Nesta seção você encontra artigos voltados para a prática de métodos ágeis.

Kanban: o ágil adaptativo

Introduzindo Kanban na equipe ágil

De que se trata o artigo?

Este artigo traz uma breve abordagem do modelo Kanban. O objetivo é apresentar o sistema Kanban e explicar sua proposta. Entender o conceito de visualização e o porquê algo tão simples pode fazer uma diferença tão grande na qualidade dos resultados.

Em que situação o tema é útil?

Nos últimos anos o conceito de metodologia ágil vem movimentando o mundo de desenvolvimento de software. Metodologias mais populares como Scrum e XP, criadas nas fabricas de software, vem ganhando cada dia mais espaço nas empresas de tecnologia. Essas ferramentas surgiram com a proposta de melhorar e agilizar os processos envolvidos no desenvolvimento de software, porém no mundo real fica claro que os processos ainda não estão "perfeitos". Mas o que fazer então, sendo que essas ferramentas estão, teoricamente, maduras e eficientes no que se propõe? Como identificar onde estão os "gargalos" que fazem as equipes falharem nos seus sprints? Podemos dizer com que o Kanban pode ajudar a identificar essas falhas e solucioná-las.

Resumo?

O método Kanban para desenvolvimento de software e processos ágeis tem como ênfase não sobrecarregar os membros que compõe a equipe de criação do produto. Por isso, o método contem princípios básicos como: a equipe ou membro deve iniciar uma nova tarefa quando é capaz de realizá-la agora, a equipe deve aceitar mudanças incrementais e evolutivas estimuladas pelo método Kanban e respeitar os atuais processos, papéis e responsabilidades. Neste sentido, este artigo irá apresentar o sistema Kanban e explicar sua proposta.

Flavio S. Mariotti
flaviomariotti@gmail.com
Especialista em Engenharia e Arquitetura de Software. Pós Graduado pelo Instituto de Pesquisa Avançada de Tecnologia IBTA em Engenharia de Software baseado em SOA. Bacharel em Sistemas de Informação pela UNIUBE e técnico em Processamento de Dados pela FEB. Consultor independente no desenvolvimento de software em arquitetura OO, SOA, GIS e Plataforma .NET.

O Kanban é baseado na ideia onde atividades em andamento devem ser limitadas. Um novo item só pode ser iniciado quando o item em andamento é finalizado ou quando uma função automática inicia o mesmo instantaneamente.

O Kanban, basicamente, tem como principal objetivo transformar o trabalho em andamento visível para toda equipe, criando um sinal visual que indica que o novo trabalho pode ou não ser iniciado e se o limite acordado para cada fase está sendo respeitado.

Neste momento, provavelmente você está se perguntando, o que isso tem de interessante? David J. Anderson teve essa mesma sensação e segundo ele "A teoria do Kanban não soa muito revolucionária nem parece afetar profundamente o desempenho, cultura, capacidade e maturidade de uma equipe e a organização na qual está inserida. Mas o impressionante é que afeta! O Kanban parece uma mudança pequena e, no entanto, muda tudo a respeito de uma empresa."

Portanto, o Kanban não é um processo e nem descreve papeis e faces para serem seguidos. Podemos dizer que o Kanban é uma abordagem para mudança gerencial do projeto, um conceito para introduzir alterações em um ciclo de desenvolvimento de software ou gerenciamento de projetos.

Os métodos ágeis fornecem transparência sobre as atividades em andamento e concluídas, e reportam métricas com velocidade. O Kanban, no entanto, vai um passo além e dá transparência ao processo e seu fluxo, expondo gargalos, filas, variabilidade e desperdícios. Portanto, tudo que impacta no desempenho da equipe de produção e para entrega de valor, fica explícito no modelo Kanban.

O que é Kanban?

O nome Kanban é de origem japonesa e sua tradução seria como "sinal" ou "cartão". Portanto, vamos chamar de sinalizador ou melhor "registro visual". O nome Kanban surgiu dos sistemas de cartão usados nas indústrias de produção, que tinham como finalidade o gerenciamento do fluxo de trabalho através da organização de desenvolvimento.

O Kanban, com seu mecanismo de sinalização, tem como objetivo apresentar uma atividade de trabalho em processo, ou seja, o número de atividades ou cartões em circulação é equivalente à capacidade do sistema.

Uma outra característica importante do modelo Kanban é o conceito de "puxar tarefa" quando há capacidade de processá-la. Esse recurso vai de encontro ao tradicional modelo de "empurrar tarefa" conforme sua demanda, mantendo assim o bom desempenho da equipe. Portanto, ao invés dos membros que produzem o produto receberem atividades conforme suas demandas, os requisitos são adicionados a lista de backlog e "puxados" pelos membros que liberam suas atividades correntes e se tornam disponíveis para iniciar uma nova tarefa.

Uma boa metáfora que descreve essa regra é imaginarmos uma rodovia que suporta até 100 veículos para manter o fluxo de trafego com um bom desempenho, porém em todos os feriados essa rodovia recebe em torno de 200 veículos. Essa demanda não suportada pela rodovia gera um congestionamento afetando consideravelmente o desempenho do trafego. Logo, não adianta empurrar um numero de atividades não suportada pela equipe, isso irá causar um "congestionamento" e afetar o desempenho de produção.

A implementação do modelo Kanban se resume em três etapas que são:

- Visualizar os processos;
- Limitar o trabalho em processo do inglês WIP (work in progress);
- Gerenciamento do lead-time, ou seja, tempo que a atividade leva para passar por todas as fases até a sua entrega.

O sistema Kanban

Para entendermos a proposta desde conceito, vamos primeiro estudar o sistema Kanban. Vamos chamar as tarefas que compõe o painel Kanban de cartões. O número de cartões representa a capacidade limite acordada em cada fase de um sistema que são colocadas em circulação.

Cada cartão funciona como um mecanismo de sinalização e o sistema só permite iniciar uma nova tarefa quando um cartão está disponível. É muito importante respeitar essa regra, e fazer com que qualquer novo trabalho espere em uma fila até que um cartão se torne disponível.

O sistema Kanban fornece um método simples, barato e fácil de implementar e rapidamente começa a apresentar resultados permitindo gerenciar o limite de atividades em andamento e garantindo o bom desempenho da equipe.

Afinal, por que usar um sistema Kanban?

Ao entender a proposta de um sistema Kanban, se torna simples perceber que o uso de um sistema prepara e limita o trabalho em andamento para uma capacidade suportada pela equipe. Esse recurso proporciona o equilíbrio da demanda de uma equipe controlando o seu rendimento, e consequentemente, acelerando sua produção.

