Gestão do desenvolvimento de software com o uso de quadro virtual Kanban

Software development management with the use of Kanban virtual board

DOI:10.34117/bjdv6n12-757

Recebimento dos originais:12/11/2020
Aceitação para publicação:30/12/2020

Tiago Martins de Alexandre
Mestrado (em curso) em Engenharia da Computação
Instituição: Universidade de São Paulo
Endereço: Av. Prof. Luciano Gualberto, tv 3 - Butantã - São Paulo - SP - 05508-010
E-mail: tiago.alexandre@alumni.usp.br

Jorge Rady de Almeida Junior
Livre Docente
Instituição: Universidade de São Paulo
Endereço: Av. Prof Luciano Gualberto travessa 3 - Butantã - São Paulo - SP - 05508-010
E-mail: jorgerady@usp.br

RESUMO
O método Kanban tem sido utilizado para evidenciar gargalos na produção de software em diversas empresas de desenvolvimento. A proposta inicial do Kanban prevê o uso de um quadro físico para controle e visualização das atividades. Esta proposta sugere que o quadro seja construído de maneira virtual tanto para melhorar o vínculo entre o que é falado e o que é escrito quanto para permitir a extração de métricas de maneira automatizada. Para endossar a proposta, foram analisados dados de aproximadamente dois anos de utilização de quadro virtual Kanban em uma empresa de desenvolvimento de software, com o objetivo de verificar a aderência das informações apresentadas no quadro em relação ao real estágio de desenvolvimento por meio da comparação desses dados com os dados obtidos nos controladores de versionamento de código fonte. Desta forma, foi possível observar algumas divergências entre as marcações o que mostra que há espaço para melhorias.

Palavras-chave: Kanban, metodologia ágil, quadro kanban, quadro virtual kanban.

ABSTRACT
The Kanban method has been used for several software development companies to expose bottle necks in their production. The initial Kanban proposal predicts the use of a physical board for controlling and visualizing activities.This proposal suggests the use of a virtual Kanban board to facilitate the managing process due automatic metrics extraction and to allow a tighter relationship between what is sad and what is done.To endorse this proposal, approximately two years of data have been gathered from a software development company to verify the people commitment to virtual board usage comparing their actions taken on the virtual board with the actual registered activity in the source code manager software. Some discrepancies were found, and this will lead to some improvement's opportunities.

Keywords: Kanban, agile methodology, kanban board, virtual kanban board.

1 INTRODUÇÃO

1.1 DESCRIÇÃO DO PROBLEMA DE PESQUISA

A crescente complexidade dos sistemas computacionais a serem desenvolvidos gerou inúmeras complicações para os desenvolvedores de software ao longo dos anos. Para lidar com sistemas complexos, é bastante comum, a proposta de sua divisão em pequenos blocos. O ato de desenvolver cada bloco pode ser representado, por uma tarefa ou atividade.

Dessa forma, um sistema é completo quando se completam todas as tarefas que compreendem o seu desenvolvimento.

É bastante comum que, após iniciada, uma tarefa permaneça cerca de 85% do tempo em alguma fila. (Vacanti, 2015).

Reduzir o tempo de fila das tarefas é a principal proposta do método Kanban.

O método Kanban, derivado do Lean, para o gerenciamento de produção de bens "não tangíveis", como software, apresenta o quadro de atividades (*Kanban Board*) como sendo a principal ferramenta para se conseguir visualizar o fluxo de trabalho de uma determinada organização. Sendo este, o primeiro dos cinco elementos de uma implantação bem-sucedida do método. (Anderson, 2010).

Kanban board expõe o fluxo de trabalho o que permite a identificação de gargalos no processo. Assim, é possível estabelecer limites no fluxo o que acelera as entregas e reduz riscos técnicos e de negócios (Middleton, 2012).

Devido a diversos fatores como, diversidade de localização física dos membros da equipe e a dificuldade de elaboração de métricas utilizando material não digitalizado, algumas empresas apresentaram propostas para virtualização do quadro (VQ) de atividades tais como *LeanKit* e *Jira*.
Uma das características do Kanban é a sua inicial adesão ao processo de trabalho que requer poucas mudanças (Nevenka, 2015). A VQ pode engessar um pouco esta adesão devido a restrições inerentes da ferramenta escolhida. Algumas ferramentas estão listadas no apêndice A.

Além disso, quadros virtuais podem tornar as reuniões diárias de acompanhamento um pouco diferentes em relação ao modelo proposto por (Anderson, 2010) em frente ao quadro físico.

O Kanban tem evoluído bastante entre as empresas de desenvolvimento de software. Muitos conceitos estão sendo testados empiricamente e os resultados das experiências estão orientando a evolução do método.

O presente trabalho propõe analisar dados de dois anos de acompanhamento de métricas de Kanban em uma empresa de desenvolvimento de software que utilizou quadro virtual durante todo o período. Os dados extraídos da ferramenta de acompanhamento virtual serão cruzados com os dados da ferramenta de versionamento de código.

A técnica do erro quadrático médio foi utilizada para avaliar se os dados informados no quadro virtual realmente representam o trabalho gerado efetivamente. O que revela o nível de confiança das informações extraídas da ferramenta. Também foi observado se, apesar de eventuais discrepâncias, pode ter acontecido uma regressão à média durante o período, o que seria uma evidência a favor da comprovação da eficácia do quadro virtual.

Contudo, não foi encontrado na literatura nenhum estudo de caso que evidencie se o uso de quadro virtual de Kanban reflete, para os gestores, o que de fato acontece no dia a dia do desenvolvimento.

1.2 OBJETIVO

Este trabalho tem como objetivo propor o uso de quadro virtual para acompanhamento do desenvolvimento de software e mostrar que pode haver divergências entre o que é falado nas reuniões diárias e o que é de fato executado na criação do software. A virtualização do quadro é um primeiro passo para melhorar este vínculo, além de viabilizar a geração automática de métricas para tomada de decisões.

No método Kanban, o desenvolvedor é responsável por indicar, no quadro físico, sempre que inicia, revisa ou conclui uma atividade por meio da mudança de sua posição. Como há reuniões diárias de acompanhamento e atualização do quadro, é normal ter, no máximo, até 1 dia de defasagem entre a movimentação das tarefas e os indicadores de versionamento de código.

Com a utilização de um quadro virtual, as movimentações são registradas e salvas de forma permanente em banco de dados. Ou seja, é criado um indicador adicional e este trabalho visa analisar a defasagem entre estes dois indicadores.

Se houver discrepância entre os indicadores, teremos um indício de que o quadro e as métricas possam não estar refletindo a realidade no grupo de desenvolvimento. O que pode gerar relatórios gerenciais equivocados e mais dificuldades no acompanhamento das atividades.

Outro objetivo desta dissertação é o de propor uma integração entre o sistema de versionamento de código e o quadro virtual de acompanhamento de atividades de modo que o quadro se torne uma consequência das ações executadas em código. Acabando com a discrepância entre os indicadores e permitindo que o quadro reflita, de fato, o que está sendo feito e quando.

