Universidade Estadual do Maranhão
Centro de Ciências Tecnológicas
Programa de Pós-Graduação em Engenharia da Computação e Sistemas

Análise da qualidade do software mínimo por meio da elucidação priorizada de requisitos

Weldys da Cruz Santos

São Luís, Junho de 2019

Weldys da Cruz Santos

Análise da qualidade do software mínimo por meio da elucidação priorizada de requisitos

Dissertação de Mestrado apresentada ao Programa de Mestrado Profissional de Engenharia da Computação e Sistemas, da Universidade Estadual do Maranhão (UEMA), como parte dos requisitos para a obtenção do título de Mestre em Engenharia da Computação e Sistemas.

Universidade Estadual do Maranhão – UEMA

Curso de Engenharia de Computação e Sistemas

Orientador: Cicero Costa Quarto

São Luís

Junho de 2019

Santos, Weldys da Cruz.
Análise da qualidade do software mínimo por meio da elucidação priorizada de requisitos / Weldys da Cruz Santos. – São Luís, 2019.
79 f.

Dissertação (Mestrado) – Universidade Estadual do Maranhão – UEMA
Curso de Engenharia de Computação e Sistemas, Junho de 2019.

Orientador: Prof. Dr. Cicero Costa Quarto

1. Engenharia de Software. 2. Engenharia de Requisitos. 3. Qualidade de Software I. Titulo

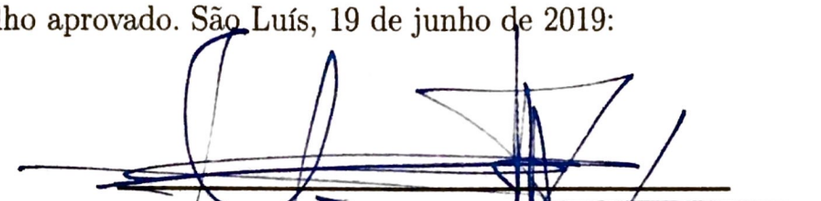
CDU 004.05

Weldys da Cruz Santos

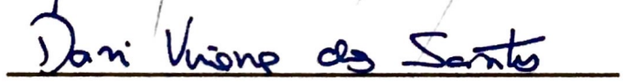
Análise da qualidade do software mínimo por meio da elucidação priorizada de requisitos

Dissertação de Mestrado apresentada ao Programa de Mestrado Profissional de Engenharia da Computação e Sistemas, da Universidade Estadual do Maranhão (UEMA), como parte dos requisitos para a obtenção do título de Mestre em Engenharia da Computação e Sistemas.

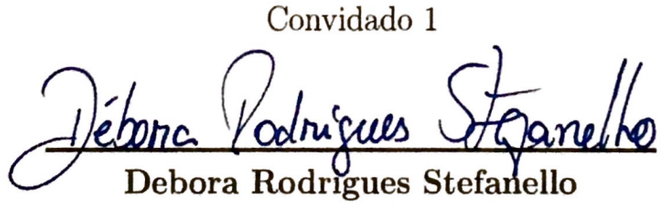
Trabalho aprovado. São Luís, 19 de junho de 2019:



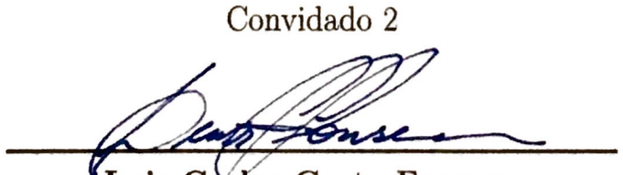
Cicero Costa Quarto
Orientador



Davi Viana dos Santos
Convidado 1



Debora Rodrigues Stefanello
Convidado 2



Luis Carlos Costa Fonseca
Convidado 3

São Luís
Junho de 2019

Aos meus pais e ao meu filho Miguel,
por me apoiarem e serem a razão da minha vida.

Agradecimentos

Agradeço meus pais pela oportunidade de desfrutar da educação e de poder chegar até este momento em minha vida. Agradeço ao meu orientador Cícero Quarto, pela paciência em corrigir e recorrigir cada ponto desta pesquisa e pelos conselhos sobre a vida acadêmica. Aos professores e outros profissionais da área que me ajudaram em pontos específicos, não só do trabalho, mas também quando foi necessário intervir em algo que precisava ser consertado. À Creative Pack, empresa no qual tenho tanto orgulho de fazer parte e que me permitiu realizar a maioria os testes que constam neste trabalho de pesquisa e que fez com que esse processo fosse muito menos doloroso. Aos meus amigos, sócios, colegas de trabalho que me incentivaram, lutaram junto comigo e suportaram cada momento difícil no processo de produção deste projeto, à Jessica Silva, que além de ter me presenteado com meu filho Miguel, permitiu que eu pudesse ter condições de chegar até o fim desta pesquisa sem maiores percalços, mas o agradecimento pessoal mais importante é para o meu filho para o qual dedico cada um dos meus passos na vida. Sem ele, muito provavelmente a realização de qualquer trabalho seria mais vazio.

Resumo

Em um mundo onde o desenvolvimento tecnológico das startups tem uma velocidade diferente para o que precisa ser desenvolvido e os futuros clientes dessas startups precisam de soluções cada vez mais específicas para ter em seus aparelhos com espaço em disco reduzido e com cada vez menos tempo para aprender a utilizar algo, esta pesquisa propôs justamente a possibilidade de criar softwares com um escopo mais reduzido, mas que atenda perfeitamente o que o usuário final precisa e o que o cliente poderá pagar, com uma otimização de tempo com elucidação de requisitos e com a análise da qualidade do software. A investigação de pesquisa realizou laboratórios com startups que pudessem abrir seus requisitos de software e que pudessem também modificar a trilha de desenvolvimento para se adequar ao método proposto. A partir das análises e discussões de resultados, o trabalho demonstrou que o software "suficientemente pronto" é viável quando os requisitos são bem coletados, analisados e descritos para o seu desenvolvimento.

Palavras-chaves: engenharia de software. engenharia de requisitos. desenvolvimento de produto de software.

Abstract

In a world where we have startups with different velocity on your needs and their future customers' needs solutions more specifics to your device with reduced diskspace and less time to learn how to do something on that. This research focus on create software reduced scope and maintaining quality as the user needs and can pay for this optimizing time with requirements elucidation and analyzing the software quality through validations with the final user. The investigation did workshops with startups who could open your software requirements and could doing change the way of development not only software, but the business itself to be compatible with this model. From the analysis and discussions of the results, this work show the software "ready enough" is viable when the requirements are quite well collected, analyzed, described and validated with success criteria.

Keywords: software engineering. requirements engineering. product development

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

MVP	Produto mínimo viável (do inglês Minimum Viable Product)

Lista de símbolos

Γ	Letra grega Gama
Λ	Lambda
ζ	Letra grega minúscula zeta
$\in$	Pertence

Sumário

Parte I

Construindo o Objeto de Pesquisa

1 Contextualização

Considerando o conceito de Bauer (1), que a Engenharia de Software é uma Ciência que aplica modelos matemáticos, especificação, desenvolvimento e manutenção de sistema de software, utilizando métodos das engenharias, haverá a premissa de que essa é uma Ciência exata, com valores esperados, ao unir determinados fatores e por isso não considera o ser humano como agente transformador em quaisquer que sejam as fases da Engenharia de Software. Levando-se em conta que o software deve ser criado para o usuário, como Krug (2) pontua, este (o usuário) se transforma no principal ator dentro de um processo de desenvolvimento de um software.

Em 2001, um movimento iniciado por renomados desenvolvedores chamado Aliança dos Ágeis criou o Manifesto para o Desenvolvimento Ágil (3), que guiaria todas as metodologias de desenvolvimento ágil existentes, sugere que o desenvolvimento de software deve ter preferência em ser guiado para pessoas e suas interações, ao invés de ferramentas e processos. Esse ponto discorre sobre como é importante que haja o melhor entendimento das pessoas sobre os problemas envolvidos no que será desenvolvido do que somente alimentar ferramentas que irão, de certa forma, automatizar o processo de entendimento do processo.

Um estudo feito por Krigsman (4), indicou um índice extremamente alto em problemas de entrega de software, chegando na casa dos 79% e isso foi percebido com diversas causas. Ainda segundo o estudo, um dos motivos está relacionado à falta de entendimento do escopo do software que deve ser desenvolvido. Esse índice chega a 40% dos problemas de software não entregue segundo o mesmo estudo, conclui este autor.

A usabilidade do software é medida basicamente pelos padrões utilizados no visual da aplicação (2), não permitindo gerar informações de usabilidade com relação à validação do software quanto ao usuário final, se aquele produto atende ao que se destina ou se ele atende aos requisitos mínimos do uso por aquele tipo de usuário ou mesmo ao grau de afetividade que aquele software criou em relação ao usuário final.

Atualmente, no Brasil existem mais de 12 mil startups segundo censo da principal associação de startups do país (5), sendo a grande maioria delas reconhecidamente de base tecnológica em que possuam algum momento de desenvolvimento de software onde estas buscam solucionar problemas específicos de pessoas ou empresas através de um modelo de negócios escalar — que permitem crescimento exponencial do negócio — com o mínimo possível de utilização de recursos financeiros sendo estes recursos próprios ou de fundos de investimento.

2 Motivação

O mercado de desenvolvimento de software está em crescimento em todo o mundo e até 2019 chegará a movimentar cerca de 2.8 trilhões de dólares (**??**).

Ainda sobre o mercado de software, considerando essa movimentação financeira, é preciso entender quanto desse montante é utilizada de forma correta, haja visto anteriormente que existe um índice de 40% de problemas com o entendimento do software e consequentemente com a acurácia dos requisitos trabalhados no projeto de software (4).

Considerando também que o mercado de startups no Brasil está em aquecimento, como mostram os dados da Associação Brasileira de Startups (ABS) (5) e que grande parte dessas startups são por base um produto de algum tipo de software e que estas startups têm por base de financiamento os recursos próprios ou de fundos de investimento, o retorno de investimento precisa ser rápido e preciso, uma vez que um novo ciclo de investimentos depende dos resultados da primeira versão da startup (6), o MVP.

