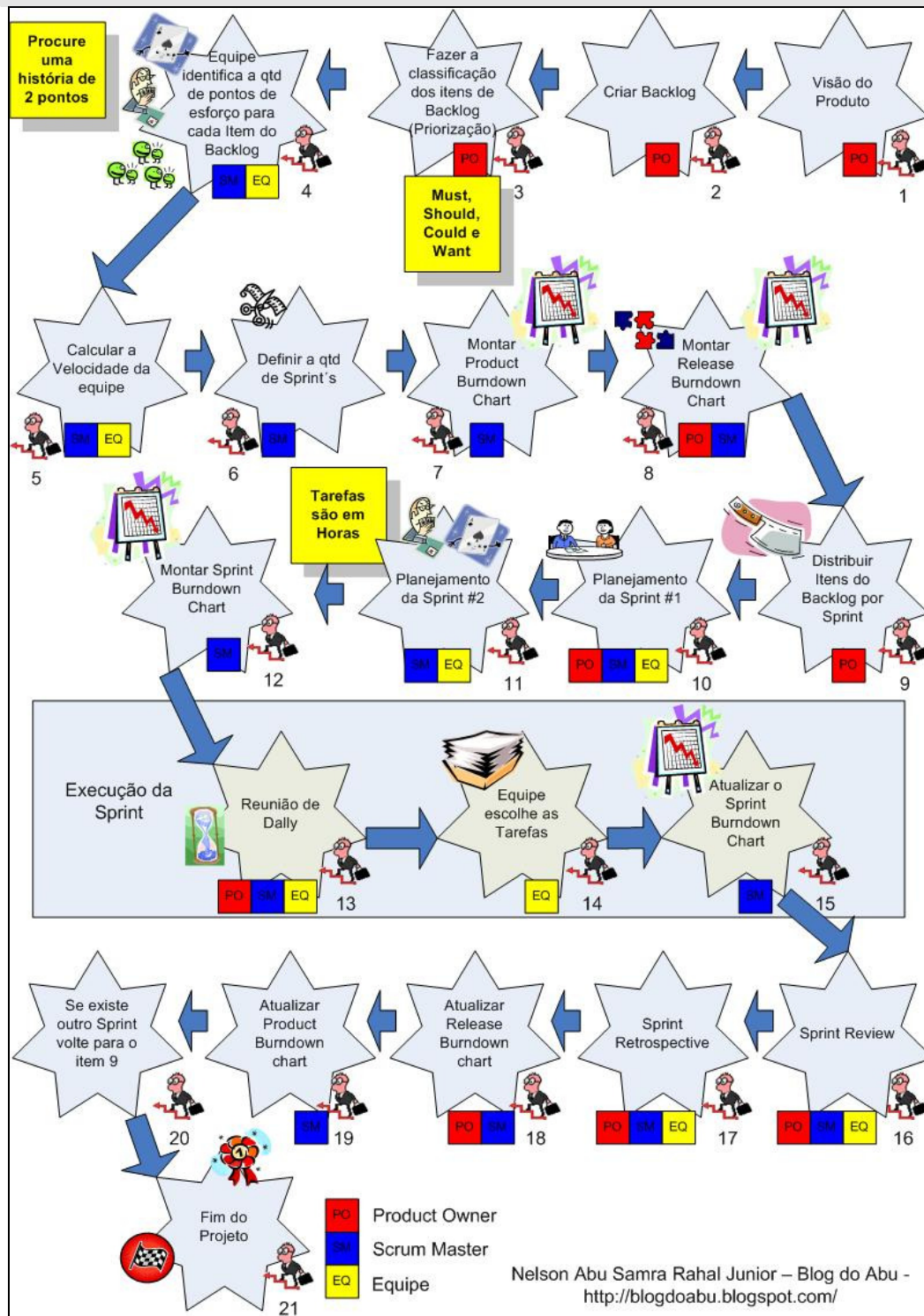


Ilustração 2 - Framework Scrum

## **Papéis no Scrum**

Scrum possui apenas três papéis, sendo eles: Product Owner, Scrum Master e a Equipe.

### **Product Owner (PO)**

Conhecido também como dono do produto é o responsável pela definição do projeto, levando a equipe de desenvolvimento o que chamamos de Visão do Produto.

Uma vez transmitida a Visão do Produto a equipe de desenvolvimento o dono do produto é responsável pela priorização dos requisitos e definição dos mesmos.

O Product Owner pode mudar as prioridades, adicionar ou remover novos requisitos conforme as suas necessidades. Ele apenas não pode mudar o trabalho que já está em codificação pela equipe de desenvolvimento.

Uma das grandes chaves de sucesso do Scrum está no Product Owner, por intermédio da definição e clareza dos requisitos. Os requisitos não devem chegar parcialmente definidos a equipe de desenvolvimento, caso isso ocorra a capacidade de produção da equipe vai diminuir.

O Product Owner também é responsável pelo retorno de investimento (ROI) do projeto. É difícil definir ROI, mas no Scrum ele se materializa com a priorização dos requisitos e codificação dos mesmos. No Scrum a entrega o mais rápido possível ao Product Owner dos módulos que ele tem necessidade caracteriza ROI e o Product Owner com o sistema em funcionamento pode avaliar o que está sendo construído ou até mesmo utilizá-lo o mais rápido possível.

Também uma das chaves de sucesso do Scrum com o Product Owner está na sua disponibilidade a equipe de desenvolvimento de software. As vezes ter o Product Owner sempre disponível é difícil, mas é uma meta a ser seguida.

Devemos lembrar que o Product Owner é o representante do cliente e a equipe de desenvolvimento de software tem que atender as suas expectativas e orientações. O Product Owner tem o poder de aceitar ou rejeitar um trabalho realizado pela equipe de desenvolvimento de software.

### **Scrum Máster (SM)**

O Scrum Máster é uma pessoa da equipe de desenvolvimento do software que possui algumas responsabilidades diferentes dos demais. Entre as responsabilidades está a de manter o processo do Scrum ativo, garantindo que o projeto vai ser realizado seguindo as boas praticas do Scrum.

Também faz parte das responsabilidades do Scrum Master identificar todos os problemas que estão ocorrendo durante o desenvolvimento do



sistema, que faz com que a capacidade de produção da equipe diminua ou até mesmo não permita que a equipe continue trabalhando. Estes problemas nos chamamos de obstáculos ou impedimentos e devem ser solucionados pelo Scrum Master. Quando o Scrum Master não tem condições de solucionar o impedimento ele deve buscar a pessoa que pode resolver o problema identificado.

A produtividade da equipe é mais uma responsabilidade do Scrum Master, pois ele deve fazer com que a equipe de desenvolvimento do sistema fique focada o máximo possível na sua área de atuação, que é a construção do software. Neste ponto nos falamos que o Scrum Master tem que ser uma blindagem da equipe, não permitindo que eventos externos atrapalhem os trabalhos que estão sendo realizados.

Um ponto importante é que o Scrum Master não tem que ter autoridade sobre a equipe, isto é, qualquer integrante da equipe de desenvolvimento de software pode ser o Scrum Master e até mesmo fazer um rodízio desta responsabilidade. A troca de Scrum Master entre as pessoas da equipe deve ser realizada com cautela, nunca a cada sprint (fase, ciclo ou iteração) de desenvolvimento e sim por projeto ou por versões (releases) do sistema.

Podemos ter também um Scrum Master coringa, para que caso o Scrum Master do projeto não possa estar presente com a equipe, uma pessoa da equipe já saiba que ele é o responsável por executar as atribuições de trabalho do Scrum Master.

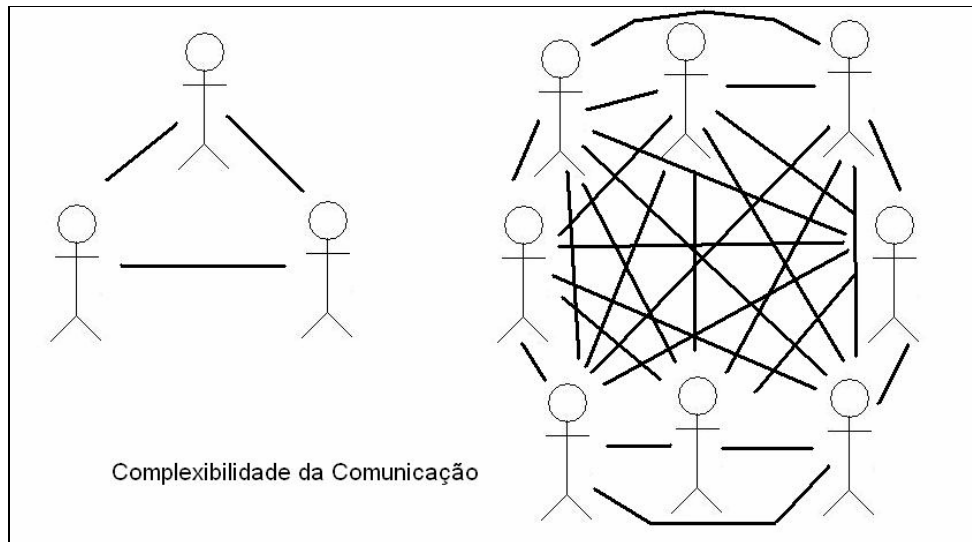
## **Equipe (EQ)**

É a responsável pela execução do projeto, sendo protegida pelo Scrum Master e tendo como foco o atendimento das necessidades do Product Owner.

Uma equipe em Scrum é constituída de no mínimo 2 pessoas e no máximo de 7 pessoas. As equipes são menores para potencializar uma das maiores armas do Scrum, a comunicação entre a equipe.

Três pessoas nos temos uma comunicação direta entre cada integrante da equipe, conforme a equipe vai aumentando a complexidade de comunicação entre todos os integrantes vai aumentando.

Equipes enxutas chegam a ter a sua capacidade de produção melhor que equipes grandes. Este aumento de velocidade ocorre em virtude da potencialização da comunicação entre os integrantes da equipe.



**Ilustração 3 - Complexidade da Comunicação**

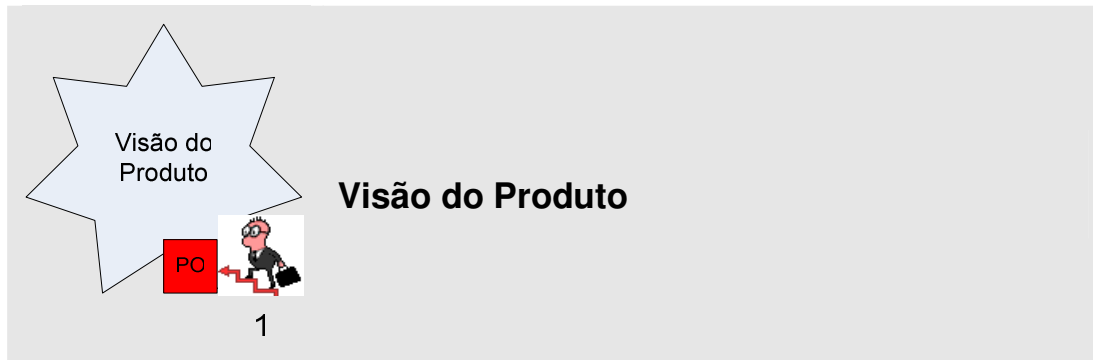
A formação da equipe é multidisciplinar, sendo ela constituída de: Engenheiros de Software, Arquitetos, Programadores, Analista, Peritos em Qualidade, Testadores, Web designers.

Mas em Scrum as equipes são generalistas e não especialistas. Todos os integrantes da equipe podem e devem desempenhar todos os papéis, conforme a necessidade e definição da equipe.

Nos utilizamos equipes multidisciplinares e generalistas para que o projeto possa ter uma velocidade de execução maior e uma redução de riscos com relação a definição do produto que está sendo realizado. Os generalistas permitem com que todas as pessoas do projeto sempre executam qualquer atividade. Com a equipe generalista nos passamos a potencializar a capacidade de produção que o nosso projeto possui, removendo o risco de um integrante da equipe ficar parado aguardando o trabalho que deve ser executado ficar disponível. Os multidisciplinares permitem a redução do risco nas definições do produto, permitindo a integração das mais diversas áreas na busca de um produto melhor.

As equipes em Scrum são auto-organizadas, isto é, elas definem a sua forma de trabalho e evoluem esta forma de trabalho pela sua própria avaliação de como está sendo executado o projeto.

Dentro da equipe de Scrum é instituído o conceito de auto-gestão, onde a equipe passa a ter autonomia de tomada de decisão para o seu foco de atuação, que é o de desenvolver o sistema solicitado pelo Product Owner.



Visão do Produto ou Escopo do projeto é a apresentação da abrangência do projeto a equipe de desenvolvimento do projeto. A visão do produto não tem o objetivo de ser uma apresentação detalhada dos requisitos e sim uma apresentação em auto nível de todos os módulos que vão ser construídos. Nesta apresentação pode ser apresentado a equipe fatores de sucesso, características de qualidade desejada, as metas e o que mais for necessário.

A visão do produto pode ser realizada varias vezes durante o projeto, não sendo uma regra a sua apresentação apenas no inicio do projeto. Com está reapresentação no decorrer do projeto diminuimos o risco do desvio do entendimento dos nossos objetivos durante a execução, fazendo com que todos mantenham o alinhamento com a “Meta” do projeto, e não apenas no inicio do projeto.

### **Executando a Visão do Produto**

Os participantes devem ser convidados com antecedência. No convite ter o local da reunião, a hora, data e o assunto da reunião. Na sala de reuniões deixar disponível para todos os presentes etiquetas coloridas colantes (post-it's), papel e canetas. A reunião pode ser filmada ou apenas o som gravado.

Durante a apresentação pode ser realizada a apresentação de slides, filmes, ou qualquer tipo de recurso que permita a transmissão de conhecimentos às pessoas que estão assistindo.

Está reunião não tem o objetivo de ser uma analise de sistemas, isto é, as pessoas que estão participando tem a liberdade de realizar perguntas, mas não existe a necessidade de se fazer uma analise detalhada dos itens apresentados. O objetivo da reunião é de apenas saber o que vamos fazer, quais as principais entregas, o que é cada modulo em auto nível e principalmente, fazer com que todos os envolvidos passem a ter conhecimento do projeto, pois este conhecimento e a participação na reunião de visão do produto leva a equipe a um comprometimento com o trabalho que vai ser realizado.

## **Exemplo de Visão do Produto**

Dia XX hora YY encontra-se reunido na sala de reuniões da empresa ACME a equipe do desenvolvimento do novo projeto da empresa. Junto com a equipe também esta presente o representante do cliente, o nosso Product Owner.

A apresentação do projeto é realizada com alguns recursos visuais e o escopo do mesmo é apresentado. O nosso projeto é de um software de venda de livros pela internet.

