FIAP ON

APRENDENDO TECNOLOGIA COM DESENVOLVIMENTO

MAPEAMENTO DE SOFTWARE

HENRIQUE RUIZ POYATOS NETO

06

LISTA DE FIGURAS

SUMÁRIO

1 MAPEAMENTO DE SOFTWARE

1.1 Introdução

Técnicas nunca são demais. Após o detalhamento do backlog e a definição da estratégia do projeto no Planejamento de *Releases*, temos que definir os *Sprints*, detalhar as atividades e conduzir as cerimônias para acompanhamento do projeto. Vamos falar sobre essas cerimônias neste capítulo.

1.2 Planejamento do *Sprint*

Dentro do ciclo do *framework Scrum*, com o *Product Backlog* já definido, inicia-se a reunião de planejamento dividida em duas etapas: a revisão da priorização do *Product Backlog* e o planejamento do *Sprint* (Figura "*Framework* Scrum").

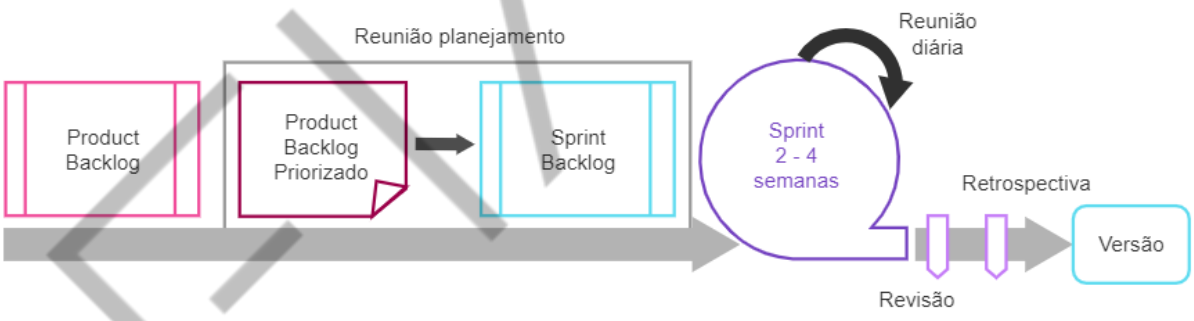

Figura 1 – *Framework* Scrum
Fonte: Elaborado pelo autor, adaptado por FIAP (2017)

Vimos, no capítulo anterior, a primeira etapa deste ciclo. Agora, veremos em detalhes como fazer o planejamento do *Sprint* que será executado.

Mas, antes disso, vamos entender o que é um *Sprint*.

Um *Sprint* é um ciclo determinado por um *timebox* representando uma iteração para a construção do produto definido. Dentro do *Sprint,* você pode usar o método que quiser para a produção, desde que o padrão de qualidade seja observado, e o prazo, respeitado.

Lembre-se de que as histórias que compõem um *Sprint* são definidas pelo *Product Owner*, e suas atividades e compromisso com o prazo são de responsabilidade do time.

No *Scrum,* cada *Sprint* é um *timebox* que dura de uma a quatro semanas e sempre deve entregar valor ao cliente. A duração de cada *Sprint* é definida para cada projeto em comum acordo entre todos os envolvidos e pode variar conforme as necessidades de cada projeto. Porém, não é comum nem recomendável que, dentro do mesmo projeto, se mude a duração do *Sprint.* Isso porque perde-se consistência, há dificuldade de medir o quanto se pode produzir e entregar em *Sprints* de durações variáveis e perde-se previsibilidade.

Lembre-se de que o conceito de *timebox* é que a data final é definitiva, não pode ser prorrogada nem antecipada sob hipótese alguma.

Um *Sprint* deve ter, no máximo, um mês de duração, porém, fora esta limitação, não há tamanho recomendado. Isso vai depender da quantidade de histórias, do time e do que espera o *Product Owner* como meta do *Sprint*.

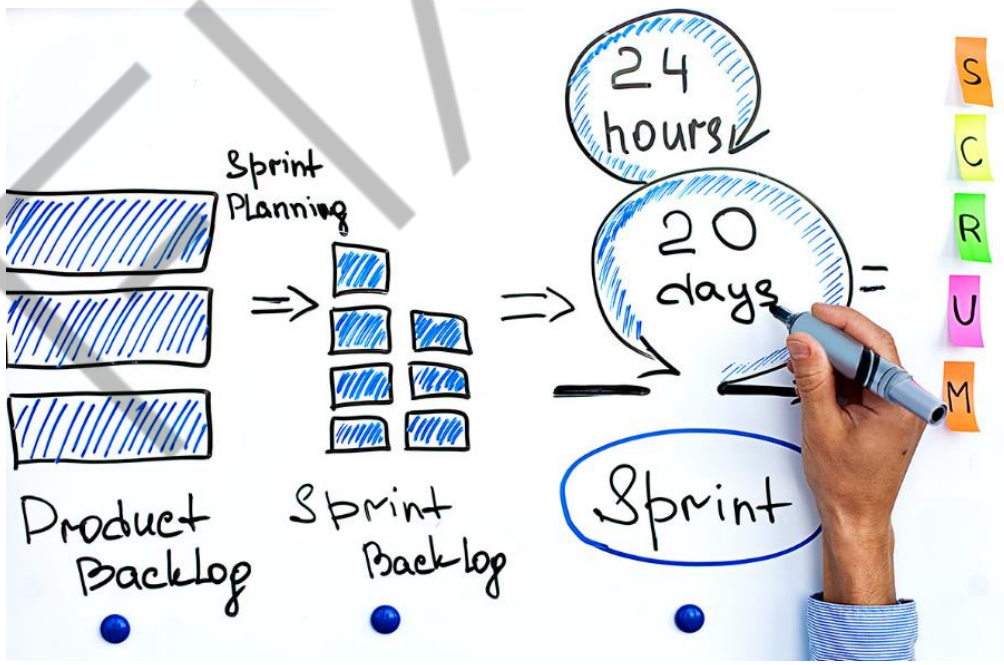

Figura 2 – *Sprint*
Fonte: Banco de Imagens Shutterstock (2017)