Considerando a contextualização e motivação, trazidas, esta pesquisa propõe um modelo de análise do processo de desenvolvimento de software em algumas startups no aspecto de como elas lidam com o escopo de software e como elas buscam a otimização para que o produto seja direcionado ao usuário final sem que haja retrabalho ou custos extras de desenvolvimento em adequar o produto a estes usuários ou sobre sua qualidade. Desse modo, o estudo é norteado na avaliação do escopo do software mínimo para o processo de desenvolvimento do software na fase de engenharia de requisitos, com base nas necessidades reais do usuário final, trabalhando principalmente nas fases de Engenharia de Requisitos e na Engenharia de Qualidade de software, quando tratamos do processo de Engenharia de Software.

Este trabalho também foi iniciado pela motivação de perceber que existem diversos trabalhos que veem a Engenharia de Requisitos como a porta de entrada para bons softwares e estes têm utilizado metodologias inovadoras para melhorar este processo. Um desses métodos é o Design Thinking, que é um modelo mental que permite modelar através de validações com o usuário, um formato de gestão de requisitos, como diz Grosskopf (7) em seu trabalho sobre o processo. E partindo deste ponto, entendendo como criar um modelo de gestão de requisitos de software que atenda tanto aos interesses das startups, como também com a realidade orçamentária destas e a expectativa do usuário com relação ao que receberá como produto.

3 Questão de pesquisa

A partir dos dados levantados, tanto com relação a problemas de grandes empresas com grandes softwares como startups que estão iniciando seu processo de desenvolvimento e entregando sua primeira versão de software, este trabalho formula a seguinte questão de pesquisa: “Como melhorar o escopo de desenvolvimento do produto de software em startups sob a perspectiva da necessidade do usuário final?”

4 Objetivos

4.1 Objetivo geral

Investigar como o software é construído a partir dos seus requisitos, através de métodos de validação de cada requisito de forma rápida e barata e analisar a qualidade do software com base na validação dos usuários e os critérios de sucesso definidos pela equipe de desenvolvimento, tendo como base, o atendimento das necessidades do usuário e como o desenvolvimento pode ser mais produtivo, através da redução o escopo de desenvolvimento, possibilitando que a startup obtenha um resultado mais rápido da validação de sua solução de software.

4.2 Objetivos específicos

- medir a qualidade dos requisitos do software através de validação
- conceber um modelo de criação de escopo de software com base na validação das necessidades do usuário
- validar este escopo com base no que a startup precisa entregar como produto e nos seus critérios de sucesso

5 Visão metodológica da pesquisa

Quanto ao que se refere à visão metodológica desta pesquisa, tem-se por conceito como Gil (8), a divisão da pesquisa em dois tipos: em relação aos objetivos e em relação aos procedimentos técnicos. Com relação aos objetivos, a pesquisa irá se dirigir ao formato em que se avançam com relação aos objetivos específicos, sendo assim, o processo decorre em se aproximar da problemática através de estudo sobre a literatura da engenharia de software, metodologias de desenvolvimento e engenharia de requisitos, além de analisar startups com casos de sucesso no desenvolvimento de software, através de estudos de caso. Além disso, há a pesquisa explicativa onde são analisados os pontos onde residem os problemas quanto às entregas e quanto ao escopo do software que está sendo desenvolvido enfatizando sempre a qualidade do produto com relação a usabilidade e a validação dos requisitos em termos de software e produto final. Além disso, também é estudado o escopo que é aplicado a cada lançamento de nova versão do software em questão. Em relação aos procedimentos técnicos, é utilizada a abordagem explicativa onde os dados que são coletados através de observação e são interpretados e transformados em técnicas para desenvolvimento de software com base na questão de pesquisa. Este formato é adotado neste trabalho por se tratar de uma análise de comportamento de startups no âmbito do desenvolvimento de software tratando também das particularidades destas, como a quantidade de desenvolvedores, a capacidade de entendimento do escopo que estão tentando absorver para o seu produto e também as condições técnicas de desenvolvimento, como complexidade de tecnologia, acessibilidade e capacidade técnica da equipe.

6 Organização do texto da dissertação

Esta pesquisa é organizada em partes e capítulos. A divisão por partes é feita para organização de blocos de capítulos, onde o leitor poderá dividir-se entre analisar os processos de pesquisa, de laboratórios e resultados por meio dos capítulos. Dessa forma, além da Parte I, já descrita, a pesquisa é organizada em mais seis partes. A parte II é reservada para a Fundamentação Teórica, onde os principais temas, tais como Engenharia de Software e Engenharia de Requisitos, além de Story Mapping, Métricas de qualidade que envolvem a pesquisa são trazidos e descritos. A concepção, desenvolvimento e implementação do modelo de engenharia de software proposto são demonstradas através da Parte IV. A Parte V é reservada para trazer o processo de validação, atrás de laboratórios e análise dos dados gerados. A parte VI traz os resultados da validação do modelo concebido em formato de comparação, comportamentos específicos das startups em cada um dos laboratórios e seus momentos de desenvolvimento. Análise e discussões de resultados são apresentadas através da Parte VII. Por fim, na Parte VIII, conclusões e perspectivas de trabalhos futuros são formuladas.

Parte II

Fundamentação teórica

7 Engenharia de Software

Pressman (9) pontua que a Engenharia de Software se define em um conjunto de métodos e práticas, criando um processo, para possibilitar que o profissional possa desenvolver um sistema de qualidade. Ainda segundo Pressman, a Engenharia de Software se divide em quatro camadas, a saber: Foco na qualidade, Processos, Métodos e Ferramentas. A seguir, as supracitadas camadas são descritas à luz de Pressman.

Foco na qualidade é a medida fundamental do processo de software, pois a Engenharia de Software, assim como todas as engenharias, precisa de um comportamento organizacional com a qualidade

Processos onde é definida a metodologia de trabalho e mantém a coerência entre todas as outras camadas da Engenharia de Software.

Métodos onde se constroem as etapas técnicas para os desenvolvimentos do software. Nela são incluídas a comunicação, a engenharia de requisitos, modelagem e desenvolvimento do software, suporte e testes. Este trabalho focará basicamente nestas camadas, com foco maior na engenharia de requisitos. Nesta fase acontecem muitos dos problemas com o desenvolvimento do software, conforme será visto mais à frente neste trabalho.

Ferramentas as ferramentas dão suporte ao que se deve ser desenvolvido, automatizando ou semi-automatizando o máximo dos processos. Isso permite que o máximo dos métodos seja aproveitado. A interação com as ferramentas, estabelece o que é chamado de Suporte de Desenvolvimento de Software, cunhado de Engenharia de Software com Auxílio de Computador. Sommerville (10) frisa que a Engenharia de Software é a disciplina da engenharia que se preocupa com todos os aspectos da produção do software desde os estágios iniciais da especificação até a sua manutenção, depois da entrega. Ainda segundo o autor, mesmo com fases definidas, não há um padrão a ser seguido rigorosamente dentro da Engenharia de Software, pois o autor defende que cada projeto tem sua forma diferente de lidar e a heterogeneidade destes é o maior desafio dentro deste campo de estudo. A Engenharia de Software possui fundamentos bem definidos e Sommerville afirma que todos estes devem estar presentes no processo de software para que este tenha sucesso ao seu final. Assim, os fundamentos são listados abaixo:

Processo o software ser desenvolvido em um processo conhecido e compreendido. Os envolvidos no processo devem planejar o processo e entender completamente o que será desenvolvido e quando deverá ser entregue. Processos podem ser diferentes para diferentes tipos de projetos

Confiança e desempenho os sistemas de software devem ser confiáveis e desempenhar de satisfatória as suas tarefas, se comportar sem falhas e estar disponível sempre que for solicitado, além de não desperdiçar recursos do sistema

Requisitos Os requisitos devem ser claros, gerenciáveis, assim como sua especificação deve ser bem clara para o público que está direcionado. Se são requisitos com alta abstração, deve ser clara para o cliente, assim como os documentos técnicos devem ser claros para os desenvolvedores.

Gestão e reuso O software deve permitir que haja desenvolvimento contínuo, sem a necessidade de reescrita ou de desperdício de recursos para aplicação de novos requisitos.

7.1 Engenharia de Requisitos

Segundo Sommerville (10), a Engenharia de Requisitos é uma área da Engenharia de Software que define as descrições do que um software pode fazer. Esses requisitos devem ser reflexo do que os usuários necessitam para ter um bom e útil produto de software em mãos. Para isso, é necessário que essas descrições estejam um documento chamado documento de requisitos. O documento de requisitos é um conjunto de declarações sobre o que um sistema deve fazer, segundo Sommerville (10), além de descrever quais os serviços e suas restrições quanto ao funcionamento. Sommerville também afirma que o conjunto de declarações que compõem o documento de requisitos possui muitas interpretações que geram em muitos casos, problemas com relação à sua interpretação, levando o software a ser desenvolvido com diferentes finalidades por conta do nível de abstração com que é tratado. Quando o documento é escrito de forma técnica, o cliente pode não ter conhecimento para validar exatamente quais os comportamentos do sistema e quando é escrito em linguagem natural, com um alto nível de abstração técnica, o software pode não conter todas as restrições e haverá um problema de desenho do escopo do que será desenvolvido. Segundo Davis (11), o documento de requisitos deve ser escrito em duas fases: a primeira onde o nível de abstração é mais alto e deve ser abrangente o suficiente para que varias empresas de desenvolvimento possam oferecer soluções iguais de forma diferente, permitindo que haja maleabilidade quanto ao que precisará ser desenvolvido e o custo envolvido no desenvolvimento deste software e a segunda fase, quando a contratação já fora efetivada e já se faz necessário o desenvolvimento mais aprofundado do documento de requisitos contendo todas as especificações técnicas feitas pela empresa contratada e validadas pela empresa contratante, a fim de remover todas as dúvidas quanto ao que deve ser desenvolvido e a como deve ser desenvolvido.

Para Pressman, a engenharia de requisitos contempla sete tarefas distintas:

1. Concepção: momento em que a primeira conversa com o cliente e são conhecidas as necessidades do negócio e do mercado em que esta futura solução estará inserida. Não são tratados aspectos técnicos do software, somente sobre a questão de mercado e o problema levantado.

2. Levantamento (elucidação): é a fase em que o projeto deve ser entendido como software e não mais somente como uma necessidade levantada. O processo não envolve apenas ouvir o cliente, mas também participar de uma investigação que permita entender mais sobre o usuário final e sobre o mercado. A tarefa de levantamento também contempla entender, priorizar e definir qual será o escopo de desenvolvimento do software. Esta fase é crítica pois há a introdução de requisitos com detalhamentos técnicos que podem ou não serem considerados suficientes.