O Product Owner apresenta nos slides dados do sistema atual e seus problemas existentes e acrescenta as expectativas com relação ao novo sistema a ser realizado.

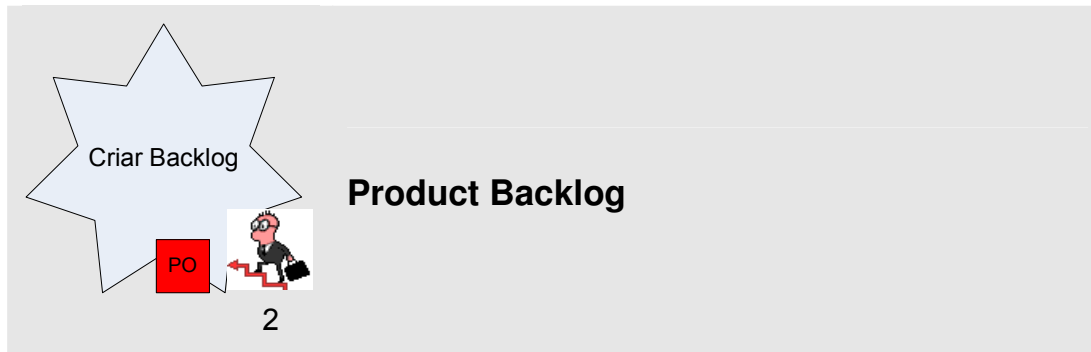
Um dos focos de sucesso está no “Carrinho de Compras”, onde o sistema atual apresenta problemas de usabilidade e de performance de execução. Também é apresentado pelo Product Owner um risco para o projeto, onde o risco apresentado é que se o “Carrinho de Compras” não for construído com foco nas expectativas do cliente o projeto não tem porque continuar.

Mais um novo risco é apresentado ao nosso projeto, nos temos uma data limite de construção do sistema, pois o cliente deseja colocar o novo software em funcionamento antes do período de compra de material escolar, para que ele “O Cliente” possa ter retorno do investimento que está sendo realizado no novo software com as vendas de seus produtos.

A equipe de desenvolvimento inicia a sabatina de perguntas ao Product Owner, para que todos possam ter o mesmo entendimento do que deve ser realizado, quando as perguntas entram em partes bem especificas, isto é, o entendimento aprofundado de um determinado modulo o Scrum Máster faz o papel de moderador e comenta que teremos o momento mais adequado para a realização da analise dos requisitos do sistema a ser desenvolvido e que este momento é para que todos saibam a onde pretendemos chegar com o nosso projeto.

Ao termino da reunião a equipe de desenvolvimento do sistema sai da reunião com as suas anotações e uma tabulação rápida destas anotações permite a criação de uma lista de requisitos em alto nível do nosso projeto.

O Scrum Máster apresenta a todos os integrantes da equipe o documento tabulado: Cadastro de Clientes, Cadastro de Produtos, Carrinho de Compras, Busca de Produtos, Categoria de Produtos, Histórico de Compras, Rastreamento de uma Compra, Pagamento por Cartão de Credito, Pagamento por Boleto Bancário e Exportação de Dados de Vendas para um sistema de ERP existente na empresa



Uma vez realizada a apresentação da Visão do Produto para os envolvidos no projeto uma relação de requisitos deve ser elaborada para que o sistema possa ser desenvolvido. Esta relação de requisitos nos chamamos de Product Backlog.

Podemos organizar a nossa lista de requisitos de varias maneiras, sendo uma organização apenas por requisitos referentes ao negocio a ser desenvolvido, ou uma lista com a relação de requisitos envolvendo o negocio a ser desenvolvido mais os requisitos não funcionais do sistema.

A estimativa do tamanho do projeto é realizada sobre os requisitos existentes na nossa lista de Product Backlog, mas para a realização desta estimativa nos utilizamos os requisitos em auto-nível chamados de “Temas”. Um “Tema” é um requisito que tem contido dentro dele requisitos menores. Nós vamos ver melhor a parte de requisitos no universo ágil no tópico denominado de historias.

Todos os envolvidos no projeto podem adicionar requisitos a nossa lista de requisitos do sistema, mas apenas o Product Owner pode priorizar os requisitos.

A priorização destes requisitos contidos no Product Backlog permite uma visão temporal de quando uma determinada funcionalidade vai ser realizada no projeto. A principio pode ser considerado um desperdício de tempo, pois no decorrer do projeto está priorização irá ser mudada conforme a necessidade do Product Owner, mas ela ajuda a todos os envolvidos a terem a visão de como o projeto vai ser executado já no início do mesmo, mesmo sabendo que podemos ter mudanças de prioridade.

### **Exemplo de Product Backlog**

Nos já temos uma relação de requisitos identificados na Visão do Produto e a qualquer momento novos requisitos podem ser adicionados ao projeto. O nosso Product Backlog não é constituído uma única vez e sim continuamente construído.

O documento gerado pela Visão do Produto possui os requisitos em auto-nível: Cadastro de Clientes, Cadastro de Produtos, Carrinho de Compras, Busca de Produtos, Categoria de Produtos, Histórico de Compras, Rastreamento de uma Compra, Pagamento por Cartão de Credito, Pagamento

por Boleto Bancário e Exportação de Dados de Vendas para um sistema de ERP existente na empresa. Para cada requisito em auto-nível, chamado de “Tema” nos passamos a adicionar os requisitos menores, denominados de historias.

Devemos lembrar que jamais podemos ignorar um requisito identificado, mesmo na visão do produto se um requisito menor for apresentado ele deve ser anotado e agrupado ao seu “Tema” principal.

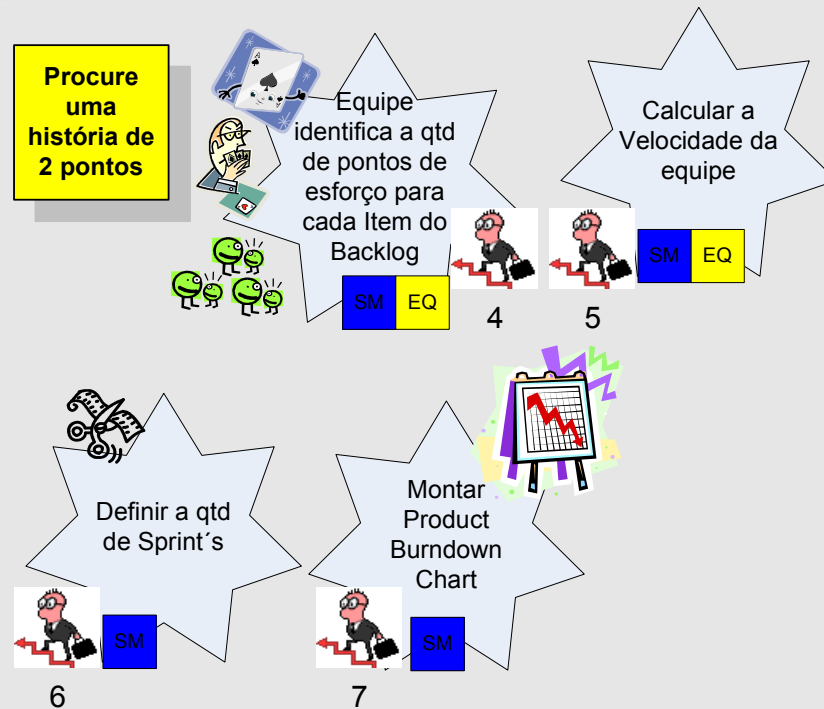
O Carrinho de Compras com o processo de análise do sistema passa a ter os requisitos menores: Adicionar produtos ao carrinho de compras, remover produtos do carrinho de compra, custo da entrega dos produtos conforme o meu CEP, desconto existentes em cada produto e itens do carrinho de compra devem permanecer no sistema até o desejo do comprador.

Vários requisitos menores foram criados pela própria equipe de desenvolvimento e o Product Owner vai validar cada um deles, desta maneira todos os envolvidos no projeto passam a ser donos do produto a ser criado.

O Tema “Cadastro de Clientes” recebeu os requisitos menores: Endereço do Cliente, Endereço de Entrega, Telefones de Contatos, Histórico de Compras.

Podemos organizar o nosso Product Backlog em uma planilha com as colunas: Tema, Historia, Pontos, Prioridade, Sprint. As colunas desta planilha vão ser definidas nos próximos tópicos.

## Estimativas, Velocidade da Equipe, Quantidade de Sprint's e Product Burndown Chart



O ato de estimar o tempo necessário para a conclusão de um projeto sempre tem uma porção de “chute”, nós apenas sabemos o tempo necessário para a realização de uma tarefa quando está tarefa já esta concluída.

Antes de sua conclusão apenas temos uma noção do tempo que vai ser necessário, ou apenas um desejo do que acreditamos ser possível de ser realizado.

Um projeto de um ano, onde existe um erro de uma hora por dia de trabalho, ao término de um ano teremos mais de 200 horas de atraso.

Em desenvolvimento ágil nós realizamos a estimativa de um projeto pela contagem de pontos, onde um ponto é uma forma de comparação de esforço necessário para a realização de um requisito.

Mas nós também temos uma estimativa em horas, que é realizada na quantidade de tarefas existente em uma Sprint.

A estimativa em pontos busca a visão do esforço do projeto e deve ser reestimada varias vezes durante o projeto, para corrigirmos imperfeições iniciais. Já a estimativa da Sprint, utilizando tarefas estimadas em horas tem o objetivo de saber se a quantidade de trabalho identificado é comportada pela equipe.

## **Contando os Pontos**

O cliente solicitou uma data de termino da entrega do projeto, para que ele possa realizar uma análise de riscos com relação a data limite da entrega do projeto e o inicio do período de vendas de material escolar.

A equipe se reúne com o Product Owner para realizar as estimativas necessárias. Os requisitos utilizados são os de tamanho de “Temas”.

Com cada um dos requisitos escritos em papeis e colocados sobre a mesa a equipe busca os de menos esforço de trabalho. Está busca tem o objetivo de escolher um requisito que vai ser utilizado como ponto de referencia, uma ancora em relação aos demais.

A participação do Product Owner é fundamental, pois a cada requisito em tamanho igual a um “Tema” a equipe pode solicitar maiores informações e o Product Owner passa a responder estas informações. Está reunião inclusive é um ótimo momento de anotarmos mais requisitos contidos num “Tema”, para atualização do nosso Product Backlog.

A equipe por consenso identifica o requisito de tamanho menor, este requisito é o de Categoria de Produtos. Para facilitar o processo de contagem de pontos buscamos os requisitos que são um pouco maior que o menor, pois sempre aparecem requisitos menores ao que nos estamos utilizando como base, isto é, ancora.

Por consenso a equipe escolhe o Cadastro de Clientes e este recebe o valor igual a dois pontos. Dois pontos neste momento da técnica de estimativa não tem nenhuma correlação com o tempo necessário para a relação do projeto, apenas representa o tamanho do esforço necessário para a construção deste “Tema”.

Com o requisito de tamanho igual a dois passamos a comparar este requisito com os demais e atribuir aos demais os valores em pontos correspondentes. A escala de pontos utilizado é a de Fibonacci, 1, 1, 2, 3, 5, 8, 13, 21, 34, ?.

O processo de contagem de pontos, nós utilizamos o Planning Poker, baralhos com os pontos de Fibonacci, para que ninguém da equipe influencie o outro. Jogamos as cartas viradas para baixo e todos ao mesmo tempo viraram as cartas para cima, onde podemos realizar a análise dos pontos jogados.

Quando as cartas possuem valores iguais significa que a equipe chegou a um consenso do tamanho do esforço, mas quando as cartas possuem valores diferentes é sinal que a equipe não esta fechada em um consenso. Neste momento entra o Scrum Máster como um facilitador perguntando a cada pessoa da equipe o porquê dos valores de suas cartas, o que cada pessoa está identificando de esforço a mais do que o outro colega de trabalho.

As duvidas, incertezas, requisitos que nos acreditamos ser verdadeiros e contidos no requisito de tamanho igual a “Temas” do requisito que está sendo estimado, nos devemos perguntar ao Product Owner, para que ele valide e deixe claro para a equipe a abrangência do esforço necessário.



Não existindo mais dúvidas as cartas são jogadas de novo e a equipe busca o consenso do esforço. O processo pode ser repetido quantas vezes forem necessárias, com o tempo a equipe acha o seu ponto de equilíbrio nas estimativas.

É importante sempre deixarmos como consenso do resultado do processo de Planning Poker as cartas de maior valor, para que todas as pessoas da equipe se sintam a vontade em executar o requisito estimado no tempo que ela acredita ser exequível.

### **Exemplo de Pontos Contados**

| <b>Tema</b>                     | <b>Pontos</b> |
|---------------------------------|---------------|
| Cadastro de Clientes            | 2             |
| Cadastro de Produtos            | 2             |
| Carrinho de Compras             | 13            |
| Busca de Produtos               | 2             |
| Categoria de Produtos           | 1             |
| Histórico de Compras            | 5             |
| Rastreamento de uma Compra      | 8             |
| Pagamento por Cartão de Crédito | 8             |
| Pagamento por Boleto Bancário   | 13            |
| Exportação de Dados p/ EPR      | 13            |
| <b>Total</b>                    | <b>67</b>     |

### **Determinando o Tempo do Projeto**

O primeiro passo para sabermos o tempo do projeto é a definição do tamanho da nossa Sprint, lembrando que uma Sprint é um tempo de trabalho, que pode ser de uma, duas, três ou quatro semanas, mas uma vez definido o tamanho da Sprint ela não deve ser modificada até o término do projeto, para não termos que regêramos gráficos e informações do projeto sobre o novo tamanho da Sprint, o que irá gerar um re-trabalho.

A equipe junto com o Product Owner determina que a Sprint vai ser de uma semana. Uma vez sabendo o tamanho da Sprint é perguntada a equipe a quantidade de pontos que a equipe consegue executar em uma Sprint. Este valor está diretamente vinculado ao que tem que ser feito e a quantidade de pessoas que a nossa equipe possui. Também neste valor está incluso o nosso processo de desenvolvimento do software, onde ele deve abranger tudo que for necessário, como testes de software, UML, documentação e o que for definido pela equipe e os interesses do cliente, representado pelo Product Owner.

A equipe definiu a quantidade de 10 pontos por Sprint, agora temos condições de dar uma previsão ao cliente do tempo necessário para o desenvolvimento do sistema, bastando apenas pegar a quantidade de pontos contados e dividir pela quantidade de pontos que a equipe é capaz de executar. A quantidade de pontos que a equipe é capaz de executar por Sprint é a “Velocidade da Equipe”.

