MINISTÉRIO DA EDUCACÃO E DO DESPORTO
UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO
PRÓ-REITORIA DE ENSINO DE GRADUACÃO
BACHARELADO EM SISTEMAS DE INFORMACÃO

Trabalho de Conclusão de Curso

Uma ferramenta para cálculo de Métricas Lean

Jessica Nunes da Silva

Recife

Agosto de 2014

Jessica Nunes da Silva

Uma ferramenta para cálculo de Métricas Lean

Orientadora: Teresa Maria de Medeiros Maciel

Coorientador: Lenildo José de Morais

Monografia apresentada ao Curso Bacharelado em Sistemas de Informação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Recife

Agosto de 2014

À Deus,

Aos meus queridos pais, José Ivanildo e Luciana,

Aos meus irmãos, Jefferson e Juliana,

Ao meu noivo, Isaildo

Aos meus orientadores, Teresa Maciel e Lenildo Morais,

Aos meus amigos.

Agradecimentos

Agradeço primeiramente a Deus que é a causa de todas as minhas conquistas e o doador de toda vida.

Agradeço também aos meus pais, José Ivanildo e Maria Luciana que desde sempre me incentivaram a não parar na caminhada redobrando seus esforços em me ajudar a chegar até aqui. E também aos meus irmãos Jefferson e Juliana que me proporcionam incríveis momentos de gozo e alegria.

Agradeço ao meu noivo, Isaildo Filho que pacientemente suportou os momentos distantes e me forneceu apoio e incentivo.

Agradeço a minha orientadora Teresa Maciel que aceitou me guiar nesse momento tão importante para minha carreira. E ao meu coorientador Lenildo Morais por compartilhar seus conhecimentos e fornecer tanta ajuda. Sem vocês dois este trabalho não seria a mesma coisa, obrigada pela confiança.

Agradeço também a banca responsável pela a avaliação do meu trabalho. Suas observações foram de suprema importância para meu aprendizado e conhecimento.

Agradeço a todos que fazem parte da Universidade Federal Rural de Pernambuco, professores, coordenadores, servidores. Esta Universidade serviu como um segundo lar para mim. Guardarei a experiência comigo pelo resto da minha vida.

Resumo

Este projeto de conclusão de curso tem como objetivo o estudo e a construção de um produto mínimo viável para Web fornecendo suporte ao cálculo de Métricas Lean.

Para alcançar tal objetivo foi realizado um estudo sobre Metodologias Ágeis, Lean e suas Métricas e também uma pesquisa sobre as ferramentas atualmente existentes para suporte ao cálculo das métricas. Dessa forma, o trabalho mostra a importância dos Métodos Ágeis no desenvolvimento de software em um cenário de constantes mudanças, alta complexidade e curtos prazos de entrega. E também a importância de medir corretamente para a obtenção de ganhos em todo o processo de desenvolvimento.

Uma ferramenta foi desenvolvida como consequência dos estudos realizados e com a proposta de facilitar o uso das Métricas Lean por toda a comunidade.

Palavras-chave: Métodos Ágeis, Lean, Métricas Lean

Abstract

This project of course conclusion aimed the study and building of a minimum viable product for Web to provide support to Lean Metrics calculation.

To achieve this objective a study on Agile, Lean and its metrics and also a research of existing tools to support the calculation of the metrics was done. Therefore the study showed the importance of agile methods in software development in an environment of constant change, high complexity and short delivery times. Also the importance of measuring correctly for obtaining gains throughout the development process.

A tool was developed as a result of accomplished studies and the proposal to facilitate the use of Lean Metrics throughout the community

Key words: Agile, Lean, Lean metrics.

Lista de Figuras

Sumário

Capítulo 1

Introdução

Este trabalho de conclusão de curso irá expor a experiência de desenvolvimento e validação de um produto mínimo viável para o cálculo de métricas Lean. Neste capítulo serão expostos aspectos para a contextualização e motivação da pesquisa realizada.

1.1 Apresentação

O Sistema Toyota de Produção, ou *Lean Manufacturing,* surgiu no Japão, após a Segunda Guerra Mundial. Em 1945, Kiichiro Toyoda havia desafiado sua companhia a se igualar às da América, mas ficou claro que a Toyota não conseguiria isso adotando o modelo de produção em massa norte-americano [1]. O modelo produção em massa exigia a construção de milhares de peças idênticas para obtenção de lucros de escala, porém a produtividade era baixa e os materiais escassos.

Embora tenha sido necessário tempo e esforço, a implementação do *Lean Manufacturing* foi primordial para o sucesso da Toyota. Através de seu fluxo de produção, *Just-In-Time,* que segundo Taiichi Ohno, desenvolvedor do Sistema de Produção Toyota é um sistema para a absoluta eliminação de desperdício [1] extinguindo estoques intermediários, e da autonomação, em japonês *Jidoka*, onde o trabalho é organizado de modo que na menor anomalia detectada ele é imediatamente interrompido, dessa forma não necessitará de inspeção.

Nos anos 70, surgia o termo Crise do Software, que designava as dificuldades enfrentadas no desenvolvimento de software, inerentes ao aumento das demandas e da complexidade delas, aliado a inexistência de técnicas apropriadas para resolver esses desafios [3]. Ainda na atualidade esta crise é vivenciada. Além do aumento grandioso da complexidade, temos também o fenômeno da urgência em se desenvolver esses softwares, por exemplo, devido às necessidades do mercado. Mesmo empresas que têm conhecimento das técnicas propostas às vezes não conseguem praticá-las por pressão do próprio cliente, por exemplo, em relação a prazos. [3]

Neste ambiente de constantes mudanças, as práticas ágeis se tornam cada vez mais presentes nos projetos de desenvolvimento de software como um caminho para reduzir desperdícios e responder com eficiência às expectativas do cliente, realizando entregas de valor e se adequando rapidamente às mudanças [4].

A utilização do *Lean Manufacturing* no desenvolvimento de software consiste na utilização dos seus princípios tendo em vista que um software também pode ser

visto como um produto. O termo *Lean Software Development* surgiu em 2003 na publicação por Tom e Mary Poppendieck de um livro de mesmo nome.

Tom e Mary Poppendieck expõem um conjunto de sete princípios do desenvolvimento *Lean* de software, eles serão expostos na seção 2.1.3 *Lean Software Development*. No 7º Princípio: Otimize o todo, é valorizado a utilização de métricas para acompanhamento do processo que valorizem o desempenho da equipe e a redução de métricas que medem o desempenho individual [4].

Existem hoje algumas ferramentas que fornecem apoio à utilização do *Lean* no desenvolvimento de projetos. *Softwares* como Kanbanize, LeanKit e Kanban Tool são utilizadas em todo o processo de desenvolvimento. Existem também Minitab e o Office Excel, que não são voltadas para o gerenciamento de projetos e sim para o uso estatístico e geral, porém também fornecem funcionalidades úteis para o uso do *Lean.* Em todos os casos é possível obter algum tipo de cálculo das métricas *Lean*, porém esta função está atrelada ao uso de outras funcionalidades do sistema. Na seção 2.1.5 Ferramentas Utilizadas para Cálculo de Métricas Lean será visto um pouco mais sobre algumas dessas ferramentas.

1.2 Relevância

Atualmente, tão rápido quanto à tecnologia evolui, a pesquisa e a possibilidade de acesso a informações precisas e confiáveis faz com que as necessidades mudem e os sistemas de software tornem-se obsoletos em menos tempo. Como consequência, a demanda por software cresce e os prazos diminuem. Essa realidade aumenta a instabilidade e torna o desenvolvimento de um software uma operação mais sensível à influência de elementos externos à equipe técnica [6].