Embora não existam padrões estabelecidos para o *Sprint*, podemos ter como orientação inicial para definir o tamanho:

- Time começando a executar projetos com *Scrum* deve usar *Sprint* de duas semanas para que as adaptações sejam mais rápidas.
- Primeiros *Sprints* de um projeto novo devem ser mais curtos, de até duas semanas, para minimizar os riscos e adaptar o time ao ambiente do projeto.
- *Product Backlog* tem muitas mudanças de priorização pelo *Product Owner,* que podem indicar indecisões ou incertezas. Nesse caso, recomenda-se usar *Sprints* mais curtos para minimizar riscos.
- Usar *Sprints* de três ou quatro semanas para reduzir o ritmo de entregas e permitir mais experimentação ou prototipagem.

Além dos requisitos de negócio, um *Sprint* pode ter atividades técnicas que devem ser colocadas nas necessidades de negócio para atender às expectativas e preocupações técnicas do time. Como, por exemplo: requisitos de desempenho, requisitos de usabilidade, requisitos de segurança, arquitetura, entre outros.

Antes de iniciar o *Sprint*, deve ser definida uma META específica que esteja alinhada à meta do *release* e atenda às expectativas do cliente. Todo o time, incluindo desenvolvedores e *Product Owner*, é responsável por definir essa meta com base no *Product Backlog* priorizado e no valor de negócio a ser desenvolvido durante o *Sprint*. A meta é o guia do time durante a execução do trabalho e deve ser atendida dentro do *timebox* que foi definido.

O *Scrum Master* deve **garantir** que nenhuma mudança possa afetar **a meta do *Sprint***.

O *Sprint Planning* é a segunda parte da reunião de planejamento e é realizado antes da execução de cada *Sprint*, quando o time e o *Product Owner* se reúnem para detalhar o que será feito durante o *timebox* de construção.

O *Sprint Planning* também é uma reunião do tipo *timebox*, definido no Scrum Guide® e devendo ter seu tempo de duração fixo (Quadro "*Timebox dos Sprint Planning"*).

Tamanho Sprint	Timebox da reunião
4 semanas	8 horas
3 semanas	6 horas
2 semanas	4 Horas

Quadro 1 – *Timebox dos Sprint Planning*
Fonte: Elaborado pelo autor (2017)

A duração desses eventos deve ser respeitada incondicionalmente por todos os envolvidos no projeto e seu cumprimento deve ser garantido pelo *Scrum Master*.

Embora as práticas de mercado sejam de *Sprints* com tamanhos entre duas e quatro semanas, caso necessário, você pode fazer um *Sprint* de uma semana para a execução de ciclos menores de atividades. Mas não pode ser maior do que quatro semanas.

Na reunião de planejamento do *Sprint,* deve ser definido o que será entregue e, em seguida, como fazer para entregar o trabalho.

Vamos analisar em etapas.

Figura 3 – Planejamento *Sprint*
Fonte: Banco de Imagens Shutterstock (2017)

1.3 Etapas do planejamento do *Sprint*

A **primeira atividade da reunião é definir a meta do *Sprint***, ou seja, uma descrição sucinta do que deve ser entregue, de valor, para o cliente. O *Product Owner* é responsável por definir o escopo fundamental do produto a ser desenvolvido para atender às expectativas dos usuários finais e do negócio. Essa meta deve ser algo tangível que possa ser avaliado pelo time.

Além de tangível, a meta deve ser realista e aceita por todos os envolvidos no planejamento. Em hipótese alguma pode ser imposta. O ideal é que todo o time discuta a meta do Sprint, baseando-se no Product Backlog priorizado, e decida o que é possível realizar dentro do *timebox* do Sprint que tenha valor para o cliente final, chegando a um consenso natural.

Figura 4 – Reunião
Fonte: Banco de Imagens Shutterstock (2017)

Com base na meta definida, o time escolhe os itens mais importantes do *Product Backlog*, que devem ser atualizados, detalhados e quebrados em histórias menores para que possam ser concluídas dentro do *Sprint*. Apesar de todas as decisões serem tomadas em conjunto por todos do time, o *Product Owner* tem uma participação especialmente relevante, pois é ele quem pode indicar o que tem maior valor para o cliente final.

Lembre-se de que:

- O *Product Backlog* pode sofrer repriorizações ou inclusão de novas histórias.
- O tamanho das histórias pode sofrer alterações à medida que os desenvolvedores conhecem mais detalhes sobre elas e planejam sua implementação.
- O time *Scrum* define sua capacidade para *Sprint*, ou seja, com o desenvolvimento e a entrega de quantos itens do *Product Backlog* o time irá se comprometer a.
- O *Product Owner* deve participar ativamente dessa fase com o time.

Após a definição conjunta da meta do *Sprint*, o time avalia e entende cada item definido do *Product Backlog* para verificar a possibilidade de entrega dentro do prazo definido. Essa decisão deve ser baseada na capacidade do time (velocidade) e no conhecimento sobre as atividades a serem desenvolvidas.

IMPORTANTE:

É essencial que o time realmente acredite que a meta possa ser cumprida e o comprometimento seja real e irrestrito com a execução das atividades.

A equipe deve decompor cada item do *Product Backlog* em atividades e fazer a estimativa de quanto tempo (em horas) ou esforço (em *story points*) vai precisar para fazer tudo o que é necessário para concluir cada atividade do *Sprint*. O resultado desse trabalho é chamado de ***Sprint Backlog*** **(**Figura “Fluxo do Planejamento do *Sprint”*).