2 REVISÃO DA LITERATURA

O método Kanban, devido às suas características, pode ser aplicado a quaisquer processos que envolve uma cadeia de valor. Assim, seu uso tem sido observado em diferentes indústrias como: Aeronáutica, Saúde, Têxtil, Recursos humanos e Software (Ahmad, Dennehy, Conboy, Oivo, 2018).

Apesar de sua larga aplicabilidade, os estudos selecionados para consulta serão os estudos em que Kanban é aplicado ao desenvolvimento de software.

2.1 EVOLUÇÃO DAS PESQUISAS SOBRE KANBAN

Em 2013, uma revisão de literatura feita sobre o método Kanban em desenvolvimento de software mostrou que as publicações sobre o tema têm aumentado consideravelmente.

Tabela 1 Principais publicações anuais

Ano	**2000-07**	**2008**	**2009**	**2010**	**2011**	**Total**
Artigos	-	1	2	6	10	19
Porcentagem	-	5%	10%	32%	53%	100%

Fonte: Ahmad, 2013

Os estudos se classificam em: relatórios de experiências (47%), estudos qualitativos (32%), estudos de simulação (10%) e estudos quantitativos (11%). Ou seja, pontuou que as publicações sobre Kanban, em sua maioria, são resultados de estudos empíricos. (Ahmad, Markkula, Oivo, 2013).

Em 2018, Ahmad fez um novo estudo exploratório sobre as pesquisas em Kanban e identificou 21 estudos primários distribuídos na seguinte forma:

Figura 1: Estudos sobre Kanban publicados por ano

Fonte: (Ahmad, Dennehy, Conboy, Oivo, 2018).

Nestes novos resultados, estão apenas listados os principais artigos que foram classificados como foco de seu estudo. Tais artigos evidenciam os benefícios do Kanban em diversos aspectos. Dos 23 artigos, 8 foram publicados em Periódicos e 15 em conferências.

Destes, Ahmad extrai algumas conclusões acerca dos desafios e benefícios do método.

2.2 PRINCIPAIS DESAFIOS APONTADOS PARA IMPLEMENTAÇÃO DO KANBAN

Alguns estudos evidenciaram dificuldades na implementação do Kanban, as principais dificuldades encontradas foram: (Ahmad, Dennehy, Conboy, Oivo, 2018):

1. Configuração e manutenção do Kanban;
2. Gestores não preparados para um novo método;
3. Pobre entendimento dos conceitos e práticas do Kanban;
4. Gestão da comunicação entre o time e o cliente;
5. Mudança da cultura da organização;
6. Falta de suporte às práticas que envolvem o Kanban;
7. Falta de treinamento;
8. Pobre conhecimento da gestão.

Os itens 1, 2, 3 e 8 são facilmente endereçados com treinamento e evidenciam falta de planejamento no processo de implantação, pois o treinamento é base fundamental de qualquer mudança.

A falta de treinamento (item 7) é decorrente do estágio do método que ainda está em processo de amadurecimento. A *Lean Kanban University* possui uma agenda regular de treinamentos que contam com empresas certificadas em vários países. Conforme o método amadurece e novas pessoas são treinadas, este problema tende a reduzir e, consequentemente, os problemas 1, 2, 3 e 8.

O item 4 é o maior desafio do Kanban e, conforme será exposto na descrição do método, o principal fator de sucesso é a comunicação e o alinhamento com o cliente. Na proposta de Anderson, há maneiras de se tratar com o cliente o modo de trabalho. Se o cliente não concordar, não há como se conseguir uma implantação 100% bem-sucedida. Este é um tipo de problema que precisa ser mitigado por meio de muitas conversas, reuniões e experimentações junto aos clientes.

O item 5 também é um fator importante, entretanto, conforme será descrito, o Kanban não impõe muitas mudanças no processo de trabalho do dia-a-dia se comparado, por exemplo, ao Scrum. O intuito de começar a mapear o fluxo de trabalho pode dar uma errônea ideia de que ser quer identificar culpados. O que não é o objetivo do método, o que deve ser muito bem esclarecido para se conseguir alinhamento da equipe. Equipes muito ineficientes também terão problemas com o método pois as ineficiências são expostas, assim como todos os demais gargalos.

2.3 PRINCIPAIS BENEFÍCIOS DO MÉTODO KANBAN

Os benefícios do método Kanban são inúmeros. A seguinte lista, proposta por (Ahmad, Dennehy, Conboy, Oivo, 2018) enumera-os da seguinte maneira, inclusive citando o número de artigos que abordam cada item:

1. Melhoria da visibilidade e da transparência (16 artigos)
2. Melhor controle as atividades dos projetos (12 artigos)
3. Identificação de impedimentos no fluxo (10 artigos)
4. Melhoria no processo de trabalho (7 artigos)
5. Redução do tempo de entrega (5 artigos)
6. Melhoria na priorização de tarefas (4 artigos)
7. Redução de defeitos e bugs (4 artigos)
8. Aumento da qualidade (4 artigos)
9. Método leve e intuitivo (4 artigos)
10. Melhoria na colaboração e comunicação (7 artigos)
11. Melhoria na motivação do time (6 artigos)
12. Coesão e formação do time (5 artigos)
13. Aumento da satisfação do cliente (6 artigos)
14. Cultura de aprendizado contínuo (5 artigos)
15. Alinhamento estratégico (3 artigos)

É importante considerar que alguns artigos citam mais de um benefício.

A redução de defeitos e bugs está intimamente ligada à melhoria de qualidade. Então, por mais específico que sejam os feedbacks, é possível juntar os itens 7 e 8 em apenas um item sobre Melhoria da Qualidade com citações em 7 artigos, dado que Middleton é citado nos dois itens. (Ahmad, Liukkunen, Markkula, 2014); (Al-Baik, Miller, 2015); (Ikonen, Kettunen, Oza, Abrahamsson, 2011); (Mahnic, 2015); (Middleton, Joyce, 2012); (Sjøberg, Johnsen, Solberg, 2012). Todos estes trabalhos apontaram que o Kanban permitiu, de certa forma, uma melhoria na qualidade.

Os itens de 1 a 4, 6, 9 a 12, 14 e 15 são os benefícios do método que levam ao resultado do aumento da qualidade (itens 7 e 8) citados nos trabalhos acima. Os itens 5, 7 e 8 levam ao 13 que é a percepção do trabalho pelo maior interessado. Estas induções ficam mais evidentes com o entendimento do método e a verificação dos demais estudos de casos.

Em indústrias de manufatura o Kanban tem facilitado a aplicação dos princípios do Lean de forma simples e efetiva. A combinação da simplicidade do Kanban com as capacidades tecnológicas de um quadro virtual facilita a automação de processos, melhora visibilidade e permite a medição de performance em tempo real (Wan, Chen, 2008)

2.4 ESTUDOS EXPERIMENTAIS SOBRE O MÉTODO

Ainda na mesma pesquisa, (Ahmad, Dennehy, Conboy, Oivo, 2018). enumera 23 relatórios de estudos de casos envolvendo o uso do Kanban.

