Melhor Scrum

Um conjunto não-oficial de dicas e idéias de como implementar bem Scrum.

Escrito por Peter Hundermark – Instrutor (CST) e Coach (CSC) Certificado Scrum.

Traduzido para o português por Samuel Gonsales – Certificado ScrumMaster (CSM).

Por que este guia?

Jim York, Instrutor (CST) e Coach (CSC) Certificado Scrum, disse:
Scrum é simples. Praticar Scrum é difícil. [SM]

Muitas pessoas que conheço dizem que elas têm dificuldade para saber como iniciar a prática de Scrum. Outros têm equipes que estão seguindo algumas práticas ágeis, mas que ainda estão longe de se tornar o que Jeff Sutherland definiu como hiper-produtivo.

Espero que este pequeno livro possa ser uma fonte de inspiração para ajudá-lo a praticar Scrum e conceitos ágeis melhor. Acima de tudo, espero poder incentivá-lo a percorrer juntamente com sua equipe e toda a sua organização para longe das velhas formas de trabalho que simplesmente não fazer bem, trabalhar e encontrar novos caminhos que levam a obter mais qualidade, entregar com mais velocidade e acima de tudo, que seja mais divertido.

Deixe-me, então, saber o que você gosta e o que não gosta para que eu possa melhorá-lo.

Agora vá e faça!

Peter Hundermark
Cape Town, Novembro de 2009
Segunda Edição

O que é Scrum?

Origens

Scrum é um framework de gerenciamento de projetos que visa o desenvolvimento de produtos complexos. Scrum tem suas origens nas áreas de gestão conhecimento, sistemas adaptativos complexos e teoria de controle empírico de processos. Foi também influenciado por padrões observados no desenvolvimento de Software e Teoria das Restrições.

Scrum e Ágil

Scrum é o mais popular dos métodos ágeis. É freqüentemente utilizado em conjunto com Extreme Programming (XP).

Qual é o problema?

- Releases demoram muito;
- A estabilização leva muito tempo;
- As alterações são difíceis de fazer;
- A Qualidade está caindo;
- Marchas fúnebres estão ferindo a moral;

Durante décadas os desenvolvedores de software têm tentado empregar métodos definidos de trabalho e gerenciamento de projetos. Os métodos definidos são adequados quando as variáveis que entram no sistema são bem definidas e o método empregado gera um resultado previsível.Desenvolvimento de software e outras formas de trabalho complexo não são adequados a tais métodos. E a alta taxa de falhas de projeto e insatisfação dos clientes ilustra isso claramente.

Como o Scrum ajuda a resolver esses problemas?

Alistair Cockburn [Cockburn 2008] descreve o desenvolvimento de software como um "jogo cooperativo de invenção e comunicação".

Metodologias tradicionais de desenvolvimento são baseadas em documentos para capturar e comunicar conhecimento de um especialista ao outro. Os ciclos de feedback são muito longos ou mesmo não existem. Décadas de projetos de baixa produtividade mostram que esta forma de trabalho conduz inevitavelmente ao fracasso.

Scrum fornece uma plataforma para as pessoas trabalharem em conjunto de forma eficaz e expõe implacavelmente todos os problemas tornando-os visíveis em seu caminho.

Melhor Scrum – 2

A Essência do Scrum

A essência do Scrum é:

- A equipe recebe metas claras;
- A equipe se organiza em torno do trabalho;
- A equipe entrega regularmente os recursos mais valiosos;
- A equipe recebe feedback de pessoas de fora;
- A equipe reflete sobre sua forma de trabalhar a fim de melhorar;
- Toda a organização tem visibilidade do progresso da equipe;
- A equipe de gestão comunica aos outros de maneira honesta sobre progressos e riscos.

Esta forma de trabalho é baseada em valores de auto-respeito, respeito pelos outros, confiança e coragem.

Por que Scrum não faz nenhuma menção a qualquer prática de desenvolvimento de software?

As ferramentas de desenvolvimento e práticas mudam e melhoram continuamente e boas equipes irão trabalhar no sentido de obter o melhor uso delas.
Scrum não tenta instruir equipes de como fazer seu trabalho. Scrum espera que as equipes façam o que for necessário para entregar o produto desejado, dando-lhes o poder para fazê-lo. Práticas e ferramentas de desenvolvimento são atualizadas constantemente e consistentemente e boas equipes trabalharão para tirar melhor proveito delas.

Aplicabilidade

Embora seja verdade que o Scrum foi usado inicialmente para desenvolver produtos de software, ele é projetado para qualquer tipo de trabalho complexo. Hoje ele é usado para gerenciar desenvolvimento de software e hardware, publicidade e marketing, igrejas e organizações inteiras.

Como Scrum é mapeado para os métodos tradicionais?

A resposta rápida é que ele não é mapeado. Métodos ágeis e Scrum são baseados em um paradigma diferente. Os fundadores Jeff Sutherland e Ken Schwaber tem repetido em diversas ocasiões que é inútil tentar mapear métodos definidos com um método empírico.

Scrum terá sucesso em minha organização?

Isso depende de você! A implementação na sua organização pode falhar por causa de uma falta de determinação das pessoas para superar os problemas que Scrum certamente irá expor. No entanto, milhares de equipes em todos os continentes e em todos os setores estão conseguindo tornar o seu mundo de trabalho melhor hoje do que era ontem.

Manifesto Ágil

Em Fevereiro de 2001, dezessete especialistas em metodologias leves se reuniram para discutir e tentar chegar a uma definição comum de trabalho. Entre eles estavam Jeff Sutherland e Ken Schwaber, fundadores do Scrum, juntamente com Mike Beedle que trabalhou no desenvolvimento dos padrões iniciais e escreveu o primeiro livro sobre Scrum. Este grupo chamava-se "A Aliança Ágil" (Agile Alliance) e concordaram com um Manifesto para Desenvolvimento Ágil de Software. Eles também definiram um conjunto de doze princípios por trás do manifesto que serão reproduzidos na página seguinte.

Comentário

O Manifesto Ágil e seus doze princípios permanecem intactos durante uma década. Eles permanecem até hoje a melhor maneira de julgar se um método realmente funciona de forma ágil ou não. A maior crítica tem sido a tendência para desenvolvimento de software, embora os métodos ágeis possam ser mais amplamente aplicados.

Nota

Scrum é um dos sabores dos métodos ágeis. A adesão ao Manifesto Ágil e todos os seus princípios não significa necessariamente que uma equipe ou uma organização está praticando Scrum. No entanto, a não adesão a qualquer um destes princípios implica que você não está praticando Scrum (ou metodologia ágil)!

Espaço para anotações

Manifesto para Desenvolvimento Ágil de Software

Estamos descobrindo maneiras melhores de desenvolver software, tanto para nossa própria experiência como para ajudar os outros. Através deste trabalho, chegamos aos seguintes valores:

- Indivíduos e interações são mais importantes que processos e ferramentas;
- Software funcionando é mais importante que documentação extensa;
- Colaboração com o cliente é mais importante que negociação de contrato;
- Responder às mudanças é mais importante que seguir um plano.