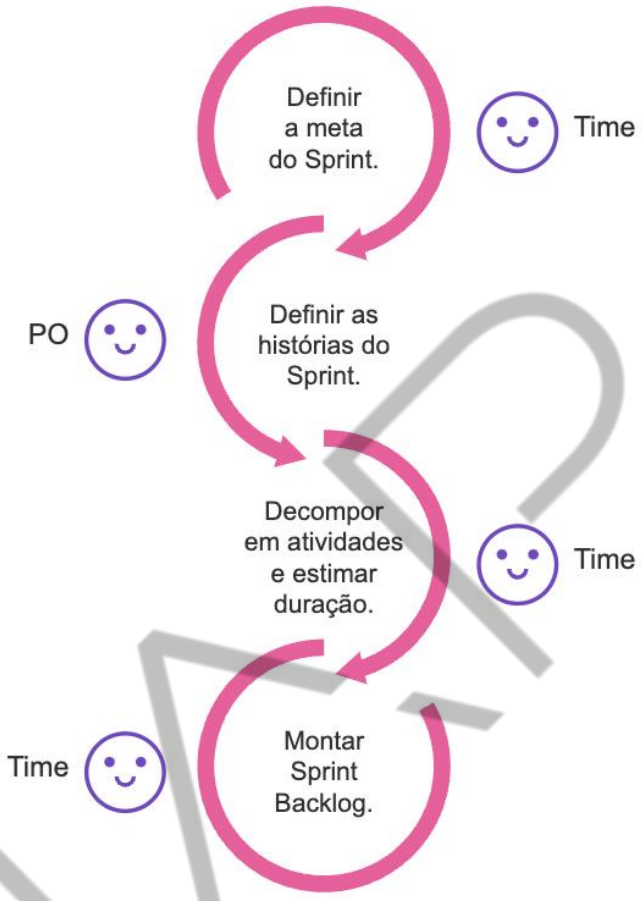

Figura 5 – Fluxo do Planejamento do *Sprint*
Fonte: Elaborado pelo autor, adaptado por FIAP (2017)

1.4 *Sprint Backlog*

O *Sprint Backlog* contém todas as atividades necessárias para completar as histórias selecionadas do *Product Backlog* e atingir a meta do Sprint. Todas as atividades devem ser estimadas em horas ou em pontos, a fim de acompanhar a evolução e o trabalho restante das atividades.

Normalmente, o *Sprint Backlog* utiliza o quadro *Kanban*, no qual as atividades são colocadas em colunas contendo a descrição da atividade e o esforço previsto pelo time (Figura "*Product Backlog* no quadro *Kanban"*).

O quadro *Kanban* é organizado em colunas que representam a ordem em que as tarefas são executadas, sendo as colunas mais comuns: A Fazer (To Do); Fazendo (Doing) e Feito (Done). Porém, novas colunas podem ser inseridas pelo time, de acordo com as características do projeto ou do fluxo específico de cada organização.

Um *Kanban* é um sistema criado na Toyota, geralmente representado por um quadro, mas também organizado por meio de software ou até mesmo uma folha de papel, em que cartões que representam o trabalho seguem um fluxo preestabelecido de estágios.

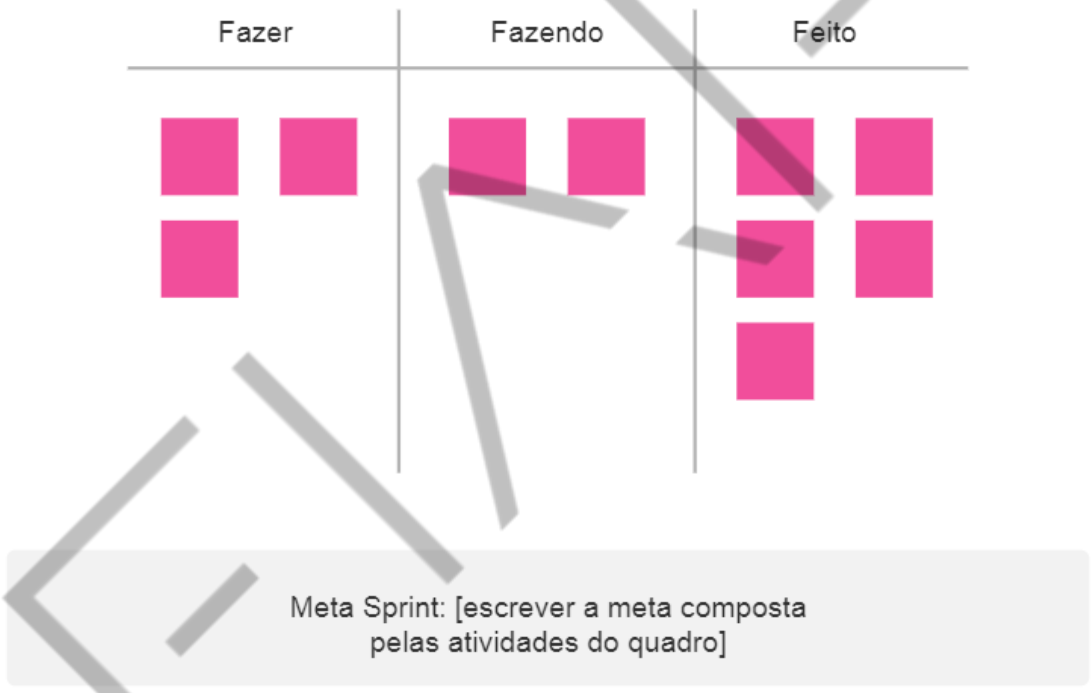

Figura 6 – *Product Backlog* no quadro *Kanban*
Fonte: Adaptado de Cultura Ágil (2018)

No quadro *Kanban,* não existe o conceito de dependências entre atividades como em um cronograma. No entanto, a priorização é feita colocando as atividades de cima para baixo, inserindo cores nos cartões ou identificando, de forma que fique claro, para o time, o que deve ser feito.

