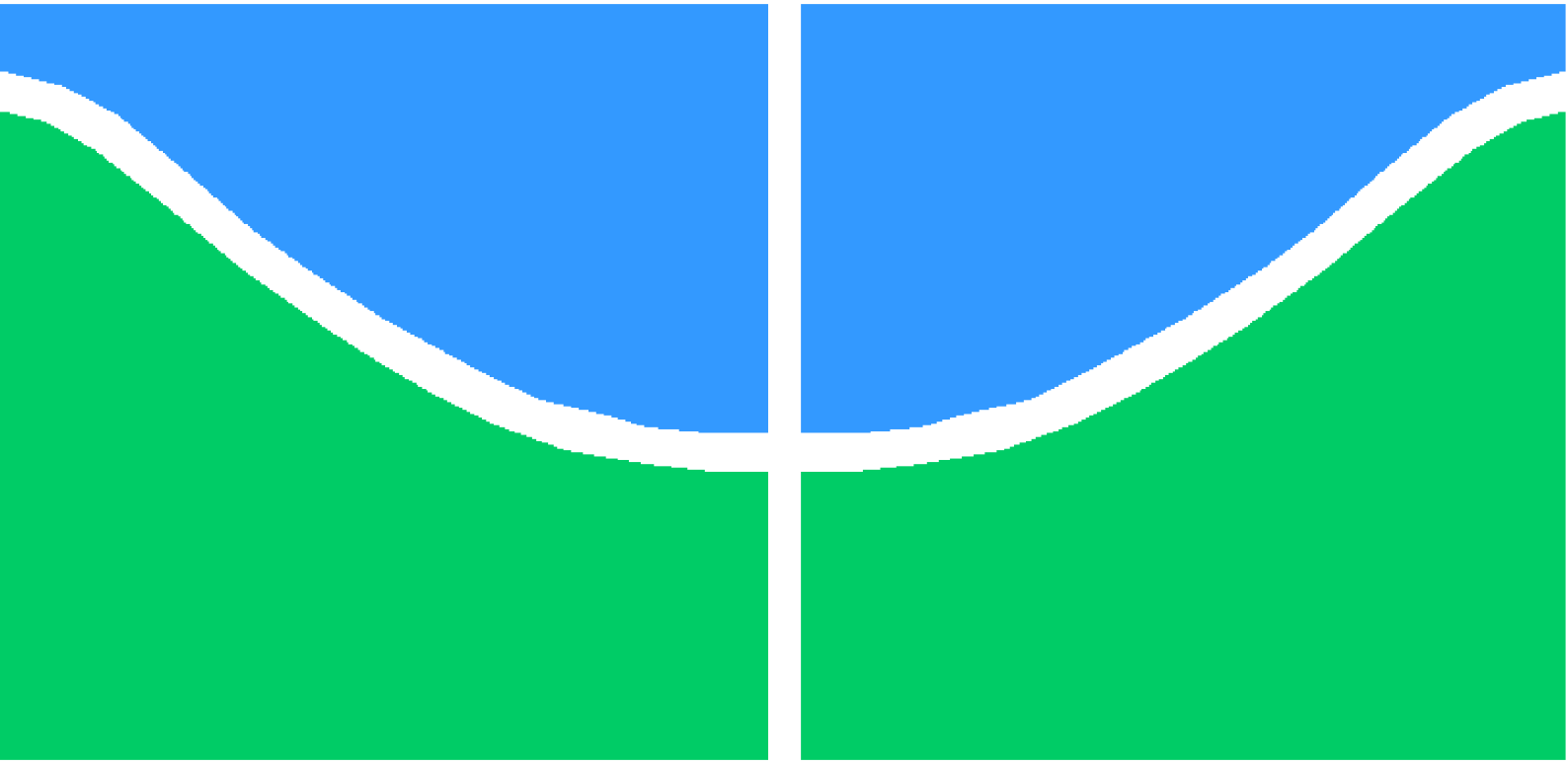

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Engenharia de Software

Comparação da acurácia de estimativas de esforço produzidas a partir de Pontos de Função Cosmic e de Story Points

Autor: Alex Cortes Alves

Orientador: Msc. Elaine Venson

Brasília, DF

2017

Alex Cortes Alves

Comparação da acurácia de estimativas de esforço produzidas a partir de Pontos de Função Cosmic e de Story Points

Monografia submetida ao curso de graduação em (Engenharia de Software) da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em (Engenharia de Software).

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Msc. Elaine Venson

Brasília, DF

2017

Alex Cortes Alves
Comparação da acurácia de estimativas de esforço produzidas a partir de Pontos de Função Cosmic e de Story Points/ Alex Cortes Alves. – Brasília, DF, 2017-
80 p. : il. (algumas color.) ; 30 cm.

Orientador: Msc. Elaine Venson

Trabalho de Conclusão de Curso – Universidade de Brasília - UnB
Faculdade UnB Gama - FGA , 2017.

1. COSMIC. 2. Story Point. I. Msc. Elaine Venson. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Comparação da acurácia de estimativas de esforço produzidas a partir de Pontos de Função Cosmic e de Story Points

CDU 02:141:005.6

Alex Cortes Alves

Comparação da acurácia de estimativas de esforço produzidas a partir de Pontos de Função Cosmic e de Story Points

Monografia submetida ao curso de graduação em (Engenharia de Software) da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em (Engenharia de Software).

Trabalho aprovado. Brasília, DF, 12 de julho de 2017:

Msc. Elaine Venson
Orientador

Dr.ª Edna Dias Canedo
Convidado 1

Msc. Ricardo Ajax Dias Kosloski
Convidado 2

Brasília, DF
2017

"It ain't about how hard you hit,
it's about how hard you can get hit
and keep moving forward.
How much you can take and keep moving forward.
That's how winning is done!
(Rocky Balboa)

Resumo

Em projetos de desenvolvimento de software ágil, os *Story Points* são geralmente utilizados como uma forma de estimar o esforço das atividades. Os *Story Points* auxiliam uma equipe na escolha de tarefas que serão executadas em uma *Sprint*, tendo o *velocity* dessa equipe, é possível tentar estimar quantos *Story Points* podem ser executados dentro de uma *Sprint*. Porém, tal técnica não permite definir o tamanho real do sistema, uma vez que a pontuação é particular de cada projeto e equipe de desenvolvimento, logo, não é possível realizar um *Benchmarking* dos projetos dentro ou fora de uma empresa. Técnicas de medição funcional como o Ponto de Função COSMIC possuem padrões definidos, fazendo com que o tamanho do software medido não dependa da opinião da equipe ou da experiência do medidor. A proposta desse trabalho é realizar um estudo de caso para comparar a acurácia das estimativas de esforço geradas a partir dessas duas técnicas de medição. Comparando os valores de estimativa e os valores reais após o desenvolvimento das atividades.

Palavras-chaves: Estimativa; Esforço; Story Point; COSMIC.

Abstract

In Agile Software Development projects, the Story Points are mainly used as a way of estimating effort of activities. The Story Points can help a team choosing the tasks that will be executed in an given Sprint, with this team's Velocity, it is possible to estimate how many Story Points can be executed within the Sprint. However, this technique can't define a software's funtional size, since this Story Points are private of each project and development team, ergo, it's not possible to Benchmark the projects inside or outside the company. Funtional Size Measurement techniques like COSMIC Function Point define patterns, wich means that the functional size of the software being measured do not relies on expert opinion. The propose of this study is a case study comparing the effort estimates accuracy of this two techniques. Comparing estimated values and the actual effort after the completition of the selected activities.

Key-words: Estimative; Effort; Story Point; COSMIC.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

COSMIC	Common Software Measurement International Consortium
ISO	International Organization for Standardization
LOC	Lines Of Code
PFC	Ponto de Função COSMIC
RUP	Rational Unified Process
SP	Story Point
US	User Story

Sumário

Capítulo 1
Introdução

1 Introdução

1.1 Considerações iniciais do capítulo

Esse capítulo apresenta o contexto, a proposta e o objetivo do trabalho, bem como a metologia abordada durante a execução do trabalho.

1.2 Contexto

Estimativas de custo e esforço de software são atividades cruciais no ciclo de vida de desenvolvimento de um software. Estimativas são as entradas-chave para alocação de recursos e definição de prazos. A falta de acurácia nas estimativas é responsável por um grande número de problemas relacionados ao gerenciamento de projetos (PEIXOTO; AUDY; PRIKLADNICKI, 2010).

Por ter uma natureza volátil e flexível, as metodologias ágeis estão sendo usadas em um grande número de projetos na indústria. No desenvolvimento ágil, o software é desenvolvido em pequenas iterações e as mudanças são bem vindas. Devido a essa natureza dinâmica, se torna muito difícil estimar custo e prazo de projeto em ambientes de desenvolvimento ágil. Os modelos de estimativa utilizados em tais metodologias não possuem nenhuma fórmula matemática para estimar esforço e custo, logo, se tornam pouco eficientes (POPLI; CHAUHAN, 2014).

Em metodologias ágeis, os requisitos são sempre descritos em *User Stories*. A *User Story* é um requisito independente, negociável, estimável, pequeno e testável. E as estimativas nessas metodologias são feitas com base nesses requisitos (POPLI; CHAUAHN, 2015).

Story Points é uma medida relativa muito usada para estimar tamanho de software em Ágil. Os times decidem o quanto vale um ponto, e pontuam as *User Stories* nas quais irão trabalhar durante a sprint atual. Porém, um time pode pontuar atividades diferentemente de outro time, não sendo possível comparar atividades executadas por times diferentes (HAMOUDA, 2014).