Ou seja, enquanto não há valor nos itens à direita, nós valorizamos mais os itens à esquerda.

Princípios por trás do Manifesto Ágil

1. Nossa maior prioridade é satisfazer o cliente através da entrega antecipada e contínua de software valioso.
2. Aceitamos que os requisitos mudem mesmo em estágios avançados de desenvolvimento. Processos ágeis aproveitam a mudança para proporcionar vantagem competitiva ao cliente.
3. Entregar software funcionando freqüentemente, a cada duas semanas ou a cada dois meses, com preferência para o prazo mais curto.
4. Os gerentes de negócios e desenvolvedores trabalham em conjunto diariamente durante o projeto.
5. Os projetos são construídos em torno de indivíduos motivados. Dê-lhes o ambiente e apoio de que precisam, e confie neles para fazer o trabalho.
6. A maneira mais eficiente e eficaz de comunicar informações para a equipe de desenvolvimento é através de conversa cara a cara.
7. O software em operação é a medida primária do progresso do projeto.
8. Processos ágeis promovem o desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante por tempo indeterminado.
9. Atenção contínua a excelência técnica e bom design aumentam a agilidade;
10. Simplicidade, ou a arte de maximizar a quantidade de trabalho não feito, é essencial.
11. As melhores arquiteturas, requisitos e projetos emergem de equipes auto-organizáveis.
12. Em intervalos regulares, a equipe reflete sobre como se tornar mais eficaz, em seguida, ajusta e melhora seu comportamento em conformidade.

Higsmith (2001)

Os papéis de Scrum

Introdução

Não há papel de Gerente de Projetos em Scrum. As responsabilidades do gerente de projeto tradicional são divididas ao longo dos três papéis na equipe Scrum:

- O Product Owner que gerencia o produto (e o retorno sobre o investimento);
- O ScrumMaster que gerencia os processos;
- A equipe que gerencia a si mesma.

Este é um desafio para os indivíduos que atualmente cumprem esses papéis e para os gestores nas organizações em que trabalham. Michele Sliger e Stacia Broderick escreveram um guia útil para a transição do Gerente de Projetos para o treinador de metodologias ágeis [Sliger e Broderick 2008].

Não há líderes designados em equipes Scrum além do Product Owner e ScrumMaster, e isso não é necessário. A necessidade de gestores de linha é reduzida, porque as equipes são administradas numa vasta gama de funções. Não é raro encontrar 50 membros de uma equipe que se reportam diretamente a um único gerente de linha em uma organização que fez a transição para Scrum.

A auto-organização

A auto-organização não implica de forma alguma em uma abordagem *laissez-faire* (do francês – deixai fazer), ao contrário, as equipes auto-organizadas são altamente disciplinadas. Uma vez que elas tenham plena autonomia, esperamos que tenham mais responsabilidade para cumprir os compromissos que assumiram. Elas são encorajadas a assumir riscos razoáveis e de aprender através de fracasso e auto-reflexão. Um alto grau de confiança e compromisso é um resultado automático de equipes verdadeiramente auto-organizadas.

Novas equipes de Scrum vão exigir algum incentivo para explorar seus limites, e superá-los, respondendo aos desafios com maior responsabilidade. Elas freqüentemente precisam superar as armadilhas que muitas vezes impõem as formas antigas e improdutivas que fizeram em seu trabalho durante anos.

A auto-organização não é uma opção em Scrum, é um princípio fundamental.Sem isso, nunca teremos equipes de alta performance. *Caveat emptor*!

Product Owner

As responsabilidades do papel do Product Owner são:

- Trabalhar em uma visão compartilhada;
- Coletar os requisitos;
- Gestão e priorização do Product Backlog;
- Aceitar o software no final de cada iteração;
- Gerenciar o plano de liberação de releases;
- Gerenciar a rentabilidade do projeto (ROI);

Metáfora: O Product Owner é um CEO.

ScrumMaster

As responsabilidades do papel do ScrumMaster são:

- Garantir um ambiente de trabalho protegido para a equipe, livre de interferências;
- Remover impedimentos;
- Manter o processo em movimento e incentivar o uso de Scrum;
- Socializar Scrum para a maior parte da organização;

Metáfora: O ScrumMaster é um facilitador, treinador e mentor!

Equipe Scrum (TIME)

As responsabilidades da função de membro da equipe Scrum ou da equipe são:

- Estimar o tamanho dos itens do backlog;
- Comprometer-se com incrementos de software a ser entregue e entregá-los;
- Acompanhar seu próprio progresso;
- A auto-organização, mas responsável perante o Product Owner para a entrega que foi comprometida.

- Os membros da equipe podem ser desenvolvedores, testadores, analistas, arquitetos, escritores, designers e até mesmo usuários. A equipe é multifuncional, o que significa que todos os seus membros possuem habilidades suficientes para fazer todo o trabalho. Não há liderança definida previamente entre os membros da equipe.

- A Equipe Scrum é composta por todos os três papéis: um Product Owner, um ScrumMaster e cinco a nove membros da equipe.

As reuniões do Sprint

O Sprint é o coração do ciclo de Scrum. É marcado pelo planejamento do Sprint no início e pela revisão do Sprint e retrospectiva do Sprint no final. O comprimento do Sprint é fixado e nunca é estendido. A maioria das equipes escolhe duas, três ou no máximo quatro semanas como sendo a duração do Sprint. Durante o Sprint a equipe realiza uma reunião diária. Cada reunião tem uma duração fixa de tempo. Isto significa ter uma duração máxima para concluir a reunião e não significa que é necessário ocupar todo o tempo fixado. Para um Sprint de 30 dias (ou quatro semanas) geralmente são dedicados 2 períodos para reunião de planejamento, bem como revisão e análise retrospectiva com duração de cerca de quatro horas cada. Para Sprints mais curtos devemos ajustar as proporções de acordo com a duração do Sprint.

Alguns dos principais atributos das várias reuniões são descritos nas seções seguintes. Inicialmente compartilho com vocês algumas experiências pessoais que eu acho que vale a pena contar.

- Acho que duas semanas para cada Sprint é um bom tempo para começar. Depois de três Sprints, deixe a equipe re-avaliar o tamanho do Sprint.
- As equipes precisam de três Sprints para compreender os novos conceitos, quebrar velhos hábitos e começar a agir como uma equipe.
- Nunca faça planejamento de Sprints na manhã de segunda-feira. A equipe ainda não está na sua melhor forma e é o dia mais comum para início de férias e eventuais doenças. Nunca faça qualquer revisão ou retrospectiva na tarde de sexta-feira. A equipe está cansada e pensando no fim de semana. Por isso, é uma boa idéia considerar o início e final de Sprints entre terça e quinta-feira.
- As equipes que executam duas semanas de Sprints podem ser tentadas a fazer as reuniões no início do Sprint em um dia. Em outras palavras, começar o dia com a revisão, em seguida a retrospectiva, depois do almoço fazer a primeira e segunda partes do planejamento. O pensamento é fazer com que todas as reuniões terminem o mais rápido possível de modo que a equipe tenha 9 dias inteiros para fazer o trabalho. Na minha experiência, há dois problemas com essa abordagem:

  ✦ A equipe não acredita que estas reuniões fazem parte do trabalho e de fato
  são a parte mais importante para o sucesso no restante do Sprint!
  ✦ Durante à última parte do dia de planejamento o time está mentalmente exausto.

