User Stories

Por que e como escrever requisitos de forma ágil?

RAFAEL HELM e DANIEL WILDT

Wildtech – start wild, keep wild

"Qualidade de software começa na especificação."

Rafael Helm.

Sobre os autores

Rafael Helm e Daniel Wildt são sócios da Wildtech, que é uma empresa de treinamento e consultoria de práticas ligadas ao desenvolvimento ágil de software.

Seu principal objetivo é ajudar pessoas a serem melhores profissionais, a realizarem mais e irem em busca daquilo que gera felicidade, além de ajudar times a melhorarem continuamente e organizações a se tornarem organizações conscientes e em busca de aprendizado contínuo.

Para falar com Rafael e Daniel bastar encontrá-los no twitter @rafaelhelm e @dwildt.

Ou se preferir mande email para contato@wildtech.com.br

Conteúdo

INTRO

Por mais que as tecnologias de desenvolvimento estejam evoluindo cada vez mais rápido, o desenvolvimento de software ainda é um processo complexo. São muitas fases envolvidas:

- Análise de negócios;
- Análise de requisitos;
- Projeto de banco de dados;
- Desenvolvimento;
- Testes;
- Implantação.

Dependendo da realidade da sua empresa ou equipe, o seu processo pode ser mais simples ou mais complexo do que o citado acima.

Mas pelo menos duas fases todos os processos de desenvolvimento de software possuem: **Especificação e desenvolvimento**.

Ao contrário do que muitos acreditam, o desenvolvimento de software não começa através das mãos do desenvolvedor quando elas iniciam a digitar o código. O desenvolvimento de software começa na fase de análise, e principalmente na especificação dos requisitos.

Não basta que sua equipe possua desenvolvedores altamente capacitados e responsáveis se a especificação que eles recebem é incompleta, superficial, ou burocrática demais.

Se a especificação do software é ruim, o resultado do trabalho provavelmente será um software igualmente ruim.

Mas é totalmente possível especificar software de uma forma muito mais efetiva, simples e até divertida do que o mercado normalmente tem feito ao longo dos anos.

Essa forma de especificação de requisitos mais eficiente se chama **User Stories**, que é uma pratica ágil de desenvolvimento de software, e é o tema central deste livro.

Ao longo dos capítulos vamos te apresentar a motivação para escrever requisitos utilizando user stories, também vamos te ensinar como escrever, além de citar ricos exemplos que poderão ser utilizados por você como guias durante suas primeiras user stories.

Importante:

Se você não tem nenhum conhecimento prévio sobre user stories, nós sugerimos que você leia o livro seguindo sua sequência natural.

Mas se você já tem uma noção sobre o assunto (ou já leu o livro), você poderá navegar diretamente até determinado capítulo para relembrar conceitos e tirar dúvidas.

Boa leitura!

Por que escrever user stories?

Ao longo dos anos temos visto muitas empresas tratarem seus desenvolvedores como funcionários de uma linha de montagem.

Ou seja, algumas empresas ainda acham que o trabalho de desenvolvimento de software é algo repetitivo, e acreditam que apenas dizer ao desenvolvedor **o que fazer** é suficiente.

Mas acontece que o desenvolvimento de software é um processo complexo, e na maioria das vezes não se trata de um trabalho repetitivo.

É comum um desenvolvedor encontrar várias formas de desenvolver uma mesma funcionalidade, e para que ele possa tomar uma decisão correta ele precisa de mais informações do que apenas saber **o que fazer**.

É importante que ele saiba **para quem** está sendo criada a nova funcionalidade.

Por exemplo, se o desenvolvedor souber que está desenvolvendo um recurso que será usado por vendedores que realizam em média 50 visitas por dia, é bem provável que ele desenvolva um design pensando mais em produtividade do que em elegância.

Também é vital que ele saiba o motivo desta funcionalidade, ou seja, **por que** esta funcionalidade está sendo desenvolvida.