Também é importante identificar, de alguma forma, sempre que uma atividade sofre mudança, uma nova é incluída ou quando é preciso retornar um passo do fluxo

de trabalho. O objetivo do *Kanban* é manter o time focado e melhorar a comunicação entre os envolvidos.

Novas atividades podem ser adicionadas ao *Sprint Backlog* durante o *Sprint,* desde que não afetem o *timebox* definido e não ameacem a entrega da meta do *Sprint,* mas novas histórias não.

1.4.1 Conceito de *Ready*

O objetivo do conceito de *ready* (DoR, em inglês *Definition of Ready*) é verificar se a história está clara e se todos os membros do time têm uma compreensão do que precisa ser feito, ou seja, se a história tem informações suficientes para começar a ser construída. Cada organização e/ou cada time define quais são os requisitos necessários para considerar que a história está OK.

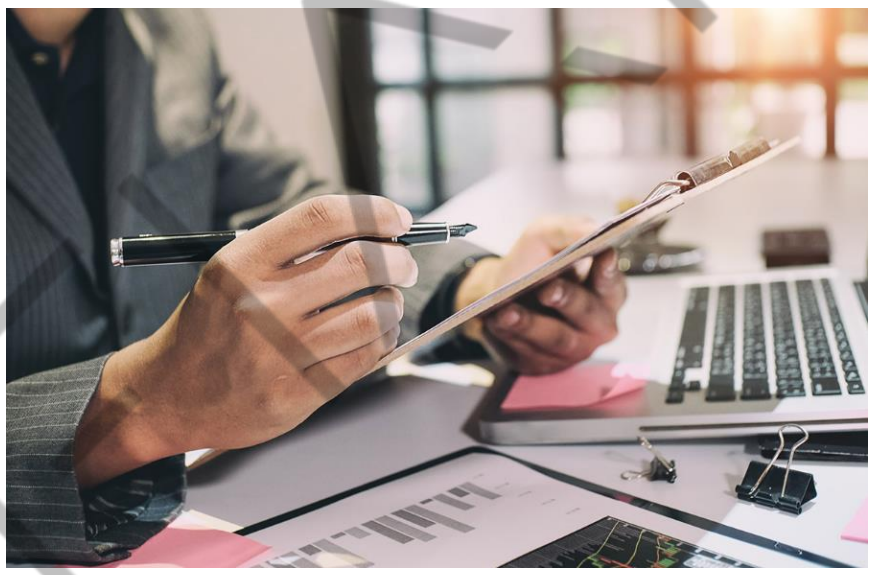

Figura 7 – *Checklist*
Fonte: Banco de Imagens Shutterstock (2017)

Normalmente, o DoR é definido por meio de um *checklist* que será aplicado sobre cada história para fazer essa verificação. Abaixo, temos uma lista de verificações que podem ser feitas em um DoR, com o time e o *Product Owner*:

- A história tem um tamanho adequado para ser detalhada?
- A história pode ser construída dentro de um *Sprint*?
- As dependências entre as histórias estão identificadas?

- Existe um esboço do produto?
- Os critérios de aceitação estão descritos de forma clara e testável?
- Os requisitos não funcionais estão identificados?
- A história não tem impedimentos?

De posse desse *checklist*, o time pode verificar as condições de execução de uma história para ajudar no momento do planejamento do *Sprint*, evitando que o *timebox* dessa reunião seja comprometido por uma série de problemas que podem ser previamente resolvidos.

Da mesma forma que todos os artefatos do *Scrum*, o DoR pode ser revisado e atualizado para melhoria da qualidade e das estimativas de tempo. Essa revisão é feita na *Retrospectiva*.

1.4.2 Reunião de Refinamento

Seu objetivo é melhorar a definição das histórias do *Product Backlog* para que esteja em um nível de maturidade melhor antes da próxima reunião de planejamento do *Sprint*.

A Reunião de Refinamento é realizada de dois a três dias antes do final do *Sprint* corrente e antes do início da próxima reunião de planejamento do *Sprint*, liderada pelo PO e com a participação de 10% a 50% do time, recomendado, respectivamente, pelos autores Mike Cohn e Ken Schwaber.

A ideia central é revisar os itens do *Product Backlog* para esclarecer todos os pontos e dúvidas do time e evitar que o PO tenha que buscar informações de última hora.

A duração de uma Reunião de Refinamento é um *timebox* de 1 a 2 horas para cada *Sprint*.

As Reuniões de Refinamento não são parte obrigatória do *framework* Scrum, porém são prática comum de mercado pelos benefícios que oferecem: mais alinhamento do time de desenvolvimento com o *Product Owner*, mais conhecimento

sobre o *Product Backlog* e como implementá-lo além de economia de tempo no planejamento do *Sprint*.

1.4.3 Conceito de PRONTO

O conceito de *pronto*, ou, no inglês, *Definition of Done* (DoD), é estabelecido pela organização e deve ser atendido minimamente por todos os times ou, caso não exista, precisa ser criado pelo time *Scrum*. Seu propósito é dar maior transparência ao processo de formalização de que um *Sprint* está concluído, além de garantir a qualidade dos itens desenvolvidos. Também tem um formato de *checklist* e deve ser aplicado antes de liberar o incremento do *Sprint* para a revisão do PO e *stakeholders* na reunião de Revisão.

O DoD deve ter o comprometimento de todo o time *Scrum* e ter suas características publicadas a todos os envolvidos.

Figura 8 – Processos concluídos
Fonte: Banco de Imagens Shutterstock (2017)

Um exemplo de *checklist* de DoD é definido abaixo:

- História definida, produto desenhado e validado pelo PO.
- História construída.
- Realizados os testes unitários sem erros.