Além do Story Point, existe um número de padrões internacionais que disponibilizam métodos de medição funcional bem documentados independente da tecnologia adotada ou do processo de desenvolvimento (COMMEYNE; ABRAN; DJOUAB, 2016). Segundo Ungan, Çizmeli e Demirörs (2014) o Ponto de Função COSMIC é uma das técnicas de medição funcional mais recente e popular, tanto na academia quanto na indústria. O software é medido em um nível de funcionalidade padronizado chamado de Processo Funcional (UNGAN; ÇIZMELI; DEMIRöRS, 2014).

O Ponto de Função COSMIC (PFC) é um padrão internacional para medida de requisitos funcionais de software. Como resultado, ele fornece um número que representa o tamanho funcional do software e pode ser usado para comparação com outros sistemas e equipes. Como exigido pela ISO, o COSMIC foi desenvolvido para ser independente de decisões de implementações contidas em qualquer artefato operacional do software. Isso significa que o tamanho do software pode ser medido tanto de um sistema já desenvolvido quanto dos requisitos de um sistema que será implementado (COMMEYNE; ABRAN; DJOUAB, 2016).

1.3 Problema

Segundo (COHN, 2005) o Story Point é uma unidade de medida que expressa o tamanho geral de uma User Story, funcionalidade ou outra peça de trabalho e não existe uma fórmula para definição do tamanho de uma história. O importante é que os valores sejam relativos, sendo que uma história definida como dois Story Points deve ser o dobro de uma história pontuada com apenas um.

O valor de Story Points significa que a equipe concorda em atribuir tais pontos a determinada User Story. Porém, os Story Points não são padronizados como uma métrica, e equipes diferentes podem não pontuar uma mesma User Story igualmente (FEHLMANN; SANTILLO, 2010).

O Story Point não indica explicitamente o tamanho do software desenvolvido, logo, não é possível determinar os níveis de produtividade alcançados em um projeto, e nem comparar a performance entre diferentes projetos e organizações (COMMEYNE; ABRAN; DJOUAB, 2016).

A questão de pesquisa desse estudo é:

Em um contexto ágil, as estimativas em Pontos de Função Cosmic geram resultados mais confiáveis do que estimativas geradas em Story Points?

1.4 Objetivo

O objetivo geral deste trabalho é realizar uma comparação da acurácia de duas técnicas de estimativa de esforço, selecionando atividades pontuadas por uma equipe ágil em *Story Points* e calculando o valor em Pontos de Função Cosmic das mesmas atividades.

Os objetivos específicos são:

- Comparar estimativas em PFC e SP;
- Medir e avaliar a acurácia dos dois métodos em contexto educacional.

1.5 Metodologia

A pesquisa proposta neste trabalho possui uma natureza aplicada, a qual tem como objetivo gerar conhecimentos para aplicação prática dirigidos à solução de problemas específicos (MORESI et al., 2003).

Do ponto de vista da abordagem, foi definida uma abordagem quantitativa, onde os resultados são traduzidos em números e requerem uso de recursos e técnicas de estatística (MORESI et al., 2003). No estudo proposto será utilizada a técnica de regressão linear, para comparar a acurácia das técnicas de estimativa de esforço descritas.

Os procedimentos técnicos selecionados para execução do trabalho são pesquisa bibliográfica e estudo de caso.

- **Pesquisa Bibliográfica:** Essa pesquisa é desenvolvida com base em material já elaborado, constituído principalmente de livros e artigos científicos (GIL, 2002). Serão pesquisados periódicos e publicações de conferências nas

principais bases científicas afim de dar suporte ao referencial teórico do trabalho.

- **Estudo de Caso:** É um método de pesquisa de campo, no qual se investiga os fenômenos à medida em que acontecem, sem que o pesquisador interfira de forma significativa (FIDEL, 1984). Este estudo de caso foi executado junto à Fábrica de Software do campus gama da Universidade de Brasília.

O estudo de caso é usado com frequência em áreas como psicologia, sociologia, ciência política, serviço social e administração. Nessas áreas o estudo de caso tem como objetivo não só aumentar o conhecimento a respeito de algo, mas também propor mudanças ao fenômeno que está sendo estudado (RUNESON et al., 2012).

Na área de Sistemas da Informação, o estudo de caso também já é mais maduro quando comparado com a Engenharia de Software. Runeson et al. (2012) apresenta propriedades dos objetos de estudo em Engenharia de Software que diferenciam de estudos em Sistemas da Informação:

- Eles são corporações privadas ou órgãos públicos que estão *desenvolvendo* software, e não que estão *utilizando-o*;
- Eles são organizações orientadas a *projeto*, e não orientadas a *linhas* ou *funções*;
- Os trabalhos em estudo são *trabalhos de engenharia avançados* realizados por pessoas altamente capacitadas, ao contrário de *trabalhos de rotina*
- Existe um objetivo de melhorar práticas de engenharia, o que significa que existe um componente de pesquisa de criação.

1.6 Organização do trabalho

Esse trabalho é organizado em quatro partes , a segunda parte apresenta conceitos sobre Estimativas de Software, a segunda parte apresenta os *Story Points*

e como pontuá-los, a terceira parte apresenta os Pontos de Função COSCMIC e o processo a ser seguido para medir um software em PFC e a última parte apresenta os resultados obtidos ao final do estudo.

Capítulo 2
Estimativa de Software

2 Estimativa de Software

2.1 Considerações Iniciais do Capítulo

Este capítulo apresenta conceitos pertinentes à área de estimativa de software, como, Medição, Métrica, Tamanho Funcional, Estimativa de Esforço e as definições de Acurácia e Precisão em estimativa de software.

2.2 Medição de Software

Medição é o processo em que números ou símbolos são atribuídos a atributos de entidades do mundo real de modo a descrevê-las de acordo com regras bem definidas. Ou seja, a medição deve capturar informações sobre atributos de uma entidade a fim de poder descrevê-la (FENTON; BIEMAN, 2014).

A medição de software é um mecanismo ideal para se obter *feedback* e avaliação. A medição e as informações levantadas são úteis para todos os envolvidos como desenvolvedores, gerentes, clientes e a própria organização (BASILI, 1992).

A medição de software é um método para se determinar quantitativamente a extensão de processos, produtos ou projetos de software. Como a indústria de software possui problemas com produtividade e qualidade, os engenheiros reconhecem a necessidade de entender e melhorar o processo de desenvolvimento para que seja possível aplicar mudanças necessárias para obter melhores níveis de produtividade e qualidade (GOPAL et al., 2002).

Na engenharia de software, a medição é uma atividade essencial. Muitos desenvolvedores de software medem características de seus software a fim de entender se os requisitos são consistentes e completos, assegurar que o *design* possui uma boa qualidade ou até saber se o software está pronto para ser entregue. Um gerente de projeto pode usar a medição para controlar prazos e custos, uma organização pode realizar medições de avaliação de seu processo de desenvolvimento ou o próprio cliente pode medir o produto entregue para avaliar se o mesmo atende a

todos os requisitos (FENTON; BIEMAN, 2014).

O tamanho funcional de um software é utilizado para avaliar e prever alguns aspectos do processo de produção como o esforço gasto em uma atividade, o custo de um sistema ou a produtividade de uma equipe de desenvolvimento. Para se medir o tamanho funcional de um sistema existem duas abordagens principais, a abordagem posterior, como *Lines Of Code* (LOC), e a abordagem anterior, que são técnicas que estimam o tamanho de um sistema baseadas nos requisitos definidos (TRAN-CAO; LEVESQUE; ABRAN, 2002).

2.3 Tamanho Funcional

O tamanho funcional de um sistema é obtido a partir da perspectiva do usuário sobre o sistema desenvolvido.

Métodos de medição de tamanho funcional como Análise de Ponto de Função e COSMIC se tornaram populares ao longo dos anos. Tais métodos apresentam vantagens sobre outras técnicas, como por exemplo, podem ser executados em fases iniciais de um projeto e são independentes de tecnologias e linguagens de programação (UNGAN; ÇIZMELI; DEMIRöRS, 2014).

- **Análise de Ponto de Função -** Publicada por Albrecht (1979), a técnica da Análise de Ponto de Função quantifica as funções contidas em um software que sejam significativas para o usuário. Ela é relacionada diretamente com os requisitos de usuário. Usando um conjunto de critérios padronizados, cada uma das funções é um índice numérico de acordo com seu tipo e complexidade. Esses índices são somados resultando em uma medida inicial de tamanho que será normalizada incorporando fatores relacionados ao software como um todo (ABOUT...,).

- **Ponto de Função COSMIC -** O método faz parte da segunda geração de medidas de tamanho funcional e fornece princípios, regras e um processo para medir um tamanho funcional padronizado. O COSMIC é associado aos movimentos de dados do sistema (*Entry, Write, Read e Exit*), em que cada

movimento de dados realizado pelo sistema deve ser contado como um PFC (Ponto de Função COSMIC) (COSMIC. . . ,).

Embora não seja considerada uma medida de tamanho funcional, em metodologias ágeis, o *Story Point* é uma unidade de medida que expressa o tamanho geral de uma *User Story*, uma *feature* ou outro tipo de peça de trabalho. Para se estimar tamanho em *Story Point* deve-se atribuir uma pontuação a uma devida atividade de trabalho, essa pontuação não é importante, o importante é que as pontuações das outras atividades sejam relativas a essa. Uma atividade de dois *Story Points* deve demandar o dobro de esforço de uma atividade de apenas um *Story Point* (COHN, 2005).