É simples deduzir que todas as pessoas produzem mais quando conseguem equilibrar a vida pessoal e profissional. O Kanban buscar atingir um ritmo sustentável de desenvolvimento para que todos os indivíduos possam alcançar esse objetivo entre vida pessoal e profissional. Segundo David J. Anderson, "O Kanban rapidamente elimina as questões que prejudicam o desempenho, e desafia uma equipe para se concentrar em resolver essas questões a fim de manter um fluxo constante de trabalho".

O Kanban atua fornecendo visibilidade nos processos, deixando explícito os problemas e prendendo o foco da equipe em qualidade. Portanto, este comportamento reflete os defeitos, pontos de sobrecarga, custos econômicos sobre o fluxo de rendimento e a variabilidade. A simples regra de limitar os trabalhos em andamento no sistema Kanban estimula maior qualidade e maior desempenho na execução de cada tarefa.

O Kanban, com a combinação de fluxo, contribui para a redução do estresse da equipe e melhora a previsibilidade e colaboração, refletindo com isso, nas datas de vencimento para entrega de tarefas. Com a equipe produzindo e cumprindo os prazos de liberação, o Kanban ajuda a fortalecer os laços de confiança dos clientes, parceiros, fornecedores e outras entidades relacionadas.

Ao aplicar o Kanban, respeitando suas pequenas exigências, o sistema tende a contribuir para a maturidade da equipe, podendo até afetar a cultura organizacional da empresa. Com a identificação de falhas, a equipe consequentemente concentra-se em uma força tarefa para resolvê-las, e por contar com maior contribuição da equipe, a tendência é de prevenir problemas futuros.

Por conta desta "filosofia", o Kanban vem mostrando eficiência e ganhando diariamente diversos profissionais que se renderam aos benefícios proporcionados por ele.

Aplicando o sistema Kanban no desenvolvimento de software

Para o desenvolvimento de software, é comum o uso de um sistema Kanban digital. Porém, pode-se manter o conceito de painel físico e digital, isso é reconhecido como boa prática uma vez que ele mantém o princípio de sinalização visual.

Algumas empresas tem implementado o Kanban físico utilizando lousas, painéis, paredes ou tabuleiros. Na verdade, não existe um objeto recomendado para usar, o importante é que este painel seja visível, atingindo o conceito de sinalizador visual. Ainda neste artigo serão apresentados alguns modelos de painel Kanban.

É importante lembrar a alguns profissionais que vem usando paredes ou até mesmo portas do escritório para sinalizar as atividades em andamento que, mesmo essa técnica conseguindo servir como sinalizador visual das atividades em andamento, não podemos considerar isso um modelo Kanban. Para ser um sistema Kanban é necessário existir a ideia de puxar tarefas, conforme o limite acordado em cada fase, ou seja, para se tonar um sistema Kanban é necessário aplicar as três etapas cruciais que são: criar o painel de visualização, limitar os processos WIP e gerenciar o lead-time, aplicando o conceito de puxar uma nova tarefa quando um cartão está disponível.

Priorização

Ao aplicar os três primeiros passos para a implementação do modelo Kanban, os resultados tendem a aparecer com: códigos de alta qualidade, lead-time de desenvolvimento relativamente curto, e controle do desempenho de produção.

O gerenciamento do limite deve ser feito de forma rígida, evitando a priorização de exceções imprevistas no negócio, e focando no desenvolvimento dos itens conforme acordado na estratégia do projeto. É recomendado que a atenção da gestão seja mais dedicada para melhorar a capacidade e previsibilidade de entrega.

Buscando a maturidade na produção

Para alcançar o adjetivo maturidade, é fundamental que a equipe primeiro busque aprender a construir códigos de alta qualidade e equilíbrio no trabalho em andamento para cumprir suas datas de entrega.

A busca pela qualidade está conectada com a velocidade no nível de produção. O desempenho da equipe de desenvolvimento pode ser fortemente beneficiada com a eliminação de retrabalhos, com isso, a equipe pode alcançar um ritmo de produção de alta performance.

Comportamento emergente com Kanban

O Kanban foi implementado na Corbis em 2007 pelo seu idealizador David J. Anderson. Este trabalho resultou em uma lista de comportamentos emergentes que vem crescendo rapidamente com novas implementações Kanban. É provável, e esperado, que esta lista cresça à medida que aprendemos mais sobre os efeitos do Kanban nas empresas. Os comportamentos que preenchem a lista atualmente são:

1. Processos limitados e adequados para cada fluxo do projeto;
2. Desenvolvimento sem a necessidade de iteração;
3. Gerenciamento do custo de implementação;
4. Valores otimizados para classes de serviços;
5. Gerenciamento de risco com alocação de capacidade;
6. Gestão quantitativa;
7. Tende a atingir outros departamentos;
8. Mescla pequenas equipes e proporciona um maior grupo de trabalho.

Sinalizador visual Kanban

O sinalizador visual Kanban funciona como uma ferramenta de sinalização de processos, deixando explícito o fluxo de valor através do processo em andamento. Para os adeptos ao Scrum, o quadro Kanban pode ser comparado ao recurso de quadro/placa Scrum para visualização de tarefas.

Assim como a sequência de colunas que representam os diferentes estados de uma tarefa existente durante o processo de desenvolvimento, o cartão ou sinalizador Kanban é movido de uma fase ou estado para outro, até que tenha sido aprovado para entrega.

Um quadro simples representando o sistema Kanban pode conter as seguintes etapas: análise, desenvolvimento, aceitação e implantação. Esse modelo, à primeira vista, pode lembrar o conceito da engenharia de cascata, porém na prática, o Kanban não atua como o cascata e evita os problemas decorrente do conceito. O Kanban tem como linha de produção a regra de limitar o processo em andamento, o WIP, essa regra evita as falhas apresentadas pela engenharia cascata.

A teoria do sistema Kanban no quadro visual é aplicada com a regra em que cada coluna terá um WIP estabelecido e representados pelo número máximo de cartões em cada fase. O cartão é composto por uma breve história do usuário, descrevendo seus requisitos. Todo cartão entra na fila de backlog e aguarda a liberação de capacidade para entrar nas colunas seguintes. Quando as atividades envolvidas com o cartão na coluna em andamento são finalizadas, o mesmo é movido para a coluna seguinte, liberando espaço para entrada de um novo cartão.

O procedimento aplicado acima gera o conceito de "puxar cartões" para inicialização. A prioridade dos cartões a serem iniciados deve seguir as exigências e estratégias do projeto.