Dando mais um exemplo: Se um desenvolvedor sabe que está alterando uma funcionalidade **por que** é necessário reduzir o tempo médio de um atendimento, ao terminar o desenvolvimento ele vai se preocupar em verificar quanto tempo leva efetivamente um atendimento com a nova interface versus o tempo deste mesmo atendimento executado na interface antiga.

Ou seja, repare que nos exemplos citados anteriormente, informar **para quem** e **por que** a funcionalidade está sendo desenvolvida ajudou o desenvolvedor a tomar decisões mais alinhadas com a necessidade do cliente. Isto tem como consequência um ganho significativo de qualidade!

Então por que escrever user stories?

Porque nós queremos que você faça certo na primeira tentativa! E o seu cliente também. :)

Existe um padrão para escrever?

Sim existem alguns padrões, mas isto não importa!

Como assim? O que importa é você entender a estrutura base de uma user story, ou seja, as informações fundamentais que precisam constar numa boa especificação de requisitos.

Como já vimos no capítulo anterior existem 3 informações que são fundamentais nas user stories, são elas:

- **Quem?** Para quem estamos desenvolvendo a funcionalidade.
- **O que?** Uma descrição resumida da funcionalidade em si.
- **Por que?** O motivo pelo qual o cliente precisa desta funcionalidade. Se possível citando o ganho de negócio.

Normalmente para responder as três perguntas citadas acima nós usamos o SENDO... POSSO... PARA QUE...

Um exemplo:

SENDO um vendedor que realiza 50 visitas por dia

POSSO consultar as últimas compras de cada cliente

PARA QUE ao chegar no cliente eu possa consultar qual foi sua última compra, e assim conseguir negociar com ele estando melhor informado.

Repare que no SENDO nós identificamos o perfil do usuário que vai usar a funcionalidade, no POSSO a funcionalidade em si que precisa ser desenvolvida e no PARA QUE a motivação da funcionalidade, incluindo o ganho de negócio.

Com estas informações, o desenvolvedor vai conseguir trabalhar "mais armado", e provavelmente vai criar uma funcionalidade mais bem elaborada do que se recebesse apenas a necessidade do cliente sem o detalhamento de **quem** vai usar e **por que** vai usar.

Entendido? Mais ainda falta uma informação muito importante, que é o **como testar?** Veremos isto no próximo capítulo.

Como testar? BDD!

No capítulo anterior entendemos melhor a importância do **quem**, **o que**, e **por que**, mas ainda falta um ponto muito importante para fecharmos a estrutura de uma boa user story: **O como testar?**

Para isto podemos usar a técnica do BDD (Behavior Driven Development) de Dan North, onde as palavras chave Dado que... Quando... Então... nos apoiam na criação de ricos cenários de teste.

Exemplos:

Cenário 1: Estoque disponível

Dado que o estoque da coca-cola é de 50 unidades

Quando informo uma venda de 40 unidades

Então a venda é registrada

E o estoque passa a ser de 10 unidades

Cenário 2: Estoque indisponível

Dado que o estoque da coca-cola é de 50 unidades

Quando informo uma venda de 60 unidades

Então a venda não é registrada

E é exibida na tela a mensagem “estoque insuficiente!”

Repare que nos exemplos anteriores nós usamos o “Dado que” para indicar o cenário atual, o “quando” para indicar a ação do usuário, e o “Então” para indicar como o software vai reagir.

Podemos também usar o “E” e o “OU” para criar cenários de teste ainda mais ricos.

Exemplos:

Cenário 1: Estoque disponível, venda limitada a 30

Dado que o estoque da coca-cola é de 50 unidades

E a venda máxima por cliente é limitada a 30 unidades

Quando informo uma venda de 20 unidades

Então a venda é registrada

E o estoque passa a ser de 30 unidades

Cenário 2: Venda com cartão indisponível para valores abaixo de 20,00

Dado que o valor da venda é de 10,00

E o valor mínimo de vendas para cartão é de 20,00

Quando informo que o meio de pagamento é cartão de crédito

OU informo que o meio de pagamento é cartão de débito

Então a venda não é registrada

E é exibida na tela a mensagem “Meio de pagamento inválido! Para valores inferiores a 20 reais somente dinheiro.”