  No entanto, como sempre, deixe a equipe experimentar esse modelo se assim o desejarem!

Sprint Planning - Parte 1

A primeira parte do planejamento do Sprint (PS1) é realmente um workshop para detalhar os requisitos. O Product Owner apresenta o conjunto de requisitos que quer que sejam implementados e a equipe faz perguntas necessárias para entender os requisitos em detalhes suficientes para entender o esforço necessário para entregar essas funcionalidades no final do Sprint.

A equipe decide o que pode entregar no Sprint, tendo em conta a duração Sprint, o tamanho da equipe e as habilidades atuais de seus membros, a sua definição de pronto, todos os feriados conhecidos ou dias de férias e quaisquer ações ou compromissos assumidos durante a retrospectiva realizada pouco antes desta reunião.

O Product Owner deve estar presente durante esta reunião para orientar a equipe na direção correta e para responder às perguntas que normalmente são muitas. O ScrumMaster deve assegurar que quaisquer outras partes interessadas necessárias para ajudar a equipe a entender os requisitos está presente ou de plantão.

Todos os novos itens do backlog serão desenvolvidos durante o Sprint atual e o que não estava estimado anteriormente será dimensionado imediatamente durante esta reunião. Esta, porém, não pode ser uma desculpa para evitar preparação do backlog - veja abaixo!

No final do SP1 a equipe se compromete com o Product Owner o que eles acreditam que pode entregar testado e funcionando. Uma equipe experiente pode usar seu histórico de velocidade para prever o que conseguirá entregar ('tempo passado'). Isto é conhecido como taxa baseada em planejamento. Minha recomendação para a maioria das equipes é fazer compromissos baseados em planejamento.

O conjunto de itens do backlog que foram confirmados pela equipe deve ser chamado de Product Backlog selecionado.

Sprint Planning - Parte 2

Se a primeira parte do planejamento do Sprint é um workshop de requisitos, a segunda parte do planejamento do Sprint (SP2) é um workshop de desenho.

Nesta reunião a equipe colabora para criar um desenho de alto nível dos recursos que se comprometeu a entregar. Um resultado desta reunião é o Sprint backlog, ou lista de tarefas que a equipe precisa executar coletivamente afim de entregar funcionalidades testadas e funcionando. Este conjunto de tarefas é chamado de Sprint backlog e é mais freqüentemente representado por uma lista de tarefas.

Durante o SP2 a equipe pode ter dúvidas adicionais sobre os requisitos. O ScrumMaster deve garantir que o Product Owner e, se necessário, outras partes interessadas estejam de plantão para respondê-las.

O desenho, como todo o resto no desenvolvimento, é emergente. Além disso, a reunião tem um tempo pré-estabelecido. Por isso, é normal a equipe não ter o desenho perfeitamente concluído nesta reunião e descobrir mais tarefas durante o Sprint. Este não é um sinal de que algo está errado. Eles devem simplesmente pegar um post-it e uma caneta e criar mais tarefas sempre que necessário durante o Sprint.

- Você saberá que o SP2 está sendo executado quando a equipe se reúne em torno do quadro branco discutindo sobre a "melhor" maneira ou sobre a maneira "certa" para implementar um recurso.

Reunião Diária do Scrum

A reunião diária do Scrum é um dos três principais pontos de inspeção e adaptação no Scrum. A equipe se reúne para se comunicar e sincronizar seus trabalhos. Uma vez que a equipe está colaborando, isto é essencial para garantir o progresso contínuo e evitar bloqueios de trabalho. A equipe também irá avaliar continuamente o seu próprio progresso no sentido de atingir sua meta no Sprint.

A reunião diária do Scrum não é para relatar o progresso para o ScrumMaster ou Product Owner ou qualquer outra pessoa interessada. O Product Owner pode participar desde que ele permaneça em posição passiva, falando apenas quando for questionado! O ScrumMaster deve garantir, usando todas as suas habilidades de facilitador, que cada membro da equipe se comprometeu a fazer algum trabalho para as próximas 24horas, que este tem como único propósito ajudar a equipe como um todo a entregar o item seguinte na lista de pendências e que os eventuais impedimentos para fazer este trabalho serão removidos para fora do caminho o mais rápido possível.

O ScrumMaster também deve garantir que a reunião seja restrita a 15 minutos, o que, surpreendentemente, é tempo suficiente.

Surpreendentemente reuniões podem ser eficazes durante esse período curto de tempo.

Jason Yip [Yip 2006] fornece um guia útil para auxiliar o ScrumMasters na realização desta reunião.

Revisão do Sprint

A revisão do Sprint (Sprint Review) é muitas vezes erroneamente denominado de demo.Embora essa reunião inclua uma demonstração das novas funcionalidades que a equipe concluiu durante o Sprint, seu principal objetivo é inspecionar o que a equipe entregou e obter feedback dos participantes para adaptar o plano para o próximo Sprint. Assim o Sprint Review é muito mais do que uma demonstração.

O foco da revisão do Sprint é o produto que a equipe está construindo. Quando perguntado sobre quem deve ser convidado para a revisão de Sprint, eu respondo "o mundo inteiro". Minha intenção aqui é ajudar o ScrumMaster e toda a organização a entender que a atenção direta e feedback de um eleitorado mais amplo da organização é fundamental para maximizar o valor que a equipe vai entregar em sucessivos Sprints. Revisões do Sprint tem muitos resultados possíveis, incluindo o cancelamento do projeto. Na maioria dos casos, a equipe está autorizada a continuar por mais um Sprint e uma meta para este próximo Sprint deve ser definida.

Espaço para anotações

Retrospectiva do Sprint

A retrospectiva do Sprint é a reunião final do Sprint. Segue-se imediatamente após a revisão do Sprint. Nunca devemos omiti-la! Considerando que a revisão do Sprint é focada no produto, a retrospectiva é focada no processo e na forma em que a equipe Scrum está trabalhando em conjunto, incluindo as suas habilidades técnicas e as práticas de desenvolvimento de software e ferramentas que estão usando. Considerando que a revisão do Sprint é aberta a todos, a retrospectiva do Sprint é restrita aos membros da equipe Scrum, que são o Product Owner, os membros da equipe de desenvolvimento e o ScrumMaster. Pessoas externas à equipe, incluindo gerentes de todos os níveis da organização são rigorosamente excluídos a não ser que sejam especificamente convidados pela equipe.

Esta regra deve ser entendida no contexto do objetivo da reunião, que é inspecionar em um nível profundo como a equipe está colaborando e tomar decisões para melhorar. Isso muitas vezes exige introspecção profunda e compartilhamento, que por sua vez requer um ambiente seguro. Em “A diretiva básica da Retrospectiva” (Retrospective Prime Directive) Norman Kerth ressalta o seguinte: "Não importa em que evoluímos, entendemos e acreditamos que cada um fazia o melhor que podia, dado o que sabia na época, suas habilidades, recursos disponíveis e a situação em que estavam." [Kerth 2001].