Exemplos de sinalizadores visuais Kanban

O quadro de sinalização visual do Kanban é uma das principais etapas propostas pela ferramenta, porém, cabe ressaltar que ao aplicar o limite de trabalho em andamento e determinar o lead-time, é importante customizar o quadro conforme suas necessidades.

Nesta etapa do artigo, será apresentado um modelo de quadro Kanban. Este exemplo será customizado conforme as necessidades da equipe. O importante é respeitar as poucas políticas exigidas pelo Kanban, e depois customizar na tentativa de acelerar e aperfeiçoar o conceito de comunicador visual.

A **Figura 1** ilustra um modelo simples de sinalizador visual Kanban. Nesta representação, fica fácil identificar o limite de cartões estabelecidos para cada fase. Este limite está representado pelos números em vermelho no cabeçalho. Os cartões ilustrados pelos retângulos representam uma breve história dos usuários, ou seja, as demandas. As imagens com formato de rosto representam os responsáveis pelos trabalhos em andamento. Portanto, este exemplo aplica as três etapas cruciais para obter os benefícios alcançados com o sistema Kanban.

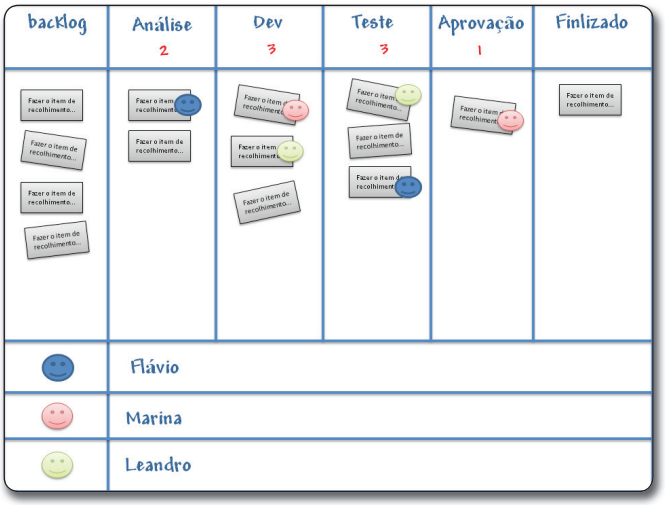

Figura 1. Ilustração de um sinalizador visual Kanban

É importante ressaltar que o desenvolvimento de um quadro Kanban irá evoluir conforme as necessidades de cada organização. Algumas empresas vêem incluindo uma coluna chamada "Refletir". Esta fase propõe uma reflexão por cada cartão que chega ao estado final do processo. Esta coluna é adicionada ao quadro na tentativa de aplicar melhoria continuada em todos os processos.

Mattias Skarin publicou recentemente em seu blog dez diferentes quadros de visualização. Na apresentação de cada modelo proposto por Mattias Skarin, fica clara a evolução dos sinalizadores conforme a necessidade de cada equipe. Conheça mais acessando o endereço: http://blog.crisp.se/2009/11/16/henrikkniberg/1258359420000.

Trabalhando com processos limitados

O Kanban vai além de rastrear e demonstrar visualmente o progresso de uma atividade em andamento. No Kanban, o conceito de limitar o que deve ser feito é aplicado em todas as colunas do quadro. Essa é uma maneira rápida de reduzir o lead-time.

Para os usuários do Scrum, essa é uma diferença fundamental entre o quadro Scrum e o quadro Kanban. Um dos desafios comuns enfrentados com o Scrum é o atraso na entrega conforme o sprint. A entrega com atraso apresenta riscos e tende a afetar o desempenho da produção, afetando significativamente o resultado de valor entregue ao cliente.

O Scrum propõe aos membros da equipe a trabalharem juntos em apenas uma necessidade antes de iniciar um novo item. O Kanban aplica essa orientação de forma implícita e explícita, definindo um limite no número de itens em andamento. Ao limitar a quantidade de trabalho em andamento, a equipe é, consequentemente, forçada a colaborar na busca por solução nos itens que apresentam riscos para o desempenho do desenvolvimento.

Outro benefício alcançado com a aplicação de limites de trabalho em andamento é o ganho do conceito de puxar novos itens, o que garante que nunca a demanda excede a capacidade de produção. É recomendado que os limites sejam estabelecidos pela equipe em colaboração e a equipe de administração ou gestão do projeto. Isso contribui para otimização no fluxo de trabalho. Essa colaboração também implica na gestão alinhada com a estratégia do negócio e proteção do limite WIP.

Benefícios alcançados com o Kanban

Alguns estudos vem mostrando os diversos benefícios alcançados pelas equipes que adotaram o Kanban. Algumas vantagens observadas são: falhas tornam-se claramente visíveis em tempo real; o benefício de encontrar os "gargalos" faz com que as pessoas passem a colaborar ainda mais para a cadeia de valor em vez de apenas fazerem a sua parte.

Um outro aspecto interessante do modelo Kanban é que ele fornece uma evolução gradual do processo cascata para o modelo de desenvolvimento ágil de software. Com isso, vem conquistando as empresas que ainda não tinham se rendido às metodologias ágeis. O fato de poder fazer desenvolvimento de software ágil, sem necessariamente ter que usar o time-box, iterações e sprints de Scrum, torna o modelo mais amigável e fácil de ser adotado.

Outro benefício relevante observado com o uso do Kanban é que, naturalmente, o conceito tende a se espalhar para outros departamentos da organização, aumentando a visibilidade de tudo o que está acontecendo na empresa.

Combinando métodos ágeis

Uma dúvida frequente nas discussões e fóruns sobre metodologias ágeis é: posso utilizar Kanban junto com meu processo atual? A resposta é simples e positiva, SIM. O Kanban vem com a proposta de agregar. Portanto, o primeiro passo é visualizar o processo atual adotado pela empresa, e implementar os conceitos Kanban para encontrar os gargalos existentes no processo.

Mitos e verdades do Kanban

Como toda metodologia, o Kanban já vem recebendo algumas características que não condizem com a realidade. Uma delas é o mito que o Kanban não é um processo com iterações. Na verdade, a iteração Kanban pode ser usada se necessária. Esse recurso é opcional, o importante é fazê-lo somente se existe uma necessidade em seu contexto.

Outro mito comum sobre o Kanban é dizer que não se usa estimativas, na verdade esse também é um item opcional, e requer cuidado com o uso desse recurso.

Um erro comum visto em debates sobre Kanban é dizer que esse modelo é melhor que Scrum, XP, RUP e etc. O Kanban é apenas mais uma ferramenta do processo, e não há tal comparação para determinar qual é melhor ou pior. Outro erro é dizer que o Kanban veio para substituir as tradicionais metodologias ágeis. Novamente cabe lembrar que o Kanban é apenas um recurso que interfere sobre o gerenciamento de fluxo de trabalho, portanto, sua proposta não é substituir nenhuma ferramenta, e sim, implementar os conceitos de mudança de unidade, aplicando o modelo visualização, limites de WIP e evoluir com seus resultados.