Mapeamento de softwareTécnicas nunca são demais...

- Funcionalidade inspecionada.
- Testes de aceitação 100% OK.

2 SABE QUANDO O CLIENTE PERGUNTA SE ESTÁ PRONTO? COMO VOCÊ SABE O QUE É PRONTO?

2.1 DoD

A diferença entre o DoR e o DoD é que o DoR é aplicado antes de o *Sprint* iniciar e o DoD é aplicado antes de o *Sprint* terminar.

2.1.2 Testes de aceitação

A elaboração dos testes de aceitação também não é uma atividade do *framework Scrum*. Porém, é uma técnica que auxilia o time na validação das funcionalidades construídas, aumentando a qualidade do produto.

Esses testes de aceitação devem ser escritos pelo *Product Owner*, com auxílio do *Scrum Master,* antes da construção do produto para permitir a avaliação preliminar do produto. A elaboração e a execução dos testes de aceitação devem fazer parte do processo de elaboração do produto.

Vamos supor que tenhamos a seguinte história:

Como um cliente cadastrado no site, posso pagar meu pedido com cartão de crédito.

Para esse caso, elaboramos vários testes que garantem o correto funcionamento da história:

- Teste de pagamento com cartão vencido.

- Teste de pagamento com cartão sem limite para valor da compra.

- Teste de pagamento com cartão expirado.

- Teste de pagamento com as bandeiras Visa e Master.

- Teste de pagamento com mais de um cartão.

- Teste de pagamento com sucesso.

- Teste de pagamento com sucesso com mais de um cartão.

Essa especificação preliminar feita pelo PO permite que o time construa e verifique o produto conforme os critérios de aceite estabelecidos, aumentando a qualidade do produto.

Podemos utilizar também o BDD (*Behavior Driven Development*) para auxiliar na elaboração dos testes, seguindo uma estrutura de linguagem natural que facilita o entendimento de todos.

Figura 9 – PO
Fonte: Banco de Imagens Shutterstock (2017)

DICA: O BDD é uma técnica ágil para testes e consiste em ter as visões técnicas e de negócio juntas, permitindo identificar os comportamentos esperados do produto em linguagem o mais natural possível. Veja mais em:

https://www.agilealliance.org/glossary/bdd/

Mapeamento de softwareTécnicas nunca são demais...

Como exemplo, para cada história do *Product Backlog*, escrevemos na seguinte estrutura:

Cenário: <descrição do teste>

Dado <um estado conhecido>

Quando <um determinado evento ocorre>

Então <isso deve ocorrer>

Exemplo de uma descrição completa para uma história:

História:

Como um funcionário,

Eu gostaria de consultar um cliente cadastrado com sucesso por CPF,

De modo que eu possa garantir que a consulta esteja correta.

Caso de teste:

Cenário: consultar cliente com sucesso por CPF

Dado: ao selecionar a opção buscar

Quando: digitado o CPF 111.111.111-11

Então: deve listar nome, CPF e data de nascimento do cliente.

O uso do BDD facilita o entendimento dos usuários e a descrição mais completa dos critérios de testes que devam ser executados.

2.2 Execução do *Sprint*

Durante a execução do *Sprint,* o time deve dar foco total para atingir a meta do *Sprint.* Nada pode atrapalhar o desempenho do time na execução do *Sprint*, momento no qual o *Scrum Master* atua fortemente como facilitador. Ele faz a interface entre o time, o *Product Owner*, o cliente e, caso necessário, outros times de desenvolvimento, além de remover impedimentos e problemas para garantir a execução dos trabalhos.

Esses impedimentos, muitas vezes, podem interromper uma atividade ou atrapalhar a sua evolução. Nesses casos, a atividade deve ser colocada em situação de impedimento e o *Scrum Master* deve atuar rapidamente.

Figura 10 – Execução do *Sprint*
Fonte: Banco de Imagens Shutterstock (2017)

O *Sprint Backlog* é um artefato que deve ser atualizado diariamente. Se um membro da equipe começa a trabalhar em uma atividade, seu nome está gravado dentro do *Sprint Backlog*. À medida que o trabalho vai evoluindo, os cartões vão mudando de coluna, e sempre que um novo trabalho é identificado, um novo cartão é criado e o quadro *Kanban* é atualizado.

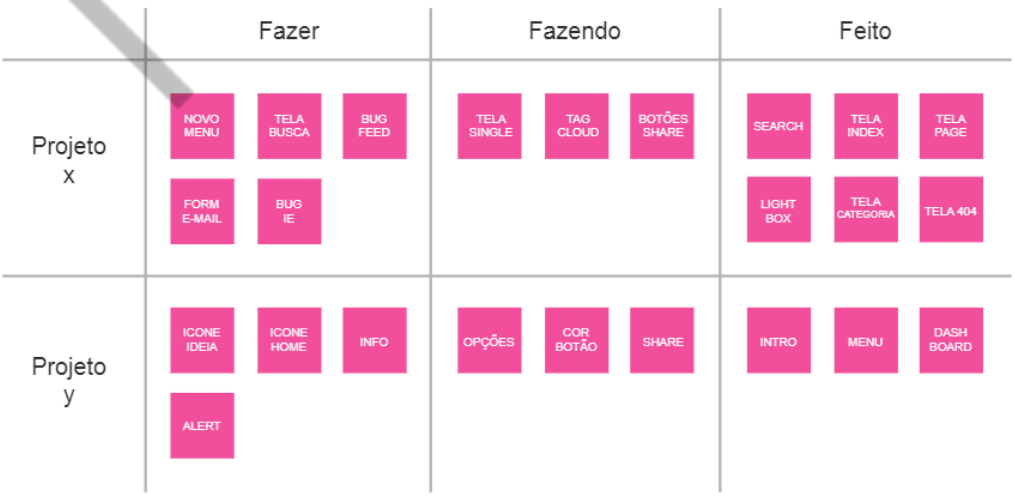

Figura 11 – Quadro Kanban durante a execução de um *Sprint*
Fonte: Cultura Ágil, adaptado por FIAP (2017)

No final do dia, todas as atividades a concluir devem ter seus esforços atualizados, permitindo que o time avalie o quanto do trabalho ainda falta ser realizado e se está adiantado ou atrasado, gerando o *Sprint Burndown*, que veremos adiante.

