Métodos Ágeis Específicos

Adoção e aplicação dos métodos Kanban e eXtreming Programing em times ágeis.

Prof.ª Carla Krieger

Propósito

Conhecer vários métodos, entre eles, os ágeis, é essencial para que o profissional possa apoiar seu time e sua organização na escolha e no tailoring de abordagens, que mais se adequem ao contexto e ao propósito a ser alcançado, alavancando resultados. Kanban e eXtreaming Programing são métodos precursores na história da agilidade.

Objetivos

- Reconhecer os conceitos básicos do método Kanban.
- Identificar os contextos de uso adequados da metodologia eXtreming Programming.

Introdução

Os métodos ágeis estão sendo cada vez mais adotados nas organizações, na busca por uma resposta mais rápida às novas necessidades do mercado, da entrega de valor e da melhoria contínua.

Existe uma variedade enorme de abordagens e práticas disponíveis a serem exploradas pelos profissionais do mercado. Neste conteúdo, abordaremos duas delas: Kanban e eXtremming Programming (XP).

O método Kanban nasceu no final dos anos 1940, na indústria, como parte do sistema Toyota de produção, com foco na otimização de fluxos de trabalho. Já a metodologia XP surgiu no final dos anos 1990, após o início da internet, que levou à necessidade de mudanças no ciclo de desenvolvimento de software.

Origem, valores e princípios do Kanban

O Kanban surgiu como um sistema de gestão visual utilizado para registrar atividades em forma de cartões. É literalmente poder ver as tarefas em um fluxo, de forma que ajude a melhorar o seu processo.

O sistema foi criado por Taiichi Ohno, em 1953, e faz parte do sistema Toyota de produção. Em 2010, David J. Anderson e Don Reinertsen adaptaram o modelo, chamando-o de "Kanban para desenvolvimento de software", combinando o **lean mindset** com a teoria das restrições, transformando-o em uma abordagem de desenvolvimento de software ágil.

Lean mindset

Combinação de crenças, suposições, atitudes e valores pessoais alinhados com o manifesto ágil.

Com o passar dos anos, devido à capacidade adaptativa e evolucionária, o método se estendeu para além da indústria de software, sendo adotado em diversas áreas, como Recursos Humanos (RH), área jurídica etc.

Vamos conhecer os diferentes Kanbans:

Kanban (Kan = Visual e Ban = Cartão)

Cartão com a atividade registrada.

Quadro Kanban

Quadro organizado por colunas, que representam um processo de trabalho pelo qual passam itens ou tarefas desenvolvidas por um grupo de pessoas ou por um time bem estruturado. É visual, podendo ser físico ou virtual, no qual as atividades ficarão explícitas a todos os membros de um time para gerenciar o fluxo de trabalho, sempre priorizando o valor.

Método Kanban

Mais do que o quadro, o método Kanban é composto por um conjunto de princípios, práticas e cadências.

O Kanban utiliza o sistema puxado. No sistema empurrado, geramos um estoque do produto, seguindo um roteiro: produzir > estocar > vender; já no sistema puxado, a demanda é gerada pelo cliente, que puxa a oferta, com adaptação do ritmo produtivo e estoques.

Quadro Kanban.

O sistema do Kanban é pautado na atitude de que todas as pessoas que contribuem para um propósito comum devem ser respeitadas. Esse sistema tem como pilar os princípios e os valores que as organizações devem adotar para, de fato, serem ágeis, por meio de **tailoring** de abordagens metodológicas adequadas a seu objetivo e contexto de atuação.

tailoring

No contexto de gerenciamento de projetos, trata-se de uma realidade na qual nada é fixo ou imutável em relação aos métodos de gerenciamento (ágeis) de projetos. Está associado à necessidade de adequação ou resiliência que as organizações (e o gestor de projeto) precisam ter quanto a determinado método (ágil) de gerenciamento de projetos, para que sejam adaptados e customizados de acordo com a realidade e o contexto de cada organização.

Os valores do Kanban incluem:

Transparência

Com troca aberta de informações e um vocabulário claro e sem ruídos, você cria transparência em todas as áreas.

Equilíbrio

A organização se tornará eficiente se equilibrar as diferentes exigências, visões e habilidades de todos os participantes entre si.

Colaboração

O método Kanban melhora a forma como as pessoas trabalham juntas. A colaboração é, portanto, um de seus pontos fortes.

Foco no cliente

Os clientes e o valor por eles percebidos são o centro natural de interesse de todas as pessoas envolvidas na organização.

Fluxo de trabalho

O trabalho representa um fluxo contínuo ou pontual de geração de valor. Um ponto de partida importante no uso do Kanban é mapear tal fluxo de trabalho, tornando-o visível, e mantê-lo.

Liderança

A competência de liderar é necessária em todos os níveis para gerar valor e alcançar um estado melhorado.

Compreensão

Kanban é um método de melhoria, e o aperfeiçoamento requer mudança, o que exige compreensão (autoconsciência). Isso significa que, primeiro, precisamos entender o trabalho e os processos envolvidos para, depois, melhorá-los. Comece sempre por como você está fazendo e entenda por que o está fazendo.

Acordo

Em qualquer situação, estabeleça políticas explícitas sobre como serão tomadas as decisões dentro de seu Kanban.

Respeito

O respeito às pessoas na forma de reconhecimento, compreensão e consideração é a base sobre a qual os outros valores se fundamentam.

Agora, vamos conhecer os princípios do Kanban:

Comece a terminar e pare de começar

Ter muitas tarefas em paralelo leva a um aumento no tempo de entrega, enquanto terminar algo que já está em andamento, antes de iniciar algo novo, possibilita pequenas entregas, agregando mais cedo valor aos clientes. No Kanban, é orientado a sempre puxarmos primeiro os cartões que estão nas etapas finais do fluxo de desenvolvimento, ou seja, a leitura e a gestão devem ser sempre da direita para a esquerda do quadro Kanban.

Faça o trabalho usando um quadro visual

Gestão à vista permite uma tomada de decisão mais rápida, melhora a comunicação e promove a transparência nas organizações.

Puxe apenas quando houver capacidade para fazer o trabalho

Puxar algo novo enquanto ainda existem tarefas a serem completadas, ocasionará gargalos, levando ao desperdício.