2.4.1 Principais desafios do Kanban

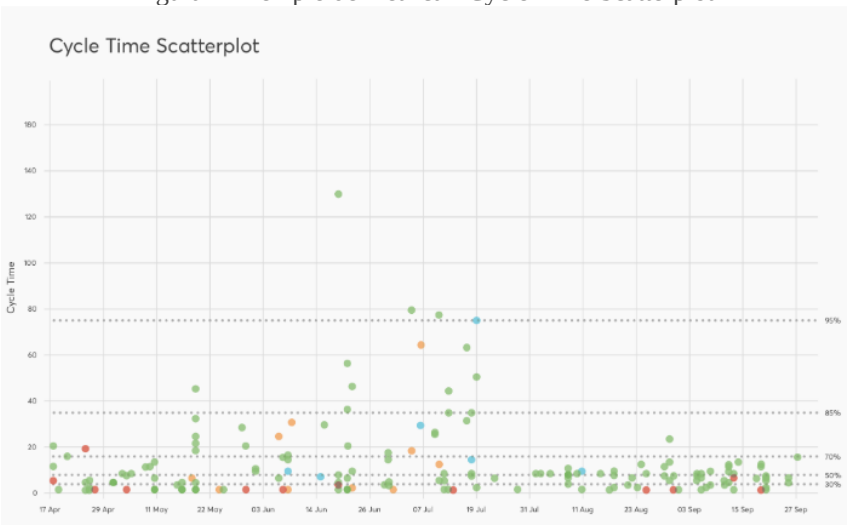
Os estudos experimentais apontaram os seguintes desafios na utilização do método Kanban.

1. Falta de boas-práticas (*guidelines*) e entendimento do método e sua implementação;
2. Avaliar o desempenho usando métricas;
3. Motivar o time para adotar novas práticas;
4. Troca de tarefas e fluxo de trabalho imprevisível;
5. Kanban requer integração com técnicas ágeis existentes, o que pode ser complicado, caro e demorado;
6. Mudança da cultura da organização;
7. Falta de habilidades especializadas e treinamento;
8. Implementação do Kanban requer profundo entendimento do Lean.

Os itens 1, 2, 5, 7, e 8 são endereçados com treinamento apropriado. O item 8 é bastante questionável pois há implantações bem-sucedidas de Kanban com equipes que apenas fizeram o treinamento de uma empresa de consultoria que apenas focou nas práticas e não em teoria pesada sobre Lean. O dia-a-dia para utilização do Kanban não requer nenhum conhecimento complexo de ser adquirido. Contudo, o desafio maior é do gestor pois é na extração das métricas e nas tomadas de decisões que se observa um dos grandes benefícios do método.

A Previsibilidade apontada no item 4 como desafio, em alguns estudos, tende a ser decorrente da falta de treinamento do gestor pois as métricas de algumas semanas de utilização podem, efetivamente, mostrar a como é a taxa de entrega da equipe e, assim, tornar as entregas previsíveis. O gráfico a seguir mostra um exemplo de como uma métrica bem estruturada permite dar previsibilidade a um projeto:

Figura 2 Exemplo de métrica - Cycle Time Scatterplot



Fonte: (Siderova, 2018)

A partir da análise deste gráfico é possível prever, com 95% de confiança que é possível entregar tarefas com até 80 horas. Este tipo de estimativa só é possível desde que as mesmas condições de projeto continuem valendo. Uma alteração na quantidade de membros da equipe, por exemplo, afetaria os resultados futuros e invalidaria as previsões.

Os itens 3 e 6 fazem parte dos desafios de qualquer mudança e de implementação de qualquer tipo de metodologia, seja Scrum, XP ou, até mesmo, RUP. Destas citadas, o Kanban é a que menos impacta no dia-a-dia dos desenvolvedores pois, devido ao método de implementação (STATIK, que será mostrado adiante) é possível iniciar com apenas 1 mudança, ou seja, a atualização diária de um quadro de atividades em uma reunião de 15 minutos. Não é necessário criar escopos, fazer reuniões de replanejamento, programação em pares, e nenhuma outra rotina diferente do que já esteja adotado na organização.

2.4.2 Principais benefícios do Kanban

Os mesmos estudos, elencam os seguintes benefícios

1. Facilita a visibilidade e suporte ao processo de tomada de decisão;
2. Desenvolve estratégias de melhorias contínuas e melhora o fluxo de trabalho;
3. Melhor entendimento do processo de desenvolvimento como um todo;
4. Melhora a previsibilidade da entrega do produto final e provê uma melhor estimativa do trabalho;
5. Redução do tempo dos ciclos de trabalho e dos tempos de cada tarefa (*lead time*);
6. Melhora o balanceamento do trabalho;
7. Garantia de melhoria das habilidades e a coesão dos times;
8. Facilita a coordenação impondo uma auto-organização;
9. Dirige e facilita a gestão da mudança da organização.

É notável o alinhamento dos resultados entre os trabalhos primários selecionados e os estudos experimentais.

Pode-se destacar um dos estudos em que (Santos, Beltrão, Souza, Travassos, 2018) foi apontado que o Kanban tem como principais benefícios, melhorar:

- Visibilidade do trabalho
- Controle das atividades e tarefas do projeto

Fluxo de trabalho

- Tempo de entrega (*Time-to-Market*)

E, como principal desafio do método, a cultura da empresa.

Não há muitas discrepâncias nas implementações. A falta de treinamento adequado da gestão e do time é bem evidente quando não há sucesso. A falta de profissionais no mercado também contribui para este fator. Trabalhos como este podem mitigar um pouco este problema.

3 MÉTODO KANBAN

3.1 INTRODUÇÃO

A palavra kanban é japonesa e significa "*Signboard".* Ou seja, cartão ou quadro contendo um texto de identificação.

Kanban, escrito com **K** maiúsculo refere-se à definição do método conforme proposta por David J. Anderson que aplica os conceitos do Lean, criado por Taiichi Ohno que desenvolveu o Sistema Toyota de Produção (TPS). Este sistema é baseado em dois conceitos:

1. Automação;
2. Produção *Just-In-Time* (JIT) cuja tradução significa: obter os recursos necessários exatamente quando eles são demandados.

Neste sistema, Ohno utiliza o **k**anban como ferramenta para seu método que revolucionou os meios de produção.

Após a publicação do livro "The Machine that Changed the World" em 1990, o termo Lean é cunhado para designar o pensamento deste novo sistema de produção que tem como conceitos (Wang, Conboy, Cawley, 2012); (Ahmad, Dennehy, Conboy, Oivo, 2018):

1. Valor: Valor conforme definido pelo cliente;
2. Cadeia de valor: O mapa que identifica cada etapa no processo e as categoriza conforme o valor que ela adiciona;
3. Fluxo: Refere-se ao fluxo contínuo da cadeia de valor no processo;
4. Puxar: Pedidos dos clientes puxam demandas por novos produtos garantindo que nada seja feito antes de ser necessário ou solicitado;
5. Perfeição: Buscar a melhoria dos processos removendo desperdícios continuamente.

(Poppendieck, 2003) fez a primeira publicação utilizando os conceitos Lean no desenvolvimento de software em 2003. Novas metodologias de desenvolvimento surgiram como XP e Scrum com o manifesto ágil. E, a partir de 2010, Anderson começa a divulgar o termo Kanban para designar este conjunto de técnicas derivadas do Lean.

Embora existam muitas derivações do método Kanban e detalhes de implementação que são aplicados em cada projeto, em cada grupo, em cada instituição, o método será descrito usando as proposições de Anderson, assim como a proposta, que será apresentada como uma evolução do método.