Boris Gloger [Gloger 2008] fornece um padrão muito simples chamado Retrospectivas de Pulsação (Heartbeat Retrospectives) para que novas equipes possam aprender rapidamente a conduzir reuniões de retrospectivas. Esther Derby e Diana Larsen [Derby e Larsen 2006] fornecem muitas atividades para facilitadores usarem durante essa reunião.

Reunião de Estimativas.

Esta reunião não é mencionada na literatura Scrum, mas é essencial se você quer conseguir um fluxo contínuo dos recursos mais valiosos feitos a partir de suas equipes.

Durante cada Sprint o Product Owner organiza uma ou duas reuniões onde devem participar a equipe e, se necessário, outras pessoas interessadas. Nesta reunião deve-se estimar o impacto de novos itens do backlog ou recalcular o tamanho de itens grandes que devem ser divididos em vários itens pequenos, de forma tal que possam ser desenvolvidos no próximo Sprint.

- As equipes precisam dedicar de 5 a 10% de seu tempo durante o Sprint à preparação para o próximo Sprint e subseqüentes. A reunião de estimativas descrita acima é um bom exemplo. Outros exemplos são as reuniões de planejamento sobre como escrever histórias de usuários. Isso é importante para evitar uma paralisação no limite (fim e início) de cada Sprint. A outra implicação, é claro é que as equipes devem gastar de 90 a 95% de seu tempo fazendo o trabalho do Sprint atual!

Artefatos Scrum

Scrum define apenas quatro artefatos:

- Product Backlog;
- Sprint Backlog;
- Gráfico Burndown;
- Backlog de Impedimento;

Scrum propositadamente não menciona nenhum outro documento e / ou artefato. Isso às vezes leva ao mal-entendido de que as equipes ágeis não precisam fazer qualquer documentação. Devemos treinar equipes para produzir apenas os artefatos que são realmente valiosos para si e para outros no futuro. Isso reduz o trabalho inútil e a desnecessária matança de florestas!

Product Backlog

O product backlog é simplesmente uma lista de itens de trabalho que precisa ser produzida ao longo do tempo. Os itens podem ser adicionados por qualquer pessoa, mas apenas o Product Owner tem o direito para determinar a ordem na qual eles vão ser executados pela equipe. Claro que um Product Owner bom vai negociar isso com as partes interessadas e com a equipe.

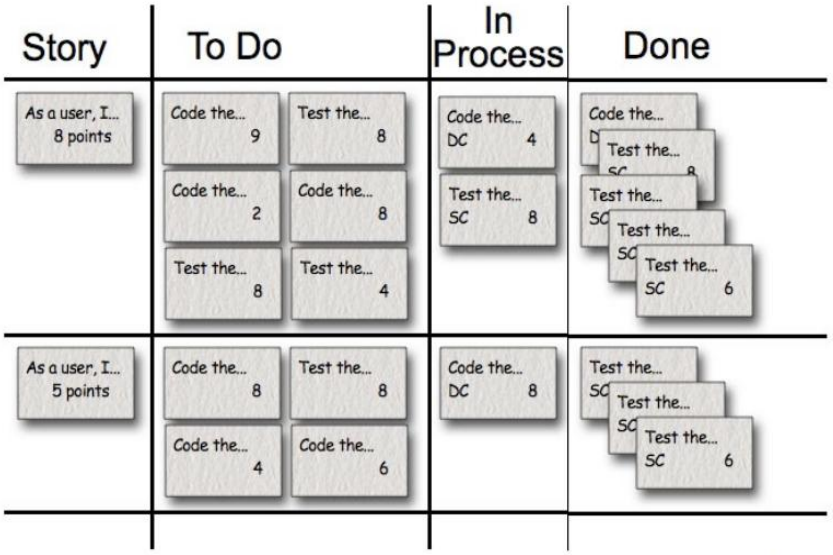
Os requisitos estão emergindo, o que significa que não sabemos ou não podemos saber de antemão todos os itens e cada uma das características que queremos no produto final. É por isso que o Product Backlog é um documento vivo que exige a revisão constante para mantê-lo atualizado e útil ao longo do tempo. Muitos novos itens serão adicionados ao longo do tempo, itens existentes serão divididos em vários itens menores, alguns itens serão removidos a partir da constatação de que não são mais necessários. Além disso, os itens devem ser estimados para determinar o custo-benefício de cada um, que influenciam diretamente na prioridade que cada um terá e seus impactos em caso de atraso.

Em praticamente todos os casos é suficiente, e geralmente preferível, criar e manter um Product Backlog num conjunto de histórias de usuários escritas em cartões de 150 x 100 mm. Ron Jeffries [Jeffries 2005] cunhou a aliteração Cartão, Confirmação, Conversação (Card, Confirmation, Conversation, - os 3 Cs) para descrever como eles devem trabalhar com as histórias de usuários. As histórias são geralmente escritas a partir da perspectiva de um usuário do produto. O livro de Mike Cohn sobre histórias de usuários [Cohn 2004] vai dizer tudo o que você precisa saber sobre elas.

Sprint Backlog

A maioria das equipes vai conhecer o Sprint Backlog como o quadro de tarefas, que é a representação física da lista de trabalho que se comprometeram a fazer durante o Sprint atual.

O quadro de tarefas é um exemplo de um Kanban, palavra japonesa que significa “sinalização visível”. Ele comunica a equipe e qualquer outra pessoa que quiser saber das tarefas planejadas para o Sprint e seu status atual.



Sprint Backlog or Task Board

Adapted from http://epf.eclipse.org

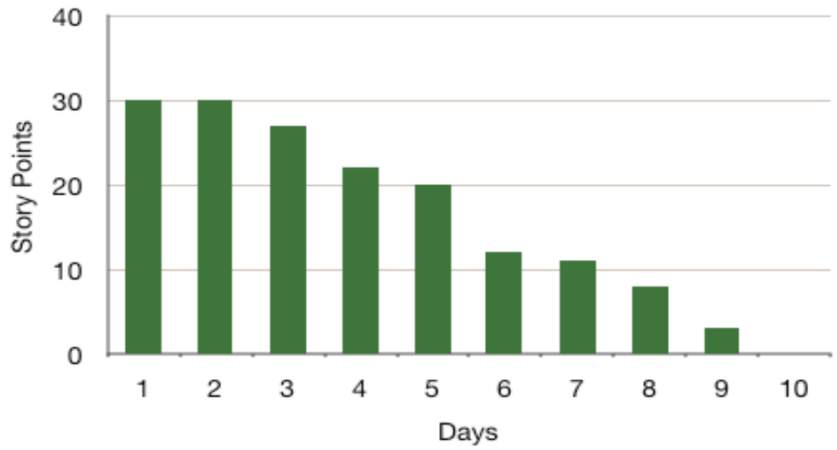
Sprint Burndown Chart

O gráfico de Sprint Burndown é projetado para ajudar a equipe a monitorar seu progresso e ser um indicador importante para saber se eles vão cumprir o seu compromisso no final do Sprint. O formato clássico exige que a equipe estime a duração de cada tarefa em horas e em uma base diária para traçar o total de horas restantes para todas as tarefas não concluídas.