As Metodologias Ágeis vêm apresentando um grande avanço em termos de utilização pela indústria e academia, tanto no Brasil como no mundo. Muito deste sucesso é pautado nas suposições de que tais metodologias e suas práticas, além de melhorar a gerência do projeto como um todo, elevam a qualidade do produto e a satisfação do cliente [9].

Como proposta dos Métodos Ágeis, o processo de desenvolvimento de *software* deve ser empírico e iterativo. Essa abordagem exige um ciclo constante de inspeção, adaptação e melhoria.

Encontrar maneiras eficazes de avaliar o processo e a equipe de desenvolvimento não é uma tarefa simples. Isso leva a uma proliferação de medidas baseadas na premissa de que se cada parte do processo for otimizada, os resultados do processo como um todo serão otimizados também [13]. A preocupação em decidir o que deve ser medido também não pode ser deixada de lado. Ao gerar medidas erradas acaba-se gerando incentivos errados.

Medir é de suma importância para a melhoria do processo de desenvolvimento. Encontrar e entender as medidas corretas para o contexto é uma tarefa árdua, mas extremamente necessária. Ao tentar micro-otimizar partes de um sistema por meio de diversas métricas, e o verdadeiro objetivo se perde em meio a tantos substitutos e a equipe perde sua capacidade de tomar decisões de balanceamento (trade-off)[13].

Assim, é importante e necessário fornecer subsídios que facilitem a utilização das métricas Lean em todo e qualquer contexto. Tendo em vista que boa parte das ferramentas existentes no presente momento está atrelada a um conjunto de outras funcionalidades ou se faz necessário à compra de uma licença para sua utilização.

1.3 Objetivos

1.3.1 Objetivo Geral

O principal objetivo deste trabalho é a construção de um produto mínimo viável para Web que auxilie na análise do processo de desenvolvimento de *software* efetuando o cálculo de Métricas *Lean* e gerando seus respectivos os gráficos.

1.3.2 Objetivos Específicos

Para alcançar o objetivo geral, foram traçados alguns objetivos específicos. São eles:

- Aprender sobre Desenvolvimento Ágil de *Software*, seus princípios, suas práticas e métricas;
- Pesquisar sobre a utilização do *Lean Software Development*, seus princípios, suas práticas e suas métricas;
- Investigar as ferramentas existentes hoje, que já fornecem algum tipo de suporte às métricas *Lean,* observando seus pontos fortes e suas lacunas para melhorias;
- Definir o escopo do produto mínimo viável dentro do que possui maior valor a partir dos resultados das pesquisas na literatura.
- Desenvolver o produto mínimo viável definido;
- Testar o produto mínimo viável em um contexto real, analisando falhas, faltas e melhorias para a ferramenta.

1.4 Metodologia

1.4.1 Revisão Bibliográfica e Referencial Teórico

Definidos os objetivos, foi realizada a revisão da literatura com a intenção de sintetizar os principais conceitos, tendências e abordagens referentes ao tema. Os assuntos de pesquisa são: *Lean Manufacturing, Lean Software Development,* Manifesto Ágil, *Scrum, Kanban,* Métricas *Lean* e o Uso de Métricas *Lean* no Desenvolvimento de *Software.*

Com base nos resultados da revisão da literatura, foi desenvolvido o referencial teórico.

1.4.2 Definição do Escopo do Produto Mínimo Viável

Baseado no referencial teórico partiu-se então para a definição do escopo do produto mínimo viável. A proposta é desenvolver o que possui maior valor primeiro, visando sua utilização como teste, e posteriormente aplicar melhorias identificadas.

1.4.3 Desenvolvimento da ferramenta

Com o escopo delimitado, iniciou-se o processo de desenvolvimento da ferramenta. Houve a definição das tecnologias que seriam utilizadas durante o desenvolvimento e a preparação do ambiente.

1.4.4 Teste da ferramenta

Com o produto mínimo viável entregue, partiu-se para o teste da ferramenta em um ambiente real de desenvolvimento de *software.*

1.4.5 Conclusões

Com base nos resultados da aplicação em um ambiente real de desenvolvimento, a aceitação da ferramenta foi analisada. Melhorias, trabalhos futuros, falhas foram levantados.

Capítulo 2

2.1 Referencial Teórico

2.1.1 Desenvolvimento Ágil de Software

Durante anos, a indústria utilizou métodos tradicionais para o desenvolvimento de software. Tais métodos eram conhecidos e utilizados no mercado e na comunidade acadêmica. Porém, percebendo que a indústria apresentava um grande número de casos de fracasso, alguns líderes experientes adotaram modos de trabalho que se opunham aos principais conceitos das metodologias tradicionais[6]. Tais modos de trabalho se mostraram bastante eficientes, mesmo não seguindo os padrões de mercado. Na medida em que eram aplicados em diversos projetos, os métodos foram aperfeiçoados e em alguns casos chegaram a se transformar em metodologias de desenvolvimento de software. Essas metodologias passaram a ser chamadas "leves ”por não utilizarem as formalidades que caracterizavam os processos tradicionais e por evitarem a burocracia imposta pela utilização de excessiva documentação [6].

Em 2001, dezessete desses líderes se reuniram em uma estação de esqui em Utah, Estados Unidos. Na ocasião eles discutiram suas formas de trabalhos, formas essas que além de eficazes eram parecidas e chegaram ao consenso de que desenvolver software é algo complexo demais para ser definido por um único processo, pois ele depende de muitas variáveis [6]. Como resultado, surgiu o Manifesto Ágil que possui 12 princípios e é representado por quatro premissas [7]:

Indivíduos e interações mais que processos e ferramentas;

Software em funcionamento mais que documentação abrangente;

Colaboração com o cliente mais que negociação de contratos;

Responder a mudanças mais que seguir o plano;

Mesmo havendo valor nos itens à direita, deve-se ser mais valorizado os itens à esquerda. Da mesma forma seguem os princípios do Manifesto Ágil citados anteriormente:

- Prioridade é satisfazer o cliente através da entrega contínua e adiantada de software com valor agregado;
- As mudanças nos requisitos são bem-vindas, mesmo tardiamente no desenvolvimento. Processos ágeis tiram vantagem das mudanças visando vantagem competitiva para o cliente;

- Frequentes entregas do software funcionando, de poucas semanas a poucos meses, com preferência à menor escala de tempo;
- As pessoas de negócio e desenvolvedores devem trabalhar diariamente em conjunto por todo o projeto;
- Os projetos devem ser construídos em torno de indivíduos motivados. Dando o ambiente e o suporte necessário e confiança para fazer o trabalho;
- O método mais eficiente e eficaz de transmitir informações para e entre uma equipe de desenvolvimento é através de conversa *face to face*;
- *Software* funcionando é a medida primária do progresso;
- Os processos ágeis promovem desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente;
- Contínua atenção a excelência técnica e bom design aumenta a agilidade.
- Simplicidade para maximizar, a quantidade de trabalho não realizado é essencial;
- As melhores arquiteturas, requisitos e designs emergem de equipes auto organizáveis;
- Em intervalos regulares, a equipe deve refletir sobre como tornar-se mais efetiva, e então, ajustar-se de acordo com seu comportamento.

O "Manifesto Ágil" não rejeita os processos e ferramentas, a documentação, a negociação de contratos ou o planejamento, mas simplesmente mostra que eles têm importância secundária quando comparado com os indivíduos e interações, com o software estar executável, com a colaboração do cliente e as respostas rápidas a mudanças e alterações. Esses conceitos aproximam-se melhor com a forma que pequenas e médias organizações trabalham e respondem a mudanças [8].

Antes de surgir o "Manifesto Ágil", surgia no Japão o *Lean Manufacturing,* também conhecido como "Sistema de Produção Toyota". Seus princípios e práticas serviram como uma das bases para o "Manifesto" e para os vários métodos de desenvolvimento ágil existentes e utilizados na atualidade. Veremos mais sobre o *Lean* na seção 2.1.2 Lean.