2.4 Estimativa de Esforço

Estimativa de esforço de software é o processo de previsão do esforço que será necessário para desenvolver o sistema (WEN et al., 2012). Este é um elemento crítico na gestão de projetos, o uso inadequado de estimativas de esforço pode levar a falhas no projeto como prazos definidos inadequadamente levando a atrasos na entrega (TARIQ et al., 2015).

Estimar o custo de um projeto de software baseado no esforço é umas das atividades mais importantes na gerência de projetos de software. Isso porque o planejamento, monitoramento e controle rigorosos do projeto não são possíveis se existirem grandes falhas na estimativa inicial. Porém, ainda não existe um modelo de estimativa de esforço foi provado consistentemente em prever o esforço de projetos de software em todas as situações (IDRI; AMAZAL; ABRAN, 2015).

Embora seja comum o uso de modelos de estimativa de esforço formais em muitas das disciplinas da engenharia, o método de estimativa de esforço prevalente em desenvolvimento de software é o baseado em julgamento, porém existem fortes evidências que as estimativas baseadas em julgamento tendem facilmente ao superotimismo (JøRGENSEN, 2016).

A partir de revisão de publicações Jorgensen (2014) descreve alguns fatos conhecidos e desconhecidos envolvendo estimativa de esforço em engenharia de

software:

- Fatos conhecidos:
  - **Não existe um "melhor"modelo ou método -** Vários estudos comparam a acurácia de modelos e métodos de estimativa e possuem grande variação de qual dos modelos possui melhor desempenho. Uma das principais razões para essa falta de estabilidade nos resultados é que as várias relações centrais, como a relação de esforço de desenvolvimento e tamanho do projeto, variam de contexto para contexto.
  - **O foco do cliente em baixos preços é a principal razão para excesso de esforço em projetos -** A tendência em subestimar o esforço de um projeto é presente particularmente em situações de concorrência de orçamento, como licitações. Em contextos menos competitivos não existe essa tendência, pode-se até observar o oposto.
  - **Intervalos de esforço mínimo e máximo são muito imitados -** Intervalos de esforço mínimo-máximo, como o intervalo de 90 por cento de confiança, são sistematicamente muito limitados para refletir a incerteza do esforço.
  - **É fácil se enganar na estimativa de trabalho e difícil de se recuperar do engano -** Talvez a maior parte do engano ocorra quando os responsáveis pela estimativa tem ciência do orçamento, expectativas do cliente, tempo disponível ou outros fatores que podem agir como âncoras de estimativas.
  - **Dados históricos relevantes e *checklists* melhoram a acurácia de estimativas -** Quando os dados históricos são relevantes e o *checklist* é adaptado para a companhia, é menos provável que atividades sejam esquecidas, e mais provável que planos de contingência aos riscos sejam adicionados baseados em experiências passadas.
  - **Combinar estimativas independentes aumenta a acurácia das estimativas -** Uma suposição-chave para a melhoria da acurácia é que as estimativas independentes possuem fontes com diferentes especialidades, experiências e processos de estimativas.

  - **Estimativas podem ser prejudiciais -** Estimativas não só preveem o futuro, como também podem afetá-lo. Estimativas baixas podem afetar a qualidade do projeto assim como gerar retrabalho em fases posteriores. Estimativas altas podem gerar baixa produtividade de acordo com a lei de Parkinson, que afirma que o trabalho se expande para preencher o tempo disponível.

- Fatos desconhecidos:

  - **Como estimar projetos super-grandes e complicados precisamente -** Mega-projetos demandam uma atenção extra à estimativa de esforço. Não só porque existem mais valores em risco, mas também existem menos dados históricos relevantes disponíveis. Muitas das atividades típicas de projetos desse porte, como demandas organizacionais com diversos *stakeholders* envolvidos, são muito complexas de se estimar porque normalmente envolvem mudanças de processos de negócio, e interações complexas com *stakeholders* e com outros sistemas já existentes.

  - **Como medir tamanho e complexidade do software para uma estimativa precisa -** Apesar de anos de pesquisa em medição de tamanho e complexidade de software nenhuma das medições propostas é muito boa quando se trata de estimativa de esforço. Alguns contextos de tamanho e complexidade podem possibilitar estimativas de esforço precisas, porém são raros.

  - **Como medir e prever produtividade -** Mesmo tento boas medições para tamanho e complexidade de software, seria necessário prever a produtividade de indivíduos ou equipes responsáveis pelo projeto. Essa predição é complicada por uma diferença surpreendentemente grande na produtividade entre desenvolvedores de software e times. Não existe um bom método para esse tipo de predição

2.5 Acurácia de Estimativa

2.5.1 Acurácia x Precisão

A Accuracy... (1994) define acurácia e precisão como:

- **Acurácia -** O grau de concordância entre o resultado experimental e o valor de referência aceito;
- **Precisão -** O grau de concordância entre resultados de testes independentes obtidos em condições estipuladas.

Em outras palavras, e no contexto de Estimativa de Software, o grau de acurácia é o quanto uma estimativa se aproxima do valor real. E o grau de precisão é o quanto o resultado de uma estimativa se aproxima de outros resultados da mesma estimativa, ou seja, espera-se que uma estimativa com alto grau de precisão obtenha os mesmo resultados sempre mesmo que não sejam próximos do valor real.

As imagens apresentadas na Fig. 1 representam graficamente os conceitos de precisão e acurácia:

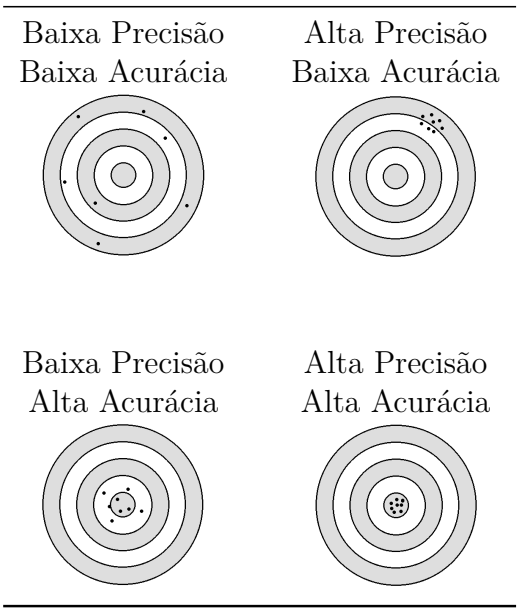

Figura 1 – Acurácia x Precisão. Fonte: Autor

O primeiro quadrante possui baixa precisão, pois os resultados não são próximos entre si, e possui baixa acurácia pois os resultados não estão próximos do alvo central.

O segundo quadrante possui alta precisão, pois os resultados apresentam valores próximos entre si, porém ainda não possuem valores próximos ao alvo centra, indicando baixa acurácia.

O terceiro quadrante apresenta resultados próximos ao alvo central, o que indica alta acurácia, porém ainda não apresenta resultados próximos entre si.

O quarto quadrante apresenta alto grau de precisão e de acurácia, pois, além dos resultados alcançarem aproximadamente os mesmos valores em todas as tentativas, eles também apresentam valores próximos ao alvo central.

2.5.2 Acurácia dos métodos de estimativa

A acurácia de uma medida depende do instrumento utilizado para se realizar a media, assim como as definições da medição (FENTON; BIEMAN, 2014).Usman et al. (2014) estudou as publicações que utilizavam técnicas de estimativa de software em projetos ágeis. Nesse estudo ele utilizou o MMRE como métrica para analisar a acurácia das técnicas de estimativa.

Nos resultados apresentados por Usman et al. (2014), o *Planning Poker*, mesmo sendo uma das técnicas mais utilizadas, apresenta acurácia de 48%, valor baixo quando comparado com estudos que utilizaram a Regressão Linear, que variam entre 66% e 90% de acurácia.

As acurácias de métodos de estimativa levantadas por Usman et al. (2014) são apresentadas na Tab. 1.

Tabela 1: Valores de acurácia alcançados. Fonte: (USMAN et al., 2014))

Técnica	**Acurácia alcançada % (MMRE)**
Planning Poker	48.0
Pontos de Caso de Uso (UCP)	10.0 - 21.0
Variações do UCP	2.0 - 11.0
Opinião de especialista	28.0 - 38.0
Regressão Linear	66.0 - 90.0
Redes Neurais	6.0 - 90.0

Capítulo 3
Story Points

3 Story Points

3.1 Considerações Iniciais do Capítulo

Este capítulo apresenta os conceitos básicos do planejamento em metodologias ágeis, mostrando o que são os *Story Points* e como é realizado o processo de planejamento *Planning Poker*.

3.2 Planejamento Ágil

O *Extreme Programming* possui dois níveis de planejamento, o planejamento de iteração e o planejamento da *release*. O planejamento da iteração é de curto prazo e muito detalhado, esse é um bom momento para se entrar em detalhes sobre como implementar uma história. O planejamento da *release* é de alto nível de longo alcance, nesse planejamento o time não se preocupa com a precisão dos detalhes e foca mais em uma visão geral do projeto (GRENNING, 2002).

Planning Poker combina a opinião da equipe, analogia e desagregação em uma atividade simples com estimativas rápidas, porém, confiáveis (COHN, 2005). Grenning (2002) afirma que a estimativa de histórias possui menos acurácia do que estimativas convencionais, porém, é melhor possuir um conjunto de estimativas com acurácia razoável do que apenas um par de estimativas precisas. E, com o tempo, a equipe deve se tornar melhor e confiar em sua intuição ao estimar atividades (GRENNING, 2002).