O resultado da operação matemática chegou ao valor de:  $67 / 10 = 6$  com resto de 7. Seis (6) é a quantidade de Sprint's necessárias para a realização do projeto. Vamos arredondar para sete (7), pois temos o resto da divisão. Sete semanas são iguais a aproximadamente dois meses de trabalho.

Podemos também realizar o cálculo da Velocidade da Equipe com as informações coletadas após a execução da primeira Sprint e com este dado calculamos o tempo de execução do projeto.

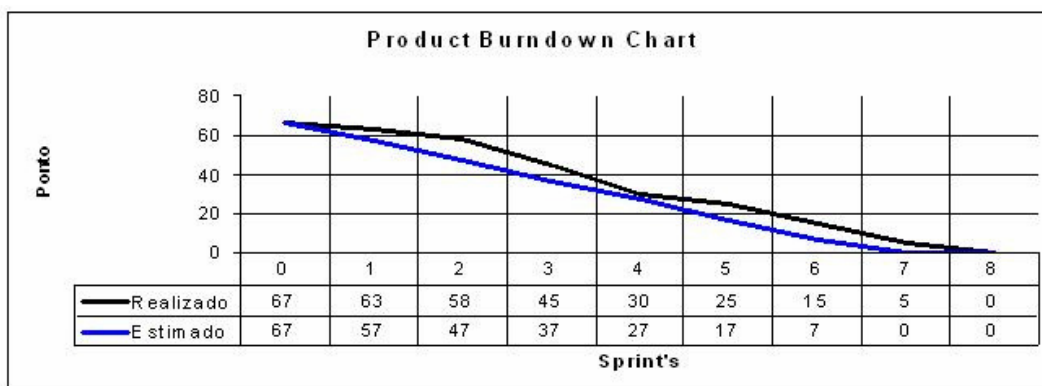
### **Product Burndown Chart**

Também conhecido como o gráfico de ciclo de vida do projeto, que tem o objetivo de mostrar como está o andamento do projeto inteiro.

Colocamos no Eixo Y os pontos do projeto e colocamos no Eixo X a quantidade de Sprint's. Passamos uma reta mostrando a meta dos pontos a serem executados pela quantidade de Sprint's do nosso projeto, respeitando a Velocidade da Equipe.

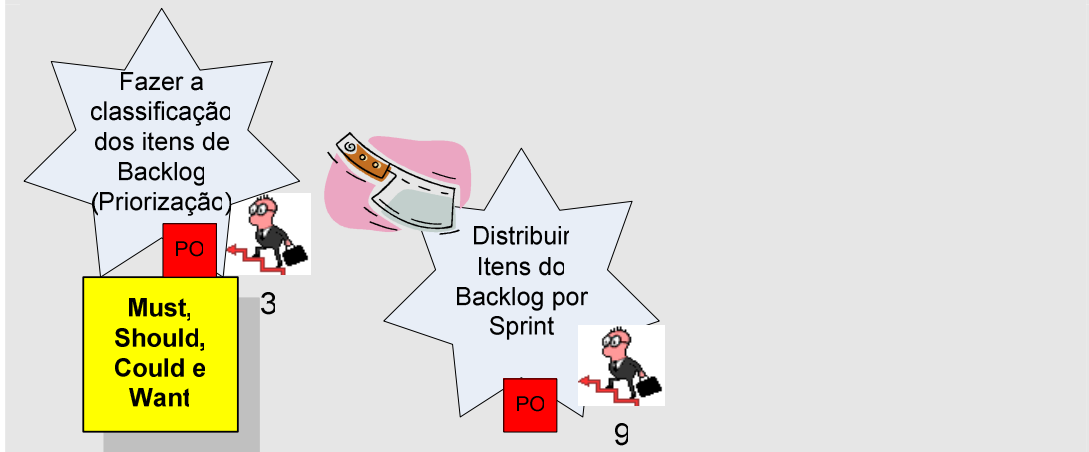
Uma segunda linha é gerada a cada execução de Sprint, com o objetivo de mostrar se a nossa meta foi alcançada ou não. A cada execução de Sprint a nossa velocidade pode ser recalculada e conseqüentemente o tempo do projeto passa a sofrer alteração.

Este gráfico fica muito bem disponibilizado em uma parede do lado da equipe de desenvolvimento. Quando todas as pessoas envolvidas no projeto, direta ou indiretamente aprenderem a ler este gráfico não existe mais a necessidade de responder aos interessados a pergunta: “Como está indo o projeto de Venda de Livros pela Internet?”, pois a informação está disponibilizada a todos, permitindo uma comunicação direta e rápida. Se o escopo do projeto passa a sofrer alterações o gráfico sofre a necessidade de ser atualizado, mas este item vai ser visto melhor nos próximos tópicos deste material.



**Ilustração 4 - Product Burndown Chart**

## **Priorização e Distribuição de Requisitos por Sprint's**



A priorização do Backlog é realizada pelo Product Owner. Esta priorização é realizada com o objetivo de selecionar os principais requisitos para construção.

Em Scrum a priorização é fundamental para a redução do risco de cancelamento do projeto, pois quanto mais rápido for entregue o que realmente é importante ao Product Owner, menor o risco do projeto ser cancelado.

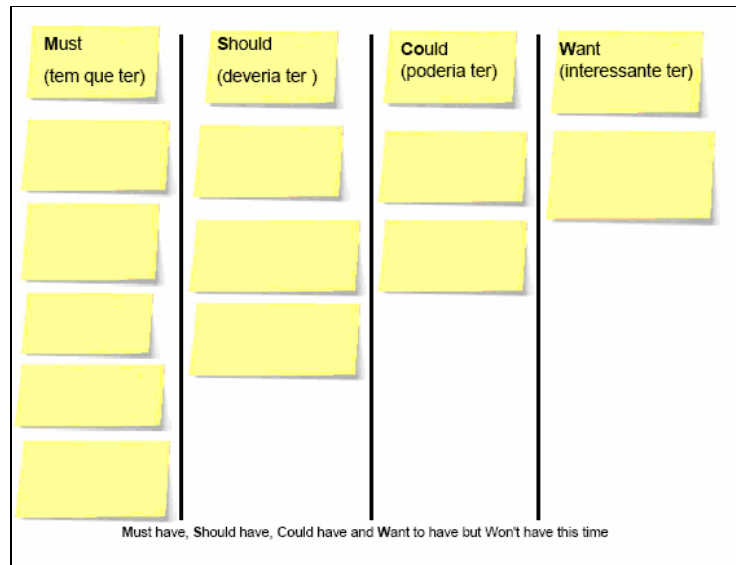
A priorização faz com que o comprometimento do Product Owner seja muito forte, pois o Product Owner é o responsável pelo retorno do investimento (ROI) e desta maneira ele que tem que ter a visão clara dos requisitos que realmente devem ser feitos antes dos outros.

Com a priorização nos passamos a ter um "Tema" do sistema, como "Cadastro de Clientes", tendo seus requisitos sendo executados conforme o desejo do Product Owner e não "todos" os requisitos sendo executados de uma única vez.

Com este modelo de trabalho um "Tema" passa a receber mais funcionalidades durante todo o desenvolvimento do projeto.

### **Executando a Priorização**

Uma boa técnica é a MoSCoW, onde os requisitos são distribuídos por Sprint respeitando a priorização.



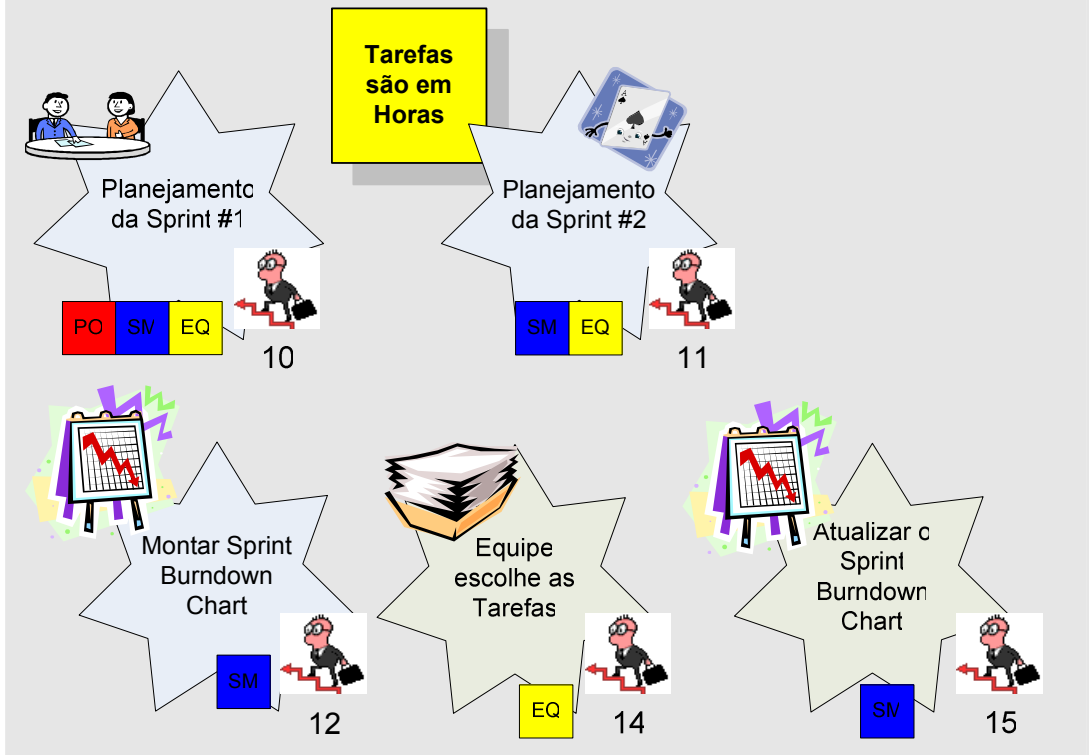
**Ilustração 5 - Priorização com MoSCoW**

Devemos executar primeiro todos os requisitos “Must”, isto é “Tem que ter” do nosso projeto. Quando não existir mais requisitos “Must” atacamos os requisitos “Should” e assim por diante, até o termino de todos os requisitos que o nosso projeto possui.

Um ponto importante do Scrum, a cada nova Sprint o Product Owner tem a liberdade de escolher os requisitos que vão ser executado na Sprint. Com esta liberdade a priorização dos requisitos no projeto é planejada Sprint a Sprint e nunca uma vez so no inicio do projeto.

Com a definição de prioridades dos requisitos que devem ser executados a cada Sprint o Product Owner busca o resultado em seu ROI (retorno de investimento), pois o cliente está recebendo sempre o que é mais importante para ele a cada Sprint.

## **Sprint, Planejamento da Sprint #1, Planejamento da Sprint #2, Sprint Burndown Chart e Execução da Sprint**



A execução do projeto é realizada em períodos de tempo que nós denominamos de Sprint. Uma Sprint tem período máximo de tempo igual a 30 dias e o período menor de tempo igual a uma semana.

Dentro de uma Sprint nós temos que executar o Framework do Scrum, com o planejamento da Sprint, as reuniões diárias, a entrega do que foi realizado no final da Sprint e a Retrospectiva, que é uma reunião de melhoria do processo pela equipe.

Devemos respeitar o período de tempo da Sprint como uma regra inegociável, onde mesmo não tendo realizado todas as atividades necessárias da Sprint nós terminamos a mesma, para que mesmo com uma entrega parcial dos resultados obtidos, seja realizada a entrega.

É importante sempre entregarmos o serviço realizado na Sprint, mesmo que este serviço não tenha sido concluído em sua totalidade.

A quantidade de serviços, isto é, requisitos que vamos implementar dentro de uma Sprint é determinada pela equipe e nunca pelo Product Owner. Quem vai realizar o trabalho deve se comprometer com a quantidade de serviço que vai realizar e nunca se comprometer com o serviço que foi determinado por outra pessoa.

Um dos segredos da Sprint está neste comprometimento, quando a equipe acredita que é possível realizar os requisitos acordados existe uma probabilidade maior de recebermos o que foi combinado.

É melhor trabalharmos com a realidade da velocidade de nossa equipe do que com o desejo que temos de velocidade. Uma equipe que sofre pressão para aumentar a velocidade acaba abrindo mão de alguma coisa e quase sempre o que a equipe abre mão é da qualidade, para que ela possa atender a velocidade esperada.

O problema de abrir mão da qualidade é que os requisitos e suas tarefas acabam não sendo fechados e retornam para a linha de produção. O nosso Product Backlog acaba recebendo os erros gerados durante o desenvolvimento das Sprint e os itens ainda não implementados. Está entrada de serviços com defeito acaba fazendo com que a equipe pare de desenvolver novos requisitos e gaste energia na correção de requisitos já implementados e que estão com defeito, o que pode levar o projeto a um fracasso.

## **Planejamento da Sprint #1**

A reunião de planejamento da Sprint é realizada com a Equipe e o Product Owner e tem o objetivo do Product Owner apresentar a equipe os requisitos que vão ser desenvolvidos na Sprint.

Este planejamento é realizado com todos os integrantes da equipe de desenvolvimento do projeto, onde passamos a ter analistas de sistemas, desenvolvedores, testadores, DBA, web-design e quem mais for necessário.

Com a união de todas estas especialidades passamos a ter um ganho melhor de análise de sistemas e uma visão mais crítica de como devemos solucionar os requisitos com a união de todas estas especialidades. O conhecimento também passa a ser compartilhado o que reduz os riscos da perda de uma pessoa na equipe, independente do motivo. Também nesta união de profissionais passamos a distribuir e potencializar habilidades a todos, onde a ação de análise de sistemas passa a ser aprendida e realizada por todos os profissionais.

Um dos grandes segredos desta reunião é o Product Owner já vir com os requisitos bem definidos, para que a equipe tenha uma compreensão bem abrangente do que deve ser feito com o requisito.

É o que eu chamo de requisitos quadrados e requisitos redondos, onde um requisito quadrado é aquele que não tem uma clareza na sua definição e faz com que a equipe tenha dificuldade de achar as tarefas necessárias para a sua implementação e durante a execução do requisito ele gera vários impedimentos. Estes impedimentos podem levar a equipe inclusive a uma implementação parcial do requisito o que leva a uma não entrega do mesmo.

Já os requisitos redondos a equipe consegue no planejamento um entendimento claro do seu objetivo, o que leva a uma definição muito boa das tarefas a serem realizadas e permite um desenvolvimento sem dúvidas. Os requisitos redondos aumentam a velocidade de desenvolvimento do sistema pela equipe, isto é, ao invés de pressionarmos a equipe para uma velocidade

maior, devemos entregar a equipe requisitos melhores, que permitira um aumento de velocidade com qualidade do produto gerado.