Na execução do *Sprint*, as mudanças são bem-vindas, mas devem ser inseridas no *Product Backlog* para posterior planejamento. Isso é feito para evitar que o *timebox* do *Sprint* tenha seu prazo comprometido por novas demandas ou que a meta do Sprint seja ameaçada, além de não desmotivar o time e cumprir os prazos.

Isso seria o mesmo cenário dos projetos tradicionais...

2.2.1 Um *Sprint* pode ser cancelado?

Em princípio, um *Sprint* só pode ser cancelado em casos extremos, senão afeta os conceitos básicos de *Scrum* de comprometimento e foco. Mas se for extremamente necessário, o *Sprint* só pode ser cancelado em uma situação:

- O *Product Owner* percebe que mudanças relevantes ocorreram e afetaram a relevância de valor de negócio da meta do *Sprint*.

Quando a meta do Sprint se torna irrelevante, os envolvidos devem iniciar, imediatamente, o planejamento do *Sprint* seguinte para evitar maiores impactos no projeto.

IMPORTANTE:

Somente o *Product Owner* tem autoridade para cancelar um *Sprint*.

2.2.2 Reunião diária

A reunião diária é de curta duração, realizada, de preferência, durante o início do dia de trabalho para melhorar a comunicação do time e permitir-lhe tomar ações

corretivas na execução dos trabalhos. Por ser uma reunião rápida que demanda atenção total de todos os membros do time, é comum que seja realizada em pé, em algum local do escritório.

Segundo o Scrum Guide®, durante essa reunião, cada membro da equipe deve falar brevemente sobre o que foi realizado no último dia de trabalho e como o time pode avançar rumo à meta do *Sprint*. É comum que times atinjam os objetivos da reunião diária, fornecendo as respostas para as três perguntas seguintes, sem desviar o foco:

- O que foi realizado desde a última reunião?
- O que vai realizar até a próxima reunião?
- Quais são os impedimentos em suas tarefas?

Todos os membros da equipe de desenvolvimento devem participar, respeitando um *timebox* de **15 minutos**.

Figura 12 – Reunião diária
Fonte: Banco de Imagens Shutterstock (2017)

Os problemas ou dúvidas que a equipe não se considera apta a resolver são classificados como **Impedimentos** e passados para o ***Scrum Master***, para que ele resolva.

A reunião diária faz parte do ciclo de inspeção e adaptação prevista nos pilares do *Scrum*.

2.3 *Burndown Chart*

O ***Burndown Chart*** é uma ferramenta de medição visual que mostra o trabalho realizado por dia contra a taxa projetada de conclusão para a versão atual do projeto. Sua finalidade é permitir a visualização do andamento do trabalho com base na programação realizada. Pode acompanhar a evolução do projeto, do *release* ou do *Sprint*.

Sua principal característica é mostrar quanto falta ser feito, e não o que já foi feito, o que proporciona uma visão real do tamanho das atividades dentro do prazo restante. Tipicamente, o *Product Burndown* é um gráfico que contém as informações do Total de Pontos a Serem Executados X A Quantidade de *Sprints*, sendo a linha de velocidade opcional durante esse ciclo (Figura "*Product Burndown"*).

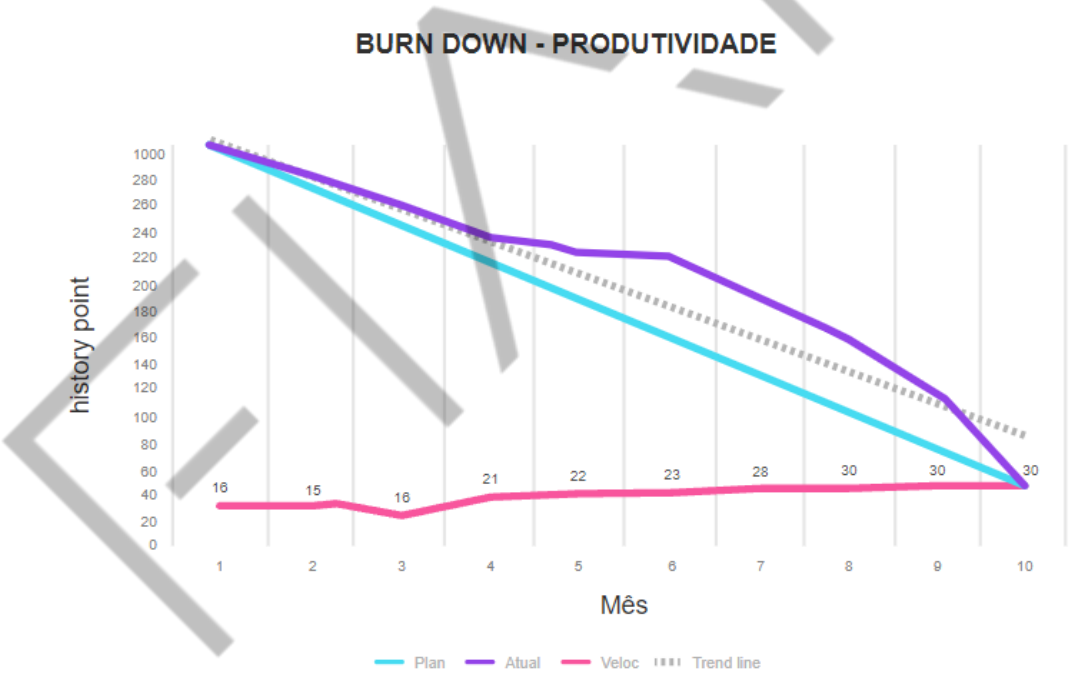

Figura 13 – *Product Burndown*
Fonte: Google Imagens (2017), adaptado por FIAP (2017)