3. Elaboração: é criado um documento técnico com todos os requisitos que devem contemplar o software, incluindo suas restrições de sistema. Nesta fase, é importante que haja a produção de diagramas de suporte para o entendimento técnico do documento

4. Negociação: fase em que o documento de requisitos é confrontado com as restrições do cliente, sejam elas de tempo ou de recursos financeiros. A priorização feita na fase de levantamento pode ser um critério para entendimento do que deve ser desenvolvido inicialmente

5. Especificação: a especificação é feita com o auxílio de descrição de cenários, gráficos incluindo também modelos de protótipo de como o software deve se comportar

6. Validação: a validação do documento de requisitos acontece para que não haja problemas de interpretação do texto, como requisitos vagos ou com duplo entendimento. É feita uma revisão técnica de todo o documento para que entregue ao cliente.

7. Gestão: a gestão passa pela necessidade dos requisitos mudaram ao longo do tempo, possibilitando ou impossibilitando melhorias no sistema de software, A gestão de requisitos acontece durante todo o ciclo de vida do sistema.

Percebe-se então, uma alta relevância acerca da Engenharia de Requisitos, por levar ser o fio condutor de todo o processo de desenvolvimento do software. Esta fase tem suma importância principalmente no que se refere à comunicação com o cliente e com o usuário final sobre quais problemas devem ser solucionados por meio de um produto de software. Neste trabalho a Engenharia de Requisitos é onde são criados tanto a lista das funcionalidades como também os critérios que nortearão o sucesso do software tanto em termos de software como processo de qualidade como quanto produto comercial.

7.2 Engenharia de Qualidade

Definir o que é qualidade de software é algo bastante complexo por se tratar de algo tão subjetivo. Segundo Crosby (12), o grau de qualidade aumenta de acordo com a conformidade dos requisitos. Porém ela continua subjetiva, principalmente quando temos que analisar qual é essa conformidade.

Em Konscianski (13), há a discussão sobre essa conformidade. A conformidade pode ser com base no que foi observado, mas ainda assim deveria considerar a taxa de erro dessa observação, dado que os requisitos também são buscados junto aos stakeholders e como Sommerville afirma, os stakeholders tem formas diferentes de ver os mesmos requisitos e isso pode gerar interferência no que é percebido pelo analista de requisitos. Dada essa falta, é necessário calcular uma taxa de erro com relação à percepção dos requisitos.

Assim, a qualidade seria "calculada" em função da diferença entre o que foi especificado e o que foi observado, mais o erro de medição que está fora do controle do analista.

Esta observação neste trabalho foi convenientemente chamada de Critérios de sucesso em termos de validação do software em relação a solução para o usuário final. Essa diferença, neste trabalho, é feita simultaneamente antes mesmo do desenvolvimento, para que haja um direcionamento quanto ao que deve ser desenvolvido ou não para aquele software.

7.2.1 Subjetividade na qualidade de software

A qualidade do software pode ser medida por uma fórmula, mas pode depender de inúmeros fatores externos ao desenvolvimento e à equipe de desenvolvimento do projeto. Um dos fatores que pode influenciar na qualidade do produto de software é a quantidade de recursos disponíveis para o desenvolvimento ou o tempo disposto em razão do tamanho do escopo. Por problemas nesta razão a qualidade pode não conseguir ser calculada da mesma forma que em situações comuns. Como dito na seção de Engenharia de Requisitos, onde as fases de elucidação, negociação e validação de requisitos devem existir no processo de software, o escopo precisa ser negociado em razão da restrição de recursos do cliente, seja ela financeira ou de tempo.

8 Startup

O termo startup, utilizado por Steve Blank (14) em seu livro, diz que startup é um momento temporário de empresa que busca um modelo de negócios repetível, lucrativo e escalável, entregando valor ao usuário a cada nova versão. Entende-se por repetível e escalável um modelo onde a startup poderá, com a mesma estrutura física ou de sistemas, atender quantidades exponenciais de usuários sem ter que dobrar sua estrutura a cada vez que dobrar a quantidade de usuários. Esse modelo se torna extremamente lucrativo quando a equipe que o desenvolve consegue perceber e atender os seus usuários e ou clientes de forma certeira.

Para que uma startup consiga obter sucesso em sua jornada, é necessário que o software que compõe o produto seja o mais próximo possível das necessidades do usuário e que isso seja desenvolvido com o menor tempo e menor margem de erro possível. É de extrema importância que o escopo a ser desenvolvido seja o mínimo e seja também o mais próximo do que se pode chamar de *killer feature* dessa startup, para garantir o engajamento máximo dentro do processo de validação.

Existem diversos tipos de startups e vários momentos em que elas pertençam, como ideação: onde ela ainda é uma ideia de projeto e que a equipe ainda está analisando quais as necessidades dos usuários elas estão buscando.

O momento dessas startups avaliadas neste trabalho está na fase de desenvolvimento da solução tecnológica que envolve software e alguma interação com usuários, como aplicativos, sejam eles: mobile, web ou desktop.

Muito embora as startups sejam empresas comuns, que tem como base uma aplicação tecnológica, é importante perceber que por conta da busca pelo modelo de negócios ideal, essa empresa precisa de um resultado o quanto antes, impedindo que um produto necessite estar totalmente pronto para validar o seu modelo. Por isso é de suma importância que o produto seja desenvolvido com o mínimo de recursos, mas suficiente para conseguir solucionar o problema a que se propõe.

9 Story Mapping

Para ajudar no processo de desenvolvimento dos requisitos de software pelas startups, foi utilizado a técnica de Story Mapping para que não só a equipe de desenvolvimento participasse do processo, mas também todos os stakeholders envolvidos no desenvolvimento do startup.

O Story Mapping foi criado por Jeff Patton (15) para entender como os usuários se comportavam com relação ao software que seria desenvolvido. Dessa forma, além de entender o processo do usuário, também se entendia o que de fato deveria existir no software a fim de atender as necessidades do usuário final.

O processo, que fora criado em 2008 especificamente para atender uma demanda de entendimento de requisitos do software Mad Mimi[1], criou um novo processo de descoberta e análise de requisitos de software. Este consiste em montar um quadro com todo o fluxo do usuário dentro do sistema de software e assim, gerar uma jornada dos usuários, com o mesmo princípio do diagrama de casos de uso da UML.

O Story Map possui uma estrutura linear de jornada, chamada de *backbone*, mas com detalhamento que torna o processo em três dimensões, uma vez que cada um dos pontos da jornada tem seu detalhamento e sua priorização logo abaixo:

A primeira etapa consiste em escrever em cada cartão os *goals* ou pontos-chave da história, bem como os usuários afetados pelo processo e estes se posicionam no topo do processo. Da mesma forma, enumerar quais são os usuários que estão envolvidos na história. Quem são as personas que utilizarão o software? Como elas se comportam? Quais as principais atividades delas no processo?

A segunda etapa é de externar qual o fluxo ótimo da aplicação. Onde se começa o uso e onde é o ponto final, tal como um algoritmo. Com o máximo de detalhes possível no sentido linear, sem descrever ou detalhar percalços ou *else-situations*. Os cartões foram colados lado a lado preenchendo uma linha de sequência com os cartões na parede. Neste ponto, está pronto o *backbone* do story map. A partir desse ponto, começa a terceira etapa onde é feito o detalhamento de cada um dos passos realizados no *backbone*. Todas as interações, caminhos alternativos e também questões de arquitetura necessárias para que o software funcionasse perfeitamente com aquela história, sempre em ordem de prioridades, de cima para baixo.

[1] http://madmimi.com

Na parte final do da sessão é traçada uma linha horizontal abaixo do nível de detalhamento que definirá qual a versão que será desenvolvida como primeiro *release* do software.

10 Métricas de qualidade

Para Qualidade de Software, existem diversos padrões de qualidade, no entanto este trabalho foca no padrão ISO 25010:2011[1] com padrão de qualidade de software e também o conceito de Brooks (16) para definição de fatores de qualidade. Esses dois padrões foram escolhidos pelo autor por se tratarem de métricas qualitativas de software que não levam em conta somente o código desenvolvido, mas também o produto que o usuário tem em mãos.

O padrão 25010:2011 é uma revisão do padrão ISO9126[2] que foi desenvolvido para atender alguns fatores de qualidade que envolvem também o usuário final, como forma de criar um padrão para todos os produtos com relação ao que ele deve seguir para obter sucesso no mercado. O padrão é dividido em dois vieses: Produto e Qualidade no Uso. Nesta pesquisa abordamos somente as métricas de qualidade de uso pois tratamos apenas a abordagem de manuseio por parte do usuário e a qualidade que o software deve conter para que este seja contemplado com uma qualidade mínima para o usuário.

Os critérios de qualidade de uso são listados abaixo:

1. Quanto à eficiência e eficácia
    - Quanto à efetividade: atributo de qualidade que mede em quanto tempo e quão simples o usuário consegue cumprir seus objetivos e atividades;
    - Quanto à eficiência: atributo de qualidade que mede a eficiência do produto através da razão entre os recursos despendidos para a taxa de precisão e a totalidade dos usuários que conseguem atingir seus objetivos dentro do produto;
2. Quanto à satisfação
    - Utilidade: medida de percepção do usuário quanto ao que ele conseguiu completar das suas atividades e a facilidade para executar tais tarefas
    - Confiança: medida de percepção que avalia quanto o usuário confia em executar suas atividades no sistema e se o produto irá se comportar como esperado durante toda a experiência;
    - Prazer: medida de percepção sobre a realização do usuário e satisfação pessoal de acordo com suas necessidades dentro do produto
    - Conforto: medida de percepção sobre conforto físico do usuário ao utilizar o produto;

[1] https://www.iso.org/standard/35733.html
[2] https://www.iso.org/standard/22749.html

3. Quanto a estar livre do risco de uso

    - Mitigação de risco econômico: medida de percepção e sensação de proteção do sistema quanto ao risco que o usuário corre no que envolve riscos financeiros, eficiência na operação, propriedade comercial, reputação ou outros recursos no contexto de uso apropriado
    - Mitigação de risco de saúde e segurança pessoal: medida de percepção e sensação de proteção do sistema quanto a potenciais riscos pessoais no contexto de uso apropriado;
    - Mitigação de risco ambiental: medida de percepção e sensação de proteção do sistema quanto a potenciais riscos de danos ambientais no contexto de uso apropriado;

4. Cobertura de contexto

    - Completitude de contexto: medida de sensação de eficácia, eficiência, estar livro do risco de uso e satisfação em todos os contextos de uso do produto;
    - Flexibilidade: medida de sensação de eficácia, eficiência, estar livre do risco de uso e satisfação em todos os contextos de uso do produto, inclusive além dos especificados nos requisitos de software

A especificação ISO 25010:2011 abrange tanto a qualidade do produto em si, quanto a qualidade no que diz respeito à interação do usuário com o produto de software. Para este trabalho, as medidas iniciais trabalhadas são (1) para produto: funcionalidade, usabilidade e segurança e (2) para qualidade de uso serão utilizadas todas as medidas. A não utilização das outras medidas não é uma recomendação ao não uso, mas porque não cabe no escopo deste trabalho que são medidas ligadas diretamente à experiência do usuário com relação ao software.