## **Planejamento da Sprint #2**

O planejamento da Sprint #2 passa a ser realizado apenas pela equipe, sem a necessidade do Product Owner. Este planejamento tem o objetivo de determinar como vamos realizar os requisitos, isto é, como vamos transformar o que foi recebido em forma de requisitos em software.

Nesta reunião definimos de maneira rápido como pode ser a tela de um sistema, o modelo de dados em auto-nível, as classes necessárias a serem desenvolvidas, as regras de negocio, etc.

Cada item identificado como sendo parte da solução do requisito passa a ser chamado de tarefas e estas tarefas recebem uma estimativa de horas.

Uma boa pratica e ter tarefas com o tempo de no mínimo de 1 hora e de no maximo um dia de trabalho. Se tivermos tarefas menores de uma hora elas podem ser agrupadas, para chegarem ao tempo de 1 hora. Uma boa dica também é agrupar as tarefas para meio período de trabalho, isto é, em um dia de 8 horas meio período seria de 4 horas. Desta maneira não perdemos tempo com estimativas de tarefas pequenas, deixando elas agrupadas e rotuladas com um conjunto de horas. Também ter agrupamento de tarefas faz com que os integrantes da equipe peguem trabalho sempre para um período de tempo ou um dia inteiro, não sendo necessário o seu deslocamento para pegar trabalhos durante o seu dia de trabalho.

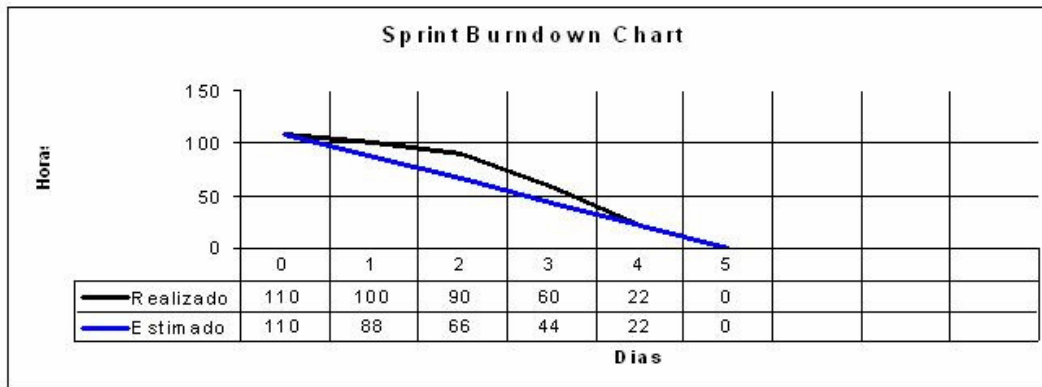
Um requisito passa a ter varias tarefas e cada tarefa passa a receber o tempo necessário para o seu desenvolvimento. A totalização das horas necessárias para a completude dos requisitos selecionados para o desenvolvimento da Sprint faz com que a equipe saiba se os itens selecionados para trabalho são suficientes para o tempo existe ou maior, menor que o tempo que temos de trabalho.

## **Sprint Burndown Chart**

Sprint Burndown Chart é o gráfico que mostra como está a execução das tarefas dentro de uma Sprint.

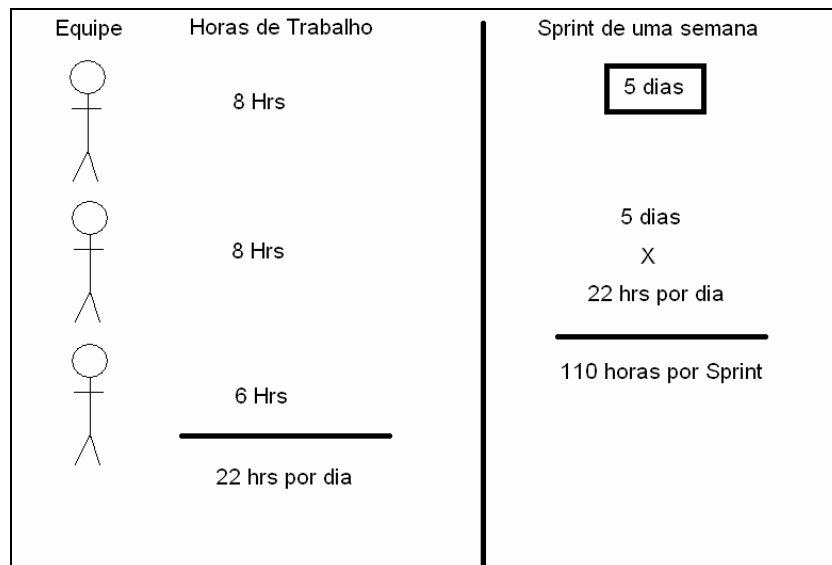
Nós temos no Eixo Y a quantidade de horas necessárias para a execução da Sprint e no Eixo X o numero de dias da nossa Sprint. A nossa velocidade é a quantidade de horas que a nossa equipe consegue executar por dia.





**Ilustração 6 - Sprint Burndown Chart**

### Calculo de Velocidade da Sprint



**Ilustração 7 - Calculo da Velocidade da Equipe na Sprint**

A capacidade de produção da nossa equipe é de 110 horas por Sprint e 22 horas por dia. A quantidade de horas das tarefas identificadas no nosso Planejamento de Sprint #2 não deve ser superior a 110 horas. Nós podemos ter mais de 110 horas de tarefas na Sprint, mas isso só ocorre quando a equipe acredita ser capaz de executar às 110 horas mais o excedente.

### Exemplo de Tarefas na Sprint

| Requisito em Tema    | Requisitos em Historias | Tarefas por Historias                        |
|----------------------|-------------------------|--|
| Cadastro de Clientes | Endereço do Cliente     | Validar CEP;<br>Validar campos obrigatórios; |

|  |                       |   |
|--|-----------------------|---|
|  | Endereço de Entrega   | Copiar o endereço do cliente;<br>Validar CEP;<br>Validar campos obrigatórios;<br>Pode ser diferente do endereço do cliente;       |
|  | Telefones de Contatos | Tipo de telefone (celular, fixo, faz);<br>Pessoal ou trabalho;<br>Telefone principal para contato;<br>DDD;<br>Numero do telefone. |
|  | Histórico de Compras  | Este item é um outro tema de historia e deve ser tratado como tal.  |

### **Exemplo de Ciclo de Vida de Uma Sprint**

O exemplo do ciclo de vida de uma Sprint apresentado tem o seu inicio em uma segunda feira, mas o inicio da Sprint pode ser em qualquer dia da semana. Também neste exemplo foi definido um período de trabalho de 8 horas, tendo o período da manha e o período da tarde, mas pode ter projetos com períodos noturnos ou de apenas um período.

Também foi colocada a reunião diária no horário da manha, mas ela pode ser realizada em qualquer horário do dia, horário este que tem que ser definido com os integrantes da equipe.

[illegible]

### Ilustração 8 - Sprint de 2 Semanas

|                        | Seg                            | Ter          | Qua          | Qui          | Sex          |
|------------------------|--------------------------------|--------------|--------------|--------------|--------------|
|                        | Planejamento da Sprint #1 e #2 | Reunião      | Reunião      | Reunião      | Reunião      |
| Mão na Massa           |                                | Mão na Massa | Mão na Massa | Mão na Massa | Mão na Massa |
| Mão na Massa           |                                | Mão na Massa | Mão na Massa | Mão na Massa | Mão na Massa |
| Mão na Massa           |                                | Mão na Massa | Mão na Massa | Mão na Massa | Mão na Massa |
| Mão na Massa           |                                | Mão na Massa | Mão na Massa | Mão na Massa | Mão na Massa |
| Review e Retrospectiva |                                |              |              |              |              |
|                        |                                |              |              |              | Sábado       |
|                        |                                |              |              |              | Domingo      |

### Ilustração 9 - Sprint de 1 Semana

## **Executando a Sprint**

Após o planejamento da Sprint #1 e #2 a equipe pode iniciar o processo de execução da Sprint. Os requisitos e suas respectivas tarefas são colocados no quadro de Taskboard, também chamado de quadro de Kanban.

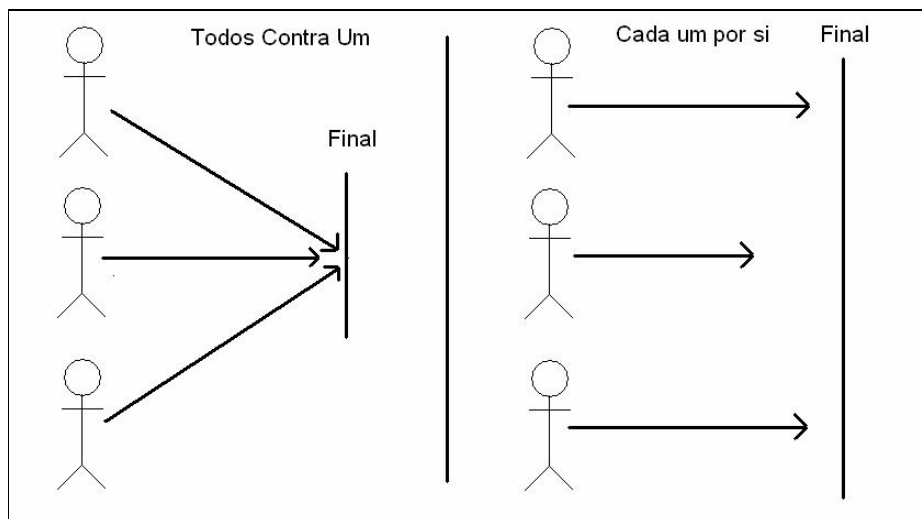
Todo dia é realizada uma reunião de alinhamento da Sprint, que vai ser vista no tópico “Reunião Diária”. A execução dos Requisitos deve ser realizada com o conceito “Todos Contra Um”, onde a equipe inteira ataca o primeiro requisito, isto é, suas tarefas.

Caso não tenha serviço para todos, o segundo requisito é aberto e os integrantes da equipe que não estavam trabalhando no primeiro requisito vão para o segundo. Caso não tenha serviço para todos, o terceiro requisito é aberto e assim por diante.

Quando um requisito é terminado ou não existe tarefa para todos, o processo de ajudar no próximo requisito ou abrir um novo requisito deve ser seguido.

A técnica de “Todos Contra Um” faz com que tenhamos ao termino da Sprint sempre requisitos prontos para serem entregues. Quando colocamos uma pessoa em cada requisito, isto é, “Cada um por si” ao termino da Sprint corremos o risco de termos todos os requisitos 90% pronto, mas nenhum realmente pronto.

Um ponto importante do “Todos Contra Um” é que a equipe em Scrum realmente é uma equipe, não buscamos mais o individualismo e sim o coletivo. Não existe mais o melhor desenvolvedor, o melhor arquiteto, o melhor testador, existe a equipe, que vai ser forte ou fraca pela união do grupo.






















**Ilustração 10 - Todos Contra Um**

É comum aparecer requisitos novos durante a execução da Sprint que está sendo executada.

Requisitos novos são ocasionados por uma análise superficial dos requisitos da Sprint, ou por uma necessidade nova do Product Owner. O requisito não planejado ocasionado por uma análise superficial, é um alerta que

não devemos jamais deixar de lado o bom trabalho de análise de sistemas. Estes requisitos não planejados vão atrapalhar a execução da Sprint por intermédio de um impedimento de falta de definição de como o sistema deve se comportar.

Mas quando um requisito novo entra na Sprint, independente do motivo, ele deve ser colocado em um espaço visível e bem sinalizado, para que todos os envolvidos no projeto saibam que demandas novas estão consumindo energia da equipe de desenvolvimento. Estas demandas não previstas podem levar a uma entrega parcial dos requisitos acordados pela Equipe com o Product Owner no início da Sprint.

| Item de Backlog   | To do   | Progress  | Waiting  | Done  |
|---|---|---|--|---|
| Requisitos  | Tarefas   | Execução  | Aguardando   | Concluídas  |
|    |  |  |  |  |
|   |  |   |  |  |
|    |  |  |  |  |
|   |  |  |  |   |
|   |  |   |  |   |
| Não Previsto  |   |   |  |   |
|    |   |   |  |   |
|     |   |   |  |   |

**Ilustração 11 - Taskbord ou Kanban**

A equipe tem autonomia de executar as tarefas do quadro de Taskboard, não sendo necessário uma pessoa ficar atribuindo estas tarefas as pessoas que vão executar.

Está autonomia ocorre porque desde o início do projeto todos os integrantes da equipe de execução do projeto estavam juntos entendendo o que deve ser feito. Também estavam juntos no Planejamento da Sprint #1 e #2. Não existe necessidade de um “controle manda faz”, a equipe sabe o que deve ser feito e não ter um controle sobre a equipe neste momento, potencializa a capacidade de execução da mesma.

O controle dentro da Sprint passa a ser compartilhado por todos da equipe, não ficando a responsabilidade para uma pessoa. Isso é o que nos chamamos de auto-gestão.



É uma reunião diária de acompanhamento de execução do projeto realizada pela equipe. Nesta reunião participa com direito a falar apenas os denominados “porcos”, “frangos” podem participar, mas não tem direito a falar nesta reunião.

Esta reunião deve ter no máximo 15 minutos de duração e deve ser conduzida pelo “Scrum Master”. Recomenda-se que esta reunião seja realizada no período matutino, bem no início das atividades da equipe. Também recomenda-se que os participantes fiquem de pé. Esta reunião não é para solução de problemas e sim de sincronismo entre os membros da equipe de como está indo a execução das atividades realizadas no “Sprint”.

O ScrumMaster deve realizar três perguntas a equipe:

- 1 – O que fiz desde a última reunião?
- 2 – O que farei ate a próxima reunião?
- 3 – Existe algum obstáculo (Impedimento)?

**Ilustração 12 - Perguntas na Reunião Diária**

### **Sete Fundamentos para reuniões diárias eficazes**

- (Standups) by Stacia L. Broderick
- O TIME acredita em auto-gestão e torna possível a sua existência.
- Todos da equipe estão comprometidos (como um time!) para as metas do Sprint.
- Eles percebem a importância da comunicação e estão utilizando o Daily Scrum para ajudar a facilitar esta comunicação.
- A equipe compreende e aceita que a ordem de, dependências entre, e necessidades das tarefas irá mudar diariamente ao longo do ciclo do Sprint. As reuniões diárias da equipe permitem gerenciar essas mudanças e reagir em conformidade.

- O TIME tem um líder efetivo "ScrumMaster" ou um líder que lhes compartilha permissões (poderes) para que eles possam tomar decisões e assumir responsabilidade com credibilidade.
- O TIME percebe a importância de tornar o trabalho mais visível; transparência melhora a natureza do relacionamento entre o TIME e o resto da organização, resultando em níveis mais elevados de confiança e colaboração.
- O TIME irá recordar durante os Eventos-marco de Processos (milestones) as definições tomadas durante a reunião diária (daily standup meeting) para tornar estes eventos o mais efetivo possível.