Importante: Você não precisa escrever os critérios de aceitação exatamente desta forma. Mas e interessante que você registre de alguma forma os testes que devem ser realizados para que a user story possa ser bem testada.

Nós particularmente gostamos muito de usar o “Dado que”, “quando”, “então”, mas fica a seu critério.

Para saber mais sobre BDD acesse a Wikipédia, lá você vai encontrar um ótimo artigo sobre o assunto.

O conceito INVEST

INVEST é um acrônimo (em inglês), que pode nos ajudar a revisar as user stories para verificar se elas foram bem escritas.

Independent (deve ser independente)

Negotiable (deve ser negociável)

Valuable (deve agregar valor para o cliente)

Estimable (deve ser possível estima-la)

Small (deve ser pequena)

Testable (deve ser testável)

Resumindo: Uma boa user story **não deve depender** de outra, deve ser possível **negocia-la** de forma que você possa alterar sua prioridade e ordem de execução com o cliente, deve **agregar valor**, deve ser possível **estima-la**, deve ser **pequena** (até para pode ser estimada), e deve ser **testável**.

Na prática em alguns casos pode ser bem difícil escrever user stories INVEST, mas com o tempo e prática vai ficando mais fácil. Então não desista. ;)

E os bugs, também viram user stories? NÃO!!!

Não! Nós não escrevemos user stories para registrar erros. User stories são uma forma ágil de especificação de **novos** requisitos, ou para especificação de **evoluções** de requisitos.

Mas isto não quer dizer que nós não vamos te mostrar uma forma efetiva de registrar relatos de bugs. ;)

Ao longo dos anos nós obtivemos muito sucesso na correção de bugs nos times que trabalhamos. Ou seja, temos conseguido resolver os bugs na primeira tentativa.

Ok, sabemos que os bugs não devem ocorrer, mas infelizmente eles ocorrem.

Então veja na página a seguir o nosso modelo para relato de bug.

LOCAL: Nome do Sistema - Módulo e Menu relacionado

VERSÃO:

Identificar em que versão do sistema envolvido o problema pode ser repetido. Importante identificar se o problema pode ser repetido na última versão.

PRÉ-CONDIÇÕES:

* Identifique o que deve estar configurado no ambiente para que o problema pode ser repetido;

* Pode ser uma lista de configurações a serem marcadas;

* Ou simplesmente a indicação de que uma base de dados específica deve ser usada.

PASSOS PARA REPRODUÇÃO DO ERRO:

1) Monte uma lista indicando os passos que devem ser realizados para repetir o erro;

2) Você pode ser específico e identificar o que deve ser preenchido em cada campo;

3) Principalmente se uma base de dados específica está sendo usada para trabalhar;

4) Deve ser possível para qualquer pessoa repetir o erro lendo esta lista de passos.

ERRO:

Mostrar o erro que está acontecendo. Pode ser com uma identificação do que está acontecendo de errado - e muito importante: mostrar contexto de negócio identificando porque a situação atual é um erro.

SITUAÇÃO DESEJADA:

Descreva a situação que o sistema de mostrar, identifique configurações que não estão sendo consideradas, mostre o que deve ser modificado pensando em regras de negócio para resolver a situação.

Segue um exemplo:

LOCAL: SoftVendas – Módulo Mobile – Tela de vendas de produtos

VERSÃO:

Identificado na última versão (03.50), o problema não ocorre em versões anteriores.

PRÉ-CONDIÇÕES:

* Acessar o ambiente de homologação;

* Logar com usuário “alfredo”, senha “xyz9988”;

PASSOS PARA REPRODUÇÃO DO ERRO:

1) Uma vez já logado no sistema mobile, acesse o menu “Vendas”;

2) Selecione um cliente qualquer e abra uma nova venda;

3) Na tela de listagem de produtos, selecione qualquer produto;

4) Após selecionar um produto informe a quantidade a ser vendida (pode ser 10), e no campo desconto informe um desconto (pode ser 10% de desconto).