Priorize valor

Faça primeiro o que entregar mais valor mais cedo.

Gerencie riscos

Ainda que estejamos inseridos em contextos de ambientes complexos de alta volatilidade, de pouca previsibilidade, uma boa análise de riscos é primordial, para, quando não for possível eliminarmos, termos respostas rápidas no caso de ocorrência de alguns deles.

Construa uma cultura de alta confiança

Devemos promover um ambiente psicologicamente seguro, no qual as pessoas possam trabalhar verdadeiramente focadas em um propósito comum, sem medos de exposições, podendo ser vulneráveis.

Responda rápida e graciosamente às mudanças

Buscamos manter um fluxo otimizado, trabalhando dentro da capacidade e nas tarefas de maior valor da entrega. Isso possibilita maior flexibilidade e adaptabilidade para atender às mudanças que surgirão durante o desenvolvimento de produtos e soluções.

Desenvolva o processo que já está em vigor

Existe uma tendência natural de que, quando estamos mapeando fluxos de trabalho, querermos inserir melhorias. Porém, a boa prática recomendada é a de realizar o mapeamento de como o time trabalha. Rode o fluxo por determinado tempo, colete métricas e realize as melhorias nos pontos de maior gargalo ou bloqueadores.

Fluxo de trabalho e métricas de eficiência

O fluxo completo da entrega, desde a idealização da nova funcionalidade ou do novo serviço até o momento em que a ideia gera de fato valor para o cliente, ou seja, o momento em que ele pode usufruir da funcionalidade no ambiente de produção (fluxo end-to-end), é o Customer Kanban.

O fluxo end-to-end é composto por duas fases diferentes do serviço, que apresentam características próprias, em termos de trabalhos a serem realizados:

1Upstream Kanban

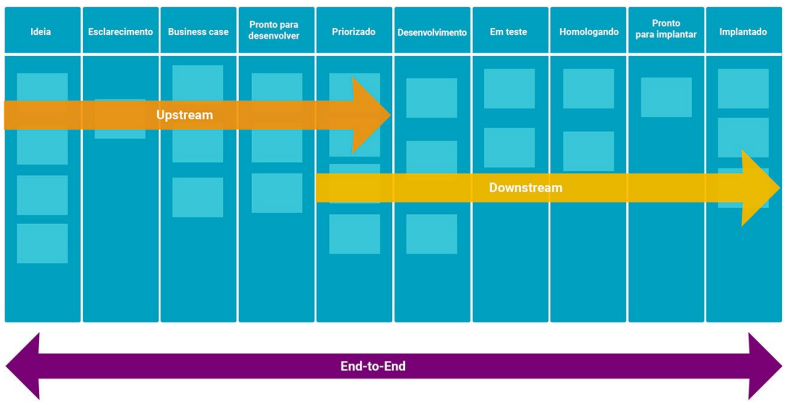
É a gestão do fluxo estruturado de trabalho para avaliar, preparar e refinar ideias, oportunidades e opções antes de chegar ao desenvolvimento, de maneira que as tarefas sejam passadas sem ruídos e na granularidade correta aos desenvolvedores, sem gerar sobrecarga. Assim, os compromissos de entregas são sustentáveis, com menor risco e menos incerteza.

2

Downstream Kanban

É a gestão do fluxo das etapas relacionadas ao desenvolvimento da solução ou a entrega. É aqui que encontramos a execução de desenvolvimento, teste, homologação e implantação, por exemplo.

Observe um quadro com melhor visualização do fluxo end-to-end:



Quadro end-to-end com a indentificação de Upstream e Downstream.

As métricas Lean são fundamentais para medir eficiência em fluxos de trabalho, independentemente da abordagem adotada (ágil, preditiva ou híbrida). Entre essas métricas, estão:

Lead time médio

Tempo entre a data de compromisso e a entrega para o cliente.

Cycle time médio

Tempo entre o compromisso e o término do desenvolvimento do item de trabalho.

Throughput

Quantidade de itens entregues em um período de tempo ou taxa de vazão.

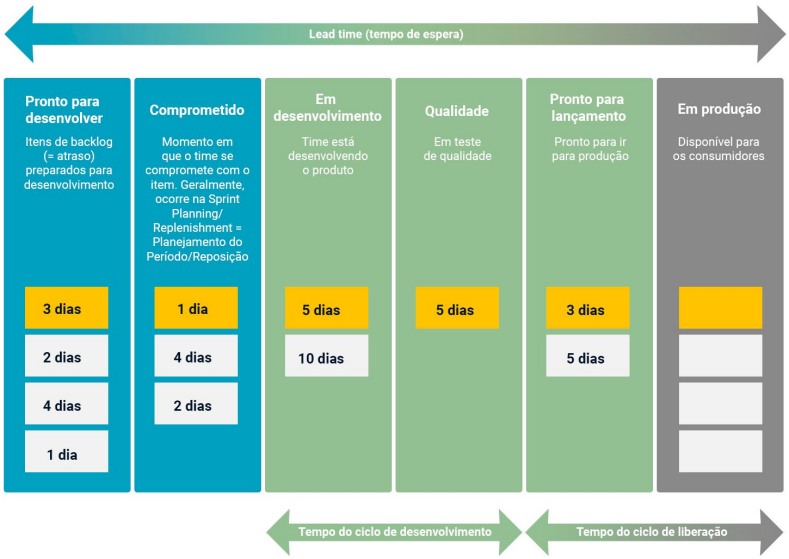
Work in Progress (WIP)

Deixa visível o trabalho em progresso, estabelecendo o número de atividades que determinado time possui, evidenciando a capacidade do fluxo de trabalho de acordo com um quadro Kanban.

Ao se limitar o WIP de um time, coloca-se em prática o conceito de finalizar mais atividades do que puxar novas e colocar várias atividades em andamento, garantindo maior foco do time. Consequentemente, atividades são encerradas com menos tempo.

A limitação do WIP permite que sejam identificados pontos de gargalos no fluxo de trabalho, evitando que determinados processos tenham sobrecarga, aumentando a frequência de entrega de valor para o cliente final.

Observe o exemplo a seguir:



Métricas Lean.