## **Impedimentos**

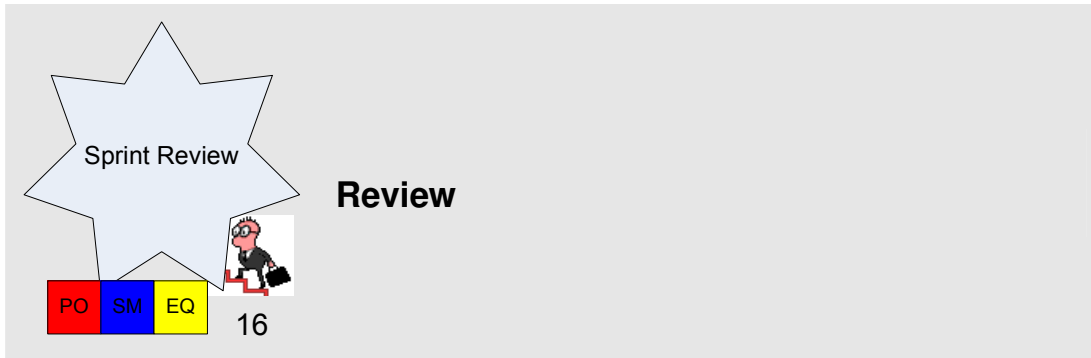
Impedimento é qualquer coisa que atrapalhe um membro da equipe de executar o trabalho. Os impedimentos podem ser identificados nas reuniões diárias, onde cada membro da equipe tem a oportunidade de comunicar o ScrumMaster do impedimento existente. O ScrumMaster é responsável pela solução dos impedimentos. O ScrumMaster pode realizar reuniões complementares para solucionar os impedimentos identificados quando as reuniões diárias não permitirem a solução.

Um dos fatores de potencialização da equipe Scrum está na ação de retirar da equipe de execução do projeto a responsabilidade de resolver um impedimento. Se deixarmos a equipe de execução do projeto focado apenas na execução e passarmos todos os impedimentos ao Scrum Máster vamos ter um ganho de velocidade e qualidade na execução dos trabalhos.

## **Dia a Dia da Reunião Diária**

Este é o nosso momento de alinhamento das atividades executadas dos nossos requisitos. Mas além do alinhamento é o momento que temos de lembrar as nossas metas. Nesta reunião vale a pena sempre lembrar ou perguntar para a equipe qual a nossa meta da Sprint, para que todos sempre saibam o valor de negocio que estamos implementando ao produto desenvolvido.

Nossa meta não é apenas fazer algoritmo e sim algoritmos alinhados com a expectativa do Product Owner, que está trazendo ganho ao sistema desenvolvido.



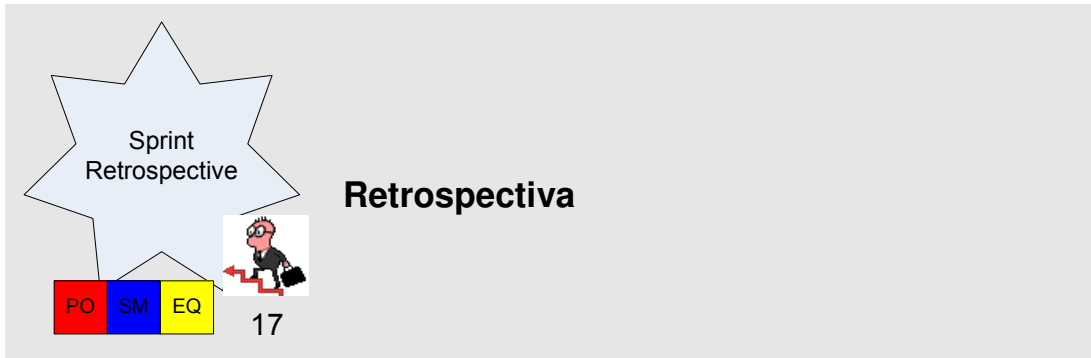
Review, também conhecido como entrega dos trabalhos realizados na Sprint pela equipe de execução do projeto para o Product Owner.

A entrega dos trabalhos realizados ao Product Owner não “deveria” ter surpresas para o mesmo, se nos temos um Product Owner presente durante o ciclo de desenvolvimento da Sprint, ele sabe o que nos estamos entregando e o que não estamos entregando. Ele também sabe os problemas que a equipe teve durante a execução da Sprint, não sendo necessária a apresentação de um “diário de bordo” com o check list de tudo que deu errado e que levou a uma entrega parcial dos trabalhos.

Na reunião de entrega será apresentado ao Product Owner o software desenvolvido para que ele aceite ou não as funcionalidades implementadas, os gráficos, os requisitos selecionados para a Sprint, o que foi feito, o que não foi feito e caso o Product Owner não tenha participado ativamente da Sprint, devemos ter o “diário de bordo” com os problemas que a equipe passou durante a execução da Sprint.

Uma boa dica para a realização da entrega do software é a equipe ter testado as funcionalidades antes de realizar a entrega, ter testado o ambiente que o sistema vai rodar e manter o foco na entrega de funcionalidades e não na evolução do sistema que está sendo entregue. Devemos entregar o que foi acordado com o Product Owner, isto é, requisitos implementados aderentes ao que nos foi passado.





A retrospectiva da equipe tem o objetivo de realizar uma melhoria contínua no processo de desenvolvimento de software que a equipe está adotando. Esta reunião é realizada ao término da Sprint, preferencialmente após a reunião de entrega (Review) ao Product Owner.

Eu particularmente gosto de colocar na retrospectiva todos os problemas que estão influenciando o projeto, tanto técnicos, quanto processuais ou até mesmo de conflito entre integrantes da equipe.

É neste momento que temos a oportunidade de resolver os problemas enfrentados durante a execução de nossa Sprint e tomarmos ações realmente efetivas para que estes problemas não ocorram mais.

Existem várias técnicas para a realização de uma retrospectiva, vamos mostrar apenas duas, mas tenho certeza que estas duas técnicas vão ser mais do que suficientes para uma boa retrospectiva de sua equipe.

## **Linha do Tempo (Timeline)**

### **Propósito**

Estimular lembranças do que aconteceu durante o desenvolvimento do trabalho. Criar uma imagem do trabalho de muitas perspectivas. Examinando hipóteses sobre quem fez o quê e quando. Ver padrões ou quando os níveis de energia mudaram. Use esse recurso para "apenas os fatos" ou fatos e sentimentos.

### **Tempo Necessário**

Trinta a noventa minutos, dependendo do tamanho do grupo e o tamanho das atividades executadas.

### **Descrição**

A equipe escrever cartas (post-its) para representar as lembranças, que foram significativas para eles, ou outro evento significativo durante a iteração, a liberação, ou projeto e depois publicá-las em (aproximadamente) por ordem cronológica. A retrospectiva apóia o líder da equipe para discutir os acontecimentos e compreender os fatos e sentimentos durante a iteração, liberação, ou projeto.

### **Passos**

**[1]**

Inicie a atividade, dizendo "Nós iremos preencher (completar) uma linha de tempo para criar uma figura completa desta iteração / release / projeto. Nos desejamos identificar a perspectiva de cada colega de trabalho.

**[2]**

Dividir a equipe em pequenos grupos, não tendo mais que cinco pessoas por grupo. Manter as pessoas que trabalharam em estreita colaboração com todos os outros juntos (afinidade por grupos). É melhor ter dois pequenos grupos que representam uma afinidade do que um grande grupo.

Todos devem pegar os post-its ou canetas, etc.

Certifique-se que cada pessoa tem uma caneta. Devemos lembrar as pessoas para escrever de maneira legível, para que todos possam ler os post-its.

**[3]**

**Descreva o processo.**

Pergunte as pessoas sobre o que ocorreu na iteração / release / projeto, resgatando todos os pontos memoráveis, significativos, eventos importantes, coisas ruins e registre esta informação em post-its um abaixo do outro. Cada post-its só pode ter uma informação.

Relembre ao grupo que o objetivo é de identificar muitas perspectivas. Caso não exista consenso sobre algum dos itens identificados em post-its não tem importância, pois se o item for importante para pelo menos uma das pessoas da equipe, já é suficiente para o seu registro.

A atividade deve durar no máximo 10 minutos.

Pode ser utilizado post-its com cores diferentes, mas deve ser colocada uma legenda que explique o significado de cada cor.

Escrever em letras legíveis.

**[4]**

Monitorar a atividade dos debates e a escrita dos post-its. Tomar cuidado para que a atividade consuma muito tempo de debate e desta maneira não gere post-its, caso isso ocorra lembre a equipe que deve iniciar o trabalho de escrever os registros nos post-its.

Quando o grupo já tiver uma pilha de cartões (post-its), convidar as pessoas para começar a publicá-las (veja a Figura).

**[5]**

Quando todas as cartas (post-its) estão destacadas, convidar a equipa a analisar a linha de tempo e ver o que outros já publicaram. Após a verificação dos itens publicados, solicite para as pessoas adicionarem novos cartões, caso alguém lembre de algum evento importante e que não foi registrado.

**[6]**

Permitir uma pausa antes de realizar a análise da linha do tempo.

## **Variações**

É possível utilizar algumas variações sobre a atividade de linha do tempo. Estas possibilidades podem ser de utilizar materiais como cartões, post-its, canetas coloridas, pontos, etc. Existem diversas formas de estimular o resgate dos fatos e sentimentos ocorridos durante a execução do projeto.

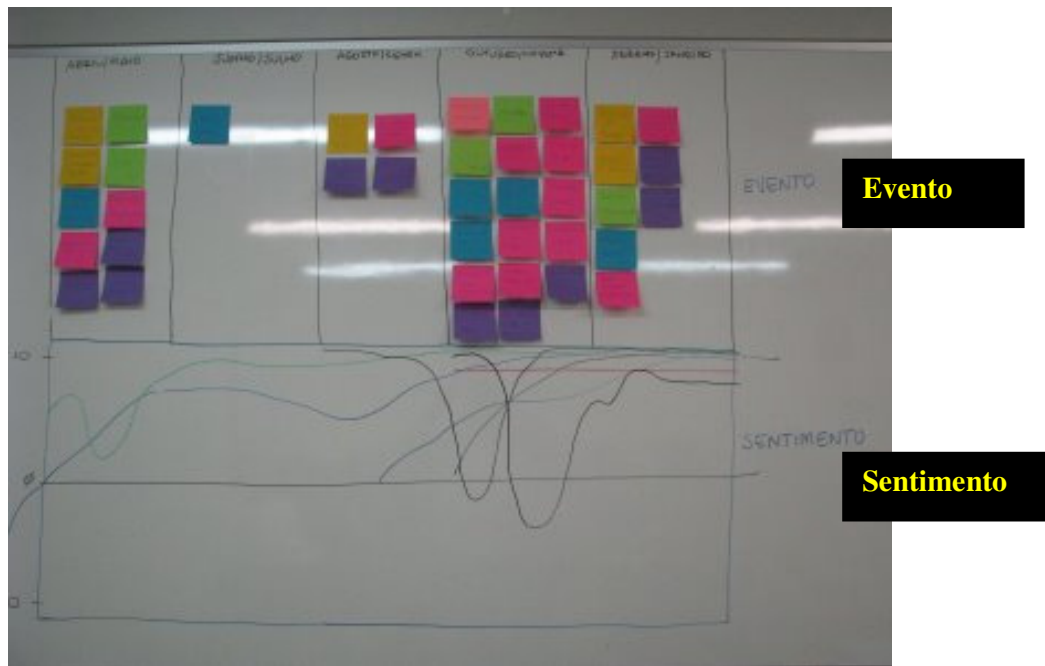
### **Por exemplo:**

- Cores para representar Sentimentos: Para se utilizado tanto para fatos e sentimentos para representar estados emocionais.
  - Azul = tristeza, raiva, mau
  - Vermelho = desafiado, estagnou
  - Verde = satisfeito, bem sucedido, energético
  - Amarelo = cauteloso, confuso
  - Purple = diversão, surpresa, humor
  - Salmão = fatigados, sublinhou
- Cores para representar Eventos: Para ser utilizado para representar tipos de eventos.
  - Amarelo = técnica ou tecnologia relacionados
  - Pink = equipe ou pessoas relacionadas
  - Verde = organização relacionados
- Cores para representar Funções: Para ser utilizado para representar funções (Papeis no projeto).
  - Azul = desenvolvedores
  - Pink = clientes
  - Verde = GQ e ensaios
  - Amarelo = escritores técnicos
- Cores para representar Temas: Para ser utilizado quando a equipe pretende concentrar em assuntos específicos, usar cores para identificar eventos relacionados a temas específicos.
  - Amarelo = equipe comunicação
  - Azul = equipamento de utilização
  - Pink = relacionamentos com os clientes
  - Verde = práticas de engenharia

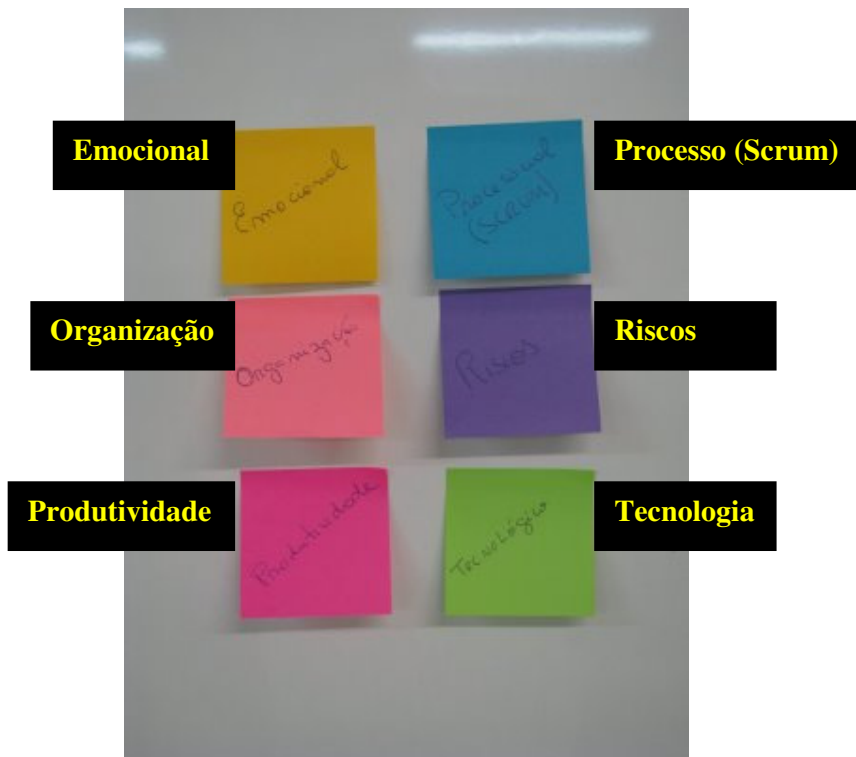
Você pode escolher o seu próprio esquema de cores com base nos cartões e post-its disponíveis para você.