2.1.2 Scrum

Em um jogo de *rugby*, "scrum" é uma forma do time recolocar a bola em jogo. Na metodologia criada por Jeff Sutherland a partir do trabalho de Nonaka e Takeuchi e formalizada em 1995 por Ken Schwaber, a equipe de desenvolvimento trabalha unida com o objetivo de entregar o software funcional de alta qualidade. Neste modelo, a equipe se compromete com um objetivo e tem autonomia para definir a tática para chegar até ele [6].

O Scrum se concentra mais nos aspectos gerenciais do desenvolvimento de software, propondo iterações de duas semanas ou 30 dias (chamados *Sprints*) com acompanhamento diário por meio das Reuniões em Pé (ou *stand-up meetings*) [13].

2.1.3 Kanban

Kanban é um método de gestão de mudanças, que dá ênfase aos seguintes princípios [14]:

- Visualizar o trabalho em andamento;
- Visualizar cada passo em sua cadeia de valor, do conceito geral até software que se possa lançar;
- Limitar o trabalho em progresso (WIP – Work in Progress), restrigindo o total de trabalho permitido para cada estágio;
- Tornar explícitas as políticas sendo seguidas;
- Medir e gerenciar o fluxo para poder tomar decisões bem embasadas, além de visualizar as consequências dessas decisões;
- Identificar oportunidades de melhorias, criando uma cultura Kaizen, na qual a melhoria contínua é de responsabilidade de todos.

2.1.4 Lean

Para competir com os fabricantes de carros americanos, em meados da década de 1940, a Toyota começou a desenvolver um sistema de produção que a permitiu crescer solidamente valendo-se de uma contínua melhoria de seu desempenho competitivo. Este sistema, conhecido como "Sistema de Produção Toyota" (TPS), tem como um de seus principais pilares, o conceito e as técnicas de produção *Just-in-time* (JIT) [9]. Além do conceito de *Just-in-Time* existe também o da Autonomação, em japonês, *Jidoka*, ou denominado em inglês de *stop-the-line*, que se preocupa em eliminar a necessidade de inspeção parando o processo de desenvolvimento imediatamente quando um erro é encontrado.

O fluxo *Just-in-Time* elimina os principais contribuidores do custo da variedade. Ele significa eliminar estoques intermediários de processo que costumam ser feitos em nome das economias de escala. A ideia por trás da Autonomação é que um sistema deve ser projetado para ser à prova de erros [1]. Ela também pode, segundo Ohno, ser definida como "automação com um toque humano" [10].

O termo *Lean Production*, ou em Português, Produção Enxuta foi utilizado para designar o Sistema de Produção Toyota pela primeira vez em 1990 no livro escrito por, James Womack, Daniel Jones e Daniel Roos, A Máquina que Mudou o Mundo.

2.1.5 Lean Software Development

De fato, a ideia de aplicar princípios *Lean* no Desenvolvimento de *Software* é mais antiga que o próprio termo Lean. Na década de 90, Robert Charrete usou o termo "Desenvolvimento Enxuto" para se referir a uma estratégia de gerenciamento de risco que traz estabilidade sobre dinâmica nas organizações as tornando mais ágeis, resistentes, e tolerantes a mudança [11].

O desenvolvimento de *software Lean* é a aplicação dos princípios da Toyota *product development system* para o desenvolvimento de software. Quando ele é aplicado corretamente, o desenvolvimento é de alta qualidade, além de ser realizado rapidamente e possuir um baixo custo. Além disso, o desenvolvimento ágil possui um grande sucesso devido ao Lean [5].

De acordo com Mary e Tom Poppendieck o desenvolvimento *Lean* de *Software* apresenta um conjunto de sete princípios que devem nortear o uso da metodologia. Princípios são verdades subjacentes que não mudam no tempo ou espaço, enquanto as práticas são a aplicação dos princípios a uma situação particular [1]. São eles:

Princípio 1: Eliminar o desperdício

Para Taiichi Ohno, desperdício é tudo o que não acrescenta valor ao produto na percepção do cliente. Para ele, o Sistema Toyota de Produção era um sistema de gerenciamento para "eliminação absoluta do desperdício" [1].

Na produção, estoque é desperdício. Existe um alto custo com o manuseio, logística, monitoramento e reabastecimento do estoque, o que acaba elevando os custos e esforços. No processo de desenvolvimento de software, funcionalidades incompletas são desperdício porque despendem esforços para serem iniciadas e não adicionam valor ao software. Pedaços de código incompletos tendem a se tornar obsoletos, mais difíceis de serem integrados e os programadores lembram menos a respeito da intenção inicial do código. Por estarem inacabadas, foi um desperdício começá-las [6].

Outro desperdício é o excesso de processos, eles demandam recursos e aumentam o tempo para a conclusão das tarefas. A criação de documentos infla o processo e causa desperdício pois eles consomem tempo para serem produzidos, sem garantias de que alguém irá lê-los. Documentos ficam desatualizados e podem ser perdidos, tornam a comunicação mais lenta e reduzem o poder comunicativo, pois são um meio de comunicação de via única no qual não é possível que escritor e leitor interajam em tempo real [6].

Entretanto, a maior fonte de desperdício no desenvolvimento de software são as funcionalidades adicionais. Somente cerca de 20% das funcionalidades e funções

em um programa personalizado típico são usadas regularmente. Algo como dois terços delas são raramente usadas [1]. Existem diversos outros tipos de desperdício, como a troca de pessoas da equipe causando perda de conhecimento, espera por requisitos, testes e *feedback*. Espera retarda o andamento do fluxo e atrasa a identificação de problemas.

Princípio 2: Integrar qualidade

De acordo com Shigeo Shingo, existem dois tipos de inspeção: inspeção após os defeitos ocorrerem e inspeção para prevenir defeitos. Se você realmente quer qualidade, não inspeciona após o ocorrido, mas, em primeiro lugar, controla as condições de forma a não permitir defeitos. Se não é possível, então você inspeciona o produto a cada pequeno passo para que os defeitos sejam identificados imediatamente após ocorrerem [1].

Um mito que na maioria das vezes está presente nas organizações é a ideia que a função dos testes é encontrar defeitos. Na realidade deve-se cultivar a prevenção dos erros promovendo processos que constroem qualidade no código desde o início, em vez de testar a qualidade no final [1].

Um dos métodos usados para inserção de qualidade no código é o TDD, do Inglês, *Test Driven Development.* Sob o *slogan* "faça certo da primeira vez" desenvolve-se a cultura de todo a escrita de código ser feita sobre testes também desenvolvidos pelo programador. Dessa maneira o trabalho dos testes não será realizado apenas no final de todo o ciclo e sim durante o próprio desenvolvimento.

Princípio 3: Criar conhecimento

O desenvolvimento de software é um processo de criação de conhecimento [1]. Lições devem ser extraídas das experiências vividas pela equipe e incorporadas ao processo fazendo com que dificuldades sejam fonte de conhecimento e contribuam para o amadurecimento da equipe e do processo [6].

É importante ter um processo de desenvolvimento que encoraje o aprendizado sistemático durante todo o ciclo de desenvolvimento [1]. Processos definidos podem engessar o aprendizado e limitar o desenvolvimento da equipe. A organização deve trabalhar para sempre melhorar seus processos, sabendo que em ambientes complexos sempre existirá problemas.

Princípio 4: Adiar comprometimento

Planeje decisões irreversíveis para o último momento possível, isto é, a última chance de tomar uma decisão antes que seja tarde demais [1]. Isso mantem a flexibilidade para adaptação a mudanças e permite que as decisões sejam apoiadas em experiências adquiridas no decorrer do processo.