Aqui, no cartão amarelo, temos os seguintes indicadores para esse item de trabalho:

Lead Time

17 dias

Development Cycle Time

10 dias

Release Cycle Time

3 dias

E a taxa de vazão (throughput) para o período foi a de 4 itens.

No entanto, esse item de trabalho é provavelmente uma parte ou um pedacinho de 1 produto. Para termos a visão completa do lead time do produto, será necessário somarmos os valores dos indicadores de cada parte que o compõe.

> Para ter a medida da performance de um time, é necessário coletar dados históricos e trabalhar com modelos estatísticos, como média, moda, mediana e percentil, mas não nos aprofundaremos em Estatística aqui. O importante é saber que a média é um caminho, porém, não pode ser usada sozinha para dar a resposta de quanto o time entrega em determinado período de tempo.

Além disso, a granularidade da quebra dos itens de trabalho (tamanho) deve ser homogeneizada, para que possamos ter uma estatística mais acurada.

Vejamos algumas dicas para limitar o WIP:

1. Estabeleça limite com certa folga para que o fluxo continue desimpedido.
2. Identifique o gargalo.
3. Ajuste os limites um por vez.
4. Observe onde o trabalho se acumula e gradualmente ajuste o limite.

O Cumulative Flow Diagram (CFD) é um dos principais gráficos de acompanhamento de eficiência, em que se registra de forma cumulativa a quantidade de itens que passam pelo fluxo de trabalho, distinguindo as etapas do fluxo (to do, doing, done) por cores.

Esse gráfico apresenta de forma acumulada informações como Cycle Time, Throghput e WIP.

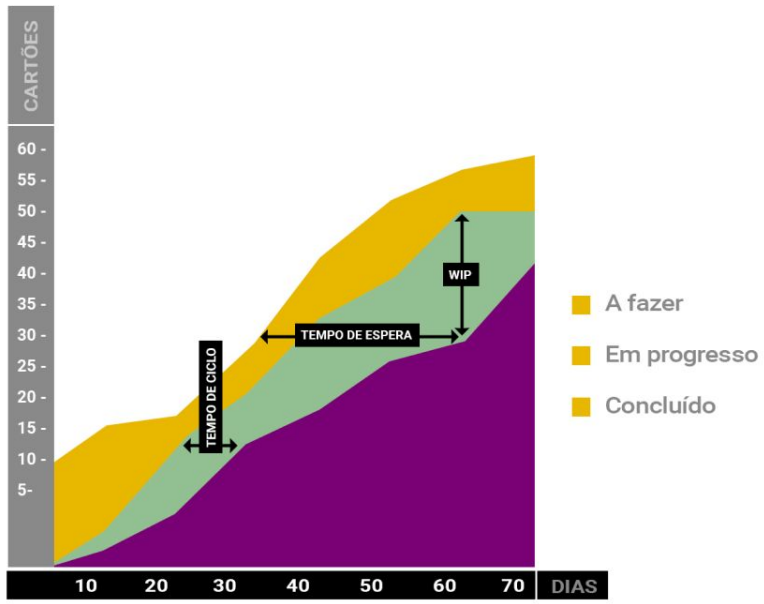


Diagrama de acúmulo no fluxo de produção.

São padrões comuns a serem observados em um CFD:

Estreitamento das faixas

Quando há uma faixa estreita no diagrama de CFD, significa que há menor volume de tarefas em paralelo àquela etapa do fluxo de trabalho.

Faixas largas

Quando há um aumento na largura da faixa de uma das etapas do fluxo de trabalho, há o indicativo de que existe algum gargalo ou bloqueio que leve à ocorrência de uma taxa de entrada mais rápida do que a taxa de saída da etapa em questão. O time deve analisar o que está causando o gargalo. Isso pode ter inúmeros significados, desde questões de capacidade até informações insuficientes sobre os itens de trabalho.

Pouca inclinação com formato praticamente plano

Quando há um desenho do gráfico praticamente plano, isso sinaliza que o trabalho não está sendo realizado. É importante o time aprofundar a investigação para entender o que está causando esse comportamento. Algumas vezes, isso pode ocorrer devido a férias ou até mesmo um período sazonal de freeze (congelamento das atividades), o que é previsto.

Práticas, cadências e papéis do Kanban

As seguintes práticas do Kanban possibilitam a obtenção de melhores resultados:

Visualize o fluxo de trabalho

O fluxo de trabalho precisa estar mapeado em um quadro para que seja possível ter sua visibilidade e, consequentemente, expor o andamento do trabalho, o que também tornará mais fácil de enxergar os pontos do fluxo que requerem melhoria.

Limite o trabalho em progresso – Work in Progress (WIP)

Se não limitarmos a quantidade de trabalho em andamento, é quase certo que teremos um sistema empurrado. Criando limite, garantimos a aplicação do sistema puxado e ganhamos mais visibilidade do problema de acúmulo de tarefas em estoque (pontos de gargalo), que, quando tratado, melhorará o tempo de entrega.

Gerencie o fluxo

No Kanban, olhamos para o fluxo de trabalho completo, e não para partes dele. O objetivo é promover uma gestão do trabalho, e não um microgerenciamento das pessoas. Em vez de buscar manter pessoas ocupadas o tempo todo, pensamos em como acelerar o ciclo de entrega de valor.

Torne as políticas de trabalho explícitas

Todos devem conhecer o processo de trabalho, suas regras, seus papéis e suas definições, pois só assim será possível que todos os envolvidos realmente estejam engajados com o processo e que exista melhoria contínua. Para haver essa transparência e clareza, é essencial que esse processo esteja registrado e acessível a todos.

Implemente ciclos de feedback

O método Kanban sugere alguns rituais para garantir os ciclos de feedback, cujo principal propósito é comparar os resultados obtidos com os esperados para realizar ajustes.

Melhore colaborativamente

Desconfortos serão gerados, e conversas começarão a ocorrer para tratá-los, com o propósito de atingir o fluxo suave e contínuo do Kanban. Isso faz parte de uma mentalidade de melhoria contínua, que torna o processo colaborativo, no qual o time identifica problemas e propõe melhorias juntos.