Conclusão

Com este artigo podemos concluir que o Kanban permite de forma efetiva visualizar o fluxo de trabalho e dividir o trabalho em partes, escrevendo cada item em um "cartão" e incluindo ele no painel de visualização.

O uso de colunas nomeadas atua como sinalizador, ilustrando onde cada item está no fluxo de trabalho. A aplicação de limite de trabalho em progresso (WIP work-in-progress) em cada coluna contribui para gestão e diminuição do lead-time.

É provável que diversas equipes de software adotem o Kanban, sendo que algumas podem adotar o Kanban definitivamente, enquanto outras equipes usarão Kanban no nível de portfólio de projetos, continuando a utilizar outras metodologias no nível de equipes pequenas.

Kanban ainda é uma ferramenta muito nova e vem se estendendo desde pequenas equipes para o projeto de portfólio até o fluxo de valor da organização. A verdade é que várias empresas vem buscando alinhar seus esforços e ganhar vantagens competitivas em seus mercados, e o Kanban, sem dúvida, pode ser uma ferramenta de auxilio na busca de uma produção com alto desempenho.

Links

Limited WIP Society
http://www.limitedwipsociety.org/

InfoQ - Kanban
http://www.infoq.com/Kanban

Kanban and Scrum making the most of both
http://www.infoq.com/minibooks/kanban-scrum-minibook

David J. Anderson, Kanban: Successful Evolutionary Change for Your Technology Business, 2010.

Jim Benson; Tonianne DeMaria Barry, Personal Kanban, 2011.

John M. Gross; Kenneth R. McInnis, Kanban Made Simple, 2008.

Dê seu feedback sobre esta edição!

A Engenharia de Software Magazine tem que ser feita ao seu gosto.
Para isso, precisamos saber o que você, leitor, acha da revista!
Dê seu voto sobre este artigo, através do link:

www.devmedia.com.br/esmag/feedback

Nesta seção você encontra artigos voltados para a prática de métodos ágeis.

Kanban no desenvolvimento de projetos de software

Entendendo os desafios e a receita para o sucesso

De que se trata o artigo?

Neste artigo será apresentado o método Kanban, uma interessante e simples abordagem para monitoramento e melhoria de processos de software que tem uma forte inspiração no Sistema Toyota de Produção.

Em que situação o tema é útil?

Se a sua empresa ou a sua equipe está com dificuldades para melhorar a forma de trabalho, não importando se ela usa ou não métodos ágeis, o Kanban pode ser uma das melhores alternativas para fazer isso com o mínimo de resistência. Além de ser um método muito simples, comparado à maioria das metodologias/frameworks de desenvolvimento atuais, suas propriedades promovem a colaboração.

Resumo?

O método Kanban para desenvolvimento de software a cada ano ganha mais destaque na indústria de TI, pois vem conseguindo promover a melhoria contínua no processo de trabalho de muitas equipes de empresas das mais variadas áreas de negócio e tamanho. Nesse artigo é apresentada uma introdução ao método Kanban, destacando os seguintes tópicos:

- Histórico e as motivações que levaram David Anderson a fazer a adaptação do método da indústria de manufatura (Toyota) para a indústria de software;
- As cinco propriedades centrais e o funcionamento do método;
- Kanban pode ser considerado um método ágil?
- Um guideline para ter sucesso com o método Kanban, focando principalmente no que deve ser priorizado para se obter as melhorias mais significativas mais cedo.

Thiago Ghisi

thiago.ghisi@gmail.com / @thiagoghisi

Pós-graduando em Gestão de Negócios (UNISUL). Bacharel em Ciência da Computação (UNISUL, 2011). Técnico em Redes de Computadores (SENAI, 2005).

É consultor certificado para implementação do MPS.BR e Sun Certified Programmer for the Java Platform, SE 6. Há 6 anos no ramo de tecnologia e desenvolvimento de software, já atuou como: Professor Voluntário de Informática, Técnico em Informática, Pesquisador de Iniciação Científica, Desenvolvedor, QA, Gerente de Projetos, Analista de Sistemas e de Requisitos, Auditor de Garantia da Qualidade (PPQA) e Analista de Processos. Possui experiência na definição e implantação de processos aderentes ao CMMI e ao MPS.BR e, em avaliações MA-MPS nível F e SCAMPI Classe A nível 2. É entusiasta em desenvolvimento ágil desde 2007. Atuamente, trabalha na Nexxera Techpeople, em Tubarão, SC.

Não há mais dúvidas de que a indústria de software é uma das mais importantes atualmente. O mercado brasileiro de software e serviços de TI, segundo o último relatório da ABES [9], é de US$ 19 bilhões e cresce de 25% a 30% ao ano desde 2004. Porém, grande parte do software produzido ainda é defeituoso, inadequado aos desejos do cliente, entregue fora do prazo e acima dos custos esperados.

Observando esses e muitos outros problemas, há mais de 10 anos, 17 profissionais da área escreveram e assinaram um manifesto: o manifesto para desenvolvimento ágil de software.

Neste, todos concordaram com alguns valores comuns, dentre eles:

- As metodologias ágeis concentram-se nas pessoas envolvidas na produção;
- Os planejamentos em longo prazo são falhos;
- É mais importante aceitar e adaptar-se a mudanças do que seguir planos rígidos;
- Software funcionando é o melhor indicador de progresso nos projetos de software.

Tendo como base grande parte dos princípios desse manifesto, bem como a filosofia Lean do Sistema Toyota de Produção (TPS – Toyota Production System), surgiu o Kanban para Desenvolvimento de Software.

A filosofia Lean tem como objetivo a constante identificação e a eliminação de qualquer espécie de desperdício no sistema de produção.

A história do Kanban para desenvolvimento de software, assim como a história da grande maioria das metodologias, modelos de maturidade, processos de desenvolvimento e processos em geral, começa com um grande desejo, muitas ideias, testes (na prática) e diversos ajustes (o método "científico") até atingir os primeiros casos de sucesso.

A principal diferença do Kanban para as demais metodologias de desenvolvimento de software atuais é que ele foi um modelo adaptado de outra indústria, a de manufatura, mais especificamente da Toyota. David Anderson [1] foi o grande responsável por essa adaptação.