Para retardar decisões durante a construção de sistemas é importante que a equipe crie a capacidade de absorver mudanças tratando os planejamentos como estratégias para atingir um objetivo e não como comprometimentos. Assim, mudanças serão vistas como oportunidades para aprender e atingir as metas [6].

Princípio 5: Entregar rápido

Rapidez entre um pedido e uma entrega e entre a entrega e as percepções de quem o solicitou permite que o cliente e desenvolvedores aprendam e melhorem através de *feedback* veloz, atualizado e confiável [6]. Com um processo iterativo, é possível construir o conhecimento a cada ciclo, evitando entregar o que o cliente não queria permitindo o adiamento da tomada de decisão para quando a equipe possuir conhecimento e experiência com o processo já vivido. Mary Poppendieck recomenda [1]:

> *"... precisamos descobrir como entregar software tão rápido que os clientes não tenham tempo de mudar de ideia."*
>
> Mary Poppendieck

Princípio 6: Respeitar as pessoas

O respeito e valorização da equipe podem ser extremamente importantes para o desenvolvimento dela. A equipe de desenvolvimento é responsável pela confecção do produto que é entregue e usado pelo cliente, o software [6]. Não existe a melhor maneira de se fazer algo, existem diversas maneiras que devem se adequar ao momento e ao contexto vivido pela equipe.

Envolver os desenvolvedores nas decisões de detalhes técnicos é fundamental para atingir a excelência. Quando dotados com a experiência necessária e guiados por um líder, eles tomarão decisões técnicas e de processos, melhores que qualquer outra pessoa poderia tomar por eles [5].

Princípio 7: Otimizar o todo

Uma organização *Lean* otimiza todo o fluxo de valor, do momento em que recebe o pedido visando uma necessidade do cliente até que o software seja implantado e a necessidade do cliente seja atendida [1]. É necessário que todo o fluxo seja otimizado. Se a equipe otimiza alguma coisa menor que isso, o fluxo de valor inteiro sofrerá.

Este princípio também trata das maneiras de medição do andamento do processo de desenvolvimento de software. O *Lean* recomenda a escolha de métricas de alto nível que sejam representativas para identificar a evolução. Essas métricas devem levar em consideração também a qualidade e a satisfação do

cliente [6]. Não é possível medir tudo, então deve-se medir o que realmente importa, utilizando métricas que valorizem o desempenho da equipe e reduzindo o número de métricas que medem o desempenho individual.

2.1.6 Métricas Baseadas no *Lean*

Os Métodos Ágeis promovem um processo empírico para o desenvolvimento de software. Essa abordagem exige um ciclo constante de inspeção, adaptação e melhoria [13]. Porém a atividade de medir o progresso em projetos de desenvolvimento de *software* muitas vezes é mal compreendida e mal aplicada.

As pessoas têm a tendência de decompor situações complexas em pequenas partes. Depois da decomposição cada pedaço é medido e cada medida é otimizada. [1]. Porém ao tentar micro-otimizar partes de um sistema por meio de diversas métricas, o verdadeiro objetivo se perde em meio a tantos substitutos e a equipe perde sua capacidade de tomar decisões de balanceamento (trade-off) [13].

Em métricas, uma regra básica a ser lembrada é que seu sistema de entrega de *software* tem capacidade limitada. Quando um sistema é pressionado além da capacidade, haverá queda na qualidade, ritmo insustentável, custos de manutenção mais altos, ou tudo isso ao mesmo tempo [14]. É de suma importância saber o que deve ser medido, a preocupação com as medidas erradas pode gerar incentivos errados, levando a consequências indesejáveis [13].

Antes de escolher a forma de medir, deve-se sempre perguntar o que será feito com a informação obtida. Se nada será mudado com a métrica, é provável que se trate de algo que não deveria estar sendo medido [14]. Existem várias métricas baseadas em *Lean,* porém algumas delas são bastante utilizadas e referenciadas na literatura, por se tratarem de métricas de alto nível que não medem o desempenho individual e sim de toda equipe. A seguir discutiremos cada uma delas.

2.1.6.1 Diagrama de Fluxo Cumulativo (*Cumulative Flow Diagram)*

Os diagramas de fluxo acumulativo (CFD – *Cumulative Flow Diagram*) apresentam a quantidade acumulativa de demandas em progresso a partir de cada fase da cadeia de valor no passar do tempo. A diferença entre duas fases adjacentes indica a quantidade de demandas que estão exatamente na 1ª fase [4]. Eles mostram essencialmente o retrato da quantidade de trabalho em progresso do sistema, para cada estágio de sua cadeia de valor, no decorrer do tempo [14]. Um exemplo de Diagrama de Fluxo Cumulativo é exibido na **Figura 3.1**

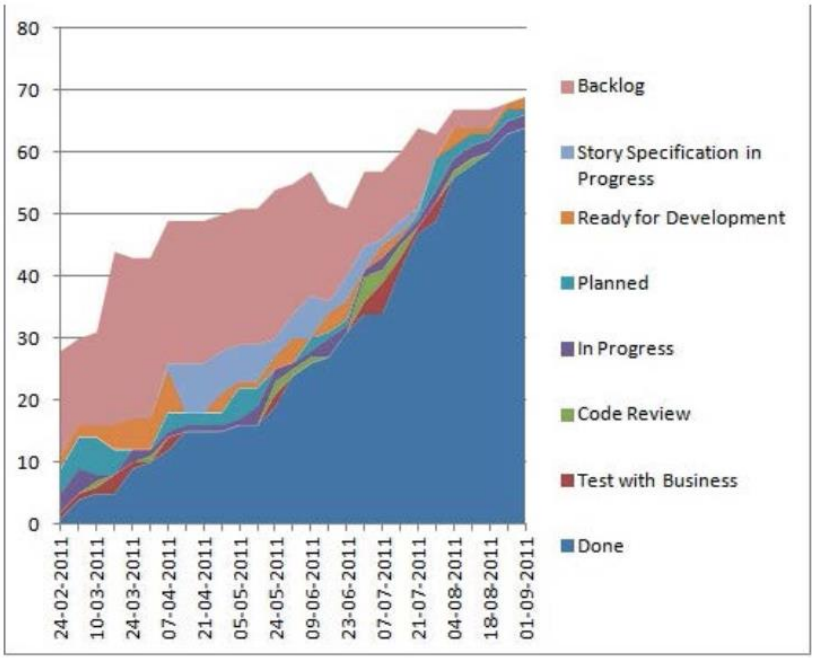

Figura 3.1 Diagrama de Fluxo Cumulativo

2.1.6.2 *Lead Time*

O objetivo desta métrica é minimizar o tempo médio de ciclo de um sistema, sob o ponto de vista do cliente final [4]. O importante é descobrir o quão rápido o sistema entrega valor de forma repetitiva e confiável [13].

O *Lead Time* mapeia a quantidade de tempo que a demanda gasta para passar por todo fluxo de valor, desde o momento que entra na primeira fase, por exemplo, é puxado pelo desenvolvedor, até o momento que entra na última fase, por exemplo, validação com o cliente. O diagrama representativo é bem parecido com o diagrama de tempo médio de ciclo que será exibido a seguir.

2.1.6.3 Tempo Médio de Ciclo (*Cycle Time)*

O tempo médio de ciclo, ou em inglês, Cycle Time, tem como objetivo minimizar o tempo médio de cada fase [4]. Ele mede o tempo que determinada demanda permaneceu em uma fase, desde o momento de sua entrada, até sua saída para a próxima fase.

O tempo médio de ciclo do sistema deve diminuir gradualmente conforme o processo é melhorado, até atingir um patamar mínimo e pode ser usada para avaliar o comprometimento das pessoas com o sistema todo [13].