O *Story Point* é uma unidade de medida que expressa o tamanho de uma *Story*, esses pontos são resultados do *Planning Poker* realizado na reunião de planejamento em metodologias ágeis (COHN, 2005).

3.3 Planning Poker

O *Planning Poker* é considerada uma técnica de estimativa em grupo construída no princípio de que mais cabeças são melhores do que uma (MAHNIČ; HOVELJA, 2012). Tem como objetivo gerar estimativas confiáveis é rápidas utilizando analogia entre as atividades e a opinião da equipe envolvida no desenvolvimento do projeto (COHN, 2005).

É realizado em dois momentos do projeto, no planejamento da *Release*, onde são estimadas todas as atividades identificadas antes do início do projeto, ainda sem grande detalhamento e com pouca precisão. E no planejamento de cada *Sprint*, onde são estimadas apenas as atividades da *Sprint* que irá começar, neste ponto já são discutidos detalhes da implementação das *User Stories*, como tem foco maior nos detalhes, a precisão da estimativa da *Sprint* tende a ser maior (COHN, 2005) (GRENNING, 2002).

Todo o time envolvido no projeto deve fazer parte do *Planning Poker*, incluindo programadores, testadores, engenheiros, analistas e *designers*, isso faz com que o planejamento tenha vários pontos de vista, gerando uma estimativa mais confiável (COHN, 2005).

É considerado que o *Planning Poker* gera melhores estimativas do que técnicas baseadas em opiniões individuais de especialistas por garantir que todos os desenvolvedores participem igualmente do processo. A participação de pessoas com pontos de vista diferentes ajuda a reduzir o super-otimismo de outras estimativas, por identificar mais problemas que podem afetar o desenvolvimento do projeto (MAHNIČ; HOVELJA, 2012).

(GRENNING, 2002) descreve as atividades do *Planning Poker* de acordo com a Fig. 2.

- **Seleção da *User Story* -** As histórias que farão parte da *Sprint* que está sendo planejada serão pontuadas e discutidas individualmente. Nessa etapa é selecionada qual *User Story* será pontuada no momento.

- **Leitura da User Story -** A *User Story* é lida e explicada pelo cliente.

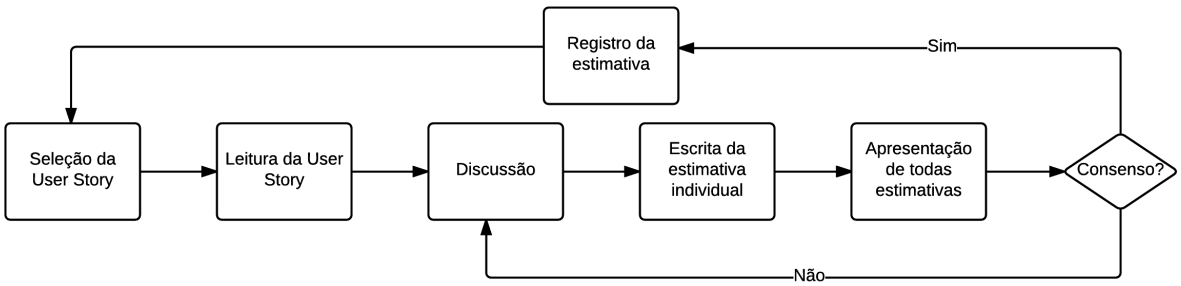

Figura 2 – Sequência de atividades do *Planning Poker*. Fonte: (GRENNING, 2002), Adaptado

- **Discussão -** A equipe de desenvolvimento deve, então, discutir os detalhes da *User Story*, apontar as visões individuais dos membros, e quais podem ser as eventuais dificuldades durante a implementação.

- **Escrita da estimativa individual -** Cada participante do *Planning Poker* deve escrever o valor de sua estimativa em *Story Points* ainda sem que os outros participantes vejam. Os valores que podem ser escolhidos são pré-definidos em 0, 1, 2, 3, 5, 8, 13, 20, 40 e 100.

- **Apresentação de todas as estimativas -** Assim que todos os participantes fizeram suas estimativas, todos apresentam os valores escolhidos ao mesmo tempo.

- **Registro da estimativa -** Assim que o time alcançar um consenso, a estimativa é registrada e é selecionada uma nova *User Story* para ser pontuada.

3.4 Story Points

O *Story Point* é uma medida relativa muito utilizada como estimativa em metodologias ágeis, o time decide o quanto vale um ponto e, baseado nesse tamanho, determina quantos pontos cada atividade possui (HAMOUDA, 2014).

Existem duas formas para se basear o tamanho de um *Story Point*, no primeiro, deve-se selecionar a menor atividade do *backlog*, e atibuir a essa o valor de 1 *Story Point*. No segundo, se é esperado que as histórias tenham tamanho entre 1 e 10 *Story Points*, por exemplo, é escolhida uma história considerada de complexidade média e atribuído um valor mediano dentro desse intervalo. E, a partir dessa definição inicial do *Story Point*, é executada a técnica de *Planning Poker* para se estimar as outras atividades da *release* ou *Sprint* (COHN, 2005).

Para se obter um planejamento mais eficiente, é importante que o tamanho estimado das histórias mantenha a relatividade, o ideal é que uma história pontuada com dois *Story Points* demande o dobro de esforço de uma atividade pontuada com apenas um (COHN, 2005). Ao decorrer do projeto a equipe se torna mais experiente para estimativa e mais intuitiva será a pontuação (GRENNING, 2002).

Segundo Cohn (2005) o *Story Point* é puramente uma medida de tamanho do trabalho a ser executado. A duração de um projeto não é estimada, e sim derivada do número total de *Story Points* dividido pelo *Velocity* da equipe de desenvolvimento (COHN, 2005).

3.4.1 Velocity

O *Velocity* é o número de histórias ou *Story Points* que a equipe pode executar no tempo de uma *Sprint*. Uma das maiores dificuldades no planejamento é prever o *velocity* do time, para isso o *Velocity* é gerado baseado em dados históricos do projeto, então o planejamento das *Sprints* é feito com base na quantidade de pontos concluídos em *Sprints* anteriores (OMANOVIC; BUZA, 2013).

No *Scrum*, o *velocity* é o termo que representa quantos *Story Points* a equipe é capaz de entregar em uma *Sprint*, ele é obtido observando *Sprints* anteriores e assumindo que a produtividade do time permanece constante (SZALVAY, 2007). Com valores constantes de *Velocity* pode-se assumir que o planejamento e a execução das *Sprints* estão bem alinhados (OMANOVIC; BUZA, 2013).

Capítulo 4
Pontos de Função COSMIC

4 Pontos de Função COSMIC

4.1 Considerações Iniciais do Capítulo

Esta capítulo apresenta um resumo sobre COSMIC em metodologias ágeis, bem como os conceitos e o processo para realização de uma estimativa em Pontos de Função COSMIC.

O COSMIC... (2014) define três fases para se realizar a medição em PFC:

- **Fase de Estratégia de Medição**;
- **Fase de Mapeamento**;
- **Fase de Medição**;

4.2 Medindo Projetos Ágeis com COSMIC

Medir projetos ágeis requer conhecimento sobre o método COSCMIC e, também, uma orientação para o tipo de aplicação que está sendo desenvolvida ou evoluída (TRUDEL; BUGLIONE, 2010).

A seguir são apresentados os desafios e os guias para medição de projetos ágeis, baseado em Trudel e Buglione (2010)

- ***Benchmarking* -** Para o propósito do *Benchmarking*, o tamanho funcional é considerado o tamanho do software entregue ao final do projeto, independentemente se o método de desenvolvimento ou de gerência foram ágeis ou não. Logo, seria de razoável que um projeto ágil fornecesse o tamanho funcional do sistema ao final do projeto;
- **Estimativa inicial e orçamento -** O primeiro desafio do gerente de um projeto ágil é obter uma estimativa inicial para um escopo definido. Em uma

etapa inicial do projeto, esse escopo pode ser definido como a lista de US com informações suficientes para se obter uma lista de funcionalidades;

- **Planejamento de Iteração e Reestimativa de Projetos -** Para um time ágil monitorar o *Velocity* em PFC por iteração, é necessário fazer uma medição de tamanho funcional ao fim de cada iteração contando cada movimento de dados que tenha sido adicionado, modificado ou retirado durante a iteração;

- **Monitoramento de Melhoria de Processo -** Times ágeis aplicam melhoria de processo contínua através de uma prática chamada "retrospectiva"que é realizada ao final de toda iteração. Um time ágil poderia monitorar seu *Velocity* em PFC entregue a cada iteração para entender melhor a eficiência de suas sessões de retrospectivas.

4.3 Fase de Estratégia de Medição

Essa fase descreve alguns parâmetros que devem ser identificados antes de se começar a medição em si:

- O *propósito da medição*;
- O *escopo geral* do software que será medido;
- Os *usuários funcionais* de cada parte do software;
- O *nível de granularidade* dos artefatos do software disponíveis;

O processo para determinação da estratégia de definição apresentada pelo COSMIC... (2014) é ilustrado na Fig. 3.

4.3.1 Determinar o Propósito da Medição

O COSMIC... (2014) diz que "Propósito de Medição"é a definição do porque uma medição é necessária e como o resultado dessa será usado.