A história começa em 2002, quando Anderson, cansado de ver equipes de desenvolvimento e departamentos inteiros de TI à mercê de outros departamentos, decide voltar seus esforços para responder a duas perguntas [1]:

1. Como proteger a minha equipe da demanda incessante de negócio e alcançar o que a comunidade ágil chama de ritmo sustentável?
2. Como adotar uma abordagem ágil em toda a empresa e superar inevitáveis resistências à mudança?

Como cita em seu último livro, Kanban: Successful Evolutionary Change for Your Technology Business [1], publicado em 2010, Anderson tinha um grande desejo: encontrar no setor de TI uma relação "ganha-ganha" entre o departamento de negócio e as equipes de desenvolvimento de software e TI.

Na tentativa de atingir esses objetivos, David idealizou, testou e falhou muitas vezes nas diversas organizações em que trabalhou. Ele notou que implantar um processo de desenvolvimento de software totalmente prescritivo, na maioria das vezes, não funcionava. Assim, chegou à conclusão que um processo precisava ser adaptado para cada situação, e que para fazer isso, era necessária uma liderança ativa em cada equipe. Porém, esta era muitas vezes inexistente. E, mesmo com uma liderança certa, ele duvidava que mudanças significativas acontecessem sem uma metodologia ou, no mínimo, sem orientações de como adaptar o processo para atender a diferentes situações.

Para David, sem um conjunto mínimo de orientações para guiar o líder, treinador ou engenheiro de processo, qualquer adaptação no processo estava susceptível a ser aplicada subjetivamente. Novos times sempre irão resistir a mudanças se você empurrar para eles processos que foram feitos para outras realidades e que tiveram bons resultados lá.

A conclusão que David chegou é que é preciso ter uma evolução com o novo time de uma forma incremental, partindo do processo que é atualmente seguido. Isso porque: "Cada time é diferente: diferentes conjuntos de habilidades técnicas, capacidades e experiência. Cada projeto é diferente: orçamento, cronograma, escopo e riscos diferentes. E, cada organização é diferente: o processo de produção de software é diferente em cada área de negócio." [1].

Esse é o principal motivo pelo qual o Kanban é um framework para melhorias. Isto é, ele orienta que o processo de trabalho deve ser customizado em cada time de cada projeto de cada organização. Ou seja, um processo não deve ter suas práticas seguidas à risca da mesma forma em todos os times, de todos os projetos, de todas as organizações do mundo, como a grande maioria das metodologias do mercado prescreve.

Em 2005, o método de trabalho de David Anderson era baseado principalmente na Teoria das Restrições (TOC – Theory of Constraints) e na FDD (Feature Driven Development).

A Teoria das Restrições foi apresentada pela primeira vez em 1984 por Eliyahu M. Goldratt, no famoso livro A Meta. Segundo David Anderson [1], "a habilidade de identificar gargalos em um sistema é o primeiro passo para entender a Teoria das Restrições". O efeito dos sistemas puxados, ou, processo de produção puxado são tópicos que também ajudam a compreender melhor a teoria. Nele, a saída de produtos acabados, tal como o software pronto para ser usado, ao final do processo de desenvolvimento, dita o ritmo da introdução de novos requisitos no sistema. Isso evita acúmulos de produtos inacabados ao longo da linha de montagem, diminuindo a quantidade de trabalho em progresso. Já a FDD é uma famosa metodologia ágil que o próprio David ajudou a criar.

Mais tarde, em 2007, após fazer algumas customizações na sua forma de trabalho inspiradas em práticas do Sistema Toyota de Produção, David apresentou nas conferências "Lean New Product Development" e "Agile 2007" os resultados preliminares do uso de Kanban na Corbis, uma empresa fundada por Bill Gates, da Microsoft.

Durante certo período, David chegou a ter dúvidas a respeito da eficiência do Sistema Toyota de Produção, mesmo com muitas pessoas falando o contrário. Porém, após conhecer um pouco mais a respeito do pensamento de Taiichi Ohno, um dos criadores de tal sistema, e a ideia por trás da cultura Kaizen (a qual será falada no próximo tópico), David reconheceu, [1] através de experiências ao longo dos cinco anos posteriores a eficiência desse sistema que originou o Kanban.

"Kanban (com K maiúsculo) é um método de mudança evolutivo para monitoramento e melhoria de processos de produção, que utiliza kanban (com k minúsculo) para auxiliar na visualização do fluxo e para permitir a criação de

um sistema puxado de trabalho além de outras ferramentas para catalisar a introdução de ideias Lean nas áreas de desenvolvimento de software e operações de TI. É um processo evolutivo e incremental. Kanban lhe permite atingir processos otimizados para contextos muito específicos, com resistência mínima e mantendo um ritmo sustentável para os trabalhadores envolvidos." [1].

O método Kanban

Como implementar mudanças continuamente no processo de trabalho da equipe com sucesso? Este é um ponto fundamental e um dos motivadores centrais das cinco propriedades do método Kanban:

1. Visualizar o fluxo de trabalho;
2. Limitar a quantidade de trabalho em andamento;
3. Medir e otimizar o fluxo de trabalho;
4. Tornar explícitas as políticas do processo;
5. Gerenciar quantitativamente.

O Kanban não é uma metodologia, mas sim um framework para implementar mudanças de forma incremental. Esse é um conceito muito importante para que se entenda o Kanban como um todo já que, quando se fala em metodologias, fala-se em conjuntos de práticas e o Kanban não tem nenhuma prática prescrita. Há, nele, somente propriedades que devem guiar a melhoria no processo atual, não importando quais práticas estejam sendo usadas.

Ao usar o Kanban, como veremos mais detalhes ao decorrer do artigo, é esperado que se consiga visualizar quais práticas estão sendo positivas e quais estão sendo negativas e isso, consequentemente, vai conduzir a mudanças, que, naturalmente adicionarão, eliminarão ou alterarão as práticas atuais de trabalho.

A forma mais comum de se conseguir visualizar o fluxo de trabalho é através de um kanban. Esse é uma espécie de quadro, que pode ser físico, normalmente colocado em uma das paredes do local onde a equipe trabalha, ou virtual. O uso do quadro virtual tem alguns prós e contras. Os pontos fortes são basicamente a facilidade de extração de métricas e o histórico. O principal ponto fraco dessa abordagem é a dificuldade que a equipe terá para analisar e evoluir conjuntamente o processo de trabalho.

Nesse quadro, inicialmente devem ser modeladas cada uma das etapas necessárias para se produzir o software, isto é, todo o workflow de trabalho. E, em cada uma dessas etapas deve ser colocado e mantido um cartão, a fim de simbolizar um trabalho que está em andamento, como podemos ver na **Figura 1**. Cada um desses cartões deve conter informações detalhadas sobre a atividade, bem como quem está desenvolvendo o item, e quando o mesmo foi iniciado.