Apesar de simples, o diagrama de tempo médio de ciclo pode trazer muitas informações e responder muitas perguntas. Observe a **Figura 3.2**

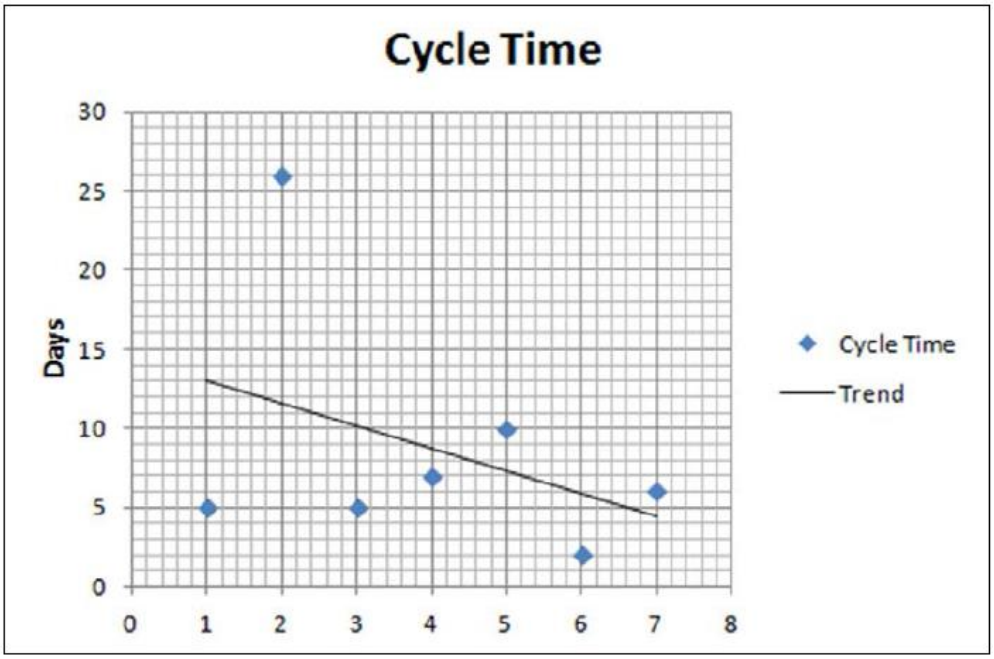

Figura 3.2 Exemplo de diagrama de tempo médio de ciclo

2.1.6.4 *Throughput*

Medir a capacidade de velocidade e o rendimento do sistema pode ajudar a definir a quantidade de trabalho que se deve aceitar e ajuda a estimar o tempo de entrega do mesmo [9].

O *Throughput* demonstra a vazão da cadeia de valor, ou seja, indica quantas demandas estamos entregando por unidade de tempo [2]. Seu cálculo é feito ao dividir-se a quantidade de demandas entregues pelo tempo total observado.

2.1.6.5 *Beat Time*

Esta métrica indica a frequência de entrega da cadeia de valor, ou seja, de quanto em quanto tempo entregamos uma demanda. O Beat Time é uma métrica importante para analisarmos a capacidade de entrega de uma equipe. Ela é calculada a partir da divisão do tempo total observado pela quantidade de demandas entregues no período [4].

2.1.7 Ferramentas Utilizadas para Cálculo de Métricas Lean

Existem inúmeras ferramentas que fornecem apoio ao desenvolvimento ágil de software, porém apenas algumas dessas ferramentas apresentam funcionalidades que calculem métricas Lean e mostrem seus gráficos e taxas. Desse modo, foram selecionadas ferramentas que proporcionam ao usuário a

oportunidade de calcular métricas baseadas no Lean e que são utilizadas pela comunidade de desenvolvimento de Software.

2.1.7.1 Minitab

Minitab é um software estatístico proprietário. Ele fornece um ambiente para análise de dados. Seus recursos envolvem tabelas estatísticas, gráficos, textos e planilhas de dados, testes de hipóteses, análise de variância, entre outros. O Minitab é bastante utilizado por pesquisadores, estudantes, engenheiros e analistas.

Sua interface é bem próxima à interface oferecida pelo Office Excel da Microsoft, porém ele oferece acuradas ferramentas para controle de qualidade, projeto de experimentos, análise de confiabilidade e estatística geral. [17].

O software é um produto proprietário e para ser utilizado se faz necessário a compra de sua licença. Ele foi projetado para o uso estatístico, logo apresenta um grande aparato de funcionalidades que provavelmente não serão utilizadas no cálculo das métricas Lean.

2.1.7.2 Kanbanize

Kanbanize é uma plataforma de gestão de projetos online. Seu funcionamento simula um quadro de visualização e acompanhamento Kanban. Com ela é possível criar projetos, e tarefas que serão apresentadas em um quadro. No próprio quadro as tarefas são manuseadas de acordo com o fluxo de valor [18].

Sua versão gratuita permite a criação ilimitada de projetos e quadros de Kanban online. Também é possível criação ilimitada de tarefas, a manipulação dos quadros entre projetos, copiando, movendo e etc. assim como a customização do fluxo de valor padrão.

Para o acompanhamento do projeto, a ferramenta fornece um módulo que gera, partindo do quadro de atividades o cálculo das Métricas, Fluxo Cumulativo, Tempo Médio de Ciclo, Distribuição de atividades, Criadas *versus* Finalizadas e Tempo de Resolução de Bloco.

Como ferramenta específica para gerenciamento de projeto Ágil, oferece funcionalidades uteis para tal dando suporte aos princípios Ágil.

Quando usada para o gerenciamento do projeto é bastante simples visualizar os gráficos, porém só é possível calcular métricas de projetos que estejam cadastrados no sistema. O cálculo é um módulo atrelado ao sistema de gerenciamento de projetos, e logo, é necessário que todo o projeto esteja sendo gerenciado através da ferramenta.

2.1.7.3 Leankit

LeanKit é um software proprietário, online que fornece apoio ao gerenciamento de processos por meio dos princípios Lean. Ele foi desenhado para a implementação do Kanban e permite a organização das atividades em cartões virtuais [19].

A ferramenta cria um modelo visual do fluxo de trabalho o que permite ver instantaneamente o que está sendo trabalhado, quem está trabalhando em o que e que ainda necessita ser feito. A manipulação das atividades é feita no próprio quadro. Se utilizando da visibilidade de um quadro é possível identificar bloqueadores e gargalos dando à equipe a compreensão compartilhada das atividades e status.

Para análise e medição a ferramenta conta com um conjunto de gráficos que são medidos diretamente dos dados inseridos no quadro. Diagrama de Fluxo Cumulativo, Tempo Médio de Ciclo, Diagrama de Eficiência e a Distribuição de atividades podem ser visualizados no LeanKit. Para sua utilização é necessário adesão de sua licença. O LeanKit também fornece uma versão free trial, porém sua validade é de apenas 30 dias.

2.1.7.4 Kaban Tool

Kanban Tool é uma ferramenta online que fornece uma solução visual para gerenciamento de processos. Com ela é possível organizar o projeto em atividades alocadas em cartões virtuais, visualizar e controlar o fluxo de trabalho através de um quadro Kanban.

A ferramenta fornece visualização do fluxo de valor, colaboração do time em tempo real permitindo o compartilhamento de quadros e documentos e o envio de notificações por e-mail. Também é conhecida por ser simples e intuitiva permitindo customização e integração com outras ferramentas [20].

Baseada nos princípios Lean, a ferramenta permite o cálculo das métricas, Fluxo Cumulativo, Tempo Médio de Ciclo e Lead Time e exibe seus respectivos diagramas. O Kanban Tool é um software proprietário, logo é necessária a compra de sua licença para sua utilização. Uma versão grátis para teste é ofertada, porém sua validade é de apenas 30 dias.

Capítulo 3

3.1. Modelagem e Desenvolvimento

3.1.1 Tecnologias para o desenvolvimento

Para a construção da ferramenta foram escolhidas a linguagem de programação PHP e as bibliotecas PHPExcel e pChart. Nesta seção iremos falar sobre tais tecnologias.