3.2 DEFINIÇÃO

Kanban (K maiúsculo) é um método evolucionário de acompanhamento de projetos que utiliza um kanban (k minúsculo) e outras ferramentas para visualizar um sistema puxado para aplicar ideias do Lean no processo de desenvolvimento tecnológico e em operações de tecnologia da informação. (Anderson, 2010).

Portanto, kanban com k minúsculo é um quadro com cartões que representam as tarefas em um fluxo de valor. O kanban representa um sistema puxado, ou seja, um sistema por onde uma nova atividade só pode ser iniciada se uma outra houver sido concluída. A conclusão de uma tarefa permite puxar uma nova e isto é o que caracteriza este tipo de sistema. Não se pode empurrar uma tarefa de modo que é possível limitar o trabalho em progresso em qualquer etapa.

Esta definição foi aceita, também por (Ahmad, Markkula, Oivo, 2013) (Dennehy, 2016), (Fitzgerald, Musiał, Stol, 2014), (Harzl, 2016), (Law, Lárusdóttir, 2015), (Mahnic, 2015), (Tripathi, Rodríguez, Ahmad, Oivo, 2015), (Rodriguez, Partanen, Kuvaja, Oivo, 2014), (Senapathi, Middleton, Evans, 2011).

3.3 IMPLEMENTAÇÃO

O método que as principais consultorias que aplicam Kanban se utilizam para ensinar os conceitos do Kanban nas empresas é chamado de STATIK. Nas próximas seções são apresentados os conceitos do Kanban por meio deste método para melhor compreensão.

3.3.1 Statik

STATIK é um acrônimo para *System Thinking Aproach to Implementing Kanban* ou, abordagem de pensamento sistêmico para implementação do Kanban.

Consiste em uma série de passos que permitem implementar o Kanban em um espaço muito curto de tempo dentro de qualquer organização.

3.3.1.1 Entender e descrever as motivações para mudanças

Geralmente, as empresas procuram algum método alternativo para controlar o desenvolvimento de software porque estão insatisfeitas com a metodologia em uso. Isso é bastante comum dado à evolução que os métodos apresentaram nos últimos anos, baseados nestas necessidades de aperfeiçoamento. O *time-to-market*, ou, tempo para o mercado é o principal motivador. Em um cenário competitivo como o atual, não entregar rapidamente pode significar a perda de boas oportunidades e, consequentemente, muito dinheiro. Para entregar mais rápido, clientes pressionam gestores, que pressionam desenvolvedores que entregam rapidamente com menor qualidade. Isso incorre em bugs

que se reproduzem rapidamente em iterações sucessivas de entrega tornando dramático os finais de muitos projetos o que gera tanto insatisfações internas e externas.

I. Insatisfações internas e externas

As insatisfações, em sua grande maioria, são o resultado de uma pressão por entregas rápidas e a falta de uma gestão eficiente de expectativas e cresce como uma bola de neve. Portanto, a primeira preocupação do método kanban é mapear as insatisfações.

II. Identificação de fontes de variabilidade

É necessário levantar todas as fontes de variabilidade que afetam a velocidade de entrega de valor. Elaborar um documento contendo-as e considerá-lo durante negociação dos prazos com o cliente. Este documento irá ajudar, também, no entendimento da capacidade.

Exemplos de fontes de variabilidade são:

- Bugs de produção que interrompem o processo de desenvolvimento e precisam ser endereçados com prioridade;
- Equipe de homologação externa da qual se é dependente e não se pode impor um tempo fixo para homologação;
- Decisões de negócios que dependem de uma resposta do cliente e que impacta no tempo de desenvolvimento;
- Dificuldade intrínseca do desenvolvimento de alguns pontos que envolvem pesquisa e estudo.

3.3.1.2 Análise de demandas e capacidades

A análise das demandas, para qualquer organização existente pode ser extraída de seu histórico recente. O número de atividades dividido pela duração do último projeto gera a métrica de quantas atividades por semana foram solicitadas. Dado que o projeto foi finalizado, tem-se também, o número de atividades realizadas. Ou seja, tem-se a capacidade histórica da equipe que participou do projeto. Neste contexto, é necessário considerar se houve mudanças na equipe e entender isso como um ponto de variabilidade nesta métrica. Outro ponto importante é entender o tipo de demandas e classificá-los.

I. Identificação dos tipos de itens de trabalho

II. Identificação de classes de serviço

3.3.1.3 Mapeamento do fluxo de trabalho

O mapeamento do fluxo de trabalho consiste em descrever em colunas todas as etapas que um objeto de valor percorre na organização desde a sua concepção até a sua entrega. À somatória destas etapas é dado o nome de Cadeia de Valor.

I. **Mapeamento da cadeia de valor**

II. **Identificando mudanças iniciais**

3.3.1.4 Desenho do quadro Kanban

O desenho do quadro é a etapa que caracteriza a aplicação do método. Consiste em compilar todos os registros levantados nas etapas anteriores e estruturá-los em um quadro que seja posicionado em um local de fácil visibilidade e acessibilidade para os desenvolvedores.

I. **Desenho da estrutura do quadro (*frame*)**

Nas linhas deve-se aplicar as classes de serviço. Como exemplo: Expedite e Standard. Nas colunas, deve-se aplicar as etapas da cadeia de valor mapeadas anteriormente. Como exemplo: Backlog, To Do, Doing, Ready to Test, Testing, Done.

3.3.1.5.Reunião de início de projeto (Kick-off)

Todos estes aspectos citados acima devem ser amplamente divulgados ao time.

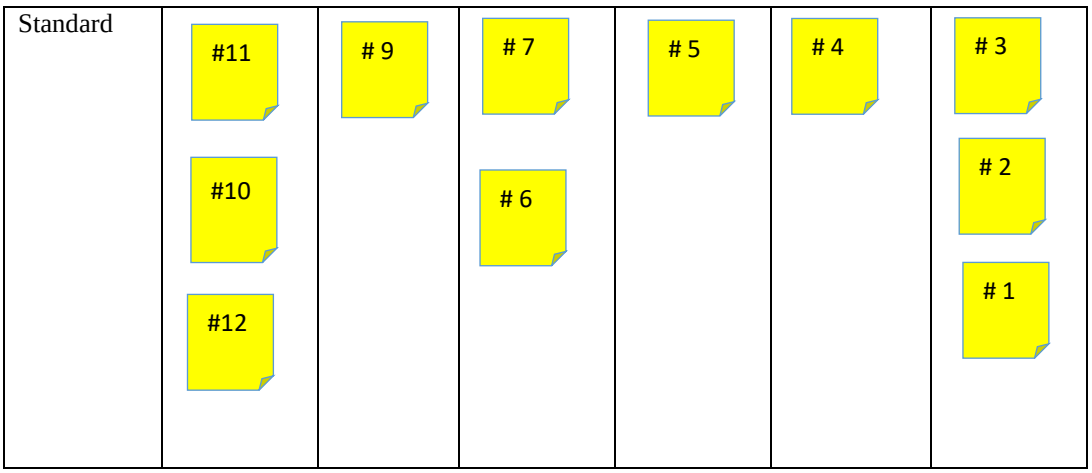
É premissa do método que a equipe seja auto gerenciável. Desde que exista um *backlog* priorizado, uma equipe madura utilizando o método Kanban deve ser capaz de definir o que precisa ser feito apenas observando o quadro Kanban e atualizando-o diariamente.