No Kanban para desenvolvimento de software, o kanban deve ser mantido e evoluído por toda a equipe, o tempo todo. A proposta baseia-se em fazer a própria equipe enxergar onde está errando e deixá-la tomar decisões seguindo um framework simples, que guiará grande parte dessas melhorias.

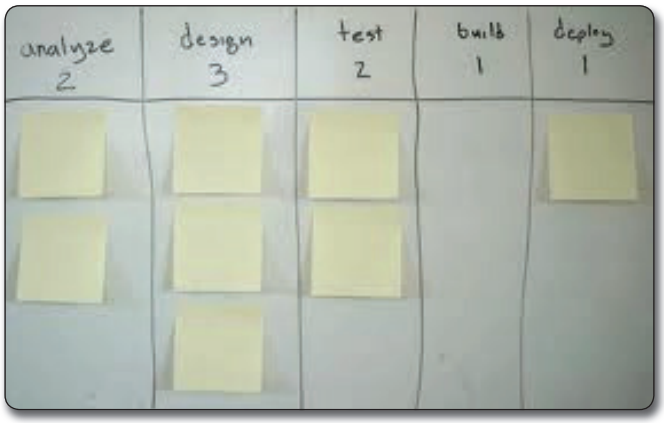

Figura 1. Kanban systems for software development [10]

Para Alisson Vale [5], em atividades que envolvem trabalho criativo, como desenvolvimento de software, o propósito do Kanban é provocar conversações sobre o sistema de trabalho.

Para limitar a quantidade de trabalho em andamento é necessário definir, monitorar e manter um limite máximo de tarefas em andamento para cada uma das etapas de trabalho mapeadas no quadro kanban, como visto na **Figura 1**.

Ao estabelecer os limites, a equipe começa a identificar onde estão os principais gargalos do processo de produção e começa a ter que terminar todo o trabalho inacabado na linha de produção para conseguir puxar mais trabalho para o gargalo. Com isso, o trabalho tende a ser finalizado de uma forma incremental e não de uma forma cascata, tornando-se assim um sistema puxado (TOC), onde é a equipe que dita a sua capacidade de produção.

O uso do sistema puxado, juntamente com a visualização de todo o processo de desenvolvimento por toda a equipe, permite implementar mudanças no processo de modo incremental. Consequentemente, há redução significativa da resistência, o que facilita o alcance do ritmo sustentável, teoria tão comentada em desenvolvimento ágil, mas para a qual poucas definições existem além das 40 horas semanais de trabalho.

Uma citação de David [1] sintetiza isso: "Um interessante efeito colateral de sistemas puxados é que eles limitam o trabalho em andamento (WIP – Work In Progress) para certa quantidade acordada, evitando assim que trabalhadores fiquem sobrecarregados.".

Outro fator importante é o ganho que se tem ao praticamente impedir que uma mesma pessoa execute várias tarefas no mesmo projeto em paralelo. Vale destacar também o processo *Just in time* (JIT), que evita o acúmulo de estoque ao longo do processo de desenvolvimento, como se faz no ciclo de vida em cascata. Nesse caso, software em estoque são todos os requisitos que ainda não foram liberados para o cliente usar.

No desenvolvimento de um software, como em qualquer outra atividade intelectual, por mais contra intuitivo que possa parecer, executa-se com mais qualidade e mais rapidamente duas tarefas fazendo-as uma de cada vez, do que as duas em paralelo o tempo todo.

Segundo Rodrigo Yoshima [7], um dos grandes pontos fortes desse método é o modelo embutido nele para trabalhar com a melhoria de processos, o modelo Kaizen. Além do modelo Kaizen, existe o modelo Kaikaku. Vamos às diferenças: "Kaikaku é uma palavra que define mudanças de processos classificadas como "melhoria radical"." [7].

Implantar o Scrum, por exemplo, exige esse cenário [7], pois o Scrum requer profundas mudanças organizacionais como a gestão abdicar de muitos instrumentos de controle, quebrar com a separação entre os grupos e mudar posições hierárquicas estabelecidas.

"Kaizen é a palavra Lean para indicar mudanças de melhoria menores e contínuas. Ao contrário do Kaikaku, Kaizen não é tão traumático, é melhor aceito por todos (gerentes inclusive) e mais simples de implementar. Tudo a nossa volta está suscetível a um evento Kaizen. Kaizen simplesmente significa mudança para melhor." [7].

Na abordagem Kanban, aplicando a cultura Kaizen, antes de mudar qualquer coisa devemos entender o ambiente de trabalho. Se filas, bloqueios, gargalos ou problemas entre áreas estão nos prejudicando, primeiro, vamos visualizar isso, convencer o grupo dos problemas e usar Kaizen para a melhoria do ambiente com pequenas mudanças incrementais e constantes. Isso irá fortalecer a cultura da empresa, pois ela compreenderá suas falhas com provas palpáveis que serão a motivação para as mudanças [7].

Segundo José Papo [8], além disso, a filosofia e práticas Lean, como é o caso do Kanban, focam na auto-organização do time, na responsabilidade conjunta pelos objetivos de negócio do projeto e na produtividade com qualidade e com ritmo sustentável. Todas elas são práticas de administração consagradas e evidenciadas por Peter Drucker para liderar trabalhadores do conhecimento.

As duas últimas propriedades do Kanban citadas (tornar explícitas as políticas do processo e gerenciar quantitativamente) são praticamente atendidas deixando claro para toda a equipe as três primeiras propriedades e reforçando que toda sugestão de otimização no processo deve ter como base modelos matemáticos que provem as métricas.

Kanban é uma metodologia ágil?

Nos últimos tempos, acompanhamos uma briga forte entre Kanban e as demais metodologias ágeis, principalmente o Scrum. A polêmica central é: quem usa Kanban é ágil?

Rodrigo Yoshima [9] explica que "o maior objetivo do Kanban é melhorar um processo existente, por pior que ele seja". Ainda segundo Yoshima, o Kanban permite melhorar até mesmo processos que seguem o ciclo de vida em cascata, fazendo eles se tornarem ágeis e podendo melhorar continuamente, e até mesmo superarem o Agile. Isso tudo graças à cultura Kaizen que o Kanban quer trazer à tona.

Outro ponto a ser destacado é que para o Kanban, diferente da grande maioria das metodologias ágeis, não importa quais práticas você irá aplicar para melhorar o seu processo, o importante é que você tenha argumentos, preferencialmente estatísticos, que expliquem o porquê dessas decisões.