ERRO:

Mesmo após informar o desconto, o valor total do produto segue sendo o mesmo que era antes (sem o desconto).

SITUAÇÃO DESEJADA:

Que o valor total do produto considere o desconto aplicado, ou seja:

Valor total do produto = (Valor unitário * Quantidade) – Desconto.

Exemplo: Se valor do produto é 90,00, e a quantidade informada é 10, e o percentual de desconto é informado é 10%, então o valor total do produto deve ser 810,00.

Algumas considerações sobre o exemplo citado:

Repare como as seções PRÉ CONDIÇÕES e PASSOS PARA A REPRODUÇÃO DO ERRO, são importantes para fazer o erro acontecer.

Perceba também que na seção SITUAÇÃO DESEJADA além de citar a explicação do que deve ocorrer nós também citamos um exemplo prático, neste caso com um exemplo real do cálculo de preço total do produto.

Ainda sobre o exemplo, verifique que não usamos emoção no relato do defeito, ou seja, não existe nenhuma frase parecida com “mais uma vez ocorreu um erro primário na aplicação”, ou “é inadmissível que erros como este ocorram numa funcionalidade tão importante do nosso software”.

Relados carregados de emoção, frustração ou cobrança não são efetivos. O importante no relato de um defeito é (1) mostrar como repetir o problema, (2) detalhar o problema, (3) apresentar o comportamento esperado.

Esperamos que este modelo de relato de bug ajude você a melhorar a qualidade da especificação dos defeitos. Afinal de contas eles não devem acontecer, mas se acontecer que pelo menos eles sejam resolvidos na primeira tentativa. :)

Exemplo: Saque no caixa eletrônico

Vamos imaginar que você trabalha em um sistema bancário de auto atendimento (caixa eletrônico).

Seu cliente envia para você um email solicitando e explicando como funciona o saque do banco:

"Olá! Precisamos disponibilizar a operação de saque no caixa eletrônico.

Segue as regras do banco para saques em caixas eletrônicos:

- Por questões de segurança o valor máximo de cada saque é de 800,00;

- Os saques só estão liberados entre 6h00min e 22h59, em qualquer dia, útil ou não;

- O saldo do cliente não pode ficar negativo, exceto se ele possuir limite de cheque especial;

- O cliente jamais poderá ultrapassar seu limite de cheque especial;

- Deve ser impresso um comprovante de saque ao final da operação, (se o cliente assim desejar)."

Como você transformaria este email do cliente em uma user story?

Segue um exemplo:

SENDO um cliente correntista do banco

POSSO sacar dinheiro em caixas eletrônicos

PARA poder comprar em estabelecimentos que não aceitam cartão de débito/crédito

Cenário 1: Horário limite

DADO QUE são 5h00

E já estou autenticado no caixa eletrônico

QUANDO solicito sacar 10,00

ENTÃO o sistema apresenta a mensagem "Os saques somente são permitidos entre 6h00min e 22h59"

E o saque não é realizado

Cenário 2: Valor máximo de saque

DADO QUE a hora atual está entre 6h00min e 22h59min

E já estou autenticado no caixa eletrônico

QUANDO solicito sacar 1.000,00

ENTÃO o sistema apresenta a mensagem "O valor de um único saque no caixa eletrônico está limitado a R$ 800,00"

E o saque não é realizado

Cenário 3: Saldo insuficiente (cliente não tem limite)

DADO QUE a hora atual está entre 6h00min e 22h59min

E já estou autenticado no caixa eletrônico

E meu saldo é +600,00

E não tenho limite de cheque especial

QUANDO solicito sacar 700,00

ENTÃO o sistema apresenta a mensagem "Saldo insuficiente"

E o saque não é realizado

Cenário 4: Saldo insuficiente (cliente tem limite)