Na reunião de início do projeto e de utilização do método, deve ficar claro o papel de cada membro em controlar a atividade que está desenvolvendo, o espírito de equipe em auxiliar os colegas em dificuldade e a busca constante por qualidade.

Na figura a seguir, pode-se observar como as tarefas ficam disposta em um determinado momento no fluxo de trabalho do dia a dia de um projeto em andamento:

Tabela 2 Etapas de desenvolvimento conforme o quadro Kanban

Projet A	Backlog	TO DO	Doing	Ready to test	Testing	Done
WIP Máx		3	4	3	3	
Expedite			# 8			



Fonte: Adaptado de (Anderson, 2010)

Neste fluxo, o WIP máximo está sendo respeitado em todas as etapas. Há possibilidade de se iniciar uma nova tarefa ou atender uma demanda urgente que apareça.

3.3.2 Métricas

Após o início do projeto, em aproximadamente duas semanas já é possível visualizar algumas métricas. As métricas permitem fazer o acompanhamento do projeto, mostrar se as decisões tomadas foram assertivas e ajudam a planejar as próximas.

As principais são (Hammerman, Sunden, 2014):

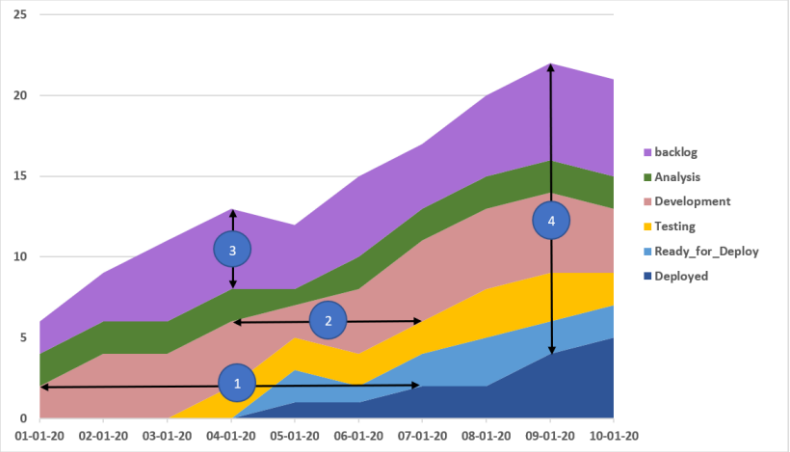
1. *Cycle and Lead Times*
2. *Throughput*
3. Issues and blocked work items
4. Due-date performance
5. Quality
6. Value demand and failure demand
7. Abandoned and discarded ideas

3.3.3 Cumulative Flow Diagram (CFD)

O CFD é útil para visualizar uma série de informações pertinentes ao andamento do projeto e serve de plano de fundo para discussões de melhorias no processo. É o diagrama mais utilizado sendo possível identificar WIP, Lead Time, cycles times e gargalos no fluxo. (Corona, Pani, 2013)

O CFD é um gráfico com uma unidade de tempo no eixo horizontal e a quantidade acumulada de tarefas segregadas por etapa no eixo vertical.

Figura 3 cumulative flow diagram



Fonte: Adaptado de (Hammarberg, Sunden, 2014)

O CFD permite a leitura imediata do Lead Time, Cycle time, Backlog e WIP (Hammarberg, Sunden, 2014).

4 PROPOSTA

4.1 UTILIZAÇÃO DE QUADRO VIRTUAL AO INVÉS DO QUADRO FÍSICO

Na proposta original de (Anderson, 2010), o quadro físico é o principal elemento, ao redor do qual, os membros do time de desenvolvimento se reúnem para fazer as discussões a respeito das suas dificuldades e, também, para atualizar o andamento das atividades nas reuniões diárias.

Dentre as consequências desta prática, pode-se citar:

1. Manutenção da atualização das atividades;
2. Busca coletiva pelos pontos de gargalo;
3. Busca coletiva para identificar impedimentos;
4. Decisão de aplicação de *Swarming* para uma determinada atividade;
5. Melhoria da visualização do fluxo de valor; (Laanti, Kangas, 2015)

Assim, alterar o método proposto em um dos seus pilares parece ser algo impossível de se pensar sem descaracterizar o método.

Entretanto, é possível utilizar recursos tecnológicos como lousas digitais ou telas sensíveis ao toque para substituir o quadro sem perda significativa de seus benefícios se sua manipulação durante as reuniões for similar ao quadro físico.

Outra possível solução é a utilização de aplicações Web para exibir o quadro nas estações de cada desenvolvedor. Tendo, cada um, o acesso ao quadro virtual por meio da sua estação de trabalho

4.4.1 telas sensível ao toque ou lousas digitais

A proposta de se utilizar um painel digital sensível ao toque não altera em nada significativo o dia-a-dia no que diz respeito à aplicação original do método. Salvo a economia com material de papelaria.

Contudo, tal proposta exige um investimento elevado, o que o torna inviável para muitas empresas.

As organizações que possuírem recursos para adquirirem lousas digitais ou monitores grandes sensíveis ao toque que possibilitem que as reuniões de Kanban aconteçam da mesma forma que aconteciam com o quadro físico se beneficiarão com os ganhos nas automações de extrações de métricas, conforme será apresentado a seguir, sem terem que passar pelas adaptações que as organizações que não podem ou não desejam fazer tal investimento necessitariam.

4.1.2 Quadro virtual compartilhado

Permitir a visualização do painel de atividades pela Web é a proposta de muitos sistemas de controle de atividades on-line que estão sendo lançados no mercado como: *Lean Kit Kanban*, *Agile Zen*, *Target Process*, *Silver Catalyst*, *RadTrack*, *Kanbanery*, *Version One*, *Jira*, *Flow.io* e *Kanban tool*.

A ausência de um quadro físico, substituído por um quadro virtual que pode ser acessado de qualquer estação de trabalho do time ou dos seus celulares implica em mudanças inevitáveis em relação à prática original do Kanban.

Ao propor a retirada do quadro físico, é necessário endereçarmos uma discussão a cada um de seus benefícios e explicarmos como eles serão supridos com a modificação. Assim, podemos mitigar os efeitos negativos desta modificação.

A reunião diária, que, acontece em torno do quadro físico, deve continuar sendo feita. Porém, cada integrante deve acompanhar a reunião a partir de sua estação de trabalho dado que não há mais um quadro disponível no ambiente.

Algumas ferramentas que podem ser utilizadas para esta reunião são: *Microsoft Skype*, *Microsoft Teams*, *Google Hangouts*, entre outras.

O coordenador, gerente ou líder técnico pode compartilhar a tela e todos podem acompanhar a reunião por meio dessa visualização.

4.1.2.1 Manutenção da atualização das atividades

A manutenção das atividades, deve poder ser feita durante as reuniões diárias. Assim, é importante escolher uma ferramenta que facilite a movimentação das atividades de maneira fluente e simples.