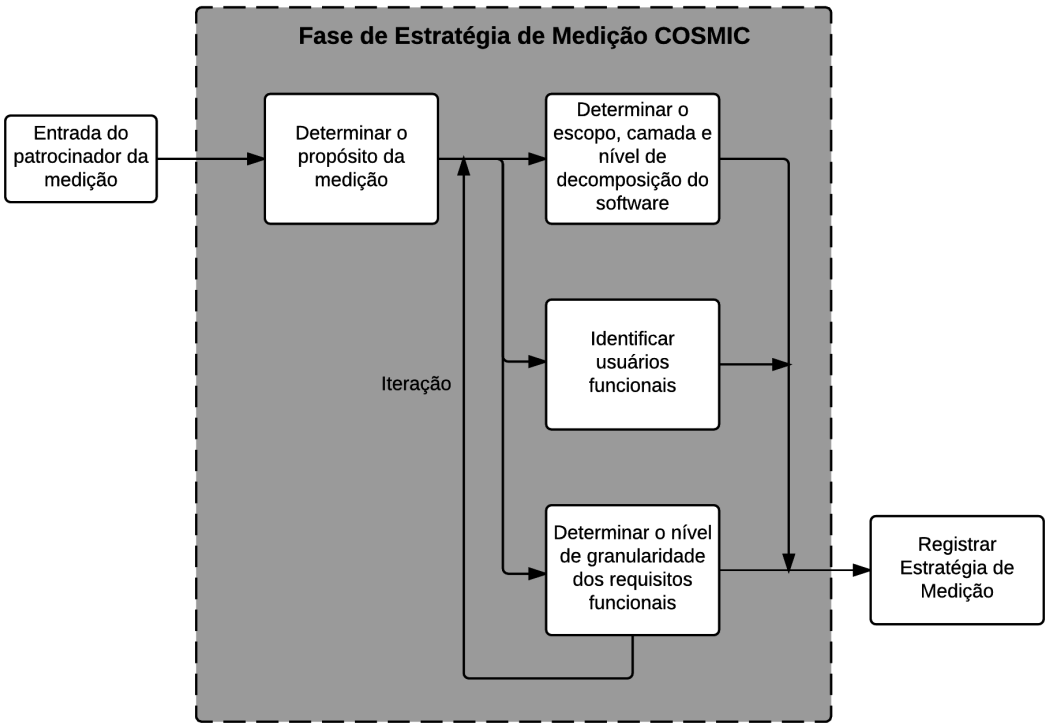

Figura 3 – Processo da Estratégia de Medição. Fonte: (COSMIC. . . , 2014)

Ao identificar o propósito da medição é possível definir, por exemplo, em que momento a medição será realizada e quais os artefatos serão necessários. No caso de estar medindo a fim se obter uma estimativa para elaborar cronograma ou fazer orçamento, o software deve ser medido ao início do ciclo de vida do projeto utilizando as especificações de requisitos funcionais do sistema.

O propósito da medição ajuda a identificar itens que serão necessários ao decorrer da atividade de medição do software:

- O escopo a ser medido e os artefatos necessários;
- Os usuários funcionais do sistema;
- Em qual momento do ciclo de vida do desenvolvimento a medição ocorrerá;

- A acurácia necessária da medição. Com isso é possível identificar se o método COSMIC padrão é suficiente ou deve-se usar uma versão de aproximação do método COSMIC.

4.3.2 Determinar o Escopo da Medição

Para o COSMIC... (2014) o "Escopo da Medição"é o conjunto de requisitos funcionais do usuário que serão inclusos na atividade de medição. Especificamente para o COSMIC, a expressão "escopo"se refere a uma parte do software que será medida individualmente.

O escopo de uma parte do software a ser medido deve ser obtido na fase de "Propósito da Medição"e não deve se estender a mais de uma camada do software.

O software definido como um escopo geral pode ser subdividido em pedaços individuais de software cada um com seu próprio escopo de medição. Alguns motivos para se fazer a subdivisão são o software está em mais de uma camada, responsabilidades organizacionais diferentes ou a necessidade de se distinguir alguns itens entregáveis.

4.3.3 Identificar os Usuários Funcionais

Diferentes tipos de usuários podem interagir com diferentes requisitos funcionais do sistema, logo, o tamanho funcional irá variar com a escolha dos usuários funcionais da medição (COSMIC..., 2014).

O usuário funcional é um tipo de usuário que faz parte dos requisitos funcionais de um software recebendo e/ou enviando dados ao sistema (COSMIC..., 2014). No método COSMIC é essencial separar os usuários funcionais dos outros possíveis usuários do sistema.

4.3.4 Identificar o Nível de Granularidade

Nas fases iniciais do desenvolvimento de um projeto de software os requisitos são especificados em alto nível e com poucos detalhes. Na medida em que o projeto avança os requisitos são refinados alcançando um menor nível e revelando

maiores detalhes do software. Esses diferentes níveis de detalhes dos requisitos são conhecidos como diferentes níveis de granularidade (COSMIC. . . , 2014).

Uma medição precisa do tamanho funcional de um software requer que os requisitos funcionais estejam em um nível de granularidade onde é possível identificar os processos funcionais e os movimentos de dados (COSMIC. . . , 2014).

Se algum requisito precisa ser medido antes que seja suficientemente detalhado ele pode ser medido utilizando uma técnica de aproximação. Essas técnicas definem como os requisitos podem ser medidos em um alto nível de granularidade(COSMIC. . . , 2014).

4.4 Fase de Mapeamento

Essa seção apresenta definições, princípios, regras e o método para o processo de mapeamento. A Fig. 4 apresenta o processo para a fase de mapeamento do COSMIC:

- Identificar os Processos Funcionais;
- Identificando objetos de interesse e grupos de dados;
- Identificando movimentos de dados;

4.4.1 Identificar os Processos Funcionais

Essa etapa consiste em identificar o conjunto de processos funcionais do software que será medido apartir dos requisitos funcionais de usuário (COSMIC. . . , 2014).

A abordagem para se identificar os processos funcionais depende dos artefatos de software que estão à disposição do medidor, o que depende do ponto do ciclo de vida do software em que a medição é requisitada e do método de desenvolvimento em uso (COSMIC. . . , 2014).

Segundo o COSMIC. . . (2014), o processo para identificação dos processos funcionais seguem os seguintes passos:

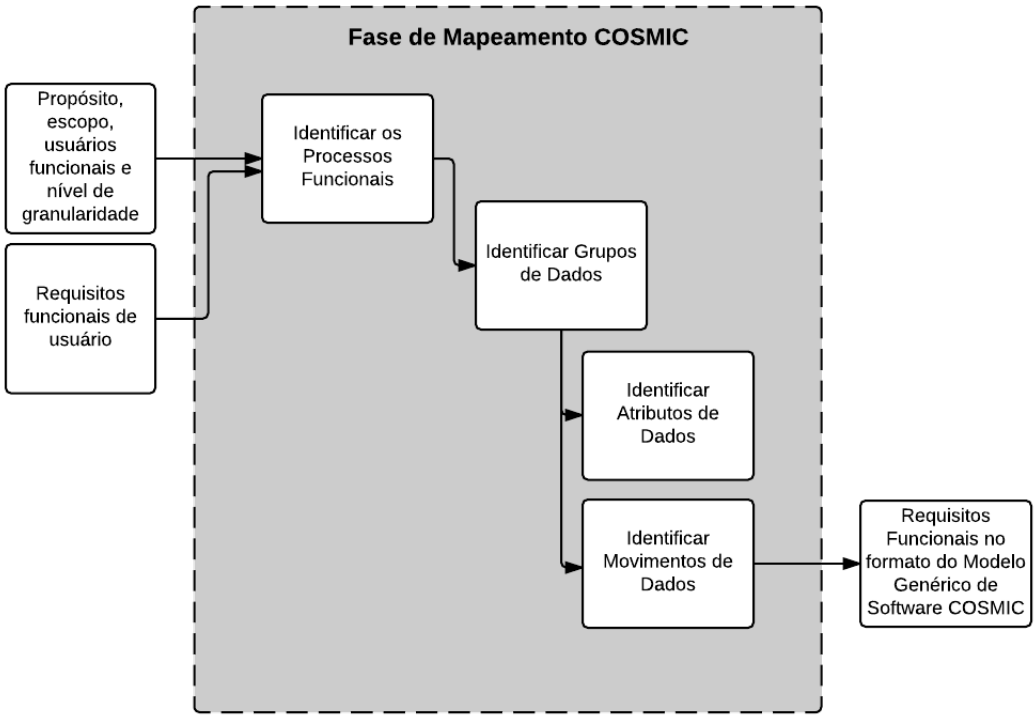

Figura 4 – Processo de mapeamento de medição. Fonte (COSMIC. . . , 2014)

- Identificar os distintos eventos no universo dos usuários funcionais em que o software a ser medido deve responder - os "Eventos Disparadores";
- Identificar quais usuários funcionais do software devem responder a cada evento disparador;
- Identificar a entrada disparadora que cada usuário funcional deve iniciar em resposta ao evento;
- Identificar os processos funcionais iniciados por cada entrada disparadora;

4.4.2 Identificando objetos de interesse e grupos de dados

Essa etapa consiste em identificar os grupos de dados referenciados pelo software a ser medido (COSMIC. . . , 2014).

As definições de objetos de interesse e grupos de dados são intencionalmente amplas para que sejam aplicadas ao maior número de contextos de software. O resultado disso acaba sendo a dificuldade em aplicar a definição e os princípios quando se está medindo um pedaço de software específico (COSMIC. . . , 2014).