Na Figura "*Product Burndown",* podemos observar a evolução de um projeto com mil pontos de história que foi concluído em dez *Sprints*. A linha azul refere-se ao planejamento original de execução; a linha roxa mostra o comportamento do desempenho do time ao longo do projeto, sendo observado que havia atrasos entre

os *Sprints* 3 e 9, mas que foi recuperado ao seu final. A linha pontilhada é a linha de tendência que permite ao time tomar as ações de recuperação. A linha magenta exibe a velocidade calculada pelo time a cada *Sprint.* Nota-se uma estabilização da velocidade a partir do quarto *Sprint*, com uma leve melhoria gradual até o final do projeto.

Por ter características mais curtas, normalmente, o *Sprint Burndown* representa Esforço X Duração ou Número de Atividades X Duração, sendo o primeiro o mais comum. Na Figura "*Sprint Burndown",* você pode notar um *Sprint Burndown* com dimensões em Horas X Dias Trabalhados, sendo um *Sprint* com um esforço previsto de 240 horas em duas semanas (dez dias).

Nesse *Burndown*, temos um desempenho do time melhor do que o planejado, como podemos perceber pela evolução da linha roxa em relação à linha de cor azul, estando a *Sprint* no sexto dia de execução, com tendência a terminar antes da data prevista.

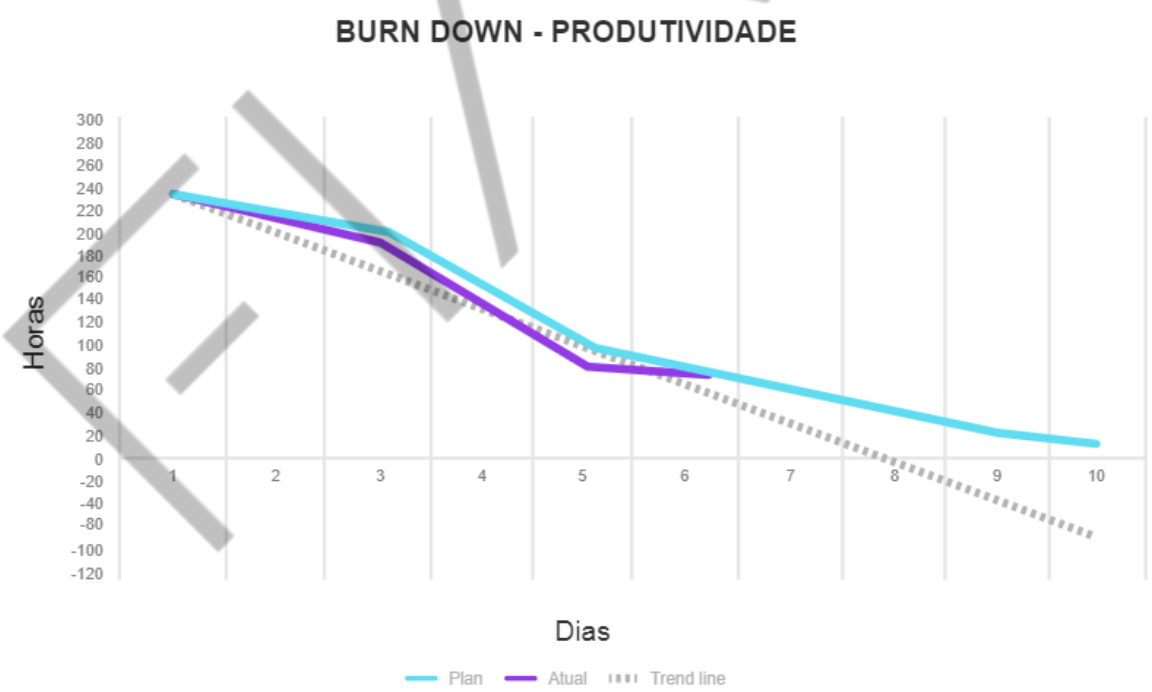

Figura 14 – *Sprint Burndown*
Fonte: Google Images (2017), adaptado por FIAP (2017)

Mapeamento de softwareTécnicas nunca são demais...

Esse gráfico deve ser atualizado ao final do dia pelo time, de acordo com a evolução do quadro Kanban, para avaliação de desempenho e verificação de ações preventivas e corretivas.

2.4 Revisão do *Sprint*

Ao final do *Sprint,* há a Revisão (*Sprint Review*), da qual todos podem participar. Seu principal objetivo é fazer a avaliação do resultado do *Sprint* pelo *Product Owner* e seus convidados (*stakeholders*).

Figura 15 – *Review Meeting*
Fonte: Banco de Imagens Shutterstock (2017)

O time é o responsável pela realização da apresentação que deve ser feita no formato de demonstração do resultado do *Sprint*. Nessa reunião, o produto deve ser mostrado funcionando, conforme as especificações.

Durante a *Review*, o *Product Owner* avalia se a meta do *Sprint* foi atingida. Então, todos os presentes na reunião discutem o que fazer em seguida, ouvindo, colaborativamente, as observações do time de desenvolvimento e o *feedback* do *Product Owner* e *stakeholders*.

Mapeamento de softwareTécnicas nunca são demais...

A *Review Meeting* também é um *timebox* definido pelo *Scrum*:

- 4 horas para *Sprint* de 4 semanas.
- 3 horas para *Sprint* de 3 semanas.
- 2 horas para *Sprint* de 2 semanas.