## **Materiais e Preparação**

Quadro branco, papel de parede, fitas, canetas coloridas, post-its coloridos, etc



**Ilustração 13 - Retrospectiva Linha do Tempo**



**Ilustração 14 - Legenda Utilizada na Linha do Tempo**

### **Resumo do Abu**

- Material
  - Quadro

- Canetas Coloridas
- Post-its
- Execução
  - Dividir as pessoas em equipes
  - Definir legenda dos cartões
  - Executar a atividade de criar os cartões
  - Criar uma coluna para todos os post-its e colar todos os post-its nesta coluna
  - Revisão da equipe para identificar se não falta nenhum evento ou sentimento
  - Dividir os post-its em colunas que representam as Sprint's / Iterações / Meses, etc
  - Criar um espaço abaixo das colunas para a adição de um gráfico emocional
  - Cada integrante da equipe desenha uma linha que represente o seu estado emocional no período (sprint)
- Legenda
  - Verde = Tecnológico
  - Amarelo = Emocional
  - Salmão = Organização
  - Rosa = Produtividade
  - Azul = Processual
  - Roxo = Riscos

## **Retrospectiva Rápida**

Está é a segunda técnica de retrospectiva que vou apresentar, é mais simples e mais rápido de ser executado.

A técnica é simples, colocamos três colunas na parede, contendo as palavras “Continuar”, “Melhorar” e “Parar” e todos da equipe escrevem os post-its contendo apenas um “Evento” e o “Evento” escrito deve ser colocada na coluna correspondente.

As colunas podem ser preenchidas simultaneamente ou coluna a coluna, conforme a condução do Scrum Máster.

Continuar é o que nos fizemos na Sprint e que vale a pena ser repetido para as demais Sprint, até o momento que continue agregando valor ao processo.

Melhorar é o que não está bom, mas é importante para o processo e desta maneira não pode ser removido das atividades que realizamos nas nossas próximas Sprint's.

Parar é o que estamos executando durante a Sprint e não está agregando valor nenhum ao processo, simplesmente não serve para nada e desta maneira não deve ser repetido para as próximas Sprint, pois está consumindo tempo da equipe e o processo ou produto não está ganhando nada com a sua execução.

| Continuar   | Melhorar  | Parar   |
|---|---|---|
|  |  |  |
|  |  |  |
|  |  |  |
|  |   |  |
|   |   |   |
|   |   |   |
|   |   |   |
|   |   |   |

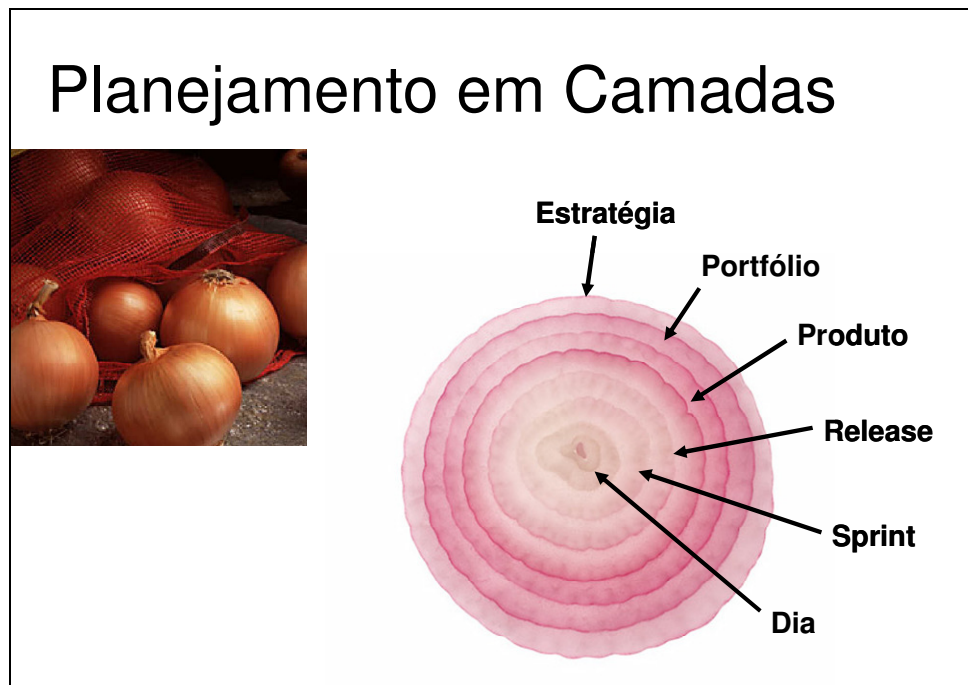
**Ilustração 15 - Retrospectiva Simples**

### **Dia a Dia Após a Retrospectiva**

O documento gerado com o resultado da Retrospectiva deve ser adicionado em local que permita a sua visualização por todos da equipe. Devemos lembrar todos os dias o que estamos buscando na nossa melhoria de processo e está busca por ser feita com as informações disponíveis a todos e dentro da reunião diária.



Release ou versão de software é o nosso planejamento de entregas de funcionalidades. Em Scrum para a equipe de desenvolvimento do sistema o que nos buscamos são metas pequenas e possíveis de serem exeqüíveis. Porém para o Product Owner o que nos planejamos são alocações de requisitos distribuídos ao longo do projeto, para que o Product Owner saiba quando ele vai ter disponível a funcionalidade desejada.



**Ilustração 16 - Planejamento de uma Release**

O planejamento em camada de cebolas representa estes dois extremos, o planejamento da equipe de execução e o planejamento do Product Owner. As duas extremidades são conduzidas até o encontro de ambas.

Quando a equipe planeja a sua Sprint, identifica as suas tarefas e executa as tarefas seguindo o seu planejamento ela está direcionando o projeto ao encontro dos interesses do Product Owner com o seu planejamento.

O foco do planejamento da equipe deve ser sempre metas exeqüíveis. Nos não focamos uma entrega do projeto inteiro ou de uma versão e sim passamos a focar as tarefas existentes em um requisito, os requisitos existentes em uma Sprint. A equipe pensar em um planejamento de longo prazo de tempo faz com que a mesma perca o ritmo, isto é, velocidade.

As tarefas quebradas em horas de no maximo um dia de trabalho faz com que todos os integrantes da equipe ao executarem as suas tarefas do dia, já estejam mantendo o planejamento para a entrega total do projeto no período acordado com o Product Owner.

| Release - 1 |      | Release - 2 |      |      | Release - 3 |      |      |      |      |
|-------------|------|-------------|------|------|-------------|------|------|------|------|
| SP01        | SP02 | SP03        | SP04 | SP05 | SP06        | SP07 | SP08 | SP09 | SP10 |

**Ilustração 17 - Sprint's por Releases**

O Product Owner ao pensar em uma Release ele deve manter o foco no retorno de investimento que ele está realizando. Eu costumo dizer que entra um saco de dinheiro e do outro lado sai algoritmo implementado. Se o algoritmo implementado não era o que o Product Owner desejava ele não vai ver ganho nenhum no que recebeu.

Valor de negocio, para obter um retorno de investimento sempre vai estar nos olhos do Product Owner.



## **Scrum de Scrum**

Scrum de Scrum é a ferramenta que nos temos de realizar um alinhamento, sincronismo entre projetos.

Se você perguntar a um integrante de um projeto “A” qual o projeto mais importante na empresa ele irá responder que é o projeto dele. Se você perguntar a um integrante de um projeto “B”, ele também irá responder que o projeto mais importante é o dele.

Não tem nada de errado com as respostas obtidas, muito pelo contrario, cada integrante de um projeto tem que ter a convicção que o seu projeto é o mais importante para a empresa.

É comum durante a execução de projetos termos a necessidade de ajuda para que o nosso projeto continue atendendo as expectativas do Product Owner. Existem impedimentos que dentro de uma estrutura funcional de uma empresa o Scrum Máster não terá condições de resolver e ele necessita escalar o problema para uma hierarquia funcional. Um bom exemplo é a necessidade de recursos externos, pessoas de outras equipes ou um recurso que deverá ser terceirizado. É neste momento que o Scrum de Scrum vem ajudar.

Scrum de Scrum funciona com a reunião de Scrum Máster para alinhamento dos projetos, solicitações de ajuda, visibilidade do caminho que a empresa está seguindo, remoção de impedimentos que o Scrum Máster não consegue resolver sozinho e o que mais for necessário.

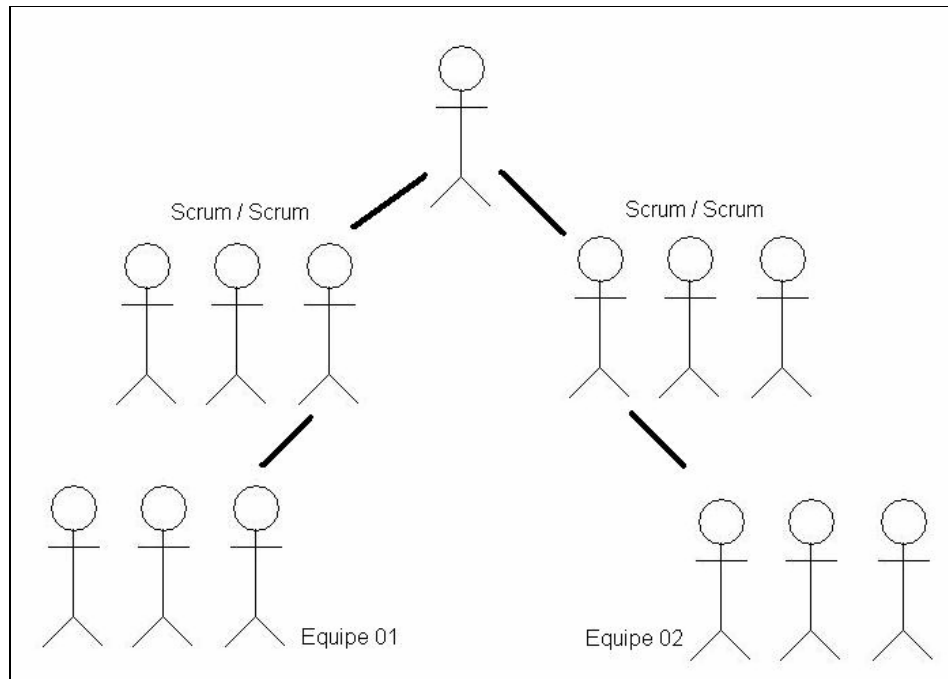
O Scrum de Scrum também permite a troca de experiências, isto é, o que uma equipe está fazendo é que está dando certo ou errado, para que modelos possam ser seguidos pelas demais equipes ou evitados por elas.

A execução do Scrum de Scrum pode ser realizada todos os dias como uma reunião diária ou por períodos de tempo.

Tem empresas que preferem realizar no primeiro momento do dia a reunião de Scrum de Scrum e depois desta reunião realizar as demais reuniões diárias das equipes, para que as informações possam ser transmitidas a todos. Isso permite um alinhamento da empresa em um curto período de tempo.

Já tem empresas que preferem após a reunião diária das equipes, para que os problemas sejam passados e o processo de solução seja iniciado no mesmo dia.

Um Scrum de Scrum utiliza todas as ferramentas do Framework do Scrum para gerenciar os projetos que estão nas equipes, é uma ótima maneira de gerar informação para levar as pessoas interessadas, que podemos materializar como exemplo: gerentes funcionais das mais diversas áreas da empresa, diretórios, presidentes, clientes, etc.



**Ilustração 18 - Scrum de Scrum**

## **Potencializando Scrum**

A potencialização do processo Scrum já foi comentada neste material, mas não custa enfatizarmos mais um pouco sobre o assunto.

Jeff Sutherland um dos criados do Scrum, escreveu um artigo fantástico sobre a união de Scrum e CMMI nível 5, onde o artigo traz uma serie de procedimentos necessários para que este casamento seja possível. Estes procedimentos para atendimento do CMMI nível 5 não fazem parte do escopo deste material.

Mas um destes procedimentos está a potencialização do Scrum para o que ele chamou de “Entendendo o Sucesso do Scrum”.

Jeff traz o conceito de “Pronto” e “Feito”, onde “Pronto” são requisitos possíveis de serem implementados e “Feito” é o requisito desenvolvido conforme os critérios de aceitação do Product Owner.

O “Feito” é a nossa coluna de “Done” de um quadro de Kanban (também chamado de Taskboard), que é definido entre a equipe e o Product Owner.

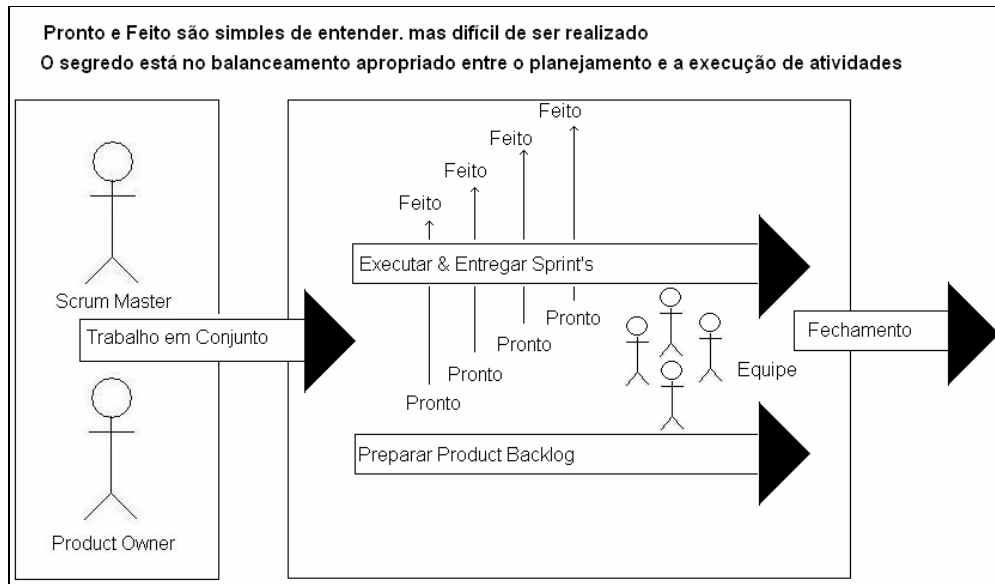
Para aumentar a velocidade da equipe com os requisitos “Prontos”, os mesmos são escritos pelo Product Owner com o auxilio do Scrum Máster, eles juntos realizam um trabalho de preparação do material da próxima Sprint e quando este material chega ao planejamento da Sprint ele possui um nível de maturidade maior, para que a equipe possa realizar a análise de sistemas do mesmo e o seu planejamento de execução.