Embora o Kanban não seja um método prescritivo, a boa prática sugere alguns papéis e algumas cadências de rituais. As cadências são divididas em dois grupos: um com foco no time e nas entregas de serviço, e outro com foco na estratégia.

Além do time, no Kanban, tipicamente, existem dois papéis sugeridos:

Service Request Manager (SRM)

Responsável por entender as necessidades e as expectativas dos clientes.

Service Delivery Manager (SDM)

Responsável pelo fluxo de trabalho, entregando os itens selecionados pelos clientes.

O que impulsionará a efetividade do fluxo de trabalho de um time Kanban será a adoção de uma cadência adequada de planejamento e sincronismo por parte do SDM e do SRM com os times, possibilitando a melhoria contínua.

Vejamos algumas sugestões de cadências:

Ritual	Foco	Descrição	Cadência sugerida
Delivery Planning	Entrega	Planejar e monitorar as entregas para os clientes, quando a equipe se compromete a liberar o trabalho finalizado em uma data específica.	Conforme necessidade
Replenishment	Entrega	Decidir, juntamente com o cliente e/ou as partes interessadas, quais itens de trabalho serão selecionados e processados no próximo período.	Semanal
Kanban meeting	Entrega	Cada pessoa envolvida no serviço ou fluxo de trabalho deve refletir sobre o progresso do trabalho e a efetividade do sistema Kanban.	Diária
Strategy Review	Estratégia	É uma reunião regular de líderes seniores para revisar e avaliar o mercado atual, a posição estratégica, a estratégia go-to-market e os KPIs, e o alinhamento da estratégia e capabilities.	Trimestral
Operations Review	Estratégia	Destina-se a cobrir serviços interdependentes, que normalmente abrangem o trabalho de 150 a 450 pessoas, uma unidade de negócios, um grande departamento ou função.	Mensal
Delivery Review	Estratégia	O objetivo é examinar e melhorar a eficácia de um serviço selecionado.	Bissemanal
Risk Review	Estratégia	O objetivo é compreender e responder aos riscos para entrega efetiva de produtos e serviços.	Mensal

Tabela: Cadências de Rituais Kanban.

Classes de serviços

As organizações são formadas por ecossistemas independentes. No Kanban, isso é reconhecido por meio de três princípios de entregas de serviços, aplicáveis a não só um serviço, mas a sua rede. De acordo com Anderson e Carmichael (2016), esses princípios são:

1. Compreender e focar nas necessidades de expectativas dos clientes.
2. Gerenciar o trabalho, deixar que as pessoas se auto-organizem em torno dele.
3. Desenvolver políticas para melhorar os resultados dos clientes e dos negócios.

Com esse foco de compreender, gerenciar e melhorar resultados, surge o conceito de classe de serviços, amplamente utilizado na comunidade Kanban.

Os itens de trabalho não terão necessariamente o mesmo nível de valor ou pressão de prazo. Seu time vai querer que isso seja refletido explicitamente em seu quadro e em outras políticas de trabalho.

> Afinal, atribuir diferentes classes de serviço a diferentes itens de trabalho facilita a adoção de uma abordagem simples e transparente, para que a equipe se auto-organize em torno do trabalho e para atender às necessidades do negócio, tornando explícitos o valor do item de trabalho, as informações de risco e as políticas.

De quebra, isso ajuda a aumentar a satisfação dos clientes, utilizando diferentes níveis de serviço para diferentes tipos de trabalho, gerenciando de forma saudável as expectativas.

No Kanban, usualmente, são adotados quatro tipos de classes de serviço:

Expedite

Itens de alta urgência, de alto impacto, que necessitam ser trabalhados e entregues o quanto antes.

Exemplo: Um bug que parou um sistema crítico em produção, uma oportunidade de venda ou criação de um produto que está com grande demanda no mercado e nas redes sociais.

Fixed Date

Fixed Date ou Date-Driven, são demandas que precisam ser entregues até um prazo fixo, geralmente relacionado a uma situação eventual ou sazonal, e que faz sentido apenas ser entregue antes do acontecimento.

Exemplo: Um hotsite de promoções para o Natal ou a algum tipo de implementação regulatória e que poderá gerar uma penalidade se não for entregue até aquela data.

Standard

Itens de trabalho padrões e orientados segundo uma priorização explícita por algum tipo de critério (urgência, valor de entrega, ordem de pedido etc.) nos quais o time trabalhará.

Exemplo: Uma nova funcionalidade de carrinho de compras em um e-commerce.

Intangibles

Demandas de trabalho cujos ganhos ou impactos são difíceis de mensurar em curto prazo, mas que a médio ou longo prazos podem causar prejuízos, caso não sejam desenvolvidos, ou trazer bons resultados.

Exemplo: Substituir uma plataforma de solução.

É importante trabalhar alguns dos aspectos que uma classe de serviço pode afetar antes de pôr em prática. De acordo com Hammarberg e Sundén (2014), esses aspectos incluem:

Visualização

A fim de ver facilmente a classe de serviço de um item de trabalho, você quer torná-lo visual para todos os membros da equipe, bem como para outras pessoas que possam estar interessadas.

Uma maneira de fazer isso é usar uma cor especial para o item de trabalho ou indicar a classe de serviço no cartão do item de trabalho de alguma forma, como, por exemplo, com um adesivo ou um ícone.

Impacto no WIP

A classe de serviço tem ou não um impacto sobre o limite WIP? Esse impacto atinge algumas colunas e não outras? Se a classe de serviço tem sua própria raia, essa raia tem um limite WIP ou um número máximo de itens? Tem um valor mínimo (limite) do WIP? Essas são perguntas que devemos fazer para que possamos impulsionar a melhoria contínua dos serviços prestados por nosso time.

Cada classe de serviço tem uma característica específica, inclusive, na maioria das vezes, com etapas diferentes no fluxo de seu desenvolvimento. Ou seja, nem todas passam, necessariamente, por todas as colunas do kanban board. E essas especificidades devem ser mapeadas na definição do WIP.

Prioridade

Como a classe é priorizada em comparação a outras classes? Os itens de trabalho dessa classe devem ser puxados antes das outras? Ela tem uma política para que outros trabalhos sejam descartados para puxar esse dado por meio do fluxo de trabalho o mais rápido possível? Talvez, uma fila do tipo First In, First Out (FIFO) — traduzido para português “primeiro a entrar, primeiro a sair” (PEPS) e significa que os primeiros itens inseridos em um fluxo de trabalho serão os primeiros a serem executados — seja uma política adequada para esse tipo de trabalho.