11 Dilema da qualidade

Em Pressman (9), há uma discussão sobre o que é um software suficientemente pronto para ir para o mercado. Um software não pode pecar pela qualidade, pois assim nenhum usuário iria conseguir usar e nenhum cliente iria querer comprar. No entanto, não é possível passar um tempo infinito tornando o software perfeito. Isso demandaria muito tempo e muito dinheiro. Ainda na obra de Pressman, há uma entrevista de Bertrand Meyer onde sugere que haja a aceitação por produzir o software "bom o suficiente". E por software "bom o suficiente", pode-se entender que é um software que atende aos requisitos de qualidade e também às especificações do software.

O software "bom o suficiente" pode atender o público — quando bem especificado — com tarefas básicas que o atenderão o usuário. Com um bom time de marketing, seria possível atingir o público ainda assim e vender este software em sua versão 1.0 e com as vendas e os aprendizados, poderá ser desenvolvida a versão 2.0, com as correções dos problemas e também as funcionalidades restantes que compõem o software.

No entanto, Pressman defende que não é uma alternativa viável quando se é uma empresa pequena que não possui de recursos para recuperar reputação caso haja um erro grave no produto e não havendo tempo para correção, podendo até mesmo fechar prematuramente a empresa ainda na versão inicial lançada.

O dilema da qualidade é, na linguagem da Engenharia de Software, um dos grandes dificultadores do que pode ser o produto "bom o suficiente"para ir ao mercado. Esse é o princípio do MVP, o produto mínimo viável que é apresentado a seguir.

12 Minimum Viable Product - MVP

Em startups, existe o conceito de MVP ou Minimum Viable Product que é uma versão inicial que é possível colocar no mercado sem que necessariamente o produto esteja totalmente acabado. O termo MVP foi cunhado por Frank Robinson (17) para definir a mínima porção de um produto que deve ir ao mercado o quanto antes para minimizar o esforço de teste com usuários e obter o mais cedo possível, os feedbacks com relação ao que deve ser modificado no desenvolvimento do produto.

Com o processo de criação do MVP, a startup pode validar com menor esforço o produto e ter retorno, em caso de venda mais cedo e mais barato. Esse esforço também pode ser convertido em risco, quando envolve um retorno de investimento mais alto e o esforço para cria-lo também é. A Figura 1 mostra o que deve ser um MVP típico. Ele deve possuir um pequeno esforço e uma taxa de Retorno de Investimento (ROI) alta. Este retorno de investimento é o quanto de retorno, a startup pode trazer aos investidores (incluindo o investimento próprio dos fundadores) em valores financeiros ou econômicos em proporção ao tempo em que esse produto é entregue para gerar esse valor. Se esse esforço passa a ser alto demais, haverá menos tempo para validação com os usuários e o retorno mesmo sendo alto, também haverá um risco muito alto. Esse risco, assim como o ROI, pode ser financeiro ou econômico.

Neste trabalho, a análise do processo de elucidação de requisitos acontece através da criação do primeiro MVP de cada startup estudada no processo. A análise de requisitos é feita através de validações das necessidades com base no conceito de cada startup. Este conceito é feito de forma prévia, onde cada startup já tem uma mínima ideia do que ele quer desenvolver e chega para o processo apenas com o intuito de definir o projeto a ser desenvolvido e não entender quais são as necessidades que este ainda contemplará.

O trabalho também busca resolver o dilema da qualidade através dos critérios de sucesso, visto mais à frente nos laboratórios e o software suficientemente pronto dentro da redução do escopo e a definição de pronto de cada lançamento de versão do produto de software.

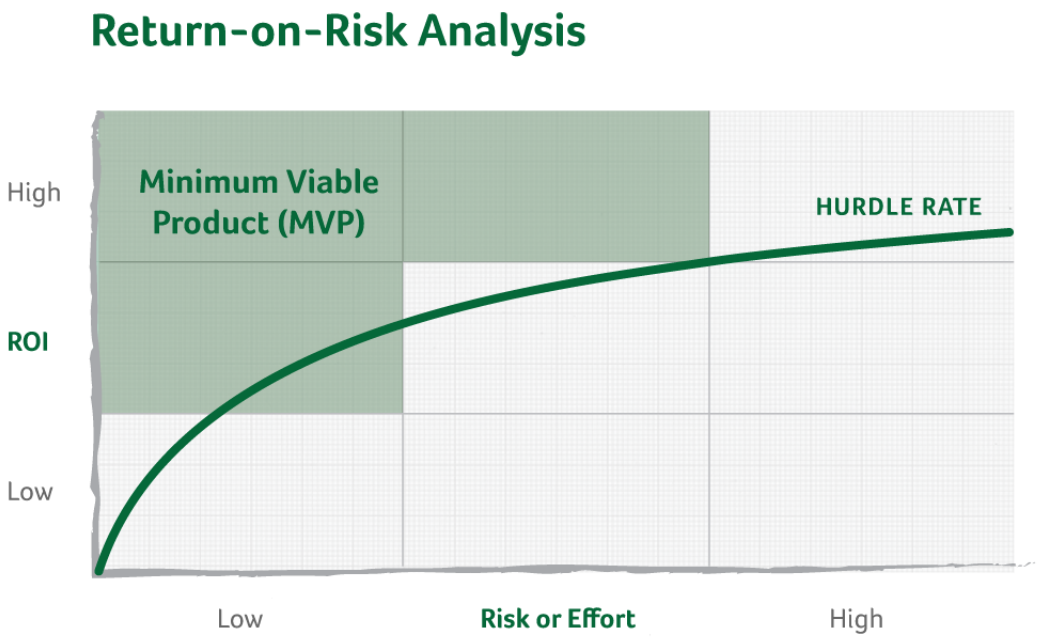


Figura 1 – Análise de risco de um MVP

Parte III

Trabalhos correlatos

13 Trabalhos correlatos

Confrontando os trabalhos relacionados trazidos e descritos com a referente pesquisa, alguns trabalhos considerados importantes foram utilizados como ferramenta ou modelo de trabalho para o estudo de caso contido neste trabalho. Alguns trabalhos trazem métodos inovadores para o desenvolvimento de software, como o processo de Design Thinking de Grosskopf (7) onde todo o processo de Engenharia de Software é revisto utilizando o modelo de Design Thinking. No entanto, esta abordagem não traz nenhuma garantia quanto ao que deve ser entregue ao usuário com relação ao software. Somente com relação ao produto.

Mesmo quando se trata de métodos ligados ao desenvolvimento de software, como Frye menciona em seu trabalho (18) utilizando Scrum como metodologia de gestão de projetos, o ponto de vista é o gerente de projetos ou o cliente. Este trabalho busca trazer não só o gerente de projetos ou o cliente como também o usuário. Como dito no Manifesto Ágil (3), as pessoas são a preferências do foco do desenvolvimento ao invés dos processos em si. O trabalho traz também um foco em desenvolvimento mais simplificado que um processo de desenvolvimento com Scrum, como no trabalho.

Em métodos que buscam reduzir o escopo do que é desenvolvido, como o Lean (19), a redução do desperdício com relação ao que deve ser desenvolvido é sempre relacionado aos processos internos da empresa que desenvolve, não necessariamente ao pensamento do software mínimo, não levando em conta a validação do MVP logo no inicio do processo. Mesmo assim, a proximidade com a obra de Poppendieck é notória, visto que o ponto mais importante deste trabalho é gerar o mínimo de desperdício com o máximo de eficácia no software desenvolvido.

Assim, as três obras citadas formam a base deste trabalho, que gerou um estudo de caso em cima de processos de startups especificamente, utilizando o menor esforço possível dentro do processo com o máximo de pessoas envolvidas no desenvolvimento deste software, além de, também, possuir o máximo de informações sobre os usuários deste produto.

Parte IV

Desenvolvimento do modelo

14 Introdução

O processo de desenvolvimento do escopo de software das startups neste trabalho é acompanhado de etapas como laboratório e monitoramento do desenvolvimento dos produtos e análise dos resultados dentro do processo de concepção, implementação e validação do modelo. Esses processos são descritos em: (i) Busca pelas startups alvo do objeto de estudo, (ii) Realização do laboratório de requisitos, (iii) Monitoramento das entregas, (iv) Validação dos critérios de sucesso do produto pelo usuário e (v) Validação do modelo propriamente dito.

O objetivo neste ponto do trabalho é desenvolver produtos iniciais de startups que já possuíssem uma clara noção do problema que estavam dispostas a resolver, mesmo ainda não tendo nenhum produto nas mãos. Para isso, foram pré-selecionadas 10 startups que se inscreveram em um programa de residência e mentorias de startups em estágio de ideação para que fossem selecionadas para o processo.

Ao final, foram escolhidas cinco startups por já possuírem tanto componentes com conhecimentos técnicos suficientes para o desenvolvimento do projeto, bem como a startup ter passado por um processo de ideação mais aprofundado. Cada uma das dez startups avaliadas passou por uma entrevista sobre os conhecimentos sobre os conceitos de startup, desenvolvimento de produtos e projetos e também foram analisadas sob o aspecto do estágio em que estavam com relação à consciência da problemática que buscavam resolver com a startup. Essa etapa foi fundamental para evitar que a startup com pouco material de subsídio sobre seu usuário final e também sobre o mercado ou sobre o real funcionamento do processo de uma startup entrasse em um processo mais avançado para elas, como o desenvolvimento de um projeto.