DADO QUE a hora atual está entre 6h00min e 22h59min

E já estou autenticado no caixa eletrônico

E meu saldo é +100,00

E meu limite de cheque especial é 500,00

QUANDO solicito sacar 700,00

ENTÃO o sistema apresenta a mensagem "Saldo insuficiente"

E o saque não é realizado

Cenário 5: Saldo disponível (sem usar limite)

DADO QUE a hora atual está entre 6h00min e 22h59min

E já estou autenticado no caixa eletrônico

E meu saldo é +600,00

QUANDO solicito sacar 200,00

ENTÃO o sistema libera o dinheiro no caixa eletrônico

E meu saldo passa a ser +400,00

E a tela de emissão de impressão de recibo é exibida

Cenário 6: Saldo disponível (usando limite)

DADO QUE a hora atual está entre 6h00min e 22h59min

E já estou autenticado no caixa eletrônico

E meu saldo é +100,00

E meu limite de cheque especial é 500,00

QUANDO solicito sacar 500,00

ENTÃO o sistema libera o dinheiro no caixa eletrônico

E meu saldo passa a ser -400,00

E a tela de emissão de impressão de recibo é exibida

Cenário 7: Emissão de recibo (confirmação de impressão)

DADO QUE meu saque foi autorizado

E a tela de impressão de recibo está sendo exibida

QUANDO eu confirmo a impressão do recibo

ENTÃO o recibo é impresso

E o sistema retorna a tela inicial do caixa eletrônico

Cenário 8: Emissão de recibo (impressão ignorada)

DADO QUE meu saque foi autorizado

E a tela de impressão de recibo está sendo exibida

QUANDO eu indico não imprimir o recibo

ENTÃO o sistema retorna a tela inicial do caixa eletrônico

Repare como a user story ficou mais rica do que o email do cliente.

Nos casos em que o sistema precisou emitir uma mensagem de erro ela já estava especificada no próprio critério de aceitação.

Em todos os casos que o saldo foi manipulado nós registramos exemplos práticos nos critérios de aceitação. Isto ajuda muito no processo de teste.

Agora pare e reflita. Comparando o email do cliente com a user story, qual especificação é mais passível de bugs?

Provavelmente o email, pois ele cita de forma superficial cada cenário de teste, enquanto que a user story detalha melhor cada um dos cenários.

Alguns Lembretes valiosos

- Qualidade de software começa na especificação.
- Se a especificação do software é ruim, o resultado do trabalho provavelmente será um software igualmente ruim.
- Além de "o que fazer" o desenvolvedor também merece saber "para quem" e "por que" cada funcionalidade será desenvolvida.
- De atenção especial aos critérios de aceitação. Eles estão diretamente ligados a como seu software será testado.
- Quando você achar que uma user story está pronta verifique se ela ficou INVEST.
- Evite ao máximo escrever user stories grandes e sempre pergunte a si mesmo se uma user story pode ser "quebrada".

Terminei o livro, e agora?

Legal você não ter abandonado a leitura do livro. Muito obrigado pela consideração!

Isto provavelmente significa que essa tal de user stories deve ter algum sentido para você. Ou quem sabe você apenas não tinha nada melhor para fazer mesmo. :)

De qualquer forma nós vamos deixar algumas sugestões sobre o que você pode fazer a partir de agora:

- Se você gostou do livro então nos ajude a divulga-lo e compartilhe o link http://historiasdeusuario.com.br no facebook, twitter, linkedin, tumbler e etc. Junte se a nós e vamos tornar as especificações de requisitos de software mais amigáveis, objetivas e divertidas.

- Mas se você não gostou do livro então nos mande email dizendo o motivo, nós prometemos não ficar magoados. Pode escrever para contato@wildtech.com.br

- Pratique! Tente escrever algumas user stories. Comece pelas mais simples. Use o exemplo do livro como guia de referência.

- Nós estamos no twitter, nos siga e manda um alô. http://twitter.com/rafaelhelm / http://twitter.com/dwildt

Vá e conte as histórias
dos seus usuários. :)