Informações sobre riscos, tais como custo do atraso, exigências de habilidades especiais e impacto técnico, são considerações típicas para tornar explícitas, para capacitar e ajudar os membros da equipe a tomar boas decisões de risco a tempo.

Fluxo de trabalho diferente

A classe deve seguir um fluxo de trabalho diferente das outras classes?

- Talvez salte uma coluna – por exemplo, nenhuma análise é necessária para itens da classe “defeito de serviço”.
- Talvez tenha uma política diferente para uma etapa do fluxo de trabalho – para defeitos, um desenvolvedor que não seja um testador, pode fazer o trabalho de um testador.
- Talvez precise de uma estimativa menos detalhada do que outras classes de serviço.

Altere o fluxo de trabalho de suas classes de serviço para otimizar o resultado dos itens pertencentes a ela. Para itens urgentes, você poderia otimizar por velocidade de entrega, enquanto itens classificados como intangíveis poderiam ter a qualidade do código como seu resultado principal.

Com esses aspectos em mente, você pode planejar a adoção de classes de serviços em sua organização ou em seu time.

Kanban

Assista agora a um vídeo em que se apresenta um profundamento dos conceitos do Kanban, entendendo sua origem e em que contextos ele se adequa melhor.

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Vem que eu te explico!

Os vídeos a seguir abordam os assuntos mais relevantes do conteúdo que você acabou de estudar.

Quando usar o método Kanban?

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Métricas Lean

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Case com Kanban

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Verificando o aprendizado

Questão 1

Você está trabalhando em um time que utiliza a metodologia Kanban, e o cliente tem a percepção de que há muita demora na entrega de seu time. Qual das ferramentas é a mais adequada para seu time analisar o cenário?

A

CFD – Cumulative Flow Diagram

B

WIP – Work in Progress

C

Gantt Chart

D

Histograma

E

Dashboard

A alternativa A está correta.

O CFD tem o histórico acumulado da quantidade de itens que passam pelo fluxo de trabalho, distinguindo as etapas do fluxo (to do, doing, done) por cores, o que possibilita melhor identificação dos pontos de gargalo que poderão ser oportunidades de melhorias para a performance do time.

Questão 2

O método Kanban é

A

um cartão com atividades registradas.

B

um quadro que contém colunas com as etapas de um fluxo de trabalho.

C

um sistema de gestão visual composto por quadro Kanban, princípio, práticas e cadências.

D

visual, podendo ser físico ou virtual.

E

uma representação da teoria das restrições.

A alternativa C está correta.

Kanban faz parte do sistema Toyota de produção, com o objetivo de ser um sistema de gestão visual, permitindo vermos as tarefas em um fluxo de trabalho, apoiando na melhoria contínua.

Origem, valores e princípios do XP

Após um projeto crítico na fabricante de automóveis Chrysler, que estava fadado ao fracasso, o americano Kent Beck aplicou, na empresa, técnicas revolucionárias de engenharia de software que deram origem ao eXtreme Programming (XP).

XP é um **framework** ágil de desenvolvimento de software com o propósito de produzir software de maior qualidade, ao mesmo tempo que proporciona melhor qualidade de vida para a equipe de desenvolvimento. É o mais específico dos frameworks ágeis, no que diz respeito às práticas de engenharia para o desenvolvimento de software.

> **Framework**
>
> Em tradução literal, significa “estrutura”. No contexto do gerenciamento de projetos ágeis, refere-se a um conjunto de regras e ferramentas predefinidas, utilizadas por qualquer pessoa para realizar o gerenciamento de um projeto ágil de forma estruturada.

As características gerais de quando o XP é indicado foram descritas por Wells (2013):

- mudança dinâmica dos requisitos de software;
- riscos causados por projetos de tempo fixo que utilizam novas tecnologias;
- pequena equipe de desenvolvimento trabalhando no mesmo local;
- tecnologia que permite testes automatizados de unidade e funcionais.

Assim como o Kanban, entre outras práticas da abordagem ágil, o XP também é pautado em valores e princípios.

Os valores do XP são:

Comunicação

O desenvolvimento de software é inerentemente um trabalho de time que depende da comunicação para transferir conhecimentos de um membro da equipe para todos os outros. O XP destaca a comunicação cara a cara (face to face), com a conversa apoiada por um quadro branco ou outro mecanismo de criação colaborativa.

Simplicidade

O time sempre se questiona sobre qual é a coisa mais simples que vai funcionar, com o intuito de evitar desperdícios e fazer apenas coisas absolutamente necessárias. Simplicidade também significa tratar apenas os requisitos que são conhecidos, não tentando prever o futuro.

Feedback

Os times podem identificar pontos de melhoria e revisar suas práticas por meio de feedbacks constantes sobre o trabalho anterior.

Coragem

Ter uma ação eficaz perante o medo é a definição de coragem no XP. Os princípios do XP são habilitadores para a coragem, possibilitando que os resultados não sejam prejudiciais para o time. Precisamos de coragem para levantar questões organizacionais que reduzem a eficácia do time, parar de fazer o que não funciona e tentar algo diferente, aceitar e agir com base em feedbacks, que nem sempre são fáceis de recebermos.

Respeito

Os membros do time precisam respeitar uns aos outros, a fim de se comunicarem de forma fluida, fornecendo e aceitando feedbacks que fortaleçam os relacionamentos e o trabalho colaborativo para identificar soluções simples.

Os valores e as práticas são conectados pelos princípios. O XP declara 14 princípios:

Humanidade

Softwares são desenvolvidos por pessoas, não máquinas. Elas têm seus próprios sentimentos e necessidades, precisando de alguns padrões básicos para realizar seu trabalho, tais como segurança (física e mental), realização, pertencimento, crescimento e intimidade.

Economia

Tudo o que um time faz deve ter valor ao negócio associado. O design incremental ajuda muito aqui, porque todas as decisões de design de menor importância são transferidas para o futuro, de modo que os recursos (dinheiro) são investidos apenas em recursos que tragam valores. O desenvolvimento de software é muito mais efetivo quando permite o ganho de dinheiro mais cedo e o gasto do dinheiro mais tarde.