A fase seguinte foi de execução dos laboratórios descritos no trabalho. Estas startups passaram por três laboratórios para elucidação de requisitos e em seguida foi feito o acompanhamento do desenvolvimento dos projetos que elas criaram. Dessa forma, está descrito o processo realizado abaixo:

Para o desenvolvimento desta pesquisa, as cinco startups no momento em que estavam a começar o desenvolvimento do software que seria seu produto e dentre as cinco, duas startups estavam com o desenvolvimento pronto, mas precisaram voltar ao planejamento pois não tinham alcançado o mercado de forma adequada, uma vez que os usuários não tinham sido alcançados.

Para as duas startups que precisaram de novo desenvolvimento, neste trabalho chamaremos de R1 e R2 e as três startups que estavam desenvolvendo pela primeira vez, chamaremos de S1, S2 e S3 por motivos de sigilo das empresas.

Estas startups foram acompanhadas durante 3 meses através de um processo de entendimento de necessidades, laboratório de requisitos com o apoio da ferramenta de story mapping e de acompanhamento até o mercado para avaliar o feedback dos usuários. O processo foi refeito 3 vezes até que todos os testes fossem satisfatórios. O processo com as startups se deu basicamente em três momentos:

- foi criado um momento de conversa com os integrantes dos projetos em que apenas se coletava informações sobre o negócio, mercado e sobre a atuação profissional dos integrantes. Esta etapa é apenas informativa e não consta no processo como parte do estudo
- as equipes passam por um laboratório de 16 horas em que através de alguns passos, desenvolviam o documento de requisitos ao final do processo através de uma jornada de projetos;
- as equipes foram acompanhadas e monitoradas quanto ao desenvolvimento do escopo por elas produzido até o momento de entrega ao usuário final, que por sua vez tinha a validação realizada sobre o que os atendia ou não.

Este processo se repetiu com todas as startups e após isso, os dados foram coletados, agrupados e qualificados. Os três laboratórios realizados tiveram um intervalo de cinco meses entre um e outro, possibilitando que as startups desenvolvessem os seus projetos por três meses e entregassem ao usuário final para validação, além de permitir criar as mudanças dentro do processo do laboratório.

15 Laboratório de projetos

Foram realizados três laboratórios para desenvolvimento deste trabalho completo. A organização do laboratório de projetos é dada da seguinte forma: Os facilitadores são uma espécie de guardiões do processo realizado no laboratório, onde estes farão a condução do processo para que as startups possam desenvolver suas atividades da melhor forma possível no laboratório. Um facilitador faz a condução com a apresentação, dinâmicas e mentoria durante o processo de learn by doing das startups, utilizando os materiais da Tabela 1.

Tabela 1 – Materiais utilizados no laboratório

Numero	Função
2	Facilitador
1	Projetor e TV
4	Startups, contendo ao menos dois desenvolvedores e um responsável pela gestão do produto
4	Blocos de cartões colantes (um de cada cor) para cada startup

O outro facilitador em sua maior parte, anota o tempo de cada processo, a evolução de cada startup, números do desenvolvimento. Tem um papel de ajudante do facilitador principal no processo.

O projetor e a TV servem para guiar visualmente através de uma apresentação de slides (anexo) onde os facilitadores demonstram os próximos passos dentro do processo com as startups.

As startups precisam participar durante as 16 horas de laboratório, sendo este dividido em dois dias consecutivos de 8 horas cada. Para a participação, é de total importância que estejam presentes a equipe de desenvolvimento e também a equipe que desenvolve o produto e faz a interface direta com o cliente, mesmo que os times sejam de uma pessoa só. Essa comunicação interna é importante pelo fato da necessidade de entendimentos sobre o que deve e o que não deve ir para as mãos dos usuários e como isso deve chegar às mãos dos usuários.

Figura 2 – Apresentação de uma das equipes no primeiro laboratório

16 Conceito do projeto

Dado o início do laboratório, as startups precisam iniciar o processo conceituando o que as suas startups fazem. Em cada startup haverá uma discussão de cerca de 1 hora onde elas definirão os seguintes pontos:

1. O que é a startup: a equipe criará uma definição onde trará a percepção do que é a solução que elas estão trazendo para o mercado e qual o seu conceito. Em um cartão deve ser suficiente para esta etapa
2. Objetivos: quais os objetivos da startup dentro do mercado? Baseado no "O que é a startup?", como que ela pretende chegar na solução, quais os objetivos dela com relação à solução dela no mercado. Os objetivos podem ser vários. No entanto, no laboratório se encoraja a chegar em no máximo três para se reduzir o escopo do que deve ser feito. Esse exercício já inicia o processo de redução do que deve ser desenvolvido na startup.
3. Diferenciais: neste quadro, a equipe precisa buscar no mercado o que já existe e quais as formas que as pessoas buscam para resolver o problema que a startup está se propondo. Parte-se do pressuposto que mesmo que exista um problema a ser resolvido, deve existir também uma solução — mesmo que improvisada — das pessoas para aquele problema. Nesta etapa, as equipes listam todos os seus diferenciais de mercado. Esses diferenciais devem ser palpáveis e menos subjetivo quanto possível, a fim de evitar duplo entendimento sobre os mesmos. Durante o laboratório se encoraja a se obter o máximo de diferenciais possíveis, pois isso fará com que a startup tenha opções de priorização daquilo que querem resolver.
4. Serviços e componentes: Nesta etapa, as startups precisam preencher um quadro que é subdividido em dois, onde se trata das funcionalidades da forma mais subjetiva possível e em linguagem natural daquilo que os usuários precisam alcançar no produto. Deve-se descrever quem é o usuário atingido e qual serviço ele vai receber. Devem ser listadas funcionalidades que atendam aos objetivos e diferenciais sem se deslocar daquilo que é a startup. O que significa que todas as etapas são interconectadas, a fim de criar uma sequência lógica na mente das equipes. Nesta etapa, as startups colocam em cada cartão uma funcionalidade-chave e para quem ela se destina.

Esta etapa, ao ser finalizada, mostra para a própria equipe o que ela quer desenvolver. É um processo de nivelamento do entendimento dentro do próprio projeto, com o objetivo de que ao chegar no processo de requisitos, estejam todos entendendo da mesma forma o projeto.

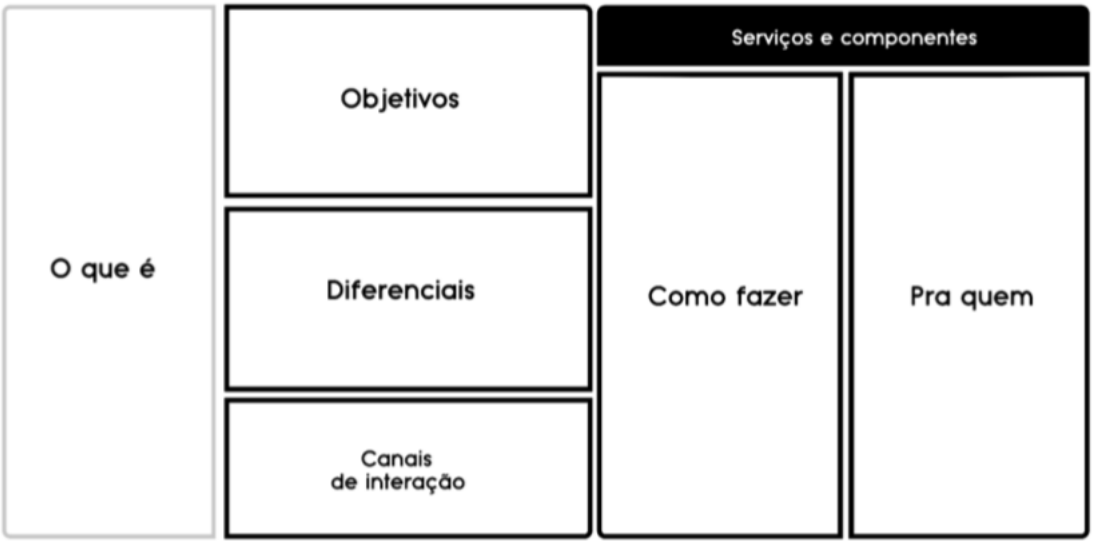


Figura 3 – Diagrama de conceito de produto

17 Requisitos com Story Mapping

Para a segunda etapa do laboratório, iniciamos o processo de requisitos utilizando a técnica de Story Mapping. Nesta etapa, as equipes precisam trabalhar no processo de contar o fluxo dos usuários como uma história linear, onde cada passo deve ser contado em um cartão colado na parede da esquerda para a direita.

O processo se inicia através da importação das informações que foram feitas na etapa anterior. Com base nos dados sobre os Serviços e Componentes que foram inseridos, traz-se os cartões para serem transformados em Goals do Story Mapping, bem como os usuários são importados para que se transformem nos atores do processo da jornada. Esta etapa leva cerca de 1 hora, pois as startups começaram a repensar quais mais atividades podem ser chave para a startup. O processo instiga os componentes a sempre repensar o que estão fazendo.

Quando as startups conseguem finalmente externalizar os usuários as atividades-chave, o processo passa a ganhar a nova etapa, que é criar o backbone do Story Map. Os integrantes nesse momento, passam a contar uma história como um fluxo linear do processo na startup desde o inicio da utilização até o fim do ciclo, passando por todos os usuários e todos os goals que o projeto possua, a fim de que todos os processos dentro do projeto fiquem visíveis, mesmo que em uma camada superior de entendimento. O processo de construção do backbone do Story Map dura cerca de uma hora, com mentoria e perguntas para os componentes das startups para que estes possam perceber mais etapas no processo.

A partir deste momento, as startups começam a trabalhar no detalhamento de cada uma das etapas. Durante esta etapa acontecem também as discussões sobre o que realmente deve ser feito em termos de produto e em termos de tecnologia dentro de cada uma das etapas da história. O laboratório fez as startups perceberem o quanto produto e tecnologia ainda estão distantes sobre a profundidade de compreensão sobre o que é necessário entregar ao cliente, pelo lado do produto e quanto deve ser feito quando tratamos de tecnologia.

O detalhamento dura cerca de 4 horas e é monitorada por mentoria, instigando os componentes a criarem mais detalhamentos sobre como deve funcionar cada etapa, pensando também em situações alternativas da história, como situações if-else de programação ou com gráficos que ilustrem como deve funcionar cada etapa. O processo de detalhamento contemplou todas as possibilidades que as startups conseguiram projetar em mente durante o laboratório.