Recomendo que as equipes construam em seu Sprint um gráfico que demonstra os pontos da história que devem ser realizados durante o Sprint atual. A lógica por trás disso é:

- A estimativa de novas tarefas e re-estimativa de tarefas que estão em curso e que exigem um esforço considerável;
- As estimativas são imprecisas;
- A estimativa em unidades de tempo nos faz lembrar dos velhos hábitos de trabalho;
- A conclusão das tarefas não entrega qualquer valor, apenas histórias concluídas entregam valor;

Portanto, em meu trabalho como consultor o Burndown Chart é semelhante ao release ou produto, exceto pelo fato que o eixo X representa dias em vez de Sprints.



Sprint Burndown Chart

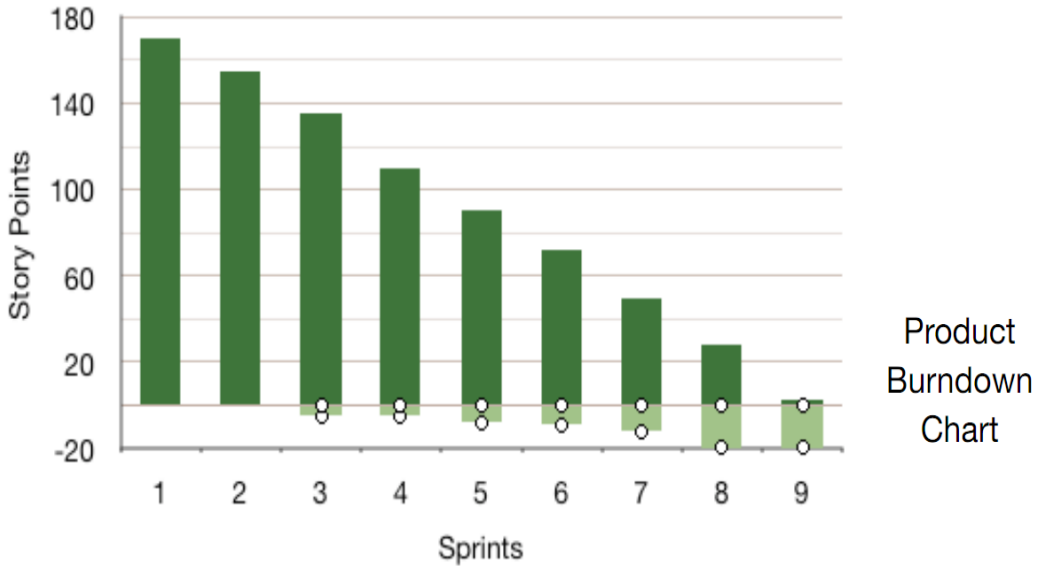
Produto ou Release Burndown Chart

O gráfico Burndown do produto, às vezes conhecido como o gráfico Burndown de release, mede a taxa de entrega de funcionalidades, testadas ao longo do tempo. Esta taxa é conhecida como a velocidade da equipe.

Uma vez que as funções variam em complexidade e, portanto, em tempo e esforço, usamos uma escala para comparar tamanho. A escala mais comum é conhecida como pontos de histórias de usuários. Uma vez que uma equipe tem trabalhado junto por alguns poucos Sprints, geralmente estabelece a sua velocidade dentro de um intervalo definido. Os Product Owners utilizam essa velocidade para prever a taxa que a equipe irá fornecer recursos no futuro, que conduz à planos de liberação de produto e / ou release mais previsíveis.

Aconselhamos que a equipe use uma forma alternativa do gráfico Burndown do produto que permita simultaneamente que os Product Owners controlem as alterações do Product Backlog. Isto é essencial dada a natureza dinâmica desta lista.

Usando esses gráficos os Product Owners são capazes de relatar o progresso, identificar as datas de lançamento do produto e prever a extensão dos mesmos.



Backlog de Impedimentos

O backlog de impedimentos é simplesmente uma lista de situações que estão impedindo a equipe de progredir. Estas são situações que o ScrumMaster deve colocar fora do caminho em sua busca incessante para ajudar a equipe a funcionar melhor. A gama de obstáculos varia de precisar consertar a máquina de café até ter que substituir o CEO! Um bom ScrumMaster tentará remover os impedimentos dentro de 24 horas desde sua identificação (OK, talvez substituir o CEO não se aplique!)

Iniciando Scrum

Ken Schwaber [Schwaber 2007] disse certa vez que não há nada que precise ser feito antes de começar a usar Scrum. Interpretamos isso para dizer que não há adaptação do processo necessário para iniciar. No entanto, praticamente nenhuma literatura auxilia nossa equipe Scrum a alcançar os primeiros sucessos sem ajuda externa.

A melhor coisa que você pode fazer é contratar um treinador (coach) experiente. Se isso não for possível, você pode tentar usar um padrão que já funciona com dezenas de equipes.

Obviamente (eu espero) você precisa de uma equipe Scrum. Isso significa ter um Product Owner, um ScrumMaster e de cinco a nove membros da equipe. Em seguida, siga essa seqüência de etapas, que serão detalhados nas próximas páginas.

1. Treinar a equipe Scrum nos princípios do Scrum;
2. Estabelecer a visão;
3. Escrever histórias de usuário para formar o product backlog;
4. Ordenar os itens do backlog por valor de negócio;
5. Estimar o tamanho dos itens do backlog;
6. Reordenar o backlog, se necessário, com base em fatores adicionais;
7. Criar o plano de release inicial;
8. Planejar o primeiro Sprint;
9. Começar os Sprints!

- Começo o meu trabalho com uma nova equipe, com um treinamento de um dia sobre os fundamentos que eles precisam saber sobre metodologias ágeis e Scrum. Isto é o suficiente para iniciar uma equipe Scrum que tenha um experiente ScrumMaster para apoiá-la durante os primeiros Sprints.

- Os passos 2 a 7 podem ser definidos confortavelmente durante uma reunião de definição do produto que precisa contar com a equipe Scrum inteira. As melhores reuniões são aquelas em que participam as partes interessadas, como arquitetos corporativos, gerentes de negócios e demais interessados no projeto.

Treinamento

No treinamento da minha equipe eu uso um monte de exercícios em grupo e jogos para ilustrar os princípios.

Tento cobrir a totalidade ou a maioria dos seguintes tópicos:

- O poder da auto-organização;
- Processos empíricos contra processos formalmente definidos;
- O valor do trabalho no mesmo espaço físico;
- A importância da confiança;
- Princípios ágeis (Manifesto Ágil);
- O fluxo do Scrum (ciclo de reuniões);
- Funções e responsabilidades (3 principais papéis do Scrum);
- Usando histórias de usuários para os requisitos;
- Estimativas usando Agile Planning Poker;
- Conceitos de tamanho e velocidade;
- Conceito de Pronto!
- Usando um painel de tarefas;
- Acompanhamento dos progressos (gráficos de burndown);
- Simulação de um Sprint inteiro.

Definindo a visão

Katzenberg e Smith [2002] confirmaram que ter objetivos claros é essencial para moldar equipes de alta performance.