Como David ressalta em seu livro [1], a abordagem evolutiva do Kanban, que promove a implementação de mudanças no processo de forma incremental, tem sido controversa na comunidade ágil de desenvolvimento de software. Isso ocorre principalmente porque se sugere que as equipes não adotem um método ou modelo de processo. Vale ressaltar que a atual indústria de serviços e ferramentas desenvolveu-se em torno de um pequeno conjunto de práticas definidas em dois populares métodos de desenvolvimento ágil: XP e Scrum. Depois, com o Kanban, os indivíduos e as equipes estão habilitados para desenvolver suas próprias soluções por meio de um processo único que evitaria a necessidade de tais serviços e ferramentas. Isso porque os mesmos requerem um novo conjunto de ferramentas e serviços muito específicos para cada realidade. Dessa forma, torna-se mais complicado para essa indústria ganhar dinheiro.

Tanta discussão acerca do assunto fez o pessoal do Kanban criar um logo de manifesto chamado: Yes We Kanban, que David [1] explica: "O slogan 'Yes We Kanban' se destina a enfatizar que você tem permissão. Você tem permissão para tentar Kanban. Você tem permissão para modificar o seu processo. Você tem permissão para ser diferente. Sua situação é única e merece o desenvolvimento de um processo adaptado e otimizado para o seu domínio, o seu fluxo de valor, os riscos que você gerencia, as habilidades de sua equipe e as demandas de seus clientes.".

Portanto, podemos dizer que o objetivo de Kanban não é tornar a sua equipe ágil, e sim, melhorar a forma de trabalho. Ele tanto pode melhorar processos ágeis como também pode melhorar processos tradicionais.

Qual é a receita para ter sucesso com Kanban?

Para que qualquer pessoa que queira ser um agente de mudança na sua organização tenha sucesso rápido (ou, uma melhoria rápida) e com baixa resistência da equipe, com foco na melhoria de processos em alguns pontos com o uso ou até mesmo sem o uso do método Kanban, é preciso seguir algumas etapas. De acordo com David, são elas:

1. Focar na qualidade;
2. Limitar a quantidade de trabalho em progresso;
3. Entregar frequentemente;
4. Balancear demanda com a capacidade máxima;
5. Priorizar tarefas;
6. Atacar fontes de variedade.

O autor orienta que essas etapas sejam seguidas na ordem em que são apresentadas. A partir de agora, passa-se a discorrer um pouco a respeito de cada uma delas.

Focando na qualidade

Como os agentes de mudanças nas empresas de TI são geralmente pessoas com um background técnico, essa tende a ser uma das etapas mais fáceis de ser implementada, principalmente por ser um problema bem entendido por todos. As outras etapas desse guia [1] tendem a ter uma implementação mais difícil porque dependem da colaboração de outras áreas

e equipes. Com isso, exigem que o agente de mudança tenha muitas habilidades de negociação, articulação e bastante inteligência emocional [1].

Os maiores geradores de retrabalho em desenvolvimento de software são os defeitos causados principalmente pela baixa qualidade das entregas. E, além de um gerador de retrabalho, baixa qualidade faz os clientes ficarem inseguros e a equipe desmotivada.

O incentivo à qualidade das entregas tem um grande impacto na produtividade das equipes com altas taxas de defeitos. Segundo David [1], em equipes verdadeiramente ruins, somente concentrando-se na qualidade pode-se obter uma melhoria de produtividade de até dez vezes.

Para David [1], tanto as técnicas de desenvolvimento ágil como as abordagens tradicionais têm seu mérito para a melhoria da qualidade. As principais práticas incentivadas por ele para a melhoria da qualidade são:

1. Escrever testes automatizados, preferencialmente antes;
2. Revisar código (Verificação);
3. Fazer atividades de análise e design do software de forma colaborativa;
4. Usar Design Patterns;
5. Usar ferramentas modernas de desenvolvimento.

Parece haver uma vantagem psicológica em pedir para os desenvolvedores escreverem testes antes, porém, é importante ressaltar que também existem inúmeros casos de sucesso com a escrita dos testes após a codificação.

Inspeções ou revisões de código ajudam a melhorar tanto a qualidade externa como, notadamente, a qualidade interna do software. Todas as técnicas têm o seu valor. Programação em par e revisão por pares são alguns exemplos. No entanto [1], inspeções de código são melhores quando são feitas em pequenas quantidades várias vezes. David menciona [1] que ele costuma encorajar as suas equipes a inspecionar código, todos os dias, por pelo menos 30 minutos.

Sem dúvidas, quando toda a equipe trabalha em conjunto na análise dos problemas para as soluções de design, a qualidade é superior do que quando apenas uma pessoa faz isso. Assim como as inspeções de código, atividades de modelagem de software devem ser feitas em pequenas quantidades, todos os dias.

Os padrões de projeto de software, mais conhecidos pelo termo original em inglês Design Patterns, descrevem soluções para problemas já conhecidos e recorrentes no desenvolvimento de software orientado a objetos. O uso de padrões de projeto garante que defeitos de design sejam eliminados já no início do projeto.

O uso de ferramentas modernas de desenvolvimento melhora a qualidade porque a grande maioria dessas inclui funções de análise de código estática e dinâmica, que evitam que os desenvolvedores introduzam problemas básicos e já bem compreendidos, como falhas de segurança, no software.

Limitando a quantidade de trabalho em andamento

É fácil especular porque limitar a quantidade de tarefas em andamento aumenta a qualidade. A complexidade do trabalho do conhecimento, como em desenvolvimento de software, cresce exponencialmente com a quantidade de trabalhos em andamento. Tanto a transferência como a descoberta de informações no desenvolvimento de software é conhecimento tácito por natureza e é criado durante sessões de trabalho colaborativo, face a face. A informação é verbal e visual, mas é em um formato casual, como um esboço em um quadro branco.

Nossas mentes têm uma capacidade limitada para armazenar conhecimento tácito. E, quanto mais tempo passa, há mais falhas para recordar detalhes precisos. Assim, uma série de erros é cometida. Equipes que trabalham de um modo ágil, em um mesmo espaço de trabalho, têm uma maior facilidade em reter o conhecimento tácito.

Mas, independentemente da forma de trabalho da equipe, o conhecimento tácito se deprecia com o passar do tempo. Por isso, tempos de espera (*lead time*) menores são essenciais para os processos que envolvem muito conhecimento tácito. O foco da redução de trabalho em andamento está diretamente relacionado com a redução dos tempos de espera (*lead time*).

Assim, podemos deduzir que haverá menor depreciação de conhecimento tácito quando temos menos trabalho em progresso – o que resultará em maior qualidade. Em resumo, reduzindo a quantidade de trabalho em andamento, melhora-se a qualidade e possibilita-se entregas mais frequentes. Isso aumenta a confiança externa na equipe.