Enquanto o processo era montado, o time de produto também conseguia validar se aquela lista de detalhes e restrições era viável do ponto de visto dos objetivos da startup.

Ao final do primeiro dia, as startups tinham o processo de Story Mapping concluído com todos os detalhamentos sobre o produto e sobre a tecnologia que era necessária para desenvolver o projeto descrito nos cartões. As paredes com os cartões afixados teriam similaridade com a imagem abaixo:

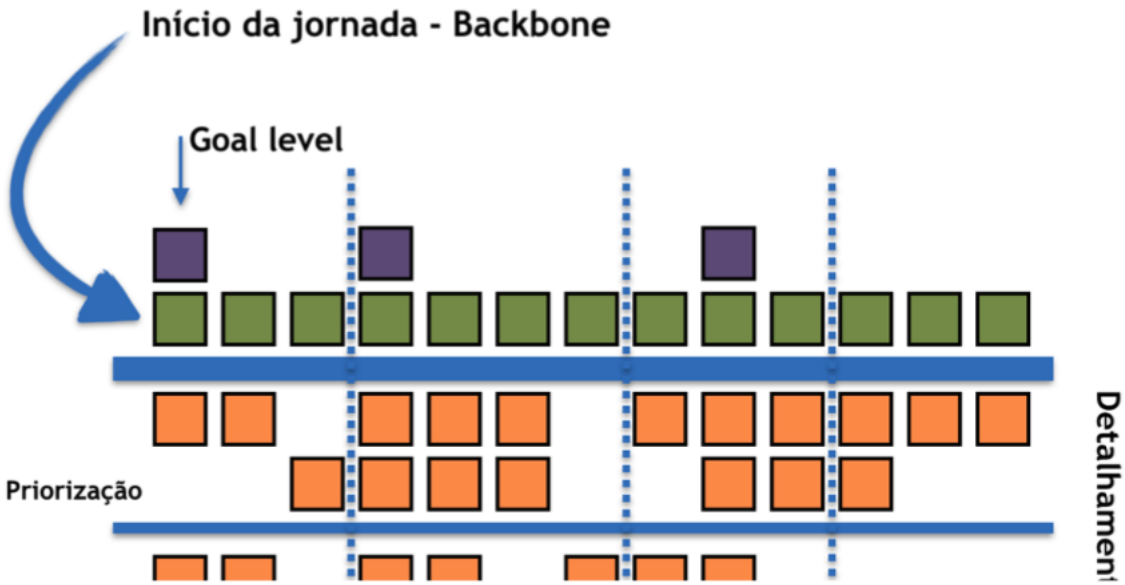


Figura 4 – Aspecto visual de um Story Mapping

Ao iniciar o segundo dia do processo, as startups tem como primeira missão, revisar todo o Story Map construído e analisar o que pode ter escapado no dia anterior e continuar as discussões do dia anterior, a fim de trazer novas ideias de um dia cansativo para o outro. Durante os laboratórios, ao serem questionados em qualquer um dos passos da jornada do Story Map, as startups sempre fizeram uma nova discussão — na maioria das vezes inédita — sobre o detalhamento da jornada. Esta tarefa de revisão é espontânea e dura cerca de uma hora.

Após este processo, começa-se a priorizar o que é mais importante para que cada etapa do backbone seja concluída, observando também os objetivos, diferenciais e o conceito da startup. O processo compreende de analisar todo o detalhamento que foi criado e priorizar quais as atividades mais importantes para que o projeto tenha sucesso. Este processo tem duração mínima de duas horas, mas nos dois laboratórios ela foi até o fim da manhã, principalmente nos casos em que as equipes eram maiores e a quantidade de cartões no detalhamento era maior.

O processo de priorização ao seu final, traz para mais próximo do backbone, os cartões que são mais importantes para que o projeto possa atingir os objetivos, respeitando os diferenciais. Assim, logo que as startups terminam a priorização, é traçada uma linha horizontal algumas linhas abaixo do backbone (o padrão para os projetos eram cinco linhas abaixo do backbone). Tudo que fora colocado entre o backbone e a linha traçada passa a se tornar a primeira versão lançada.

Da mesma forma, são traçadas outras linhas para demonstrar os próximos releases dos projetos. Da mesma forma como foi criado o cronograma de priorização do projeto para o primeiro release, também é feito para os demais. Mesmo que a priorização inicial tenha acontecido para a primeira etapa, as subsequentes passam por uma repriorização para adequar àquilo que elas pretendem lançar aos usuários em seguida ao lançamento da primeira versão. Dessa forma, o plano de entregas das startups está concluído. Dentro do laboratório é importante frisar que os releases que são feitos após os primeiros podem sofrer alterações severas pelo fato de serem resultantes de feedback dos usuários que terão uma versão (a primeira versão) em mãos.

Figura 5 – Imagem de um Story Mapping criado em laboratório

18 Priorização de escopo e validação

As quatro últimas horas do laboratório são dedicadas ao fechamento das especificações dos requisitos, onde as tarefas do detalhamento são colocadas dentro de cartões previamente montados, com ponto descritivos para o uma *user story*. Uma *user story* é basicamente um cartão com um requisito escrito e descrito com a visão do usuário. Neste cartão deve conter um nome para a estória, um usuário como *persona* ou um usuário identificado, um objetivo claro que ele deve atingir com aquele requisito e quais as tarefas do usuário que devem ser completadas para que aquela estória tenha todos os requisitos atendidos. Essas *user stories* servem para compor o backlog ou a coleção de funcionalidades que serão desenvolvidas.

Esses cartões devem ser inseridos em outro quadro, chamado quadro de projetos (*Project Board*), onde as *user stories* são colocadas já priorizadas para fins de organização do que será desenvolvido tanto para gestão de projetos como na gestão de produto, considerando sempre que o software em questão é um produto.

No cartão de *user story*, se deve preencher o nome da estória que é a referência que aquela estória tem e a melhor forma de nomear é o ponto da história do backbone do story map. Em seguida é preenchido o usuário afetado dentro do story mapping o usuário que é alcançado no processo. Caso sejam dois ou mais usuários, a *user story* deve ser desmembrada para cada usuário, deve-se criar um cartão para cada objetivo de usuário.

Figura 6 – Cartão de user story

Um ponto importante que fora acordado no laboratório para a priorização dos backlogs, era analisar a tabela de conceito (ver Figura 3) e ver quais são os cartões que mais rapidamente atenderão os Objetivos e Diferenciais e quais os custos (financeiros ou

econômicos) para que aquelas tarefas sejam realizadas em menos tempo e o software tenha o maximo de aproveitamento quando for para o mercado.

Com todos os cartões priorizados transformados em cartões, as startups tem a tarefa de realizar a transferências dos cartões para o *Project Board* que é o quadro onde será feita o acompanhamento do que está sendo desenvolvido, no sentido macro da metodologia.

A priorização do que será desenvolvido segue um padrão que permite que a startup possa definir não só o que será desenvolvido como também quando estará pronto e também quais os critérios de avaliação para considerar que aquela versão entregue obteve sucesso quanto aos usuários que irão trabalhar com aquele produto.

Cada startup após a preencher o Backlog Priorizado tem agora que interpretar o que vai ser entregue ao final do período de desenvolvimento. Duas coisas são necessárias neste passo: Preencher a *Definição de Pronto* e *Critérios de Sucesso* que definem as entregas e quais serão as métricas da startup para cada entrega.

Como *Definição de Pronto*, entende-se por quais são os critérios para que aquela versão esteja de fato pronta e que não há mais nenhuma necessidade de modificações ou de aperfeiçoamentos naquele momento do desenvolvimento. Esta definição ajuda a startup sobre o momento de parar o desenvolvimento e considerar que aquela versão é aceitável do ponto de vista de validação do usuário. A startup poderá definir mais de uma definição de pronto. Quanto mais definições de pronto complementares, melhor declarados estarão essas definições. Para a definição de pronto, trata-se como uma definição na Gestão de Projetos da startup.

Para os *Critérios de Sucesso*, são definidas as medidas que serão utilizadas para validar aquela versão com o usuário final que ao receber o produto poderá utilizar, validar e retornar informações para a startup, que colherá os feedbacks e irá avaliar se aquela versão teve sucesso quanto ao seu usuário ou não. Os *Critérios de Sucesso* podem ser múltiplos, assim como a Definição de Pronto, no entanto, o custo para ter múltiplos *Critérios de Sucesso* é maior, visto que para cada critério, é necessário que a startup crie métricas específicas que elas nem sempre possuem prontas para sua primeira versão, principalmente. Para os *Critérios de Sucesso*, trata- se como uma definição na Gestão de Produtos da startup.

Com este processo, o laboratório é encerrado e começa-se então a etapa de acompanhamento das startups. Cada semana foi feita uma reunião de retrospectiva sobre o que havia sido entregue e o que estava atrasando. As startups começaram a desenvolver seus produtos de acordo com o escopo estabelecido no *Backlog Priorizado* e com suas metodologias favoritas de microgerenciamento de projetos para que este não atrasasse a entrega. Assim persistiu durante as 14 semanas seguintes.

19 Aplicação de métricas de qualidade

Cada Critério de Sucesso precisava ser acompanhado de métricas que precisam ser aplicadas ao usuário. Algumas startups, mesmo apresentando alguma dificuldade para definir o que seria a validação com os usuários, conseguiram criar os critérios e assim foi percebido algum padrão aplicável a todas elas inconscientemente.

A maior parte dos critérios de sucesso teve como base a relação do usuário com o produto, como o sucessso na realização de determinadas tarefas, como a geração de um relatório específico pedido por usuários ou um *output* desejado pelos usuários em determinado ponto de uso do sistema. Essas métricas devem atender as políticas de Métricas de Qualidade, conforme o capítulo 10. Estas métricas não foram inseridas compulsoriamente, mas foram sugeridas de acordo com o critério de sucesso que cada startup cogitou para seu modelo de negócios.

A análise dessas métricas se deu especificamente através de validação e testes dos envolvidos no processo com os seus usuários. O grau de aprovação fora definido também por elas.