Quando se tem a necessidade de analisar um grupo de atributos de dados que é movido para dentro ou para fora do processo funcional ou movido por um processo funcional para um armazenamento persistente, é de crítica importância decidir se todos os atributos transmitem um único objeto de interesse (COSMIC. . . , 2014).

4.4.3 Identificando movimentos de dados

Essa etapa consiste em identificar os movimentos de dados (*Entry, Exit, Read e Warite*) para cada processo funcional (COSMIC. . . , 2014).

As definições dos tipos de movimentos de dados segundo COSMIC. . . (2014):

- ***Entry (E)*** **-** Um movimento de dados que move um grupo de dados de um processo funcional através da fronteira para um processo funcional onde é requisitado;
- ***Exit (X)*** **-** Um movimento de dados que move um grupo de dados de um processo funcional através da fronteira para um usuário funcional que o requisita;
- ***Read (R)*** **-** Um movimento de dados que move um grupo de dados de um armazenamento persistente para um processo funcional que o requisita;
- ***Write (W)*** **-** Um movimento de dados que move um grupo de dados que está em um processo funcional para o armazenamento persistente;

4.5 Fase de Medição

Este é o último passo do processo de medição COSMIC. Primeiro, a unidade de medida é definida (um movimento de dado é medido como um CFP). Em

seguida são definidas regras para se atribuir um tamanho ao requisito funcional de usuário do software sendo medido (COSMIC..., 2014).

O método geral para se medir um pedaço de software quando seus requisitos funcionais de usuário já foram expressados na forma do modelo genérico de software COSMIC é apresentado na Fig. 5.

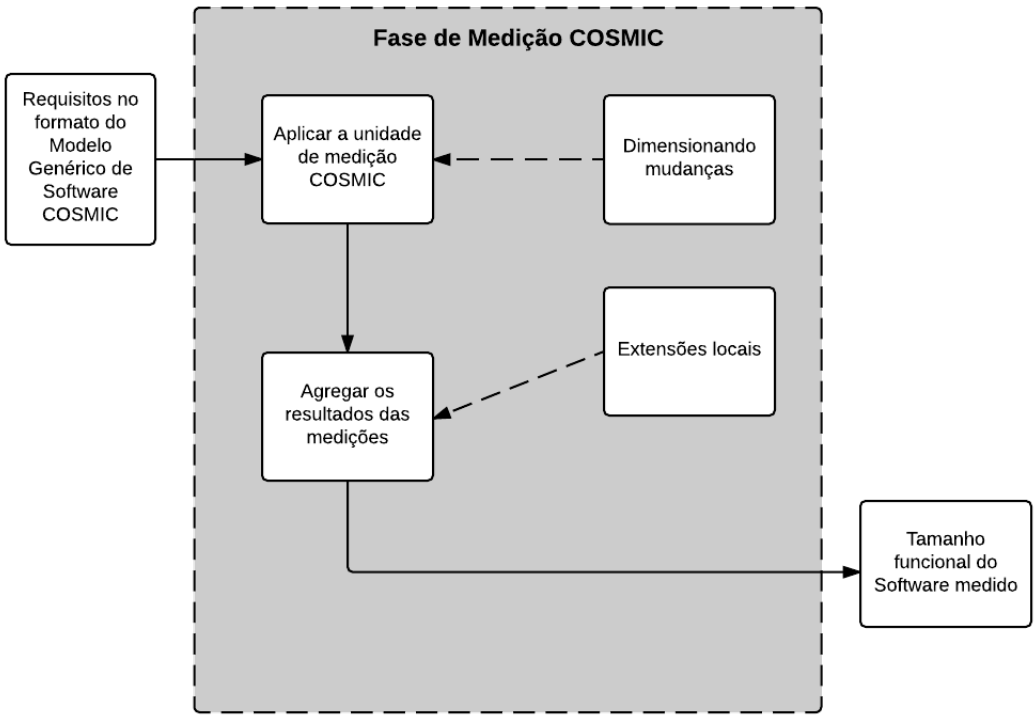

Figura 5 – Processo geral da fase de medição COSMIC. Fonte (COSMIC..., 2014)

4.5.1 Aplicando a função de medição

A definição de 1 CFP, segundo o COSMIC... (2014), é o tamanho de um movimento de dados. Assim, cada movimento de dado (Entrada, Saída, Leitura ou Escrita) que tenha necessidade de ser adicionado, modificado ou deletado do software sendo medido, também é contado como 1 CFP.

4.5.2 Agregando os resultados das medições

Essa etapa consiste em agregar os tamanhos de todos os movimentos de dados identificados em um único valor de tamanho funcional (COSMIC..., 2014).

O COSMIC... (2014) define as seguintes regras para a agregação dos resultados das medidas:

- Para qualquer processo funcional, os tamanhos de cada movimento de dados devem ser somados para se obter um único valor de tamanho funcional em CFP;
- Para qualquer processo funcional, o tamanho das mudanças dos requisitos funcionais devem ser agregados dos tamanhos dos movimentos de dados que foram adicionados, modificados ou deletados no processo funcional para se obter um tamanho da mudança em CFP;
- O tamanho de um pedaço de software dentro de um escopo definido deve ser obtido através da soma dos tamanhos dos processos funcionais que o constituem;
- O tamanho de qualquer mudança em um pedaço de software dentro de um escopo definido, deve ser obtido através da soma dos tamanhos das mudanças dos processos funcionais;
- Tamanhos de pedaços de software ou de mudanças podem ser somados apenas se forem medidos no mesmo nível de granularidade de processos funcionais;

Capítulo 5
Execução do Estudo

5 Esxecução do Estudo

5.1 Considerações Iniciais do Capítulo

Neste capítulo serão apresentados o processo e os passos seguidos no estudo de caso deste trabalho. Bem como a caracterização do projeto e equipe estudados.

5.2 Etapas de execução do estudo de caso

O processo seguido para execução do estudo de caso proposto é descrito na Fig. 6.

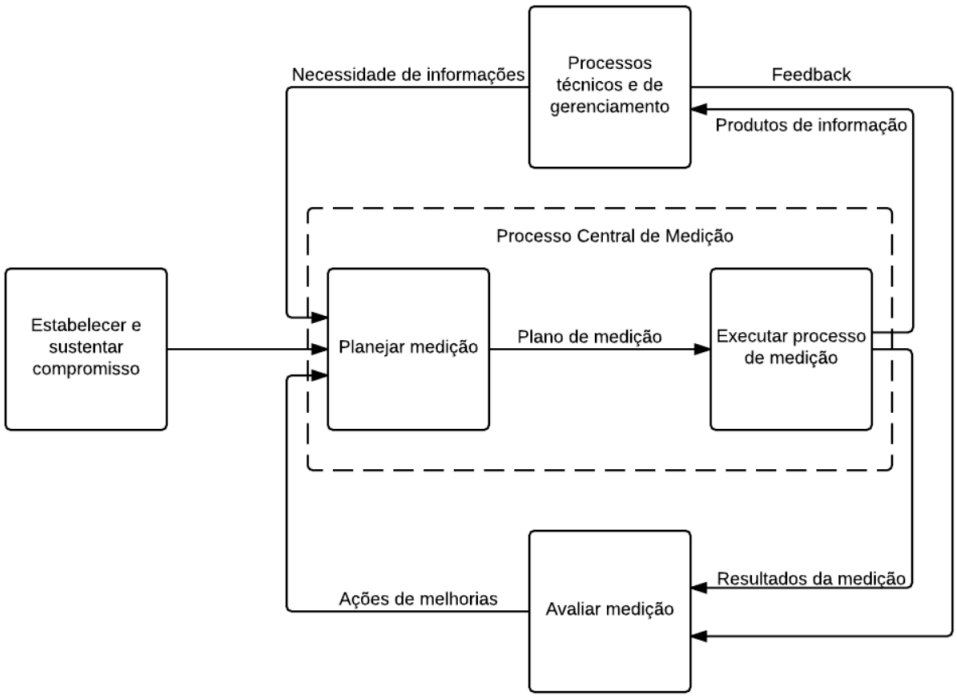

Figura 6 – Fluxograma das atividades de medição de software. Fonte (SYSTEMS. . . , 2008)

Para que fossem alcançados os objetivos do estudo, as etapas do processo apresentado incluem atividades que foram executadas conforme descrito abaixo:

- **Estabelecer e sustentar compromisso:**
  - **Selecionar organização -** Foi selecionada a Fábrica de Software do campus FGA da Universidade de Brasília. Onde fazem uso de metodologias ágeis e foi possível ter acesso à todos os artefatos necessários para a conclusão do estudo.
  - **Selecionar projeto -** Dentre os projetos em andamento, foi escolhido o sistema de Perícia Médica, pois estava em fase de desenvolvimento mais avançada em relação aos outros projetos disponíveis para estudo.
- **Planejar medição:**
  - **Selecionar funcionalidades a serem estudadas -** Foram selecionadas as novas funcionalidades previstas para as Sprints 6, 7 e 9, excluindo do estudo Histórias De Usuário técnicas e de manutenção.
  - **Definir as métricas a serem utilizadas -** Para o estudo, as métricas relevantes foram o Story Point (essa, de responsabilidade da equipe de desenvolvimento), o Ponto de Função Cosmic e o esforço real de cada atividade.
- **Executar processo de medição:**
  - **Acompanhar e registrar a pontuação -** As atividades escolhidas, foram pontuadas pela equipe de desenvolvimento, assim como as demais atividades da *Sprint*. A atividade de planejamento (*Planning Poker*) foi acompanhada e as pontuações definidas para as atividades selecionadas foram registradas.
  - **Realizar a contagem de Pontos de Função Cosmic -** A partir das *User Stories* e de detalhes fornecidos pela própria equipe do projeto, foram calculados os PFC das funcionalidade desenvolvidas.