Benefícios mútuos

Todas as atividades realizadas pelo time devem trazer benefícios a todos, tanto no presente quanto no futuro. Desenvolver testes automatizados hoje, que ajudam a projetar e implementar recursos, também ajudará os novos colaboradores do time a entenderem um sistema.

Semelhanças

Copiar uma solução de um problema semelhante pode ajudar, mesmo em uma escala diferente. Entretanto, devemos ter atenção ao decidirmos realizar essa ação, pois copiar uma solução de um problema semelhante poderá não funcionar se o contexto for diferente.

Melhorias

“O ótimo é o inimigo do bom”. Não há perfeição em desenvolvimento de software, projetos, processos, implementações etc. Tudo depende do contexto e de uma situação. Devemos ser pragmáticos, fazendo o melhor hoje, e estar preparados para o amanhã. Ser um perfeccionista pode trazer uma perda para a organização e um trabalho inútil para o time.

Diversidade

É necessário buscar sempre a diversidade ao formar um time. Ter diferentes conjuntos de habilidades, incluindo habilidades técnicas e soft skills (comportamentais), experiências em diferentes áreas e contextos, possibilita perspectivas distintas e ajuda a encontrar a melhor abordagem para cada situação. Conflito é uma coisa comum em times com opiniões diversas, mas em um time que possua o valor “respeito”, o processo realmente acaba em consentimento.

Reflexão

Um bom time aprende tanto com os triunfos quanto com seus fracassos. Analisar e aprender com os erros é um diferencial entre um bom time e um mediano. Retrospectiva deve fazer parte da rotina do time, não apenas em final de **sprints**.

Sprints

Períodos necessários para a entrega de parte de um projeto que utiliza o framework de desenvolvimento ágil scrum para gerenciamento do projeto.

Fluxo

O time deve buscar um fluxo contínuo de entrega de valores, com um ritmo que seja sustentável ao longo do tempo. O adiamento da implantação em produção só leva à ampliação do risco de defeitos e à perda do **time to market**.

Time to market

Em tradução literal, significa “tempo para comercialização”. Refere-se ao período desde a concepção de um produto até sua liberação para ser utilizado pelo mercado.

Oportunidade

Problemas são parte do desenvolvimento de software, não importa o quanto um time tente evitá-los. Eles aparecerão cedo ou tarde. Importante é perceber que os problemas são oportunidades para melhorar e aprender.

Redundância

No XP, a solução é aplicar muitas medidas de qualidade – programação de pares, testes, integração contínua –, cada uma como linha de defesa, que, juntas, formam uma muralha virtualmente impenetrável.

Falha

Não devemos desperdiçar tempo com discussões, e sim ter coragem de experimentar alternativas, quando o caminho não for claro. Depois, escolha a melhor e coloque-a em produção. Se a implementação de todas as alternativas for demorar muito, escolha um subconjunto delas ou faça protótipos.

Qualidade

A qualidade não é negociável no XP. Podemos limitar o escopo, mas não cortar a qualidade. Manter a qualidade do código ajuda a estabelecer um fluxo contínuo de entregas. Quando estamos implementando XP, devemos fazer isso em pequenos passos.

Não basta tentar reduzir seu período de entrega para a metade e ver o que acontece. Implementar XP com grandes pedaços de requisitos não vai funcionar. Devemos buscar uma granularidade mínima da entrega de valor.

Baby steps

As organizações podem adicionar, de forma incremental, as práticas do XP.

Responsabilidade

A responsabilidade não pode ser imposta, mas aceita. Criar um ambiente em que as pessoas possam colaborar, estimando, projetando, implementando e testando tarefas, mostra que elas têm coragem de ser responsáveis pela entrega.

Práticas, cadências e papéis do XP

Com o passar do tempo, as práticas do XP evoluíram, sempre olhando para a melhoria contínua, com foco na entrega de valor.

Confira na imagem uma série de práticas que, no XP, possibilitam a integração e a entrega contínua.

Práticas do XP.

Vamos agora conhecer melhor essas práticas:

Time Coeso (Whole Team)

A equipe de desenvolvimento é composta pelo cliente e pelo time de desenvolvimento.

Jogo de Planejamento (Planning Game)

Desenvolvedores e cliente reúnem-se para priorizar funcionalidades. O cliente identifica prioridades e os desenvolvedores as estimam. Ao final de cada iteração, o cliente recebe novas funcionalidades, completamente testadas e prontas para serem disponibilizadas à produção. Histórias de usuários devem estar prontas antes de cada planning. O desenvolvimento ocorre em iterações semanais.

Pequenas Versões (Small Releases)

A liberação de pequenas versões funcionais do produto auxilia muito no processo de aceitação por parte do cliente.

Testes de Aceitação (Customer Tests)

São testes desenvolvidos pelo cliente em conjunto com analistas e engenheiros de teste, para aceitar determinado requisito do sistema.

Padrões de Codificação (Coding Standard)

São acordos feitos pelos membros dos times, a fim de estabelecer regras e padrões de códigos para programar, nos quais todos devem estar comprometidos com o que foi definido, de modo a ter uma boa fluidez nas entregas.

Ritmo Sustentável (Sustainable Pace)

Trabalhar com foco na qualidade, buscando ter ritmo de trabalho saudável. Busca-se um ambiente que promova um trabalho motivado sempre.

Metáfora (Metaphor)

É preciso traduzir as palavras do cliente para o significado que ele espera que seja entregue.

Integração Contínua (Continuous Integration)

Sempre que produzir uma nova funcionalidade, ela deve ser integrada no repositório do projeto para ser validada pelos membros da equipe. Nesse repositório, ocorrem testes e coleta de feedback sobre a qualidade técnica da solução.

Posse Coletiva (Collective Ownership)

O código fonte não tem dono, e ninguém precisa pedir permissão para modificá-lo. Toda a equipe conhece todas as partes do sistema.

Desenvolvimento Orientado a Testes (Test Driven Development)

Primeiro, são criados os testes unitários (unit tests). Depois, o código é desenvolvido para que os testes funcionem. Os testes se tornam parte da definição e documentação dos requisitos e isso também ocorre com os testes funcionais.