Jeff faz a observação que: “Pronto e Feito são simples de entender, mas difícil de ser realizado” e “O segredo está no balanceamento apropriado entre o planejamento e a execução de atividades”.

O link do material é:

<http://jeffsutherland.com/scrum/PracticalRoadmapMunich20091020.pdf>

Os requisitos “Prontos” é o que eu chamei neste material de requisitos quadrados e redondos. Onde um requisito quadrado traz problemas na análise de sistema, planejamento e execução. Já os requisitos redondos permitem a sua análise, planejamento e execução em velocidade e qualidade pela equipe de desenvolvimento do projeto. Toda vez que um requisito quadrado entra na execução da Sprint ele é um candidato a gerar impedimentos e conseqüentemente trazer problemas na execução da Sprint, trazendo impacto na velocidade de produção da equipe.



**Ilustração 19 - Trabalho em conjunto entre Product Owner e Scrum Máster**

## **Épicos, Temas e Historias**

### **Cartão de Historia**

Cartões de história é uma técnica de captura de requisitos de sistemas, esta técnica surgiu com a metodologia de desenvolvimento chamada de XP (extreme programming), porém podemos utilizar esta técnica com qualquer metodologia de desenvolvimento de sistemas ágeis.

Como “Cartões” têm o objetivo de capturar os requisitos, eles se concentram em “quem, o quê e porquê de um recurso, e não como”. Neste ponto ele é igual a varias técnicas de levantamento de requisitos, onde tentamos identificar o que tem que ser feito e não como será feito.

O cartão de história tem que ser escrito a partir de uma perspectiva do usuário. Desta maneira o cartão não deve possuir jargões técnicos ou informações de design, ou qualquer outra coisa que não seja informações de negócio. Um cartão deve ser escrito com a linguagem no negócio, para que seja compreendido por todos.

### **Exemplo de Cartão de Historia**

*Como um professor, eu gostaria de lançar as notas de meus alunos, para que os alunos possam saber como estão indo na minha matéria.*

*Como um aluno, eu gostaria de visualizar as minhas notas de modo que eu possa saber o meu desempenho nas disciplinas.*

Observe que em ambos os casos estamos buscando um linguajar de negócio e buscando as necessidades do sistema (em nível de negócio – o que tem que ser feito), mas em nenhum dos exemplos entramos em uma linguagem técnica e nem falamos “como vamos fazer”.

### **Template de Cartão de Historia**

Como um [usuário papel], quero [meta], para que eu possa [motivo].

A primeira vista podemos achar que a parte do [motivo] não é necessária, mas a utilização do [motivo] nos leva a um entendimento melhor do objetivo pelo qual o cartão é útil. Ele nos ajuda a entender melhor como o requisito funciona e nos da uma idéia melhor para outras características do sistema.

Em Scrum nós buscamos identificar os cartões no início do projeto, para que ele de origem ao “Product Backlog”. Com os cartões identificados podemos dar origem a priorização e a estimativa dos mesmos.

Uma vez os cartões priorizados para execução em uma Sprint eles devem ter os seus detalhes (tarefas) identificados. Não devemos identificar os detalhes de cartões que não fazem parte da Sprint, desta maneira evitamos o gasto de energia na Sprint corrente em trabalhos que fazem parte de outro momento temporal.

## **Organizando Cartões de Historia**

Um cartão de história deveria ter pelo menos três itens: Informações do cartão, dados de conversa com o usuário e testes de validação do cartão.

Informações do cartão são: nome, descrição da história, um número de referencia, estimativa, etc.

Conversas são anotações (lembretes) da história e não uma especificação completa. Estas anotações servem de apoio para a obtenção de mais detalhes no momento de desenvolvimento, identificando como o software deve funcionar. Tudo que ajude a explicar a funcionalidade deve ser colocado como um item de conversa. É como um conjunto de palavras chaves, mas não tem a necessidade de ficar restrito a uma palavra, pode ser colocada (escrito) como uma frase.

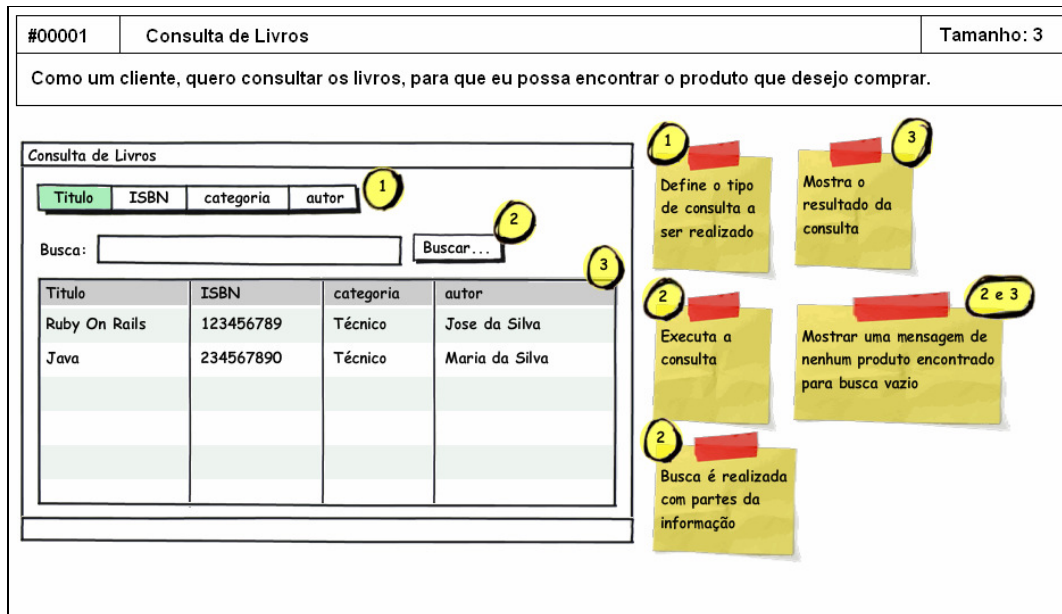
Confirmação (Testes) são os casos de testes no verso do cartão. Os testes possibilitam o entendimento melhor do cartão, ajuda na busca de cenários que os usuários, desenvolvedores / analistas podem não ter pensado. Um cartão (requisito) tem que ser possível de ser testado.

Um cartão de história tem que possuir uma única responsabilidade, ter cartões de história com mais de uma responsabilidade dificulta o entendimento, a estimativa e a priorização. Cartões pequenos nos ajudam a realizar uma administração (gestão) do projeto.

Link de Referencia:

<http://kw-agiledevelopment.blogspot.com/2008/01/user-stories-answers-on-postcard-please.html>

<http://kw-agiledevelopment.blogspot.com/2008/01/user-stories.html>



2 e 3

Mostrar uma mensagem de nenhum produto encontrado para busca vazia

**Ilustração 20 – Exemplo de Cartão de Historia**

Fonte: <http://kw-agiledevelopment.blogspot.com/2008/01/example-of-user-story.html>

Observe que o desenho que representa a frente do cartão tem os dados de identificação do cartão de história. Estes dados são: o numero (0001), o titulo (User Login), a história do cartão (As a [register user]...) e ainda dados das conversas com o usuário, onde temos um esboço de como deve funcionar a tela e anotações do comportamento esperado.

Na segunda imagem nós temos as informações dos testes (confirmação), mostrando os cenários possíveis e esperados pelo usuário.

|   |
|---|
| <p><b>Comportamento de Sucesso</b></p> <p>Entra com parte do nome do livro – Pressiona Buscar – Mostra os livros encontrados<br/>         Entra com o nome completo do livro – Pressiona Buscar – Mostra o livro encontrado<br/>         Entra com o ISBN – Pressiona Buscar – Mostra o livro encontrado<br/>         Entra com parte do nome do autor – Pressiona Buscar – Mostra os livros encontrados<br/>         Entra com o nome completo do autor – Pressiona Buscar – Mostra o livro encontrado</p> <p><b>Comportamento de Erros</b></p> <p>Entra com parte do ISBN – Pressiona Buscar – Mostra mensagem de erro<br/>         Resultado da busca vazio – Mostra mensagem de erro<br/>         Campo de Busca vazio – Pressiona Buscar – Mostra mensagem de erro</p> |
|---|

**Ilustração 21 - Exemplo de Testes do Cartão de Historia**

## **Exemplos de Cartão de Historia**

Eu estou publicando uma relação de cartões de história utilizados em um sistema de venda de livros pela internet. Estes cartões são apenas de exemplo. Eu tenho utilizado o tema Venda de Livros pela internet em meus treinamentos de Scrum.

Pode ser observado que alguns cartões possuem responsabilidades simples e outros possuem uma complexidade muito maior.

MODELO: Como um [usuário papel], quero [meta], para que eu possa [motivo].

### **Clientes**

- Como um cliente, quero consultar os livros, para que eu possa encontrar o produto que desejo comprar.
  - Conversa: Titulo, ISBN, categoria, autor, gênero
- Como um cliente, quero que o produto seja acompanhado com o valor de desconto para compra a vista, para que eu tenha a visibilidade da diferença monetária do produto a vista ou a prazo.
- Como um cliente, quero que os produtos selecionados para compra fiquem armazenados como um carrinho de compras, para que eu possa visualizar todos os meus produtos e o preço total.
- Como um cliente, quero que o sistema mostre o valor da postagem, para que eu possa saber quanto vai ser a taxa de entrega.
- Como um cliente, quero ter os comentários referentes aos livros consultados, para que eu possa saber a opinião de quem já leu
- Como um cliente, quero consultar a relação dos livros mais vendidos, para que eu possa ter uma visibilidade dos livros preferidos pelo mercado
- Como um cliente, quero realizar o pagamento em bloqueto bancário, para os pagamentos a vista
- Como um cliente, quero realizar o pagamento por cartão de credito, para pagamentos a vista e parcelados
- Como um cliente, quero visualizar os produtos acompanhados por uma fotografia, para que eu possa ter certeza que é o produto que eu procuro
- Como um cliente, quero ter acesso a uma área privada, para que nesta área eu possa acompanhar a situação do meu pedido.
- Como um cliente, quero ter acesso a uma área privada, para que eu possa atualizar os meus dados pessoais
- Como um cliente, quero ter acesso a uma área privada, para que eu possa ter acesso as informações de minhas compras anteriores
- Como um cliente, quero ter os meus dados pessoais cadastrados, para que estes dados possam ser utilizados como endereço de entrega dos produtos



- Como um cliente, quero poder determinar o endereço de entrega do produto, para que eu defina o endereço do destinatário

### **Administrador do Sistema**

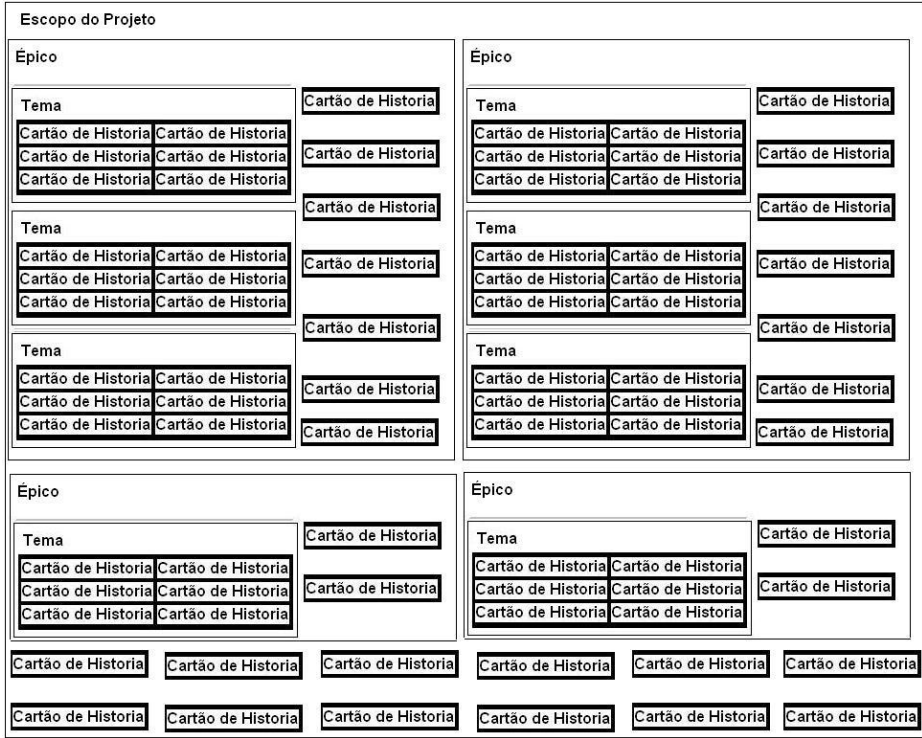
- Como um adm, desejo cadastrar usuários para o sistema, para que eu possa ter controle das pessoas que possuem acesso as funcionalidades
- Como um adm, desejo que o sistema gere log de todas as ações realizadas no sistema, para que eu possa realizar um auditoria de inclusão, alteração ou exclusão de dados

### **Módulo administrativo**

- Como um usuário administrativo, quero tirar um relatório com a relação de produtos vendidos em um intervalo de tempo, para que eu possa ter visibilidade das vendas realizadas
- Como um usuário administrativo, quero tirar um relatório com o montante financeiro realizado por período, para que eu possa saber o faturamento
- Como um usuário administrativo, quero tirar um relatório dos produtos mais vendidos, para que eu possa saber as tendências do mercado
- Como um usuário administrativo, quero cadastrar os parceiros (fornecedores), para que eu tenha informações de contato
- Como um usuário administrativo, quero importar a relação de novos produtos fornecido pelo parceiro, para que eu não tenha que cadastrar os produtos manualmente

### **Escopo do Projeto**

Vamos fazer a definição das Categorias dos Cartões de História. Um projeto possui um escopo e dentro deste escopo nós temos requisitos do tamanho de um Épico, Tema ou Cartão de História.



### Ilustração 22 - Escopo do Projeto

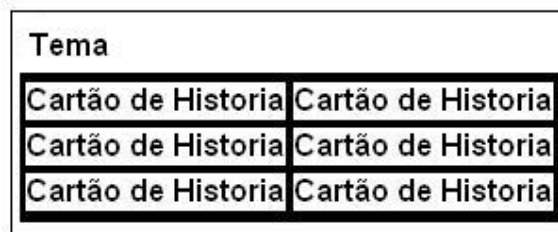
## Épico, Tema e Historia

Épico é um Cartão de História muito grande.