Além de reduzir a quantidade de trabalho em andamento, é importante reduzir o tempo de uma iteração, pois isso também trará um impacto positivo significativo na qualidade. Segundo David [1], parece que existe uma relação entre a quantidade de trabalho em andamento e a qualidade, ou seja, defeitos vão aumentar com o aumento da quantidade de WIP. Portanto, faz sentido que iterações de duas semanas sejam melhores do que iterações de quatro semanas e que iterações de uma semana sejam melhores ainda. Iterações mais curtas irão resultar em entregas de maior qualidade.

Seguindo a lógica das evidências apresentadas, se é sabido que limitar o WIP irá melhorar a qualidade, por que não introduzir política explícita para isso, deixando assim os gerentes livres para se concentrarem em outras atividades? Essa é justamente uma das propriedades do Kanban. Entretanto David afirma [1] que ainda não existe nenhuma evidência científica desse resultado que foi observado apenas empiricamente.

Entregando frequentemente

Para entendermos a importância dessa etapa, David apresenta uma excelente analogia em seu livro [1]: "Quando eu ensino isso nas aulas, eu gosto de perguntar às mulheres da classe o que elas pensam sobre duas situações depois de ter um primeiro encontro com um cara:

- Situação 1: Eles tiveram um bom encontro, mas depois disso ele não dá sinais de vida a ela durante duas semanas. Mas, então, ele aparece em sua porta com um ramo de flores e um pedido de desculpas;
- Situação 2: Eles tiveram um bom encontro e na mesma noite a caminho de casa ele envia uma mensagem de texto a ela

dizendo: "Eu me diverti muito esta noite. Eu realmente quero me encontrar com você novamente. Posso ligar para você amanhã? E, esse mesmo cara, segue ligando e enviando mensagens dia após dia.

Qual cara vocês acham que elas preferem?".

Pequenos e frequentes gestos não custam quase nada, entretanto, eles constroem mais confiança do que grandes e caros gestos ocasionalmente. Portanto, realizar frequentemente pequenas entregas de alta qualidade constrói mais confiança para as equipes parceiras do que entregas maiores, mas com menos frequência.

Balanceando a demanda com a capacidade máxima

Construir um consenso em torno da necessidade de equilibrar a demanda contra a capacidade da equipe é crucial. No entanto, para isso, é preciso resolver problemas como a disfunção entre os papéis e as responsabilidades dos membros da equipe.

Essa etapa implica definir a taxa em que a equipe aceita novos requisitos no processo de desenvolvimento de software para corresponder com a capacidade em que a equipe pode entregar código de qualidade.

Quando se faz isso, fixa-se efetivamente a quantidade de trabalho em andamento, permitindo efetivamente que seja a equipe que crie a demanda de acordo com a sua capacidade (sistema puxado).

Priorizando tarefas

Priorizar é trabalho da área de negócios, e não da área de TI da organização. Assim, não deveria ser da competência de um gerente técnico fazer isso. Entretanto, infelizmente, é comum a área de negócios delegar a responsabilidade e deixar um gerente técnico priorizar o trabalho e depois culpar que o gerente fez escolhas erradas.

Neste ponto, a atenção da gerência deve-se voltar para otimizar o valor entregue ao invés de meramente a quantidade de código entregue.

Atacando fontes de variedade

Podemos entender como fontes de variedade tudo aquilo que de alguma forma prejudica o desempenho do processo de produção.

Atacar as fontes de variedade é a última etapa da receita porque alguns tipos de variedade exigem profundas mudanças de comportamento e, consequentemente, uma alta resistência da equipe.

A dica para implementar essa etapa é focar nas fontes de variedade que requerem pequenas mudanças de comportamento e que podem ser aceitas facilmente pela equipe.

Segundo David [1], não se deve focar nessa etapa sem antes implementar e dominar as primeiras cinco etapas da receita.

Resumindo, essa receita é a forma que David [1] acredita que uma equipe de desenvolvimento de software deve amadurecer: "Em primeiro lugar, aprender a construir código de alta qualidade. Em seguida, reduzir o trabalho em andamento, encurtar os lead times, e entregar (software funcionando) com frequência. Em seguida, equilibrar a demanda contra a capacidade máxima, limitar a quantidade de trabalho em andamento, e criar folga para liberar capacidade, o que permitirá a melhoria contínua. Então, com isso funcionando razoavelmente e continuamente otimizando a capacidade de desenvolvimento de software, melhore a priorização para otimizar a entrega de valor.".

O Kanban permite que você implemente todas as seis etapas dessa receita.

Conclusão

É interessante conhecer as motivações que levaram David Anderson a chegar até o método Kanban para o Desenvolvimento de Software e, através da sua receita, entender o porque da maioria das propriedades por trás do Kanban. Essa metodologia tem sido um assunto recorrente e tende a continuar sendo, já que é uma forma eficiente de melhorar continuamente a área.

Se a sua empresa está com dificuldades para melhorar, não importa se ela adota métodos ágeis ou não, o Kanban pode ser uma das alternativas para fazer isso com o mínimo de resistência, pois, além de toda a simplicidade do método, suas propriedades promovem a colaboração.

Referências

1. Anderson, D. (2010) Kanban: Successful Evolutionary Change for Your Technology Business. Washington, Blue Hole Press.

2. Martin, R. (2011) The Clean Coder: A Code of Conduct for Professional Programmers. Indiana, Prentice Hall.

3. Anderson, D. (2003) Agile Management for Software Engineering: Applying the Theory of Constraints for Business Results. New Jersey, Prentice Hall.

4. Bernabó, J. (2011) Mais comprometimento = menos produtividade? http://www.teamware.com.br/blog/mais-comprometimento-menos-produtividade/

5. Vale, A. (2011) Kanban Explicado. http://www.slideshare.net/alissonvale/kanban-explicado

6. Wikipédia. TOC. http://pt.wikipedia.org/wiki/Teoria_das_restri%C3%A7%C3%B5es

7. Yoshima, R. (2011) Kaikaku – Kaizen, http://blog.aspercom.com.br/2011/09/09/kaikaku-kaizen/

8. Papo, J. (2010) Porque Lean/Agile funcionam? http://josepaulopapo.blogspot.com/2010/03/agile-lean-funciona-por-que.html

9. ABES (2011). O Mercado Brasileiro de Software em 2011. http://www.abes.org.br/UserFiles/Image/PDFs/Mercado_BR2011.pdf

10. Corey Ladas (2007). Kanban systems for software development (Imagem) http://leansoftwareengineering.com/wp-content/uploads/2007/08/kanban1.png