3.1.1.1 PHP

PHP é a abreviação de *Hypertext Preprocessor* ou em tradução livre "Pré-processador de Hipertexto". É uma linguagem de script de código aberto que tem como objetivo primário a geração de conteúdo dinâmico para páginas da internet. Isto quer dizer que ao invés de criar um programa para gerar e imprimir HTML, você pode escrever HTML com o código PHP embutido para gerar conteúdo dinâmico.[15].

De acordo com Tiobe Software [16], PHP está entre as dez linguagens de programação mais utilizadas no ano de 2014. Partindo da versão 5 a linguagem suporta orientação objetos de forma consistente. Por ser um software livre, não possui custo de licença, o que significa que pode ser utilizada sem nenhum custo adicional.

PHP possui vasta documentação que pode ser facilmente encontrada na internet. Seu suporte é feito por meio de comunidades de usuários. Algumas das vantagens em utilizar PHP podem ser listadas [15]:

- Manipulação de arquivos como texto plano, tipo PDF, dodumentos DOC entre outro, incluindo sua criação, exclusão, alteração etc.
- Geração de imagens dinâmicas para valiadação de formulários;
- Criptografia de dados;
- Definção de cookies e sessões;
- Definição de interfaces para Webservices;
- Manipulação de arquivos XML;
- Suporte a vários bancos com acesso nativo como: DB2, *Firebird, Infomix, interbase,* MySQL, *Oracle,* PostgreSQL, SQL *Server* etc;
- Suporte a vários padrões e interfaces como: COM, CORBA, POP3, MAP etc;
- Independêcia de plataforma: Windows, Linux, Unix, Mac etc;
- Suporte a Orientação Objetos consistente na versão 5;
- Curva de aprendizagem reduzida para iniciantes;

- Integração com vários servidores web como: Apache, IIS, Xitami, entre outros;
- Servidores de hospedagem tanto gratuitos como pagos em grande quantidade no Brasil e no mundo;
- Documentação oficial em Português

3.1.1.2 PHPExcel

PHPExcel é uma classe escrita em PHP que fornece funcionalidades para a manipulação de arquivos baseados em XML. Com ela é possível ler, escrever, manipular células em planilhas eletrônicas utilizando o PHP.

A biblioteca foi usada para capturar dos arquivos os dados para posteriormente serem calculadas as Métricas *Lean*.

3.1.1.3 pChart

pChart é um *framework* em PHP que ajuda a criar gráficos e imagens diretamente do servidor Web. Ele permite que os gráficos gerados sejam exibidos na tela, enviados por e-mail e também inseridos em arquivos PDF.

A biblioteca foi utilizada para gerar os gráficos de acordo com os dados passados depois do cálculo das Métricas.

3.1.2 Estórias de Usuário

Para definição do escopo e elicitação dos requisitos foi utilizada a técnica de Estórias de Usuários. Foi então criada uma estória de usuário geral exibida na **Figura 3.3**, que foi posteriormente dividida em estórias menores.

	Eu como usuário
	Desejo fazer upload de arquivo contendo os identificadores, datas, status e tipo das atividades presentes no processo de desenvolvimento que irei medir.
	Para calcular as Métricas Lean Fluxo Cumulativo, Lead Time, Tempo Médio de Ciclo, Throughput e Beat time, gerando seus respectivos gráficos.

Figura 3.3 – Estória de Usuário Geral

O usuário deve poder visualizar o Diagrama de Fluxo Cumulativo, o gráfico do *Lead Time,* e as taxas de *Throughput* e *Beat Time,* estórias exibidas na **Figura 3.4**, partindo dos dados que ele inseriu no arquivo que foi enviado à ferramenta.

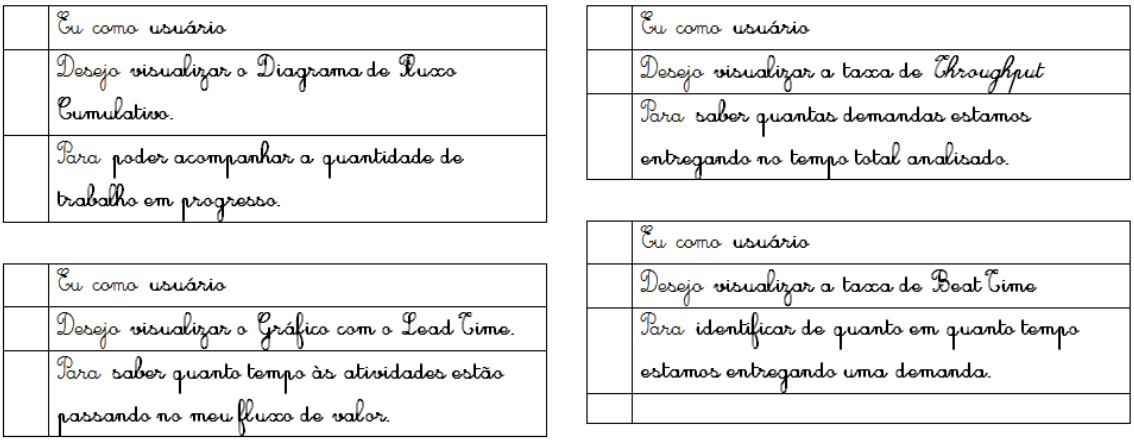

Figura 3.4 – Estórias de Usuário de algumas Métricas

Da mesma forma o usuário deve poder visualizar o gráfico do Tempo Médio de Ciclo como exibido na **Figura 3.5**.

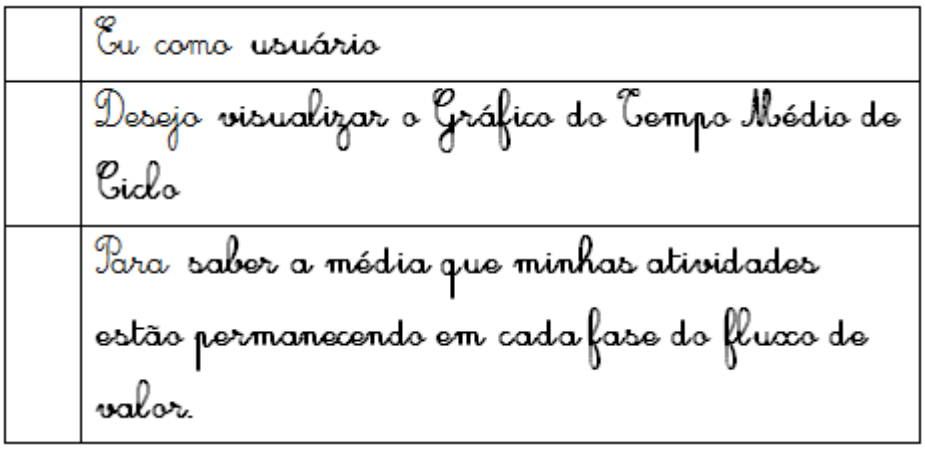

Figura 3.5 – Estória de Usuário do Tempo Médio de Ciclo

Possuindo o escopo e as tecnologias para o início do desenvolvimento, a ferramenta começou a ser implementada.

3.2 Teste da Ferramenta

Para a contribuição da melhoria e ajustes da ferramenta, ela foi testada em um ambiente de desenvolvimento de *software.* Assim, a ferramenta foi publicada na Web e disponibilizada para utilização.

3.2.1 Contextualização da Empresa

A empresa onde a ferramenta foi testada é uma empresa de desenvolvimento de *software* especializada em Computação em Nuvem localizada em Recife – Pernambuco.

Situada no Porto Digital, a empresa implementa vários projetos simultaneamente, em todos, Métodos Ágeis são utilizados no processo de desenvolvimento. A empresa também utiliza uma ferramenta online para controle de mudanças e gerenciamento de falhas.