Refatoração (Refactoring)

É um processo que permite a melhoria contínua da programação – base para um design evolutivo –, mitigando a introdução de erros e mantendo a compatibilidade com o código já existente. Um ponto de atenção é que a refatoração segura somente ocorrerá se a solução tiver uma boa cobertura de testes automatizados.

Projeto Simples (Simple Design)

Simplicidade é um dos princípios da XP. Projeto simples significa desenvolver o mínimo necessário, de acordo com a expectativa trabalhada junto ao cliente. Em outras palavras, caso o cliente tenha pedido que, na primeira versão, apenas o usuário “teste” possa entrar no sistema, ainda que sem senha, e, assim, ter acesso a todo o sistema, o time fará o código exato para que essa funcionalidade seja implementada.

Programação em Pares (Pair Programming)

É a programação em par/dupla, em que dois desenvolvedores atuam juntos na solução: um fica à frente, fazendo a codificação, e o outro acompanha, ajudando o primeiro. Dessa forma, o programa sempre é revisado por duas pessoas, evitando e diminuindo a possibilidade de defeitos.

Atenção

Além das práticas que impulsionam a integração, o XP também sugere que os times realizem reuniões em pé (stand-up meeting), para que não se perca o foco nos assuntos relevantes. São reuniões rápidas, que abordam tarefas realizadas e a realizar pela equipe, além de levantamento de impedimentos que deverão ser tratados em um fórum à parte.

Definição de papéis

No XP, assim como no Kanban, não existe uma rigidez quanto à definição de papéis, mas é importante reconhecermos que os papéis podem ser úteis para os times iniciantes, até que passem a atrapalhar a colaboração.

Vejamos os papéis comumente associados ao XP:

Cliente

Deve ser alguém que conhece o negócio, suas regras e tenha a capacidade de definir prioridades. Quando isso não for possível com membros do time, essa função poderá ser realizada por um representante do próprio cliente.

Desenvolvedores/Time

Em uma equipe de XP, os desenvolvedores estimam o esforço necessário para a entrega de cada funcionalidade, tarefas e histórias, incluindo a escrita de testes automatizados.

Coach

Não é primordial ter um coach, e muitos times alcançaram sucesso sem um. No entanto, ter alguém que conhece bem as boas práticas e os valores do XP para sinalizar ao time quando algo não está de acordo ajuda muito na melhoria contínua. Esse papel, geralmente, é desempenhado por algum dos desenvolvedores do time. Quando todos são experientes em XP, pode ser feito revezamento nesse papel.

Tracker

O tracker coleta as métricas de projeto e conversa com cada membro da equipe para identificar bloqueios no caminho e descobrir soluções. As métricas podem ser rastreadas no próprio código ou em conversas com os desenvolvedores.

O ideal é que um tracker trabalhe em média com três métricas, para fazer um bom acompanhamento e não onerar o fluxo de trabalho. Se, com o passar do tempo, alguma métrica se mantiver constante e não trouxer informações relevantes, poderá ser substituída por outra.

Desenvolvimento Orientado a Testes ou Test Driven Development (TDD)

Vamos explorar um pouco mais o conceito de TDD, que surgiu como prática fundamental no XP e tem tido adesão em grande escala nas organizações.

> Também chamado de Test First Development, é uma abordagem de desenvolvimento de software, na qual os casos de teste são desenvolvidos para especificar e validar o que o código fará. Baseia-se em um ciclo curto de repetições.

O TDD começa com o projeto e o desenvolvimento de testes para cada funcionalidade do produto. A prática instrui os desenvolvedores a escreverem o novo código somente se um teste automatizado falhar, evitando a duplicação do código. Em outros termos, significa escrever e corrigir os testes falhados antes de escrever um novo código (antes do desenvolvimento).

Existem dois níveis de adoção de TDD:

Aceitação TDD (ATDD)

Também conhecido como Behavioral Driven Development (BDD). Com o ATDD, escrevemos um único teste de aceitação. Esse teste deverá cumprir a exigência da especificação ou satisfazer o comportamento do produto. Depois disso, são escritos apenas os códigos de produção/funcionalidade suficientes para cumprirem o teste de aceitação, que foca no comportamento geral do sistema.

Desenvolvedor TDD

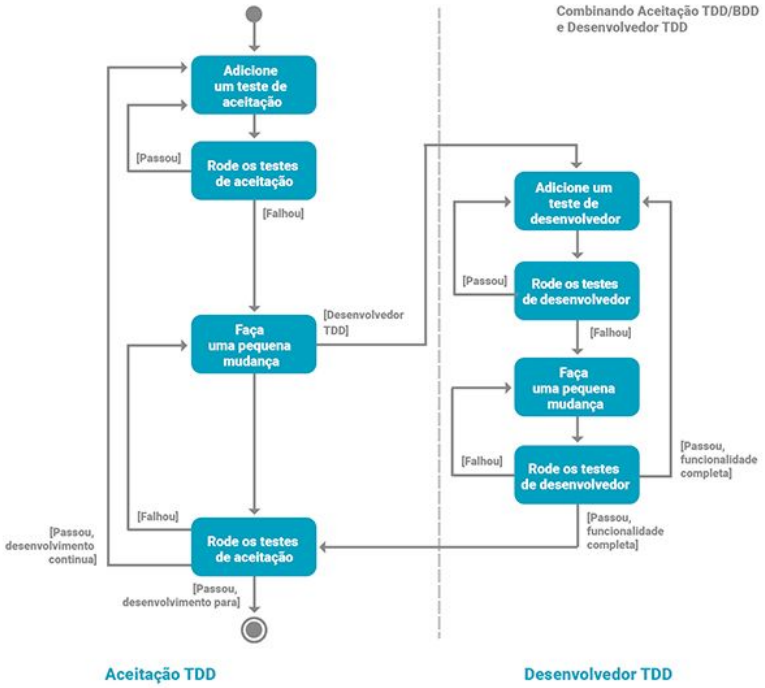
Com o Desenvolvedor TDD, escrevemos um único teste de desenvolvedor (teste unitário) e, depois, apenas o código de produção suficiente para cumprir esse teste. O teste unitário foca em cada pequena funcionalidade do produto.