4.1.2.2 Busca coletiva pelos pontos de gargalo

Como todos os integrantes do time podem acessar o quadro virtual, todos podem ter a mesma visualização sobre o fluxo para tentarem identificar os gargalos. Dessa forma, a identificação dos gargalos não deve ser prejudicada.

4.1.2.3 Busca coletiva para identificar impedimentos

As ferramentas de comunicação permitem que todos os integrantes conversem em tempo real. Até mesmo quem está remoto pode compartilhar suas opiniões e relatar suas dificuldades para todos. Assim, é possível identificar eventuais impedimentos com a mesma eficiência.

4.1.2.4 Decisão de aplicação de Swarming para uma determinada atividade

Quando um impedimento é identificado, é possível atribuir mais pessoas para ajudar. As ferramentas podem prover estas funcionalidades para que esta ajuda fique exposta de maneira clara para todos. Caso contrário, ainda é possível atribuir uma observação na própria tarefa.

4.1.2.5.Melhoria da comunicação

Desde que agendadas de maneira rotineira, as reuniões via ferramentas de comunicação proveem os mesmos resultados do que as reuniões presenciais.

4.2 EVIDÊNCIAS QUE SUPORTAM A PROPOSTA

Foram encontrados diversos relados na bibliografia sobre empresas que já utilizam quadros virtuais para auxiliar em suas atividades. Muitas delas com equipes remotas e que demandam esta alternativa. Há relatos, também, de uso misto, sendo o quadro virtual, apenas uma réplica para viabilizar métricas e permitir o acesso remoto.

Esta proposta defende que o método Kanban não mais necessita da utilização de um quadro físico para funcionar. Sendo este, um meio utilizado massivamente pelas consultorias de implementação do método para permitir a introdução e o treinamento básico dos gestores e desenvolvedores que não estavam acostumados com a prática. O estudo de caso que será apresentado a seguir, extraído de uma empresa de grande porte da vertical de alimentos do varejo, mostra uma utilização bem sucedida dos quadros virtuais durante um período de, aproximadamente, dois anos de observação.

5 ESTUDO DE CASO

O estudo de caso consiste em acompanhar a execução de um projeto de desenvolvimento de software que envolve a evolução de um sistema existente (C/C++) adaptando-o a novas tecnologias (C#) e a uma nova arquitetura (micro serviços) em uma empresa de grande porte.

5.1 CARACTERÍSTICAS DA ORGANIZAÇÃO E DO TIME DE DESENVOLVIMENTO

A organização possui diversos produtos e times específicos para atender cada um deles. O projeto foi executado por uma equipe de um dos produtos, possuindo, aproximadamente, 15 pessoas, 2 coordenadores, 1 gerente e 1 gerente sênior. Dos desenvolvedores, nenhum de nível Junior, 2 plenos e o restante seniores. Todos possuem conhecimento básico do método Kanban, do sistema de versionamento *git* e da ferramenta de controle de projetos *Jira*.

5.1.1 Processo de Desenvolvimento

O processo de desenvolvimento da organização de uma determinada tarefa se inicia na sua priorização.

Quando uma tarefa é priorizada ela entra no quadro Kanban e o tempo de desenvolvimento começa a contar. O primeiro desenvolvedor livre puxa a tarefa e inicia o desenvolvimento. O desenvolvedor deve indicar que iniciou a tarefa no *Jira*.

Ao completar a tarefa, o desenvolvedor deve confirmá-la (commitar) no *git* e indicar a sua conclusão no *Jira*. Um outro desenvolvedor deve, então, revisar a tarefa e indicar no Jira o término da revisão. Dessa forma, um dos coordenadores ou os gerentes podem aplicar a alteração no código de desenvolvimento (merge na branch de desenvolvimento).

5.2 ACOMPANHAMENTO DO PROJETO

Durante todo o projeto, os coordenadores, juntamente com seus desenvolvedores faziam uma reunião diária onde o progresso de cada um era reportado e os avanços eram registrados no Jira. Na reunião, era também possível, conversar sobre dúvidas particulares e repriorizar tarefas.

É importante ressaltar que, nos dias em que as reuniões não aconteciam, algumas atividades eram atualizadas com atraso no dia seguinte quando a reunião acontecia e o desenvolvedor reportava ou era lembrado pelos colegas ou pelo coordenador.

5.3 CONTROLE DE VERSÃO

O versionamento de código fonte é um assunto bastante antigo e já é de comum senso a sua necessidade, principalmente, em projetos de grande escala. No entanto, apresentamos esta breve

descrição para tornar mais claro como o software de controle de versão foi utilizado para levantar as métricas do estudo de caso.

O software de controle de versão define como “Master” a “linha” imaginária do tempo que contém o código consolidado.

Quando uma tarefa é iniciada, um ramo (branch) é criado, a partir do código consolidado (Master) e as alterações são registradas com segurança e de maneira isolada.

O software não registra o momento de criação do branch, e sim, o primeiro registro (commit) associado a este branch. Esta informação pode variar muito dependendo da maneira como o desenvolvedor prefere trabalhar, ou seja, ele pode fazer muito trabalho em usa máquina local sem enviar para o servidor por vários dias e enviar tudo quando estiver prestes a terminar a tarefa.

Dessa forma, adotou-se a mesma data registrada pelo quadro de atividades, como a data de criação do ramo (Branch).

A 2ª etapa que foi observada neste software não está representada na tabela, mas está nos registros, é a data de abertura do pedido de revisão (merge-request). Quando o desenvolvedor faz esta requisição significa que ele já terminou o desenvolvimento da tarefe e agora solicita a aprovação de um revisor. (No quadro Kanban, é quando ele muda a atividade para a coluna “Aguardando Revisão”.

A terceira etapa é o “merge” ou união do código desenvolvido na tarefa com o código fonte principal consolidado. Que representa a última atividade de desenvolvimento. A tarefa agora, aguarda os testes ou é concluída se for uma tarefa técnica ou não-funcional do ponto de vista de negócio.

5.4 GERAÇÃO DO BACKLOG

Após a quebra os requisitos e as definições das atividades pelos coordenadores, o backlog é gerado e priorizado. Não há um momento no tempo específico para se revisar o backlog. Porém, sempre que há um evento externo que exija uma alteração, o backlog pode ser modificado. Esta é uma das características do Kanban. Nenhuma mudança na direção e nos requisitos do projeto afeta negativamente o time de desenvolvimento. Tudo o que está no backlog pode ser modificado, removido ou substituído sem prejuízo ou até mesmo ciência deste time que apenas se preocupa com as atividades que eles se comprometeram a fazer (coluna “TO DO”) em diante.

5.5 DEFINIÇÃO DE PRONTO

O time de desenvolvimento possui um checklist que precisa ser atendido para considerar uma tarefa como pronta. Entre outros itens do checklist, estão:

- Revisão do código por um ou mais pares;
- Completude de 95% do código com testes unitários;
- Atualização do andamento da tarefa no quadro de acompanhamento *(Jira)*;

- Atualização do código no controlador de versão *(Git)* com indicação da tarefa pertinente.

Há ainda outros itens, mas são irrelevantes para este estudo.