3.2.2 Utilização da Ferramenta

A ferramenta foi testada em um dos projetos desenvolvidos pela empresa, o responsável pelo acompanhamento do projeto visitou a seção "Entenda como Funciona" que pode ser vista na **Figura 3.6** e observou como funciona a ferramenta e o que ele necessitava fazer para obter o cálculo das métricas. Ele utilizou a ferramenta de gerenciamento de mudanças para exportar as atividades do Sprint que ele desejava analisar para uma planilha eletrônica

Figura 3.6 – Seção Entenda como Funciona

Com os dados exportados, foi necessário realizar pequenas adaptações no arquivo de acordo com o modelo exibido na seção "Formatação" da própria ferramenta, **Figura 3.7**. Essas adaptações são necessárias para que o algoritmo reconheça o fluxo de valor presente no documento, dependendo de onde o usuário recolha seus dados inúmeros formatos podem existir.

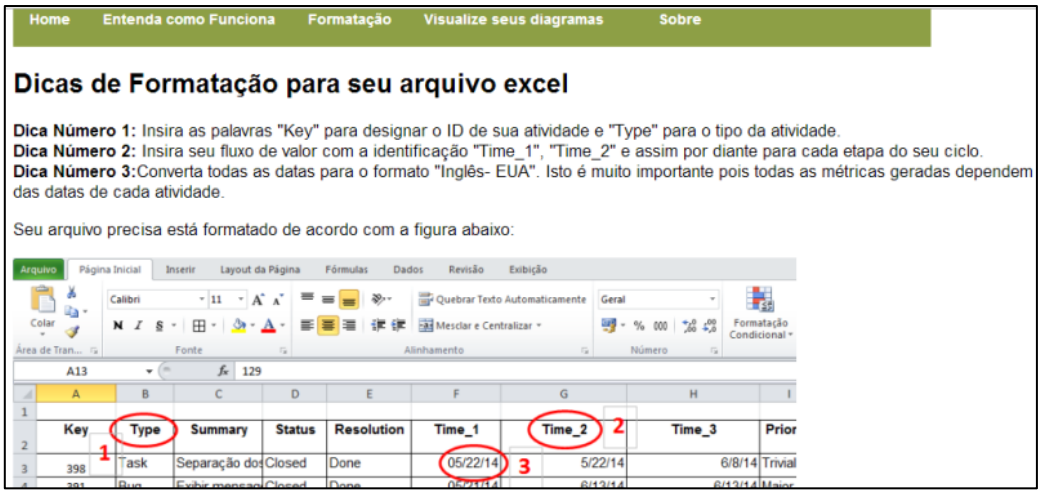

Figura 3.7 – Seção Formatação

Depois de realizadas as adaptações solicitadas pela ferramenta o usuário fez o envio do arquivo com seus dados como mostra a **Figura 3.8**.

Ao realizar o *upload,* as métricas são calculadas no *backend* da aplicação de maneira transparente para o usuário. Não é mais necessária nenhuma ação do usuário apenas acessar o menu "Visualize seus Gráficos", em cada subseção é exibido o gráfico da respectiva Métrica.

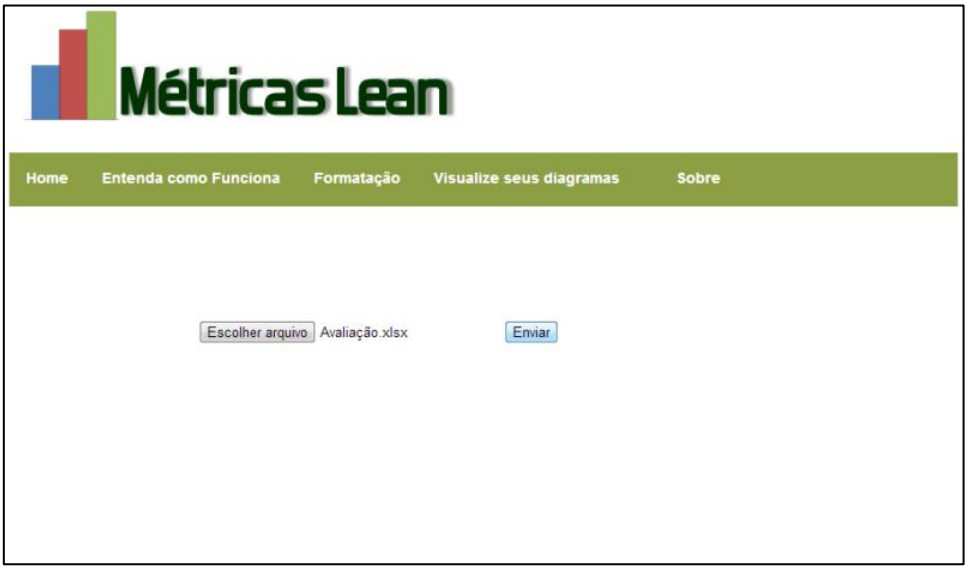

Figura 3.8 – Upload de arquivo com dados para cálculo das Métricas

A aplicação exibe o Diagrama de Fluxo Cumulativo conforme a **Figura 3.9** e permite ao usuário salvar a imagem em seu disco local. Com porte deste diagrama é possível visualizar a quantidade de trabalho em cada estado do fluxo de valor.

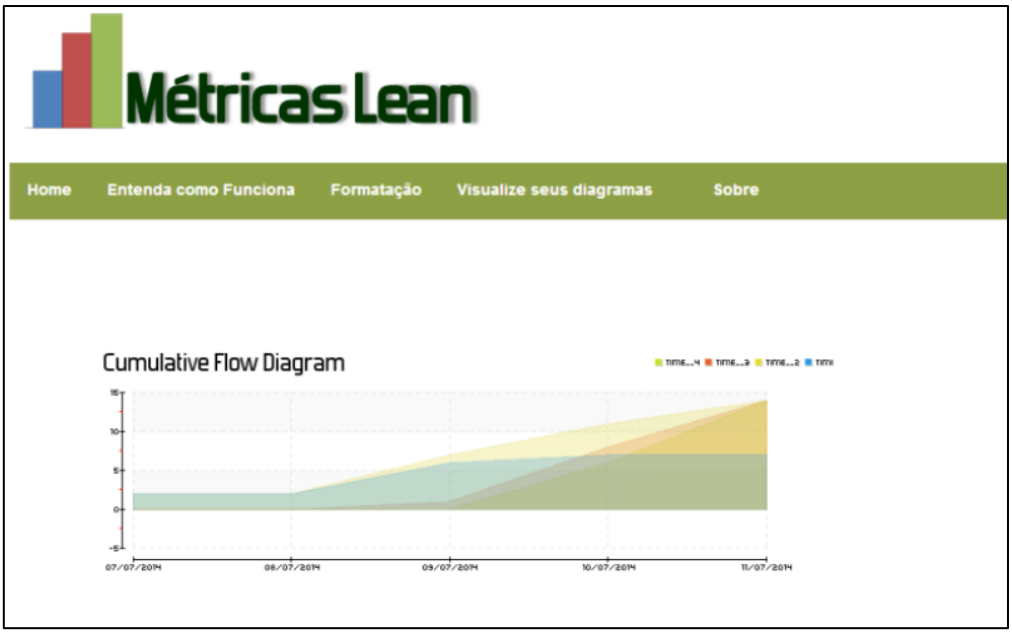

Figura 3.9 – Diagrama de Fluxo Cumulativo

Na **Figura 3.10** apresenta as taxas de *Throughput* e *Beat Time,* tais taxas demonstram a vazão do clico de desenvolvimento. Com elas é possível saber, quanto trabalho, ou demandas está sendo entregue em determinado espaço de tempo e quanto tempo é necessário para entregar determinada quantidade de trabalho ou demanda.