Parte V

Validação do método

20 Introdução

Após o laboratório em que as startups montaram seus planos de requisitos e definiram o escopo de desenvolvimento do seus projetos, cria-se o momento onde deve-se validar se essa elucidação de requisitos de fato fez sentido no que tange o alcance ao usuário final e a resolução dos problemas que o produto se propôs a resolver. As etapas seguintes são consideradas:

1. Primeiras semanas de desenvolvimento, onde as startups de fato desenvolveram os MVP que colocariam no mercado, sempre respeitando a definição de pronto ora proposto;

2. O desenvolvimento foi interrompido no momento em que a definição de pronto seria atingido e assim, a etapa de desenvolvimento estava concluída para que a versão fosse colocada no mercado para validação dos critérios de sucesso;

3. Com o software no mercado, os critérios de sucesso devem ser validados com os usuários, conforme os critérios de qualidade do capítulo 10;

4. Ao final do período de validação, foram conferidos se cada um dos critérios de sucesso fora validado e se estes estavam de acordo com o que a startup validou.

Os dados foram compilados de acordo com as *user stories* de cada startup e a sua relação com os critérios de sucesso, tamanho do escopo e se as *user stories* foram realmente desenvolvidas para alcançar os critérios de sucesso. O modelo completo foi repetido mais uma vez para um segundo lançamento. O critério de avaliação dos critérios de sucesso é a composição de conformidade com objetivos, diferenciais e serviços e componentes do produto conceitual, seguindo a Tabela 16. A validação deve ser feita pela própria equipe.

Para o processo inteiro de desenvolvimento dos requisitos até a validação do método, se passaram 14 semanas entre o primeiro laboratório e o segundo, que envolveu na maioria das startups, uma continuidade.

O método se mostrou bastante eficaz para a equipe de desenvolvimento, pois além de gerar uma maior visibilidade dos requisitos, houve um aproveitamento alto das especificações que conseguiram ser desenvolvidas e entregues ao usuário. Houve uma pequena taxa de descarte de especificações por mudanças ao longo dos projetos em todas as startups, através de um taxa média de 13%, o que manteve o grau de motivação da equipe de desenvolvimento e menos interrupções sobre o que deveria ser desenvolvido. Quanto aos critérios de sucesso, houve um aproveitamento alto, visto que as startups

conseguiram em sua maioria, atingir os critérios de sucesso por elas proposto, conforme a Tabela 2.

Tabela 2 – Primeiro laboratório, total de user stories

	Backbone	Detalhamento	US Total	US desenvolvidos	US descartados	Aproveitamento
R1	76	192	123	64,06%	20	83,74%
R2	52	129	102	79,07%	4	96,08%
S1	58	153	101	66,01%	23	77,23%
S2	48	120	88	73,33%	10	88,64%
S3	24	67	48	71,64%	3	93,75%

Leva-se em conta o grande aproveitamento das *user stories* devido ao entendimento do todo pelas equipes. Ao criar o Story Map, os membros das equipes passam a entender o todo e não ficam brechas obscuras sobre o que o projeto deve ou não deve ter. Em um formato de laboratório, torna mais encorajador ter o processo de discussão entre todos os membros de forma igualitária, com as possibilidades de haver nãos, resolução de conflitos quanto a entendimentos e assertividade quanto ao que vai ser desenvolvido.

Quanto aos usuários que recebem a aplicação ao final do processo, leva-se em conta que ao mesmo tempo que o Story Mapping auxilia no processo de entendimento do todo, também é importante saber que as startups estão priorizando requisitos que estão ligados diretamente ao que eles se conceituam, tem como objetivo e criaram na jornada do usuário. Se torna mais simples entender o que a startup é e o que ela entrega ao usuário para que se saiba exatamente o que deve ser entregue ao usuário. A validação se tornou maior quando foi percebido que houve uma pequena taxa de requisitos que voltaram ao processo de entendimento ou de refinamento para ser refeita ou entendida novamente. Isso mostra que as startups conseguiram entregar funcionalidades que os usuários realmente utilizaram, não descartaram e as startups ainda puderam progredir com o mínimo de prejuízo de recriação de funcionalidades ou de novas interpretações sobre o que os usuários pediam.

21 Coleta de dados

Para coleta dos dados, foram utilizadas cinco startups em fase de projetos — no qual estão iniciando o desenvolvimento do seu MVP — durante um laboratório de 16 horas. Também foi acompanhados o processo de entrega ao usuário e o trabalho de entendimento sobre como o software era lançado para eles. Para cada startup, foram feitas anotações sobre a evolução do desenvolvimento do software e também sobre como este software estava sendo entregue aos seus usuários.

Durante o processo de 12 semanas, as startups avaliaram todos os critérios de sucesso e validaram as definições de pronto. A cada semana de entrega, foi avaliada a definição de pronto. Ao final de cada 2 semanas, era feita entrega ao usuário e esta entrega se avaliou os critérios de sucesso de cada startup de acordo com o que havia sido pensado no início do ciclo de duas semanas.

No final de cada ciclo também é feita uma retrospectiva no qual as validações de critérios de sucesso são feitas. Utilizando o quadro de projetos, é possível inserir as lições aprendidas sobre o que obteve sucesso e o que não obteve sucesso nas validações. Essas validações podem ser feitas tanto no sentido de projetos quanto de produto. A cada lição aprendida, a startup pode preencher com cartões onde foram sucessos atingidos e onde não foram atingidos.

Aqueles sucessos não-atingidos são trazidos para os membros e rediscutido no sentido do que deve ser alterado na jornada através do Story Map, seja na jornada em si ou no detalhamento da mesma.

Em nenhuma das startups, houve uma mudança na primeira fase — no conceito, objetivos ou diferenciais propostos. Somente alterações na jornada, seja no backbone ou no detalhamento dela. O aproveitamento aumentou em cada uma das iterações dos ciclos de duas semanas. O primeiro variou entre 83% a 99% de aproveitamento das stories que foram criadas dentro da jornada, conforme a Tabela 2.

22 Desenvolvimento de métricas

Para o desenvolvimento das métricas das startups com base nas entregas, foi criado um processo dentro do laboratório onde estas startups desenvolviam os Critérios de Sucesso das atividades e assim conseguir atribuir como elas poderiam entregar melhor o software aos seus usuários. As métricas foram criadas para ter critérios subjetivos, dando a possibilidade de serem revistas e a régua ser delimitada de forma dinâmica, atendendo assim como um índice para as startups quanto ao que elas entendem ser necessárias para definir a qualidade do produto de software que estão entregando.

O processo de desenvolvimento das métricas obedeceu aos critérios de desenvolvimento da qualidade do software, como os processos ISO para validação de qualidade.

Parte VI

Apresentação dos resultados

23 Para o laboratório

Os laboratórios foram dedicados a ajudar as startups a encontrarem exatamente quais eram seus requisitos para o desenvolvimento do processo de software. Nesse laboratório, foram definidas as user stories e também o escopo do que deveria ser desenvolvido. Todas as startups conseguiram detalhar, com algum processo de ajuda, os pontos que tornariam facilmente entendidos os requisitos do software. Todas as startups, através de análise empírica, perceberam que o software que pretendiam desenvolver era muito maior do que o que inicialmente pensaram.

Outro resultado que chamou a atenção, foi a quantidade de user stories desenvolvida por cada startup, em que das cinco startups, no mínimo 73% das users stories foram desenvolvidas de fato, o que fez com que a Definição de pronto do MVP fosse muito próximo de atingido. Durante o processo, as startups também passaram por percepções de mudança do escopo, devido a dificuldades de entendimento de algumas tarefas ou de processos dentro do software, como mostra a Tabela 2. Houve, no segundo laboratório, um aumento na quantidade de user stories e uma diminuição na taxa de desperdício das tarefas, caindo pra menos de 10%, como mostra a Tabela 3.

Tabela 3 – Segundo laboratório, total de user stories

	Backbone	Detalhamento	US Total	US desenvolvidos	US descartados	Aproveitamento
R1	92	268	221	82,46%	1	99,55%
R2	68	178	150	84,27%	5	96,67%
S1	62	195	150	76,92%	8	94,67%
S2	75	160	129	80,63%	9	93,02%
S3	53	175	159	90,86%	14	91,19%

24 Para os critérios de sucesso

Os critérios de sucesso anteriormente definido com base nas métricas de qualidade, descritos na Seção V onde as próprias startups definiam com base em seus conceitos, quais as melhores métricas de qualidade para o produto em si, atingiram os resultados através do uso do software e da capacidade de venda do produto no mercado para os usuários. Para cada critério, as startups elaboraram uma lista de requisitos para cada critério, este subjetivo, onde seriam validados ou não.

Com o escopo reduzido, já no primeiro laboratório, os critérios de sucesso definidos, foram alcançados em no mínimo 55% nas startups, chegando a 100% no caso de uma das startups. Esse resultado de 100% foi considerado fora da curva por contar com apenas um terço dos critérios de sucesso da startup que mais definiu critérios e isso é demonstrado na tabela 4. No segundo laboratório, os critérios foram melhor definidos, muito pela experiência da própria equipe ao ter se preparado melhor para o laboratório.

No segundo laboratório (Tabela 5), houve um expressivo aumento na quantidade de critérios definidos, muito por já haver uma experiência dos participantes dentro do processo. A taxa de sucesso também aumentou significativamente, onde, no mínimo houve um aproveitamento das taxas de sucesso em 80%.

Tabela 4 – Critérios de sucesso do laboratório 1

	Quantidade de critérios de sucesso	Sucesso atingido	Sucesso não-atingido	Taxa de sucesso
R1	12	7	5	60%
R2	11	6	5	55%
S1	8	7	1	80%
S2	9	6	3	66%
S3	4	4	0	100%

Tabela 5 – Critérios de sucesso do laboratório 2

	Quantidade de critérios de sucesso	Sucesso atingido	Sucesso não-atingido	Taxa de sucesso
R1	25	24	1	95,83%
R2	20	18	2	88,89%
S1	22	21	1	95,24%
S2	18	15	3	80%
S3	18	16	2	88,50%

Parte VII

Análise e discussões dos resultados

Dado que as startups estavam criando o seu primeiro MVP quando iniciaram o processo e que basicamente elas só tinham a ideia ainda em formação do que deveria ser desenvolvido, o aproveitamento de mais de 80% de tarefas foi considerado extremamente satisfatório. Alguns dos projetos terminaram o processo sem muitas *features* desejadas pelos fundadores, em alguns casos pelo fato de que aquela funcionalidade não ser igualmente desejada pelos usuários no primeiro momento. Essa visão, trouxe o desenvolvimento realmente para o campo da realidade no que diz respeito ao que deve ser desenvolvido e não para o que queremos que seja desenvolvido.