O Product Owner geralmente irá partilhar a sua visão do produto. Uma técnica que uso é pedir a cada membro da equipe para escrever a sua versão da visão. Então fazemos uma formação em pares e pedimos para chegar a uma visão que combina para ambos. O processo continua com dois pares de trabalho da mesma maneira e assim, até que a equipe completa conclui com uma visão única.

Este exercício usa o poder do grupo em conjunto, resultando em maior compromisso com a visão obtida. A equipe deve apresentar a visão de modo proeminente em seu ambiente de trabalho. O exercício de definição da visão pode ocupar cerca de três horas.

Criando o Product Backlog

A próxima etapa é a realização de uma reunião para escrever histórias de usuários. É vantajoso envolver partes interessadas no negócio nessa reunião. Certamente toda a equipe Scrum deverá ser envolvida. Claro, o ScrumMaster será o facilitador.

Escrever boas histórias de usuários é simplesmente uma questão de prática. O princípio de Pareto (regra 80/20) também se aplica nessa etapa, assim como se aplica em diversos outros casos.

Mike Cohn [Cohn 2004] tem proporcionado um excelente guia para escrever histórias de usuário. Eu encorajo todos os Product Owners a terem uma cópia desse exemplar sempre à mão!

Template: Em um determinado papel, eu quero funcionalidade por uma razão.
Exemplo: Como um programador, eu quero café para ficar acordado.

Bill Wake [Wake 2003] sugere a utilização do acrônimo INVEST para lembrar os atributos que uma boa história de usuário deve ter.

Independent – Independente – pode ser implementada em qualquer ordem;
Negotiable – Negociável ou não negociável;
Valuable – Valiosa para o cliente;
Estimatable – Suficientemente estimável para classificar e agendar;
Small – Pequeno e com breves descrições;
Testable – Testável – eu poderia escrever um teste para essa funcionalidade.

No mínimo, o backlog deve conter elementos suficientes para a equipe planejar o primeiro Sprint. Comumente, o backlog contém todos os itens que compõem o próximo release do produto.

Como referência, o backlog deve conter 80% de tarefas que possam ser concluídas até o final do primeiro dia. A primeira parte do dia dois pode ser usada para adicionar os 20% restante e para resolver quaisquer questões pendentes. A pausa durante os dois dias é muito importante, pois pode desencadear uma nova reflexão sobre o trabalho e gerar novas idéias.

Ordenando o backlog

O backlog deve ser classificado por valor de negócio. Isto é muito mais fácil de falar do que fazer. Mike Cohn [Cohn 2006] descreve dois métodos para priorizar: o modelo Kano de satisfação do cliente e a abordagem Wiegers de ponderação relativa. O que eu gosto sobre estes dois métodos é que não só consideram o benefício esperado para ter a funcionalidade, mas também leva em conta a penalização por não ter tal funcionalidade. Isto é particularmente útil quando o backlog contém itens técnicos, cujo valor do negócio não é imediatamente óbvio.

No mínimo precisamos do julgamento subjetivo do Product Owner para avaliar uma funcionalidade em detrimento de outra. É preferível uma abordagem quantitativa utilizando uma técnica semelhante ao Planning Poker.

Não importa como será executada a ordenação do backlog é uma das principais responsabilidades do Product Owner.

Estimando o Backlog

A estimativa tem sido sempre a chave para a condução eficaz do planejamento de projetos. Gerentes de projeto passam semanas avaliando modelos complexos com dados históricos.

A dura realidade é que o melhor dessas técnicas não proporciona melhores resultados que técnicas muito simples, como o Planning Poker ou estimativa por afinidade. O Planning Poker funciona em parte porque está embasado em uma sólida teoria, mas principalmente porque as pessoas que fazem as estimativas são as mesmas que realizam o trabalho. Quem teria pensado nisso? O Planning Poker é rápido. Uma equipe com experiência estima em média um item a cada 3 minutos. O Planning Poker é preciso. As estimativas obtidas usando Planning Poker são tão boas quanto às obtidas com métodos tradicionais. E, sobretudo, o Planning Poker é divertido. Remove parte do tédio associado esta fase.
A estimativa por afinidade é ainda mais rápida do que o Planning Poker. É ótima para começar quando temos um backlog grande e o tempo é mais importante do que a precisão e compartilhamento de informações.

No entanto, ainda precisamos entender alguns conceitos-chave sobre estimativa ágil:

‣ Nós dimensionamos os itens do backlog através de comparação com outros itens. Por quê? Porque, como seres humanos encontramos resultados mais naturais e mais confiáveis nas comparações. Portanto, é fácil concordar que "uma determinada história de usuário é o dobro da complexidade de outra história" mesmo que ainda não saiba quanto tempo cada história vai usar para ser implementada.
‣ Nós dimensionamos os itens do backlog usando unidades de complexidade ao invés de tempo. Por quê? Porque permite separar a taxa que uma equipe trabalha a partir do tamanho ou complexidade do trabalho. Isso nos protege de ter que mudar nossas estimativas de acordo com quem faz o trabalho, ou de acordo com as habilidades e capacidades de mudança da equipe ao longo do tempo. Usamos pontos da história de usuário como unidades de medida.
‣ Usamos uma escala não-linear para calibrar que a diferença entre um '1' e um '2' é, obviamente, mais significativa relativamente, do que entre '20' e '21'. Minha preferência é usar os números pseudo-Fibonacci: 1, 2, 3, 5, 8, 13, 20, 40 e 100. E definindo de 1 a 13 como o intervalo de tamanho de características que uma equipe pode entregar em um Sprint. Os números mais elevados são reservados para grandes histórias (épicos) que precisarão ser divididas em histórias menores antes que elas possam ser adotadas em um Sprint.
‣ Relatamos nossas estimativas de tamanho de tempo por meio de velocidade, que é a taxa na qual uma equipe pode entregar funcionalidades testadas ao Product Owner. Dizemos que uma equipe tem uma velocidade de '25', quando eles são capazes de entregar no final de cada Sprint histórias cujos tamanhos, em média, somam 25 pontos.

- Não é uma boa idéia comparar as estimativas entre equipes, a menos que estejam trabalhando no mesmo backlog. Esta é uma questão de escala Scrum e é facilmente resolvida, mas não é um tópico para este pequeno guia.

Reordenando o Backlog

Após o dimensionamento dos itens do backlog podemos descobrir que alguns itens devem ser reordenados. Devemos levar em conta, além do valor de negócio, os seguintes fatores:

- Tamanho: naturalmente vamos optar por implementar uma história pequena e simples frente à uma história grande e complexa, se eles têm valor de negócio similar;
- Aprendizagem: podemos optar por implementar uma história mais cedo que ajudará a equipe a aprender sobre o domínio do negócio ou uma nova tecnologia;
- Risco: podemos optar por implementar uma história no início, porque isso vai mitigar um risco identificado. Os exemplos mais óbvios são os itens que ajudarão a estabelecer limites de desempenho e escalabilidade.

Devemos sempre lembrar que o Product Owner tem palavra final sobre a ordenação do backlog.