O objetivo principal da ATDD e do TDD é especificar os requisitos de forma detalhada e executáveis para uma solução com a mentalidade de Just In Time (JIT), o que significa alocar esforços apenas naqueles requisitos necessários ao produto naquele momento, aumentando a eficiência.

O fluxo típico de construção no ATDD tem quatro etapas:

1. Adicionar um teste.
2. Executar todos os testes e ver se algum novo teste falha.
3. Escrever algum código/altere um código para que seja correto, ou seja, refatorar.
4. Repetir o processo.

Como podemos ver no esquema a seguir, essas etapas possuem conexão direta com o fluxo de trabalho de um desenvolvedor TDD – prática que tem como pilar o princípio XP da qualidade.



Fluxo típico de ATDD com TDD.

De acordo com Prikladnicki, Willi e Milani (2014), são alguns benefícios da prática do TDD:

Qualidade embutida

O programador se torna responsável pela qualidade de seu código, diferentemente de outras abordagens, em que é responsabilidade de outra pessoa ou área. O fato de o código estar lado a lado com seus testes impulsiona a garantia da qualidade, como se fosse uma parte embutida do próprio sistema.

Como existem testes associados a cada pedaço de código, é possível ter a certeza de que todo o código teve um teste executado, em que os erros, geralmente, são encontrados no início do desenvolvimento.

Design evolutivo

No XP, não se espera que o design seja conhecido de forma antecipada. Mesmo que não se tenha decidido por uma solução tão boa no começo do desenvolvimento do projeto, quando se usa TDD, temos um aprendizado real contínuo, obtido durante o desenvolvimento em cada uma das iterações. Isso nos permite rever estruturas e refatorar os códigos.

Adaptabilidade

O processo iterativo e incremental possibilita que adaptemos a solução a cada nova iteração. Uma vez que as funcionalidades são testadas logo após sua criação, passamos a ter um feedback quase instantâneo de seu funcionamento, o que permite correções ou até mesmo mudanças mais cedo.

Fluxo

O ciclo de desenvolvimento de software torna-se mais fluido com o XP, por ser uma abordagem sistemática, que trata da atividade de programação em seus detalhes.

Impulsiona uma otimização do fluxo por meio da eliminação de desperdícios, como, por exemplo, os próprios testes servindo como documentação do software, uma vez que, ao criá-los, é possível entender facilmente como o código funciona.

Por produzir um código flexível e limpo, gasta-se menos tempo com a correção de bugs e a implementação de novas funcionalidades, levando a uma maior produtividade do time.

À primeira vista, pode ser estranho escrever testes sem ao menos ter a respectiva funcionalidade criada. Mas a vantagem desse processo é justamente facilitar a criação do código e evitar a inserção de erros e defeitos no sistema. Além disso, o próprio teste já pode ser utilizado como um tipo de documentação do código desenvolvido, ajudando na otimização do fluxo de desenvolvimento.

eXtreme Programming

Assista agora a um vídeo em que se apresenta um profundamento dos conceitos do XP, entendendo sua origem e em que contextos ele se adequa melhor.

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Vem que eu te explico!

Os vídeos a seguir abordam os assuntos mais relevantes do conteúdo que você acabou de estudar.

Quando usar XP?

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

A importância do TDD

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Case com XP

Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Verificando o aprendizado

Questão 1

Seu time percebe que algo não está de acordo com as práticas e valores do XP, impactando o processo de melhoria contínua. Esse é um tipo de problema que pode ser liderado (facilitado) por qual papel do XP?

A

Tracker

B

Desenvolvedor

C

Cliente

D

Coach

E

Usuário

A alternativa D está correta.

Em times maduros, os próprios desenvolvedores podem desempenhar as funções desse papel, inclusive intercalando entre os membros do time.

Questão 2

Qual princípio do XP promove a melhoria contínua de um software, mantendo a compatibilidade com o software preexistente?

A

Pequenos desenvolvimentos

B

Programação em pares

C

Refatoração

D

Simplicidade

E

Reprogramação

A alternativa C está correta.

A refatoração é um processo em que os desenvolvedores revisam e ajustam códigos, quando necessário, para manter um código que seja o mais enxuto e confiável possível.

Considerações finais

Como vimos, tanto o Kanban quanto o XP têm foco em otimização de fluxo de trabalho, em busca da melhoria contínua e entrega de valor.

O que norteará a adoção de um desses métodos é o contexto em que o produto é desenvolvido, uma vez que a natureza do produto, a maturidade do time e o mindset da organização podem ser habilitadores ou ofensores na adoção de métodos ágeis.

Esses dois métodos podem ser adotados ao mesmo tempo por um time ágil, por meio de tailoring e experiências curtas, com feedbacks rápidos sobre o que funciona e o que não funciona no time.

Tudo isso nos leva à reflexão de que quando a forma de pensar tem sua base muito forte nos princípios ágeis, o time encontra a melhor maneira de realizar o valor.

Podcast

Ouça agora um podcast em que são reforçados os principais conceitos tanto do KANBAN e do XP.

Conteúdo interativo

Acesse a versão digital para ouvir o áudio.

Explore +

Para saber como as práticas de XP atuais foram originalmente descritas, pesquise e acesse o material **What is Extreme Programming**? de Ron Jeffries (O que é programação extrema?).

Referências

ANDERSON, D. J.; CARMICHAEL, A. **Essential Kanban condensed**. Seattle: Lean Kanban University, 2016.

CARDOSO, V. (org.). **TimeDev**: muito mais do que código. Joinville: Clube de Autores, 2021.

HAMMARBERG, M.; SUNDÉN, J. **Kanban in action**. New York: Manning Publications, 2014.

MUNIZ, A. *et al.* **Jornada Kanban na prática**: unindo teoria e prática para acelerar o aprendizado para quem está iniciando. Rio de Janeiro: Brasport, 2021.

PRIKLADNICKI, R.; WILLI, R.; MILANI, F. (org.). **Métodos ágeis para desenvolvimento de software**. Porto Alegre: Bookman, 2014.

WELLS, D. **Extreme Programming**: a gentle introduction. [S. l.]: XP, 2013.