Esse modelo de pensamento criou automaticamente um formato de escopo reduzido, ainda mais após o Story Mapping ter tornado a jornada do usuário muito mais visual e mais fácil de ser mensurada onde tratamos do que é todo o escopo do software.

Existiram severas diferenças de dados entre os dois laboratórios e o acompanhamento das taxas de sucesso dos critérios definidos, no entanto, ao definir um escopo muito menor, com um estudo feito em laboratório, as startups tiveram um tempo muito menor, de cerca de 14 semanas, entre o momento em que definiram os primeiros critérios de sucesso e o segundo laboratório. A taxa de 55% nos critérios de sucesso definidos no primeiro laboratório no diz que mais da metade dos critérios definidos em um escopo extremamente reduzido e que fora desenvolvido em menos de três meses, pôde criar um produto de software usável para o usuário final através dos critérios de sucesso definidos pela startup como métricas de produto.

O ponto observado com maior importância em todo o estudo fora que o alastramento do conhecimento sobre as regras de negócio por parte de desenvolvedores e dos próprios fundadores da startup. Esse conhecimento se mostrou fundamental, visto que no segundo laboratório, com mais critérios de sucesso, já entendendo e conhecendo o usuário final.

A percepção das startups quanto à qualidade dos requisitos também chama a atenção, principalmente no que se refere a taxa de desperdício de user stories escritas durante o processo. O primeiro laboratório, onde as startup não tinham a ideia completa do software que estava construindo, houve um desperdício maior das user stories, taxa esta que baixou drasticamente no segundo laboratório, o que confirma que estas startups não só passaram a entender melhor o produto que estavam desenvolvendo como também que houve uma continuidade no projeto, indicando que estas estavam desenvolvendo o produto certo.

Dentro do que fora planejado para executar os estudos e avaliar se de fato o melhor estudo do escopo permitiria que o produto desenvolvido, teria maior qualidade tanto por parte do resultado de trabalho dos desenvolvedores como pela validação pelo uso pelo usuário final, o estudo se mostrou satisfatório quando tratamos de startups que ainda buscam o modelo ideal de negócios.

No laboratório que fora executado, os próprios times passaram a ter uma melhor observância quanto ao que deveria ser desenvolvido e como deveria ser desenvolvido, bem como o que deveria ser melhor priorizado. Através desse processo, onde todo ele passa a ser visível tanto para que está desenvolvendo quanto para quem gere o projeto ou o produto de software, ficou mais simples de toda a startup saber quais os novos rumos deveriam ser tomados em caso da necessidade de melhorias, mudanças de rumo quando as validações não são supridas e também quando o que foi desenvolvido alcançou o objetivo proposto.

Foi observado também que as startups passaram a utilizar do modelo também nas versões seguintes, quando já não havia a obrigação de trabalhar para entender o negócio. Estas startups utilizaram o processo para se resignificar como negócio sempre que executavam uma etapa de validação, tornando mais simples o processo de criação de novas funcionalidades. Essa continuidade deu sentido ao processo quanto à necessidade de um facilitador para guiar o processo de elucidação de requisitos. Todo o processo pode ser continuado sem a necessidade de um facilitador desde então. Porém, o primeiro laboratório necessita de um ponto focal para conduzir o laboratório e também ser o ponto de apoio do método e também um mediador entre os componentes das startups envolvidas no processo.

Parte VIII

Conclusões e trabalhos futuros

A partir das análises e discussões de resultados, o trabalho demonstrou que o software mínimo pode ser validado e colocado em produção quando os requisitos são bem coletados, analisados e descritos para o seu desenvolvimento.

A metodologia adotada permitiu que a startup entenda melhor quais as necessidades dos seus usuários e assim conquistar a parcela de mercado que ela deseja e também gerar métricas do produto de software com base no sentimento do usuário com relação às suas percepções no uso do software. Por parte das métricas, é possível através do padrão ISO25010:2011, definir métricas subjetivas para entender se o produto de software está atendendo os usuários em cada uma das suas entregas.

Considerando que ao desenvolver os requisitos de software com base no uso do Story Mapping, ao se criar os user stories necessários para elucidar, desenvolver, detalhar e priorizar o que deve ser desenvolvido com base nos conceitos que a startup possui para a contemplação do produto de software.

Foi identificado que a quantidade de cartões do detalhamento no processo de Story Mapping não necessariamente significa que o software está exatamente detalhado em seu escopo, porém o escopo fora melhor definido quando houve a priorização mais correta do que deveria ser desenvolvido. Essa priorização foi fundamentalmente feita quando houve a consulta em cada cartão criado ao diagrama de conceito da startup.

Como perspectiva de trabalhos futuros, o trabalho precisa se integrar mais às metodologias de desenvolvimento como Scrum e Kanban, permitindo que seja possível acompanhar mais internamente como está sendo desenvolvida cada uma das user stories inseridas no conjunto de atividades de cada ciclo. Dessa forma, os testes poderão ser mais aprofundados através do método de Specification by Example que permite que sejam especificados também os testes que devem estar no desenvolvimento do software.

Também há a perspectiva de trabalho futuro relacionar este processo com métricas ISO de qualidade e avaliar o acompanhamento com as startups com relação ao padrão de qualidade, indicando também os índices de qualidade com cada software desenvolvido. Assim, podendo ser atestada a qualidade do software com relação a cada item da implementação de qualidade, gerando um padrão específico de qualidade de software específico para startups que poderão assegurar a qualidade do produto de software desenvolvido para os momentos de investimento e segurança para os usuários que poderão avaliar melhor o produto que estarão utilizando.

Parte IX

Referências bibliográficas

Referências

1 GOLDBERG, R. Software engineering: An emerging discipline. *IBM Syst. J.*, IBM Corp., Riverton, NJ, USA, v. 25, n. 3-4, p. 334–353, set. 1986. ISSN 0018-8670. Disponível em: <http://dx.doi.org/10.1147/sj.253.0334>. Citado na página 3.

2 KRUG, S. *Don't make me think! a common sense approach to Web usability*. 2nd ed. ed. Berkeley, Calif: New Riders Pub, 2006. ISBN 978-0-321-34475-5. Citado na página 3.

3 MANIFESTO para Desenvolvimento Ágil de Software. 2001. Disponível em: <http://agilemanifesto.org/iso/ptbr/manifesto.html>. Citado 2 vezes nas páginas 3 e 33.

4 KRIGSMAN, M. *Study: 68 percent of IT projects fail*. 2009. Disponível em: <https://www.zdnet.com/article/study-68-percent-of-it-projects-fail/>. Citado 2 vezes nas páginas 3 e 5.

5 ABSTARTUPS. *Estatísticas*. 2018. Disponível em: <https://startupbase.com.br/home/stats>. Citado 2 vezes nas páginas 3 e 5.

6 PESQUISA mostra que 30% das startups não conseguem se manter no mercado. Disponível em: <https://epocanegocios.globo.com/Empreendedorismo/noticia/2018/02/pesquisa-mostra-que-30-das-startups-nao-conseguem-se-manter-no-mercado.html>. Citado na página 5.

7 GROSSKOPF, A. et al. Design Thinking implemented in Software Engineering Tools. p. 17, 2010. Citado 2 vezes nas páginas 5 e 33.

8 GIL, A. C. *Métodos e técnicas de pesquisa social*. São Paulo: Atlas, 2008. OCLC: 298931695. ISBN 978-85-224-5142-5. Citado na página 11.

9 PRESSMAN, R.; MAXIM, B. *Engenharia de Software - 8ª Edição*. [S.l.]: McGraw Hill Brasil, 2016. ISBN 978-85-8055-534-9. Citado 2 vezes nas páginas 17 e 27.

10 SOMMERVILLE, I. *Engenharia de software*. São Paulo: Pearson Prentice Hall, 2011. OCLC: 940079598. ISBN 978-85-7936-108-1. Citado 2 vezes nas páginas 17 e 18.

11 DAVIS, A. M. *Software requirements: objects, functions, and states*. Rev. [S.l.]: PTR Prentice Hall, 1993. ISBN 978-0-13-805763-3. Citado na página 18.

12 CROSBY, P. B. *Completeness: quality for the 21st century*. New York, New York, U.S.A: Dutton, 1992. ISBN 978-0-525-93475-2. Citado na página 20.

13 KOSCIANSKI, A.; SOARES, M. d. S. *Qualidade de software: aprenda as metodologias e técnicas mais modernas para o desenvolvimento de software*. [S.l.]: Novatec, 2007. ISBN 978-85-7522-112-9. Citado na página 20.

14 BLANK, S.; BLANK, S.; DORF, B. *The Startup Owner's Manual: The Step-by-step Guide for Building a Great Company*. [S.l.]: KS Ranch, Incorporated, 2012. ISBN 978-0-9849993-0-9. Citado na página 21.

15 PATTON, J.; ECONOMY, P. *User Story Mapping: Discover the Whole Story, Build the Right Product*. [S.l.]: O'Reilly Media, 2014. ISBN 978-1-4919-0486-2. Citado na página 23.

16 BROOKS, F. P. Three great challenges for half-century-old computer science. *Journal of the ACM*, v. 50, n. 1, p. 25–26, Jan 2003. ISSN 00045411. Citado na página 25.

17 MINIMUM Viable Product. Disponível em: <https://www.syncdev.com/minimum-viable-product/>. Citado na página 29.

18 FRYE, U.; INGE, T. The integration of design thinking and lean software development from the perspective of product owners and scrum masters. p. 64, 2013. Citado na página 33.

19 POPPENDIECK, T.; POPPENDIECK, M. *Implementando o desenvolvimento Lean de software: Do conceito ao dinheiro*. 1. ed. Porto Alegre: Bookman, 2011. ISBN 978-85-7780-756-7. Citado na página 33.

Anexos

ANEXO A – Publicações relacionadas

Tabela 6 – Publicações relacionadas

Titulo	Evento	Url	Tipo
Study about software project management with Design Thinking	EATIS'18	https://dl.acm.org/citation.cfm?id=3293643	Artigo
Desenvolvimento de requisitos com Design Thinking	FISL'18	http://fisl18.softwarelivre.org/index.php/en/	Palestra
Análise de Requisitos de software com Story Maps	ENUCOMPI'18	https://enucompi.com.br/wp-content/uploads/2019/01/anais_Enucompi_2018.pdf	Artigo