5.6 REUNIÕES DIÁRIAS

As reuniões diárias aconteceram via ferramenta de comunicação de grupo (Skype for business), no caso, com cada desenvolvedor em sua máquina. Nas reuniões, os seguintes itens eram discutidos:

- Dificuldades individuais eventuais
- Próximas atividades e priorização
- Melhorias nos fluxos e nos processos
- Compartilhamento de conhecimentos pontuais de alta relevância
- Atualização no *Jira* das tarefas em atraso.

O projeto seguiu por 2 anos e durante este tempo, centenas de tarefas foram executadas. Diversos bugs foram captados pelo time de qualidade e foram introduzidos e priorizados no backlog. Bugs críticos receberam prioridade máxima e entravam sempre como alta prioridade.

Após este período, foi possível montar comandos específicos nas ferramentas utilizadas para se extrair informações do projeto a fim de se realizar as devidas análises.

5.7 EXTRAÇÃO E ANÁLISE DOS DADOS

Tanto os sistemas Git e Jira utilizados no projeto fornecem APIs para extração de dados. Os dados extraídos do Git foram os seguintes:

Tabela 3: Dados extraídos da ferramenta de controle de versão

Name – Nome da branch Project – Nome do projeto Repo – Nome do repositório Title – Título da branch Description – Descrição da branch Id – identificação da branch Created – Data de criação Updated – Data do último update FromRef – Nome da branch que originou o último *pull-request* deste branch. ToRef – Nome da branch de destino do último *pull-request* desta branch. FromRef – Nome de exibição da branch que originou o último *pull-request* deste branch. ToRef – Nome de exibição da branch de destino do último *pull-request* desta branch.	public class Branch { [JsonProperty("Name")] public string Name { get; set; } [JsonProperty("project")] public string Project { get; set; } [JsonProperty("repo")] public string Repo { get; set; } [JsonProperty("Title")] public string Title { get; set; } [JsonProperty("Description")] public string Description { get; set; } [JsonProperty("Id")] public string Id { get; set; } [JsonProperty("Created")]

	public string Created { get; set; } [JsonProperty("Updated")] public string Updated { get; set; } [JsonProperty("FromRef")] public string FromRef { get; set; } [JsonProperty("ToRef")] public string ToRef { get; set; } [JsonProperty("FromRefDisplayId")] public string FromRefDisplayId { get; set; } [JsonProperty("ToRefDisplayId")] public string ToRefDisplayId { get; set; } }

Fonte: próprio autor

Os dados extraídos do Jira foram os seguintes:

Tabela 4: Dados extraídos da ferramenta de controle de projeto

Id – Número de identificação da tarefa no Jira **IssueType** – Tipo de tarefa (correção de bug ou nova funcionalidade) **CreatedAt** – Data da criação da tarefa no Jira **InProgress** – Data de início de desenvolvimento. **WaitingReview** – Data do término do desenvolvimento (envio para revisão) **UnderReview** – Data do início da revisão. **Closed** – Data do término da tarefa **ReadyForTesting** – Data do término da tarefa (enviada para outra equipe).	public class Task { [JsonProperty("Id")] public string Id { get; set; } [JsonProperty("IssueType")] public string IssueType { get; set; } [JsonProperty("CreatedAt")] public string CreatedAt { get; set; } [JsonProperty("InProgress")] public string InProgress { get; set; } [JsonProperty("WaitingReview")] public string WaitingReview { get; set; } [JsonProperty("UnderReview")] public string UnderReview { get; set; } [JsonProperty("Closed")] public string Closed { get; set; } [JsonProperty("Ready for Testing")] public string ReadyForTesting { get; set; }

Fonte: Próprio autor

De acordo com os processos pré-definidos, o nome (display-id) das branchs (git) devem conter o mesmo padrão de nomes das tarefas cadastradas no Jira.

Número de tarefas obtidas por meio das APIs no Jira: 212

Número de branches encontradas por meio das APIs no Git: 436

Número de tarefas consistentes e com dados completos encontradas em ambas as bases para efeito de comparação: 101

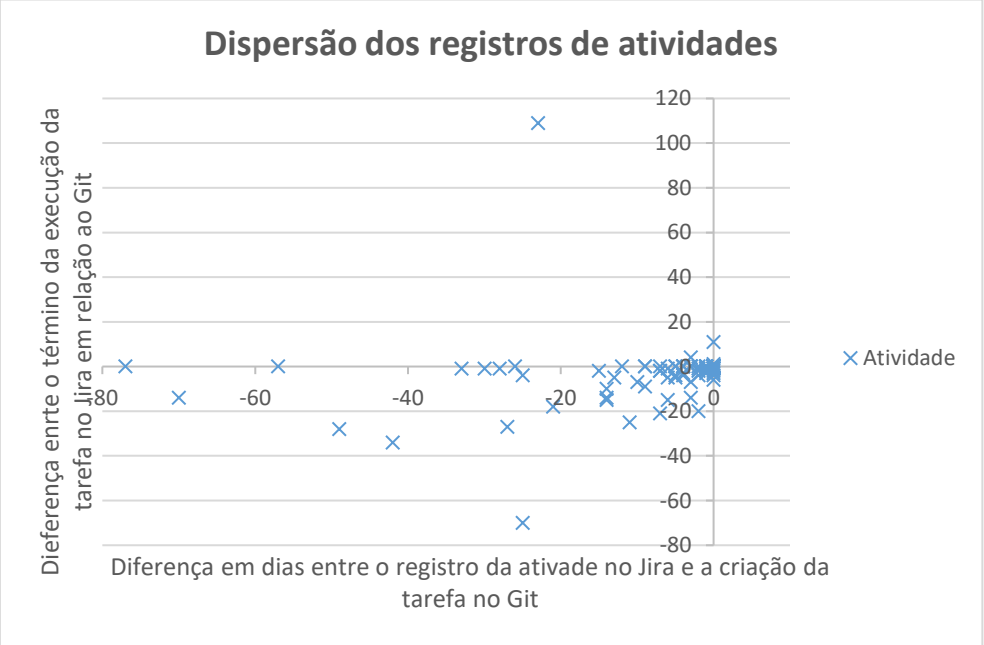
(Alexandre, 2020)

6 RESULTADOS

Após o processamento das métricas colhidas durante a execução do projeto, foi possível observar, naturalmente, que as tarefas, em sua maioria, foram iniciadas no mesmo dia em que havia o comprometimento com a mesma ou no dia seguinte. Em alguns casos, o atraso foi maior e apareceram alguns outliers que, na análise qualitativa, foram identificados como assuntos postergados logo depois do início do desenvolvimento em que o desenvolvedor entendeu repensar na continuação da tarefa. Esta análise aparece na parte horizontal do gráfico da Figura 4.

Além da observação do início das tarefas, foi possível comparar o término efetivo das mesmas (*git*) com o término anunciado pelo desenvolvedor (*Jira*). Ou seja, quanto tempo a tarefa demorou para ser comunicada de seu encerramento, após o seu encerramento efetivo. Da mesma forma, os resultados foram bastante alinhados. A maioria foi comunicada no mesmo dia, conforme o eixo vertical do gráfico da Figura 4.