Tema é um conjunto de Cartões de História Correlacionados, isto é, é um cartão de história que possui vários cartões de história dentro dele mesmo, mas que todas estas histórias internas estão correlacionadas ao mesmo assunto.



**Ilustração 23 - Épico com vários temas e histórias**



**Ilustração 24 - Tema com as Historias correlacionadas**

## **Trabalhando com Temas**

Como trabalhar com cartões de História que possuem vários requisitos?

Vamos pegar o exemplo:

- Como um cliente, quero consultar os livros, para que eu possa encontrar o produto que desejo comprar.
  - Conversa: Título, ISBN, categoria, autor, gênero

Uma boa pratica é sempre buscar o equilíbrio das coisas, neste cartão podemos executar 2 técnicas, onde a primeira é gerar vários cartões menores.

Exemplo:

- Como um cliente, quero consultar os livros por titulo, para que eu possa encontrar o produto que desejo comprar.
- Como um cliente, quero consultar livros por ISBN, para que eu possa encontrar o produto que desejo comprar.
- Como um cliente, quero consultar os livros por categorias, para que eu possa encontrar o produto que desejo comprar.

- Como um cliente, quero consultar os livros por gênero, para que eu possa encontrar o produto que desejo comprar.

Quebrar em cartões menores possibilita a criação de testes, o entendimento do cartão e a negociação de escopo com o Product Owner.

A segunda técnica é deixar o cartão original como ele está e por intermédio de conversas identificarmos todos os detalhes (abrangência) do cartão.

Neste exemplo eu prefiro a primeira técnica, no início ele gera mais trabalho para a criação dos cartões com as funcionalidades mais granulares (unitárias), mas esta granularidade permite uma distribuição do serviço pela equipe, uma estimativa melhor, teste mais simples e negociação com o Product Owner.

### **Exemplos de Testes criados para Cartões de História**

Segue uma relação de exemplos de Testes para Cartões de História. O template de como escrever os testes foi tirado do blog do Alexandre Magno.

Referencia: <http://amagno.blogspot.com/2006/09/apresentandofdd.html>

#### **Template: [ação] [resultado] [objeto]**

##### **[Exemplo 1] – [Cartão de História]**

Como um cliente, quero que o produto seja acompanhado com o valor de desconto para compra a vista, para que eu tenha a visibilidade da diferença monetária do produto a vista ou a prazo.

#### **[Testes] - Template: [ação] [resultado] [objeto]**

- Mostrar valor do desconto de venda a vista
- Mostrar percentual do desconto de venda a vista
- Mostrar o valor do produto sem desconto de venda a vista
- Mostrar o valor do produto com desconto de venda a vista
- Validar valor do produto é maior que o valor do desconto para evitar um valor de desconto maior que o valor de venda
- Validar valor do produto maior que zero para evitar produtos sem preço de venda
- Mostrar valor do desconto de venda a prazo
- Mostrar percentual do desconto de venda a prazo
- Mostrar o valor do produto sem desconto na venda a prazo
- Mostrar o valor do produto com desconto na venda a prazo

##### **[Exemplo 2] – [Cartão de História]**

Como um cliente, quero que os produtos selecionados para compra fiquem armazenados como um carrinho de compras, para que eu possa visualizar todos os meus produtos e o preço total.

[Observação] Cartão de História muito grande, devemos quebrar em Cartões menores e criar para cada Cartão menor os testes correspondentes.

**[Testes] - Template: [ação] [resultado] [objeto]**

- Cancelado

**[Exemplo 3] – [Cartão de História]**

Como um cliente, quero que o sistema mostre o valor da postagem, para que eu possa saber quanto vai ser a taxa de entrega.

**[Testes] - Template: [ação] [resultado] [objeto]**

- Mostrar valor da postagem de correio para vendas realizadas na mesma cidade
- Mostrar valor da postagem de correio para vendas realizadas no mesmo estado
- Mostrar valor da postagem de correio para vendas realizadas em estados diferentes
- Mostrar valor da postagem de correio para vendas internacionais
- Mostrar valor da postagem de correio para vendas isentas de taxa de entrega
- Calcular valor da postagem de correio para cada produto existente no carrinho de compras
- Mostrar mensagem de erro para cep's inválidos
- Mostrar mensagem de erro para cep's vazio

**[Exemplo 4] – [Cartão de História]**

Como um cliente, quero ter os comentários referentes aos livros consultados, para que eu possa saber a opinião de quem já leu

**[Testes] - Template: [ação] [resultado] [objeto]**

- Mostrar comentário realizado para livro
- Mostrar mais de um comentário realizado para o livro
- Mostrar inexistência de comentário para um livro
- Visualizar marcador de mais comentários que o valor maximo permitido para visualização para o livro

**[Exemplo 5] – [Cartão de História]**

Como um cliente, quero consultar a relação dos livros mais vendidos, para que eu possa ter uma visibilidade dos livros preferidos pelo mercado

**[Testes] - Template: [ação] [resultado] [objeto]**

- Mostrar a relação dos 10 livros mais vendidos

- Mostrar vazio a relação dos livros mais vendidos
- Mostrar a relação dos livros mais vendidos conforme a customização
- Mostrar a relação dos livros mais vendidos com o valor de customização de quantidade igual à zero
- Remover da relação dos livros mais vendidos os livros que possuem quantidade de venda inferior ao valor informado na customização.

Este teste é um forte candidato a cartão de história

Também nestes testes o conceito “customização” apareceu, e este conceito pode, ou melhor, deve dar origem a um cartão de história

**[Exemplo 6] – [Cartão de História]**

Como um cliente, quero realizar o pagamento em bloqueto bancário, para os pagamentos a vista

**[Testes] - Template: [ação] [resultado] [objeto]**

- Gerar bloqueto com valor da venda
- Checar se o valor do bloqueto é igual ao valor da venda
- Validar data de vencimento tem que ser igual a data de emissão do bloqueto
- Gerar bloqueto para o Banco do Brasil
- Gerar bloqueto para o Banco Caixa Econômica

Dica:

1 – Padronizar os verbos, afinal mostrar e visualizar pode ter o mesmo comportamento.

2 – Padronizar o objeto, desta maneira podemos já identificar nomes fortes que vão dar origem ao nosso modelo de domínio.

## **Templates de Documentos**

Vou apresentar alguns documentos que eu criei para serem utilizados no nosso dia a dia de Visão do Produto, Planejamento de Sprint, Retrospectiva e Review.

Estes documentos devem ser adaptados à realidade de cada projeto, é apenas uma sugestão de uso.

Embora estes documentos não agreguem valor direto ao produto que está sendo desenvolvido, eles garantem uma comunicação e um certo oficialismo na gestão do projeto. Teremos o projeto bem documentado traz confiança entre as partes. A documentação deve deixar claro o escopo, o que foi acordado por Sprint, as entregas realizadas, as melhorias do processo de desenvolvimento da equipe, as alterações de escopo e requisitos e o que for necessário. A questão não é deixar de documentar ou documentar ao extremo e sim a busca do ponto de equilíbrio que permite o projeto fluir e ao mesmo tempo termos os registros de como ele esta sendo executado.

Talvez alguns neste momento estejam achando este tópico deste material inapropriado, mas eu como Gerente de Projetos e já tendo gerenciado vários projetos como Gestor utilizando técnicas do PMBOK ou como um Scrum Máster de equipe, tenho a convicção que é melhor documentarmos do que deixar de lado a documentação. Não ter documentação mais cedo ou mais tarde nos iremos nos arrepender de ter deixado de documentar e quando mais necessitarmos não terá como recuperar os fatores históricos do nosso projeto.

### **Visão do Produto**

Ata de Visão do Produto

Reunião: \_\_/\_\_/\_\_\_\_

Participantes: \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_,

Datas Importantes do Projeto

Final do Projeto: \_\_/\_\_/\_\_\_\_

Release 1: \_\_/\_\_/\_\_\_\_, Release 2: \_\_/\_\_/\_\_\_\_, Release 3: \_\_/\_\_/\_\_\_\_

Temas Apresentados

---

---

---

Expectativa de Temas por Release

Critérios de Sucesso do Projeto

Riscos Identificados

Alinhamento de Tarefas

|                                  |  |
|----------------------------------|--|
| Reunião                          | Data da realização da reunião.   |
| Participantes                    | Pessoas que participaram da reunião.   |
| Final do Projeto                 | Data prevista, desejada, para o termino do projeto.  |
| Release                          | Datas previstas, desejadas, para o recebimento de um grupo de funcionalidades.   |
| Temas Apresentados               | Historias de tamanho igual a temas apresentadas pelo Product Owner.  |
| Expectativa de Temas por Release | Distribuição de temas por release.   |
| Critérios de Sucesso do Projeto  | Identificação de expectativas do Product Owner. Um bom item para identificação de critérios de qualidade do produto.       |
| Riscos Identificados             | Apontamentos de riscos do projeto identificados por todos da reunião.  |
| Alinhamento de Tarefas           | Apontamento de tarefas que são identificadas durante a reunião e que pertencem a equipe de desenvolvimento do projeto ou o |



|  |  |
|--|--|
|  | Product Owner. Estas tarefas necessitam de um responsável para a sua execução. |
|--|--|

## Planejamento da Sprint #1

Ata: Planejamento da Sprint – XX

Data: \_\_/\_\_/\_\_ – \_\_:\_\_/\_\_:\_\_

Participantes: \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_,

Release – \_\_/\_\_/\_\_

Temas do Release

---

---

---

Dados da Sprint

Data de Inicio: \_\_/\_\_/\_\_ - Data de Terminio: \_\_/\_\_/\_\_

Tamanho da Sprint: \_\_ semanas

TimeBox:

---

---

---

Capacidade da Equipe: \_\_\_\_ hrs / sprint

Temas / Historias da Sprint

Tema: \_\_\_\_\_

Historia: \_\_\_\_\_

Conversas:

---

---

---

Tema: \_\_\_\_\_

Historia: \_\_\_\_\_

Conversas:

---

---

---

|                             |  |
|-----------------------------|--|
| Tema: _____                 |  |
| Historia: _____             |  |
| Conversas:                  |  |
| _____                       |  |
| _____                       |  |
| _____                       |  |
| _____                       |  |
| Planejamento da Sprint – XX | Numero da Sprint   |
| Data                        | Data e hora da reunião   |
| Participantes               | Pessoas que participaram da reunião.   |
| Release                     | Data do release que está Sprint pertence. Este é um mecanismo de manter o foco.  |
| Temas do Release            | Temas de historia que fazem parte da realse. Este é um mecanismo de manter o foco.   |
| Dados da Sprint             | Data de início e termino da Sprint. Quantidade de semanas da Sprint. Timebox: Pessoas que vão participar da execução da Sprint. Quantidade de horas por dia que cada profissional vai ter. Total de horas de produção que a Sprint possui.   |
| Temas / Historias da Sprint | <p>Anotações do processo de analise de sistema realizado pela equipe junto com o Product Owner.</p> <p>Um tema pode ter varias historias e cada historia tem que ter as suas anotações de conversas. Estas anotações vão ser confrontadas com o produto entrega na Review da Sprint.</p> |

## **Planejamento da Sprint #2**

Não existe necessidade de um documento de registro da Sprint Planning #2. Caso o desenvolvimento do sistema não seja junto do Product Owner, softwares podem ser utilizados para mostrar o andamento da execução do Sprint Planning #2.

## Review

Ata: Review da Sprint – XX

Data: \_\_/\_\_/\_\_\_\_ – \_\_:\_\_/\_\_:\_\_

Participantes: \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_,

Entregas

---

---

---

Não Aceito ou Observações

---

---

---

|                           |  |
|---------------------------|--|
| Review da Sprint – XX     | Numero da Sprint   |
| Data                      | Data e hora da reunião   |
| Participantes             | Pessoas que participaram da reunião.   |
| Entregas                  | O que esta sendo entregue, sempre com o foco no que foi acordado no Planejamento da Sprint.<br><br>Podemos ter entregas não acordadas.   |
| Não Aceito ou Observações | Para cada item entregue devemos anotar o que não está sendo aceito, ou itens de melhoria. Itens de melhoria vão ser cadastrados como novos itens de backlog. Itens não aceitos são “variáveis” importantes para a reunião de retrospectiva da equipe, para realizarmos a melhoria do processo. |

## Retrospectiva

Ata: Retrospectiva da Sprint – XX

Data: \_\_/\_\_/\_\_\_\_ – \_\_:\_\_/\_\_:\_\_

Participantes: \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_,

Não Continua

---

---

---

Melhorar

---

---

---

Continuar

---

---

---

|                              |  |
|------------------------------|--|
| Retrospectiva da Sprint – XX | Numero da Sprint   |
| Data                         | Data e hora da reunião   |
| Participantes                | Pessoas que participaram da reunião.   |
| Não Continua                 | O que não devemos continuar na nossa Sprint e todos devem ficar atentos em não repetirmos estes itens durante o nosso dia a dia de trabalho. |
| Melhorar                     | O que nos fizemos que ficou bom, deve ser continuado nas próximas Sprint's, porem devem ser melhorados, para ficar cada vez melhor.          |
| Continuar                    | O que estamos fazendo está bom e devemos continuar executando.   |

## **Softwares Para Gerenciamento Ágil de Projetos**

<http://www.targetprocess.com/>  
<http://acunote.com>  
<http://www.versionone.com/communityedition.asp>  
<http://www.danube.com/scrumworks/basic/license>  
<http://code.google.com/p/storyverse/>  
<http://studios.thoughtworks.com/mingle-project-intelligence>  
<http://www.scrumforteamssystem.com>  
<http://www.projectkoach.com/>  
<http://www.xplanner.org/>  
<http://www.dotproject.net>  
<http://www.egroupware.org/>  
<http://www.horde.org/groupware/>  
<http://www.rallydev.com/>  
<http://www.ivis.com/>  
<http://trac.edgewall.org/>  
<http://www.agile42.com/cms/pages/>  
<http://virtual-planner.palepurple.co.uk/>  
<http://trichord.change-vision.com/en/index.html>  
<http://retrospectiva.org/overview>  
<http://www.agilezen.com/>  
<http://www.digaboard.net/>  
[http://confluence.atlassian.com/display/GH/Adding+Constraints+to+your+Task+Board+Columns+\(Kanban\)](http://confluence.atlassian.com/display/GH/Adding+Constraints+to+your+Task+Board+Columns+(Kanban))  
<http://leankitkanban.com/>  
<http://code.qbranch.se/post/listTag?selectedTag=qanban>  
<http://www.toolsforagile.com/>  
<http://www.pivotaltracker.com>  
<http://www.redmine.org>  
<http://pronto.bluesoft.com.br/>  
<http://pangoscrum.com/>  
<http://www.firescrum.com/>