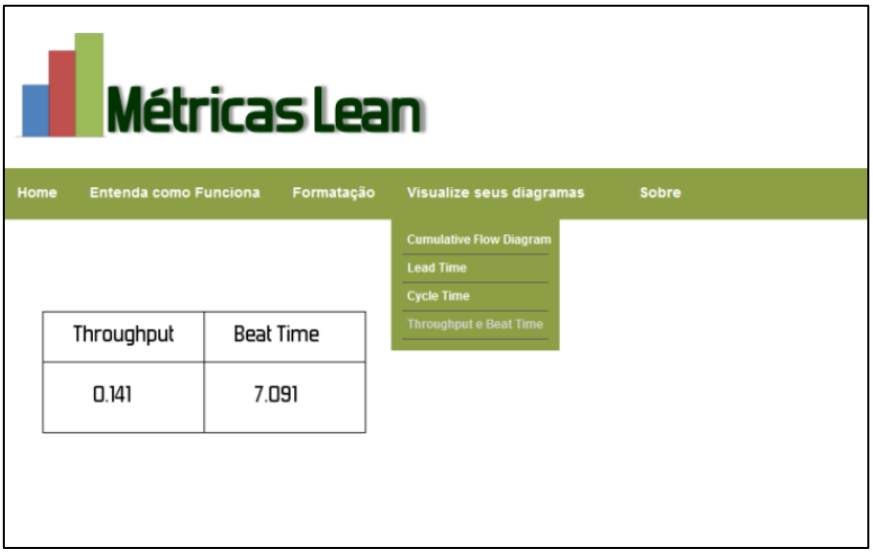

Figura 3.10 – Tela das taxas Throughput e Beat Time

O diagrama do Tempo Médio de Ciclo apresentado pela ferramenta pode ser visualizado na **Figura 3.11**. Essa métrica é importante para a redução gradual do tempo médio em que as atividades permanecem em cada fase. É possível perceber

o comprometimento da equipe à medida que o processo é melhorado e o tempo médio de ciclo é minimizado.

Figura 3.11 – Tela do Diagrama de Tempo Médio de Ciclo

A **Figura 3.12** exibe o Diagrama de Lead Time, métrica também calculada pela ferramenta. O Lead Time é importante para observar o tempo médio que cada atividade permanece em todo o ciclo. Desse modo é possível saber o quão rápido está o sistema de entrega de valor para o cliente.

Figura 3.12 – Tela do Diagrama de Lead Time

Capítulo 4

4.1 Conclusões

Este capítulo descreve a conclusão da sobre todo trabalho desenvolvido, o que inclui as contribuições obtidas e os trabalhos futuros.

4.1.1 Contribuições Obtidas

Este trabalho propôs o estudo e a construção de um produto mínimo viável para Web que forneça suporte ao cálculo de Métricas Lean. Para isso mostrou a importância dos Métodos Ágeis no desenvolvimento de *software* no cenário atual, de constantes mudanças, alta complexidade e curtos prazos de entrega.

Para cooperar com o desenvolvimento da equipe e com a melhoria contínua do processo, medir corretamente é extremamente importante, assim algumas Métricas Lean que ajudam na otimização do todo ao invés de medir individualmente foram exibidas.

Observando as ferramentas existentes na atualidade foi visto que ótimas opções existem, porém são pagas e necessitam de ser utilizadas em sua totalidade para fornecer o cálculo das métricas. A funcionalidade de calcular métricas está atrelada ao de gerenciar as atividades, por exemplo.

Com o intuito de fornecer subsídios para o cálculo das Métricas Lean, uma ferramenta foi proposta e desenvolvida, partindo do resultado do estudo e da observação das ferramentas existentes hoje. Com a pretensão de obter cenários de melhorias para o projeto, a ferramenta foi testada em um ambiente de desenvolvimento de software.

Desse modo este trabalho atingiu seus objetivos, contribuindo com o conhecimento na academia e na indústria, incialmente pelo estudo realizado e consequentemente pela ferramenta proposta e desenvolvida.

4.1.2 Trabalhos Futuros

Para a evolução deste trabalho, considerando também o resultado do teste realizado, pretende-se desenvolver algumas melhorias na ferramenta. O cálculo das métricas dividindo as atividades por suas complexidades e a diminuição de adaptações necessárias no documento são algumas das melhorias.

O desenvolvimento da uma interface amigável que proporcione fácil interação com o usuário, adição de outras métricas e permissão ao usuário selecionar as métricas cabíveis ao seu cenário.

Possibilitar maior dinamismo no momento de selecionar as datas e atividades para cálculo das métricas. Permitir a definição de datas no Diagrama de Fluxo Cumulativo é também uma melhoria para o trabalho desenvolvido.

Referências

[1] T. Poppendieck e M. Poppendieck. **Implementando o desenvolvimento Lean de software: Do conceito ao dinheiro**. Bookman, 2011.

[2] T. Poppendieck and M. Poppendieck. **Lean Software Development: An Agile Toolkit**. Addison-Wesley Professional, 2003.

[3] H. E. Júnior. **Engenharia de Software na Prática**. Novatec Editora, 2010.

[4] L. R. Galvão. **Aplicação de Métricas Lean para análise e Melhoria em Processos de Manutenção de Software**. Centro de Estudos e Sistemas Avançados do Recife – C.E.S.A.R., 2014.

[5] A. C. Fabel e H. M. Silveira. **Metodologias Ágeis no contexto de desenvolvimento de software: XP, Scrum e Lean**. Universidade Estadual de Campinas - UNICAMP, 2010.

[6] D. L. B. Filho. **Experiências com o desenvolvimento ágil**. Dissertação de Mestrado, Universidade de São Paulo – USP, 2008.

[7] Manifesto Ágil. Disponível em http://agilemanifesto.org/history.html Acessado em: 13 de Julho de 2014.

[8] M. S. Soares. **Comparação entre Metodologias Ágeis e Tradicionais para Desenvolvimento de Software**. Universidade Presidente Antônio Carlos – Unipac.

[9] K. G. S. Oliveira. **Análise do impacto da utilização de Lean e Kaban em serviços de software**. Universidade Federal Rural de Pernambuco – UFRPE, 2014.

[10] Toyota Production System. Disponível em http://www.toyota-global.com/company/vision_philosophy/toyota_production_system/jidoka.html. Acessado em: 13 de Julho de 2014.

[11] M. Poppendieck. **Lean Software Development: A Tutorial. IEEE**, 2012.

[12] M. Poppendieck. **Lean Software Development. 29th International Conference on Software Engineering**. IEEE, 2007.

[13] D. T. Sato. **Uso Eficaz de Métricas em Métodos Ágeis de Desenvolvimento de Software**. Dissertação de Mestrado. Universidade de São Paulo – USP, 2007.

[14] J. Boeg. **Kanban em 10 Passos, Otimizando o Fluxo de Trabalhos em Sistemas de entrega de Software**. InfoQ Brasil, C4Media, 2012.

[15] A. A. de Melo e M. G. F. Nascimento. **PHP Profissional, aprenda a desenvolver sistemas profissionais orientados a objetos com padrões de projeto.** Novatec Editora, 2007.

[16] Tiobe Software. Disponível em http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html. Acessado em 24 de Julho de 2014.

[17] Minitab 17. Disponível em http://www.minitab.com/pt-BR/products/minitab/features/. Acessado em 22 de Julho de 2014.

[18] Kanbanize Software. Disponível em https://kanbanize.com/ctrl_functionality. Acessado em 22 de Julho de 2014.

[19] LeanKit. Disponível em http://leankit.com/product . Acessado em 22 de Julho de 2014.

[20] Kanban Tool. Disponível em http://kanbantool.com/kanban-analytics-and-metrics. Acessado em 22 de Julho de 2014.