Release Planning

Uma vez que tenhamos ordenado e avaliado o backlog, o próximo passo é criar o plano inicial de liberação (Release Planning). Para fazer isso precisamos de uma estimativa da velocidade de nossa equipe. No entanto, como ainda não fizemos nenhum trabalho nesse momento, usamos uma técnica simples, conhecida como planejamento baseado em compromisso.

Primeiro, é claro, devemos ter a certeza de saber o tamanho da equipe durante o Sprint e se qualquer membro da equipe será afastado por licença, etc. Temos que definir o tamanho do Sprint. Geralmente recomendo que seja de duas semanas para uma nova equipe. Temos, então, que criar uma definição de pronto para a equipe - o que dá a equipe a noção de dizer que completou uma história.

O ScrumMaster agora pega o primeiro item a partir do topo do backlog e questiona à equipe: "vocês podem completar este item durante o Sprint?". Ele continua fazendo isso até que a equipe decide não adicionar mais itens ao Sprint. Contando os pontos de história de todos os itens comprometidos, a equipe tem a sua estimativa inicial de velocidade.

Este valor de velocidade é utilizado para distribuir os itens restantes do backlog ao longo de sucessivos Sprints. Esta lista de itens ligados à Sprints é o nosso Release Planning inicial. Ele é exato? Talvez não, mas é provavelmente mais aceitável que qualquer coisa que um gerente de projetos possa produzir em um estágio inicial de um projeto. Uma vez que começamos a trabalhar e completar um Sprint atrás do outro, vamos começar a traçar a nossa velocidade atual e usar isto como um indicador de produção futura. Assim, o plano de lançamento (Release Planning) é uma entidade viva que se torna mais e mais confiável à medida que progredimos.

- Não deixe que o release planning seja desnecessariamente longo. É possível preparar um backlog suficiente para o primeiro sprint em apenas dois dias para quase qualquer combinação de equipe e produto.

Localização física de equipes Scrum

Alistair Cockburn [Cockburn 2007] define o desenvolvimento de software como um jogo de invenção, cooperação e comunicação. O manifesto ágil diz que os desenvolvedores e representantes das empresas devem trabalhar juntos em uma base diária e que conversas frente a frente são a melhor forma de transferir informações.

Sem dúvida o mais importante para uma equipe colaborar de forma eficaz é o compartilhamento no mesmo espaço físico. Um estudo sobre as equipes de desenvolvimento de software com equipes que compartilham o mesmo espaço físico [Teasley, Covi, Krishnan, & Olson, 2000] mostraram que elas tinham duas vezes mais produtividade que outras equipes onde os membros foram separados uns dos outros.

A definição que eu uso para "compartilhar o mesmo espaço físico" é que os membros da equipe estejam sentados no máximo num raio de 6 metros de distância. Essa definição resulta na existência de um limite para o número de pessoas que podem se acomodar em tal espaço. Na prática, este número é de 8 a 10 pessoas, que felizmente corresponde ao tamanho máximo de uma equipe Scrum.

Além disso, cada membro da equipe deve ser capaz de ver todos os outros membros com o simples gesto de virar a cabeça ou virar sua cadeira giratória para o lado. Isto significa que não devem haver divisórias entre as mesas - Adeus Dilbertville!

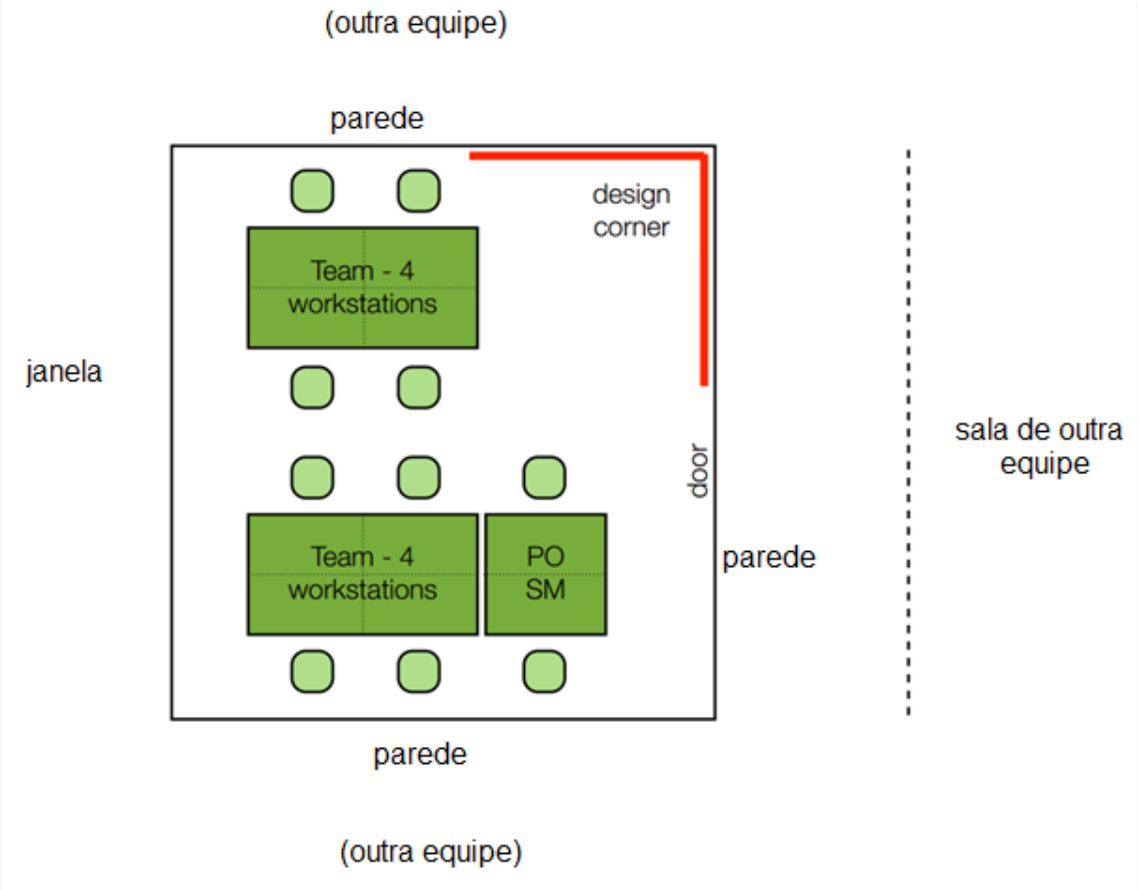
Sem nenhuma boa razão além do hábito, as organizações dificilmente são relutantes em mudar o ambiente de trabalho para facilitar essa colaboração. Isto é, há evidências que mostram que os espaços das equipes não necessitam ser ampliados, podendo manter os layouts atuais.

Qualquer proposta para uma reestruturação do espaço físico pode ser criticada pela administração como uma ameaça a flexibilidade. Isto não é necessariamente a verdade, pois a flexibilidade é necessária na estrutura organizacional, e não no escritório!

Minha experiência é que 90% das novas equipes de Scrum em organizações que se esforçam para praticar metodologias ágeis tem áreas de trabalho simplistas e que estão se esforçando para no horizonte de um ano realizar as mudanças necessárias para desenvolver um ambiente lúdico. E uma vez que essas mudanças são feitas, todos concordam que converge para melhoria imediata e mensurável. Por que isso ocorre?