2.4.1 Resultados da revisão

Durante a reunião de revisão, além da avaliação, outras atividades são realizadas, segundo o Scrum Guide®:

- Retornar as funcionalidades não terminadas ao *Product Backlog* e repriorizá-las.
- Remover do *Product Backlog* funcionalidades que foram finalizadas.
- Trabalhar com o *Scrum Master* para reformular a equipe, caso necessário.
- Repriorização do *Product Backlog*.
- Solicitar o fechamento de um *Release* com as funcionalidades demonstradas, sozinhas ou com o incremento de *Sprints* anteriores.
- Autorizar ou não autorizar o próximo *Sprint*.
- Acrescentar ao *Product Backlog* novos itens, com base no *feedback* obtido durante a Revisão.

2.4.2 Retrospectiva

Após a reunião de revisão, é realizada a reunião da Retrospectiva do *Sprint*, com a participação do time *Scrum* e o *Scrum Master* como facilitador. Nesta reunião, todos os membros da equipe refletem sobre o *Sprint* passado e verificam três coisas:

- O que correu bem durante o *Sprint*?
- O que não correu bem durante o *Sprint*?
- Quais melhorias poderiam ser feitas para o próximo *Sprint*?

Figura 16 – Retrospectiva
Fonte: Banco de Imagens Shutterstock (2017)

Esta reunião é essencial para o processo de melhoria contínua do projeto.

É nela que o time se rearranja, estabelece novos padrões e corrige os problemas, permitindo à equipe melhorar a sua produtividade e qualidade.

A retrospectiva também é um *timebox* definido pelo *Scrum*:

- 3 horas para *Sprint* de 4 semanas.
- 2 horas para *Sprint* de 3 semanas.
- 1,5 hora para *Sprint* de 2 semanas.

2.5 Dicas

Há uma grande resistência em utilizar *Scrum* em projetos grandes. Isso é possível, mas não quer dizer que seja fácil de ser implementado.

O conceito para trabalhar com grandes times é chamado de *Scrum of Scrums*, pois no *Scrum* temos limite de times com até dez pessoas. Portanto,

quando falamos em escalar o *Scrum*, estamos falando de criar todos os times de dez pessoas que forem necessários para atender à demanda do projeto.

Nesse cenário de vários times trabalhando ao mesmo tempo, cada time terá seu *Scrum Master* e seu *Product Owner*, mas podemos ter vários problemas decorrentes disso, entre eles:

- Conflitos de liderança.
- Falhas de comunicação.
- Planejamento descentralizado.
- Duplicação de esforço e retrabalho.
- Problemas com a integração dos produtos.
- Dependências entre as atividades.
- Difícil controle de mudanças.

Para solucionar esses problemas, existem várias sugestões, inclusive, um guia específico só para tratar desse assunto, chamado de *Nexus Guide*.

Vamos listar alguns tópicos que ajudam na solução de alguns itens:

- Dividir o escopo do projeto em incrementos mais independentes possíveis entre si, limitando as dependências entre os times.
- Criar papéis para a centralização de decisões no nível do *Scrum Master* e do *Product Owner*.
- Criar um *Product Backlog* único e compartilhado entre todos os times.
- Criar marcos (pontos de controle) para a unificação de *releases* e publicação de uma nova versão.
- Fazer reuniões diárias conjuntas para que representantes de outros times possam participar, com o objetivo de compartilhar problemas, dependências e conhecimento.
- Criar uma reunião de *Scrum* de *Scrums* para o acompanhamento das atividades, verificação de dependências e centralização das mudanças.

Não é o objetivo, aqui, esgotar este assunto, mas passar uma visão geral e pontos críticos que devem ser observados em eventual necessidade de utilizar vários times.

IMPORTANTE:

Mais detalhes para escalar o SCRUM, consulte o *Nexus Guide* no site www.scrum.org.

Algumas ferramentas gratuitas:

- **ScrumHalf –** Brasileira, facilita o uso do Scrum. Tem um quadro Kanban virtual e propicia a colaboração e o acompanhamento da equipe. Simples de dar manutenção e priorizar o *Product Backlog,* gera o *Burn Down Chart*, entre outros: <http://myscrumhalf.com/?lang=pt>.
- **Trello** não é uma ferramenta de Scrum, mas automatiza o quadro *Kanban*: <https://trello.com/>.

REFERÊNCIAS

AGILE ALLIANCE. **Behavior Driven Development (BDD).** Disponível em: <https://www.agilealliance.org/glossary/bdd/>. Acesso em: 10 dez. 2020.

COHN, M. **Desenvolvimento com Scrum**: aplicando métodos ágeis com sucesso. Cidade, 2000. GLOSSARY.

EQUIPE CULTURA ÁGIL. **Management 3.0**: o método ágil de Gestão Empresarial. Disponível em: <http://www.culturaagil.com.br>. Acesso em: 10 dez. 2020.

SCHWABER, K.; SUTHERLAND, J. **Scrum Guide**. Scrum Org, 2020.

SCHWABER, K.; SUTHERLAND, J. **Nexus Guide**. Scrum Org, 2016.

SCHWABER, K.; SUTHERLAND, J. **Software em 30 dias**. Nova Jersey. John Wiley, 2012.

SCRUM. [s.d.]. Disponível em: <http://www.scrum.org>. Acesso em: 10 dez. 2020.

SCRUM. [s.d.]. ALLIANCE. [s.d.]. Disponível em: <https://www.scrumalliance.org/>. Acesso em: 10 dez. 2020.

WEST, D. **Updates to the Scrum Guide – The 5 Scrum values take center stage**. 6 jul. 2016. Disponível em: <https://blog.scrum.org/updates-scrum-guide-5-scrum-values-take-center-stage/>. Acesso em: 10 dez. 2020.