Figura 4 Comparação entre as marcações efetuadas manualmente pela ferramenta de controle de projeto (JIRA) e os registros efetivamente observados durante o projeto (*Git*)



Fonte: Próprio autor

Com estes resultados foi possível concluir, que o uso do quadro virtual de acompanhamento de atividades (Jira) mostrou na maioria das vezes, que as atividades marcadas como iniciadas ou finalizadas estavam, de fato com este status. Independentemente da eficácia dos quadros virtuais é possível concluir que o quadro virtual permitiu um acompanhamento correto do andamento do projeto. Além disso, permitiu a extração automatizada das informações supracitadas e de outras análises que permitem aprofundar a empresa no desenvolvimento de níveis mais avançados de maturidade dentro do Kanban.

6.1 TRABALHOS FUTUROS

É possível criar integrações entre as ferramentas de versionamento de código e a ferramenta de controle de projeto de modo que a tarefa mude de status automaticamente conforme algumas etapas do desenvolvimento são cumpridas.

Dessa forma, haveria uma consistência fiel entre os cartões das atividades que estão vinculadas ao desenvolvimento de código e a conclusão efetiva do trabalho. Mitigando, assim, o risco do erro humano no registro das atividades.

6.2 CONCLUSÃO

Idealizado por (Anderson, 2010), o uso do método Kanban aplicado ao desenvolvimento de software obteve diversos resultados positivos e tem evoluído ao longo dos anos. Atraiu autores e é, talvez o método que mais cresce dentro do movimento ágil (Corona, Pani, 2013).

A troca do quadro físico pelo quadro virtual é mais um importante marco evolutivo para o método pois viabiliza a extração automatizada de métricas e a redução do erro humano durante as atualizações diárias.

Outra importante contribuição da virtualização é a criação de oportunidades para integrações entre o quadro e as ferramentas de versionamento de código fonte e de entrega contínua.

Com informações mais consistentes, os gestores e equipes terão maior confiança e segurança para realizarem suas tarefas.

REFERÊNCIAS

Ahmad, M.O., Dennehy, D., Conboy, K., Oivo, M., 2018. Kanban in software engineering: A systematic mapping study. The Journal of Systems and Software 137: 96-113. DOI https://doi.org/10.1016/j.jss.2017.11.045
0164-1212

Ahmad, M. O., Liukkunen K., Markkula J., 2014. Student perceptions and attitudes towards the software factory as a learning environment. International conference on Global Engineering Education: 422–428.

Ahmad, M. O., Markkula, J., Oivo, M., 2013. Kanban in software development: A systematic literature review. *2013 39th Euromicro Conference on Software Engineering and Advanced Applications*, 9–16. http://doi.org/10.1109/SEAA.2013.28.

Al-Baik, O., and Miller J., 2015. The Kanban approach, between agility and leanness: a systematic review. Journal of Empirical Software Engineering: 20(6): 1861–1897.

Alexandre, T., 2020. Kanban tasks opening and close dates differences between git and Jira (Version 1.0.0) [Data set]. Zenodo. http://doi.org/10.5281/zenodo.4317101

Anderson, D. J. 2010. Kanban: Successful Evolutionary Change for Your Technology Business. Blue Hole Press.

Corona, E., and Pani, F. E., 2013. A review of Lean-Kanban approaches in the software development. Transactions on Information Science and Applications: 10(1):1–13.

Dennehy, D. and Conboy, K., 2016. Going with the flow: An activity theory analysis of flow techniques in software development. Journal of Systems and Software.

Fitzgerald, B., Musiał, M. and Stol, K. J., 2014. Evidence-based decision making in lean software project management. 36th International Conference on Software Engineering Companion: 93–102. ACM.

Hammarberg, M., Sunden, J. 2014. Kanban in Action. (1st. ed.). Manning Publications Co., USA. ISBN:978-1-61729-105-0

Harzl, A. 2016. May. Combining FOSS and Kanban: An action research. In IFIP International Conference on Open Source Systems: 71–84. P12. HeikkiläVT., Paasivaara, M., Lassenius, C., 2016. Teaching university students Kanban with a collaborative board game. 38th International Conference on Software Engineering Companion: 471–480.

Ikonen, M., 2011. On the impact of Kanban on software project work: An empirical case study investigation. 16th International Conference on Engineering of Complex Computer Systems: 305–314.

Laanti, M., and Kangas, M. 2015. Is agile portfolio management following the principles of large-scale agile? Case study in Finnish Broadcasting Company Yle. In IEEE Agile Conference. 92–96.

Law, E. L. C., Lárusdóttir, M. K., 2015. Whose experience do we care about? analysis of the fitness of scrum and Kanban to user experience. International Journal of Human - Computer Interaction: 31(9): 584–602.

Mahnic, V., 2015. From Scrum to Kanban: introducing lean principles to a software engineering capstone course. International Journal of Engineering Education.

Middleton, P. and Joyce, D. 2012. Lean Software Management: BBC Worldwide Case Study, IEEE Transactions on Engineering Management, vol. 59, no.1, pp. 20-32, Feb.

Nevenka, K., Saso K., 2015. Usage of Kanban Methodology at Software Development Teams. Journal of applied economics and business. Vol 3: 25-34.

Poppendieck, M., & Poppendieck, T. 2003. Lean Software Development: An Agile Toolkit: An Agile Toolkit. Addison-Wesley.

Rodríguez, P., Partanen, J., Kuvaja, P. Oivo, M., 2014. Combining lean thinking and agile methods for software development: A case study of a Finnish provider of wireless embedded systems detailed. 47th Hawaii International Conference on System Sciences: 4770–4779.

Santos, P. S. M., Beltrão, A. C., Souza, B. P., Travassos, G. H., 2018. On the benefits and challenges of using kanban in software engineering: a structured synthesis study. Journal of Software Engineering Research and Development. https://doi.org/10.1186/s40411-018-0057-1.

Senapathi, M., Middleton, P., Evans, G., 2011. Factors affecting effectiveness of agile usage–insights from the BBC Worldwide case study. International Conference on Agile Software Development, Springer Berlin Heidelberg: 132–145.

Siderova S., "Maximize Customer Satisfaction: Kanban Cycle Time". Disponível em <https://getnave.com/blog/kanban-cycle-time/>. Último acesso em 21/04/2020.

Sjøberg, D. I., Johnsen, A. Solberg, J., 2012. Quantifying the effect of using Kanban versus scrum: A case study. IEEE software: 29(5): 47–53.

Tripathi N., Rodríguez P., Ahmad, M. O., Oivo, M., 2015. Scaling Kanban for software development in a multisite organization: challenges and potential solutions. International Conference on Agile Software Development, Springer International Publishing: 178–190.

Vacanti, S.D. 2015. Actionable Agile Metrics for Predictability: An Introduction Inc. Actionable Agile Metrics for Predictability: An Introduction, Leanpub.

Wan, H., Chen, F.F. 2008. A Web-based Kanban system for job dispatching, tracking, and performance monitoring. Int J Adv Manuf Technol () 38: 995.

Wang, X., Conboy, K., Cawley, O., 2012. An experience report analysis of the application of lean approaches in agile software development. Journal of Systems and Software. 1287-1299.

Womack, J., Jones, D. & Roos D. 1990. The Machine That Changed the World. ASIN: B001D1SRRS.