É com a esperança de que esta resistência absurda diminua com o tempo que ofereço aqui um esquema simples de trabalho para uma equipe Scrum. Esse esquema vai além do escopo deste pequeno guia e para fornecer todo o raciocínio por trás deste projeto, você terá que contratar-me!

- Salas de equipes reduzem significativamente a necessidade de usar salas de reuniões para fazer o Sprint Planning, reuniões diárias e revisões em seu escritório.
- Saiba de antemão que o designer ou arquiteto que trabalha para a sua empresa não vai ser de grande ajuda. Normalmente ele não estudou para criar espaços para o trabalho em equipe.
- O custo das mudanças será recuperado no prazo de uma semana a um mês! Parece inacreditável, não é?



- Alguns parâmetros de montagem da sala:

  ✦ As mesas devem ser retangulares para facilitar o trabalho em pares;
  ✦ Mesas de 1,5-1,8 m x 0,8 m são mesas de bom tamanho;
  ✦ Entre uma mesa e uma parede recomendo 1,2 m de espaço;
  ✦ O canto de desenho deve medir cerca de 3 x 3 m;
  ✦ O tamanho típico da sala deve ter 7 x 8 m = 50 a 60 m²;
  ✦ O ScrumMaster deve ver toda à equipe;
  ✦ O Product Owner tem um local não permanente;
  ✦ As paredes com janelas internas criam barreiras de ruído e dão espaço suficiente para radiadores de informação sem reduzir a luz;

Métricas

É razoável esperar de qualquer gestor que ele queira, dentro de um período razoável, algumas medidas para ser capaz de aferir os resultados de uma equipe ou da transição da organização para metodologias ágeis.

Felizmente, existem métricas que são fáceis e rápidas de implementar. Baseado em minha própria pesquisa e de outros profissionais ágeis, eu descrevi um conjunto de métricas que sua equipe pode e deve implementar [Hundermark 2009]. Logo que a equipe entende o básico do Scrum e tem alguma idéia de sua velocidade - provavelmente no final do seu terceiro Sprint – será o momento para iniciar a medição. Na verdade, você pode executar pesquisas de satisfação com clientes e equipes antes mesmo de começar com Scrum!

Sem mais explicações aqui, o conjunto de métricas que eu recomendo para uma implementação inicial é:

- Pesquisas com clientes e equipes;
- Gráfico de velocidade;
- Gráfico Burndown;
- A execução de testes automatizados;
- Dívida Técnica;
- Trabalho em processo;
- Tempo médio do ciclo de conclusão de histórias de usuários;

E uma vez que a equipe foi capaz de colocar em algum tipo de medida para o valor do negócio, adicione o seguinte:

- Custo por Sprint ou pontos da história;
- Valor real entregue;
- ROI – Retorno sobre Investimento ou NPV – Valor Presente Líquido (Net Present Value).

Espaço para anotações

Coaching

O que faz um treinador (coach)?

> “Coaching Scrum é definido como um compromisso com uma ou mais organizações / equipes durante o qual o coach (treinador) atua como um mentor ou facilitador para as equipes melhorarem sua compreensão e aplicação de Scrum para chegar aos seus objetivos. O trabalho de orientação (mentoring) inclui indivíduos e equipes, facilitação do processo de desenvolvimento, facilitação organizacional, alinhamento e interação com todos os níveis de liderança dentro de uma organização. Pode também incluir orientação em métodos relacionados com a eficácia do Scrum, os princípios descritos no Manifesto Ágil, os princípios Lean, e práticas de Extreme Programming.”
>
> *Certified Scrum Coach Application, Scrum Alliance*

Um treinador (coach) orienta indivíduos dentro de uma organização das seguintes maneiras:

- Conselho e consultoria para melhorar e acelerar o processo de auto-descoberta;
- Facilitar a adoção, implementação e aprendizagem Scrum;
- Liderança Ágil com base em um estilo de liderança servidora;
- Desenvolvimento organizacional para melhorar as habilidades, recursos e criatividade.

Por que minha organização precisa de coaching?

Desenvolvimento de software e outros tipos de trabalhos realizados por trabalhadores do conhecimento se baseia no conhecimento tácito, em oposição com o conhecimento formal ou explícito. O conhecimento tácito é difícil de se transmitir de pessoa para pessoa e geralmente é obtido através da experiência pessoal. Um exemplo poderia ser o de aprender a andar de bicicleta.

O papel do ScrumMaster deve conduzir a adoção e definição do processo, tanto para a equipe Scrum como para o restante da organização. Formações teóricas, como por exemplo curso para Certificação ScrumMaster, é um meio insuficiente para para adquirir o conhecimento tácito necessário para utilizar eficazmente metodologias ágeis e Scrum na prática. A equipe e a organização precisam das habilidades de um profissional experiente, para guiá-los através do labirinto de desafios que enfrentarão durante a transição para uma forma mais ágil de trabalhar.

A resposta mais freqüente quando perguntamos aos gestores das organizações o que fariam para terem uma transição mais eficaz para metodologias ágeis e Scrum é "Faríamos mais Coaching”.

Referências

1. Cockburn, Alistair (2007). Agile Software Development: The Cooperative Game (2nd edition). Addison Wesley

2. Cohn, Mike (2004). User Stories Applied. Addison Wesley.

3. Cohn, Mike (2006). Agile Estimating and Planning. Prentice Hall.

4. Gloger, Boris (2008). Heartbeat Retrospectives. http://borisgloger.com/2008/04/24/heart-beat-retrospectives-1-introduction/.

5. Highsmith, Jim, et al (2001). Manifesto for Agile Software Development. http://www.agilemanifesto.org/.

6. Hundermark, Peter (2009). Measuring for Results. http://www.scrumsense.com/coaching/measuring-for-results.

7. Jeffries, Ron (2001). Card, Conversation, Confirmation. http://www.xprogramming.com/xpmag/expCardConversationConfirmation.

8. Katzenberg, Jon R. And Smith, Douglas K. (2002). The Wisdom of Teams. Collins.

9. Nonaka, Ikujiro and Takeuchi, Hirotaka (1986). The New, New Product Development Game. Harvard Business Review, Jan-Feb 1986.

10. Sliger, Michele and Broderick, Stacia (2008). The Software Project Manager's Bridge to Agility. Addison Wesley.

11. Schwaber, Ken (2007). The Enterprise and Scrum. Microsoft Press.

12. Schwaber, Ken (2009). Scrum Guide. http://www.scrumalliance.com/resources].

13. Teasley, Covi, Krishnan, & Olson (2000). How Does Radical Collocation Help a Team Succeed? Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work (pp. 339 - 346). New York: ACM.

14. Wake, William (2003). INVEST in Good Stories, and SMART Tasks. http://xp123.com/xplor/xp0308/index.shtml

15. Yip, Jason (2006). It's Not Just Standing Up: Patterns of Daily Stand-up Meetings. http://martinfowler.com/articles/itsNotJustStandingUp.html.