  - **Coletar valores de esforço real -** Durante o desenvolvimento das atividades, cada desenvolvedor registrou o esforço em horas de desenvolvimento das mesmas. O esforço final de cada US era informado para registro no estudo antes do planejamento da próxima Sprint..

- **Avaliar medição**

  - **Realizar cálculo de regressão linear com valores estimados e reais -** Foi utilizada a técnica estatística de Regressão Linear para analisar a variação do esforço real de cada atividade em comparação aos valores de esforço estimados. A variação entre os esforços é indicada pelo MMRE calculado a partir do esforço real e do tamanho estimado, mais detalhado na seção seguinte.
  - **Analisar valores de acurácia *Story Points* vs. Pontos de Função Cosmic -** Os gráficos com valores obtidos da regressão linear são apresentados e analisados ao final desse estudo.
  - **Gerar modelo de estimativa para o próximo ciclo -** Com os cálculos de regressão linear foi possível, ao final de cada ciclo, obter uma fórmula que poderia ser utilizada para estimar o esforço das atividades da próxima *Sprint*.

- **Processos técnicos e de gerenciamento**

  - **Apresentar Resultados -** Os resultados alcançados pelo estudo serão apresentados aos responsáveis dos projetos para que se possa identificar possíveis melhorias no processo de estimativa dentro da organização.

5.3 Objeto de Estudo

A Fábrica de Software estudada está inserida em um contexto educacional, onde todos os membros da equipe são alunos do curso de Engenharia de Software da Universidade de Brasília. A equipe é composta por cinco estudantes todos entre o quinto e o nono semestre da graduação. Todos os membros já cursaram disciplinas que envolvem desenvolvimento de software em ambiente ágil

O projeto alvo do estudo é o módulo mobile do sistema de Perícia Médica, que é desenvolvido junto ao Cebraspe, que tem objetivo de facilitar o processo de perícia médica em candidatos de concursos. O módulo mobile envolve a execução da perícia médica, nele são identificados os candidatos e inseridos os dados referentes à perícia gerando uma ficha que deverá ser assinada pela junta médica responsável.

5.4 Análise da acurácia das medições

Commeyne, Abran e Djouab (2016) e Ungan, Çizmeli e Demirörs (2014) apresentam estudos similares ao proposto, em que comparam a acurácia das estimativas em COSCMIC e *Story Point* utilizando a técnica de regressão linear.

Porém, em seu estudo,Commeyne, Abran e Djouab (2016) assumem que 1 SP equivale a 1 dia de trabalho, ou 8 horas de esforço. Neste trabalho não será considerada tal suposição, pois é possível utilizar a técnica de regressão linear também ao *Story Point* para verificar o grau de correção entre o SP estimado e o esforço real. Ungan, Çizmeli e Demirörs (2014) não evidencia como é estimado o esforço para a técnica de estimativa ágil.

Tanto Commeyne, Abran e Djouab (2016) quanto Ungan, Çizmeli e Demirörs (2014) apresentam conclusões semelhantes, onde as estimativas a partir dos Pontos de Função COSMIC alcançam uma melhor acurácia.

Baseados nos estudos de Commeyne, Abran e Djouab (2016) e Ungan, Çizmeli e Demirörs (2014) e apartir de uma adaptação do processo de medição estabelecido na norma ISO 15939. Para se calcular a acurácia das estimativas obtidas através do estudo de caso, foi utilizada a técnica estimativa de regressão linear, descrita a seguir.

No processo de tomada de decisões, muitas vezes é necessário fazer previsões. Se torna muito mais fácil tomar decisões a respeito de uma variável quando é possível estabelecer uma relação entre essa e alguma outra variável conhecida. Para que seja estabelecida tal relação, é necessário que haja uma relação de causa-efeito entre elas, isto é, a variação de uma pode ser atribuída a variação da outra (REIS, 2008).

Para o contexto desse trabalho, a relação entre estimativa e esforço real, é representada pela regressão linear como feito por (COMMEYNE; ABRAN; DJOUAB, 2016), calculada a partir das pontuações em Pontos de Função Cosmic, *Story Points* e o esforço real de cada atividade.

A regressão nasce da tentativa de relacionar um conjunto de observações de certas variáveis, com as leituras de uma certa grandeza. O objetivo pode ser explicativo (demonstrar uma relação matemática que pode indicar, mas não provar, uma relação de causa-efeito) ou preditivo (obter uma relação que permita, prever o valor correspondente sem a necessidade de se medir) (MATOS, 1995).

Para se calcular a regressão linear, é necessário conhecer a reta de regressão $Y = a + bX$, onde:

- Y é a variável dependente, no caso, o esforço.
- X é a variável independente, no caso, as pontuações em Pontos de Função Cosmic ou Story Points.
- a e b são dados pelas equações 5.1 e 5.2:

$$a = \bar{Y} - b\bar{X} \tag{5.1}$$

$$b = \frac{n\sum(X_i Y_i) - \sum X_i \sum Y_i}{n\sum X_i^2 - (\sum X_i)^2} \tag{5.2}$$

- $\bar{Y}$ é a média dos valores de Y conhecidos, ou seja, média dos valores de esforço real das atividades selecionadas.
- $\bar{X}$ é a média dos valores de X conhecidos, ou seja, média dos Pontos de Função Cosmic estimados ou dos *Story Points* pontuados para as atividades selecionadas.

Após a criação da reta de regressão, calcula-se o erro relativo, que é o quanto cada atividade varia da média. O erro relativo é calculado para cada atividade de acordo com a equação 5.3.

$$ErroRelativo = \frac{(EsforçoReal - EsforçoEstimado)}{EsforçoReal} \tag{5.3}$$

E então é calculado o MMRE, do inglês *Mean Magnitude of Relative Error* (Magnitude Média do Erro Relativo), que mede a diferença entre o esforço estimado e o real relativo ao esforço real. Calcula-se o MMRE de acordo com a equação 5.4

$$MMRE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{(EsforçoReal - EsforçoEstimado)}{EsforçoReal}\right| \tag{5.4}$$

Capítulo 6
Resultados

6 Resultados

6.1 Considerações Iniciais do Capítulo

Neste capítulo são apresentados os resultados obtidos no estudo de caso. Foram estudadas três Sprints (6, 7 e 9), sendo os valores incrementados à cada Sprint estudada. Para cada Sprint, serão mostradas as etapas do estudo.

6.2 Escolha das Histórias de Usuário

Foram selecionadas histórias de usuário de novas funcionalidades do sistema Perícia Médica. As US escolhidas são referentes as Sprints 6, 7 e 9, executadas nos períodos de 27/03 a 10/04, 10/04 a 24/04 e 15/05 a 02/06 respectivamente.A Sprint 8 foi excluída dos estudo pois se tratava de uma manutenção no banco de dados do sistema.

6.3 Sprint 6

6.3.1 Dados da Sprint

Os dados coletados referentes aos Story Points medidos e ao esforço real da Sprint 6 bem como os PFC medidos são apresentados na Tab. 2:

Tabela 2: Dados coletados na Sprint 6. Fonte: Autor

User Story	**Story Points**	**PFC**	**Esforço Real (h)**
US 19	8	5	6,5
US 20	8	10	9
US 21	5	3	5,5
US 22	5	5	4

6.3.2 Cálculo da Regressão Linear

Com os dados coletados e calculados anteriormente, foram feitos dois cálculos de Regressão Linear, um para Pontos de Função COSMIC e um para os Story Points. E ao final de cada Sprint é feita a comparação do MMRE das duas regressões para indicar qual obteve maior acurácia por Sprint.

6.3.2.1 Pontos de Função COSMIC

Após os cálculos de regressão com os valores obtidos na Sprint 6, foi obtida a seguinte reta de regressão:

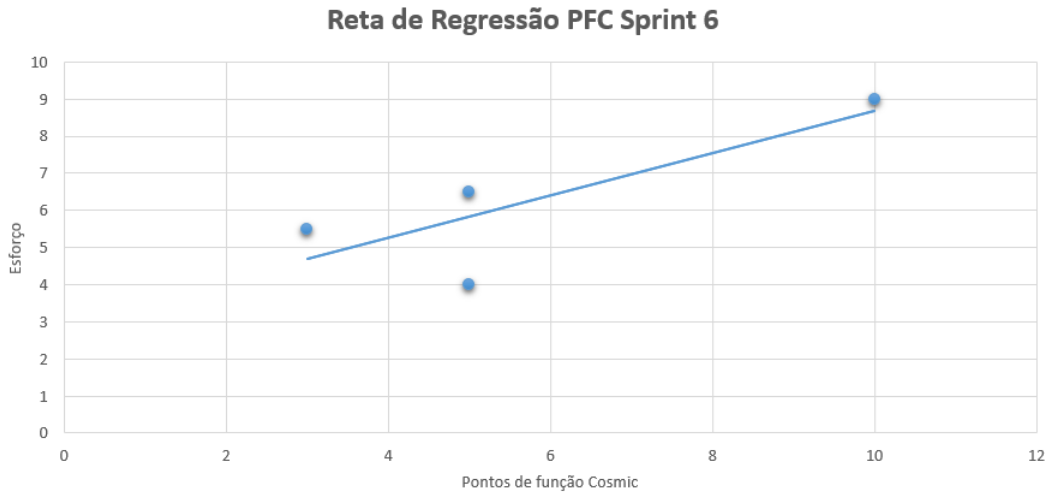

Figura 7 – Reta de regressão para PFC referente à Sprint 6. Fonte: Autor

A regressão linear dos Pontos de Função Cosmic da Sprint 6 obteve como resultados a Equação da Reta e o MMRE seguintes:

$$Y = 2.77 + 0.57X \tag{6.1}$$

$$MMRE = 18,62\% \tag{6.2}$$

6.3.2.2 Story Points

Com os dados coletados na Sprint 6 obteve-se a seguinte reta de regressão para os Story Points:

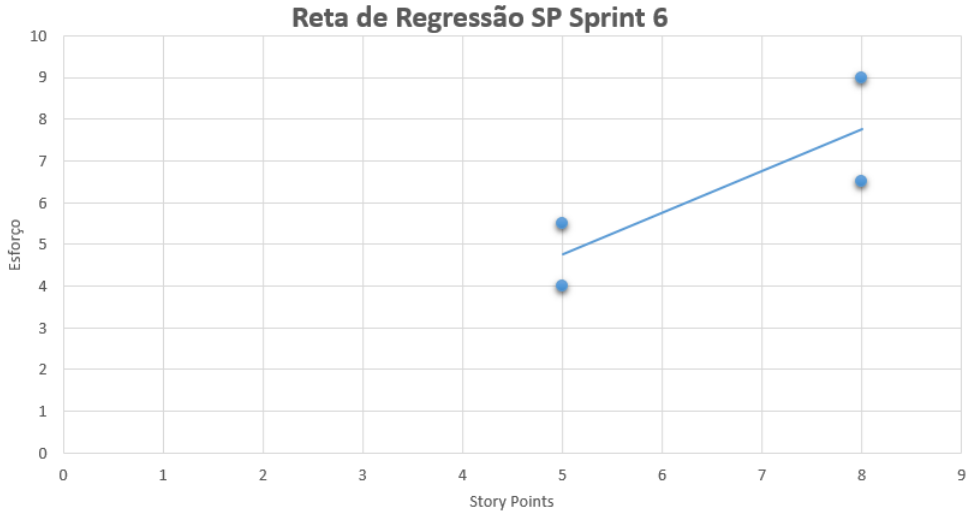

Figura 8 – Reta de regressão para SP referente à Sprint 6. Fonte: Autor

Foram calculados os seguintes valores para a equação da reta e MMRE:

$$Y = -0.25 + X \tag{6.3}$$

$$MMRE = 16,38\% \tag{6.4}$$

6.3.2.3 Resultado da Sprint

Analisando os valores da primeira Sprint estudada é possível observar que o MMRE de 16,38% das estimativas com Story Points indica melhores resultados quando comparados com os 18,62% de MMRE das estimativas com Pontos de Função COSMIC.

6.4 Sprint 7

6.4.1 Dados da Sprint

Os dados coletados na Sprint 7 foram combinados aos dados da Sprint 6 e os valores foram recalculados. Dessa forma sempre que uma Sprint é finalizada, ela é utilizada para refinar a equação da reta calculada ao final da Sprint anterior. Os dados das duas Sprints são apresentados na Tab. 3:

Tabela 3: Dados coletados na Sprint 7. Fonte: Autor

User Story	**Story Points**	**PFC**	**Esforço Real (h)**
US 19	8	5	6,5
US 20	8	10	9
US 21	5	3	5,5
US 22	5	5	4
US 23	2	6	4
US 24	3	5	8
US 25	3	11	14

6.4.2 Cálculo da Regressão Linear

Com os dados acumulados das duas Sprints, foram refeitos os cálculos de Regressão Linear para Pontos de Função COSMIC e para os Story Points. E ao final da Sprint é possível observar como se comportam os valores de regressão ao se adicionar os valores das novas Sprints.

6.4.2.1 Pontos de Função COSMIC

Após os cálculos de regressão com os valores obtidos nas Sprint 7 e 6, foi obtida a seguinte reta de regressão:

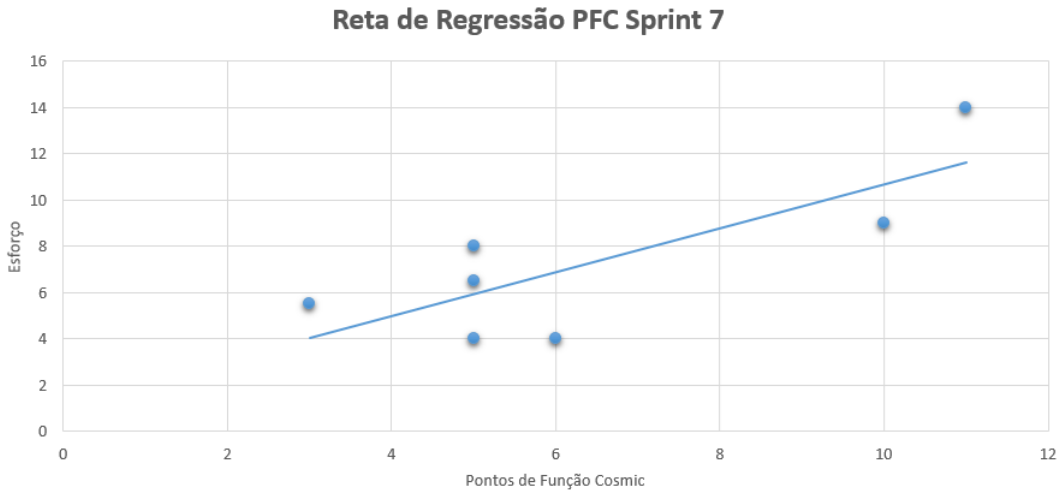

Figura 9 – Reta de regressão para PFC referente à Sprint 7. Fonte: Autor

A regressão linear dos pontos de função cosmic da Sprint 7 obteve como resultados a Equação da Reta e o MMRE seguintes:

$$Y = 1,176 + 0,95X \tag{6.5}$$

$$MMRE = 31,03\% \tag{6.6}$$

6.4.2.2 Story Points

Com os dados coletados na Sprint 7 obteve-se a seguinte reta de regressão para os Story Points:

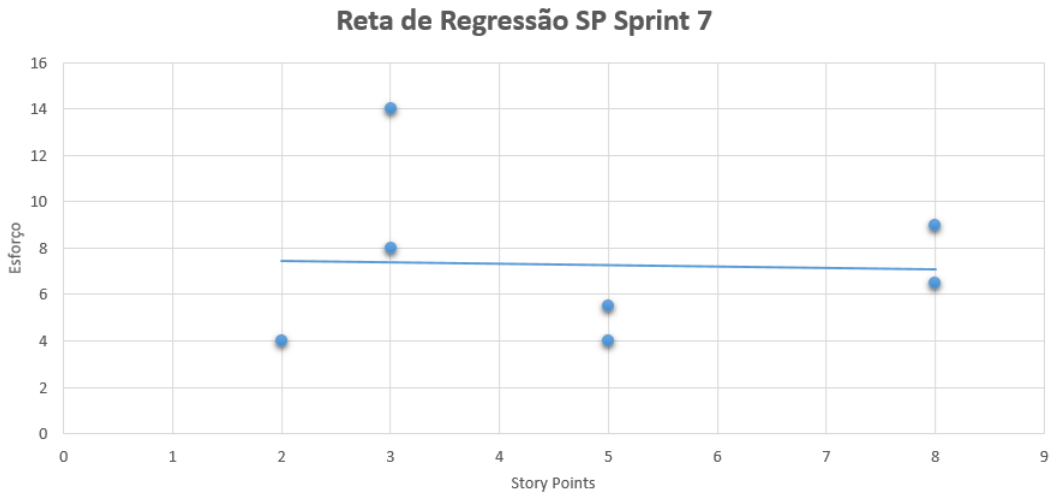

Figura 10 – Reta de regressão para SP referente à Sprint 7. Fonte: Autor

Foram calculados os seguintes valores para a equação da reta e MMRE:

$$Y = 7,594 - 0.063X \tag{6.7}$$

$$MMRE = 40,82\% \tag{6.8}$$

6.4.2.3 Resultado da Sprint

Observando os valores observa-se que o erro médio da regressão linear com Pontos de Função Cosmic variou dos 18,62% para 31,03%, apesar do erro ter aumentado esse valor passa a ser melhor quando comparado com a regressão com Story Points, que variou de 16,38% para 40,82%.

6.5 Sprint 9

6.5.1 Dados da Sprint

Sendo a Sprint 9 a última a ser analisada nesse estudo, os valores calculados nas outras Sprints foram recalculados juntamente com os valores obtidos nesta

Sprint. A Tab. 4 apresenta todos os valores calculados e coletados no estudo:

Tabela 4: Dados coletados na Sprint 9. Fonte: Autor

User Story	**Story Points**	**PFC**	**Esforço Real (h)**
US 19	8	5	6,5
US 20	8	10	9
US 21	5	3	5,5
US 22	5	5	4
US 23	2	6	4
US 24	3	5	8
US 25	3	11	14
US 42	33	29	26

6.5.2 Cálculo da Regressão Linear

Com os dados apresentados na Tab. 4, foram refeitos os cálculos das regressões.

6.5.2.1 Pontos de Função COSMIC

Após os cálculos de Regressão Linear apartir dos valores de Pontos de Função Cosmic das 3 Sprints estudadas foi obtida a seguinte reta de regressão:

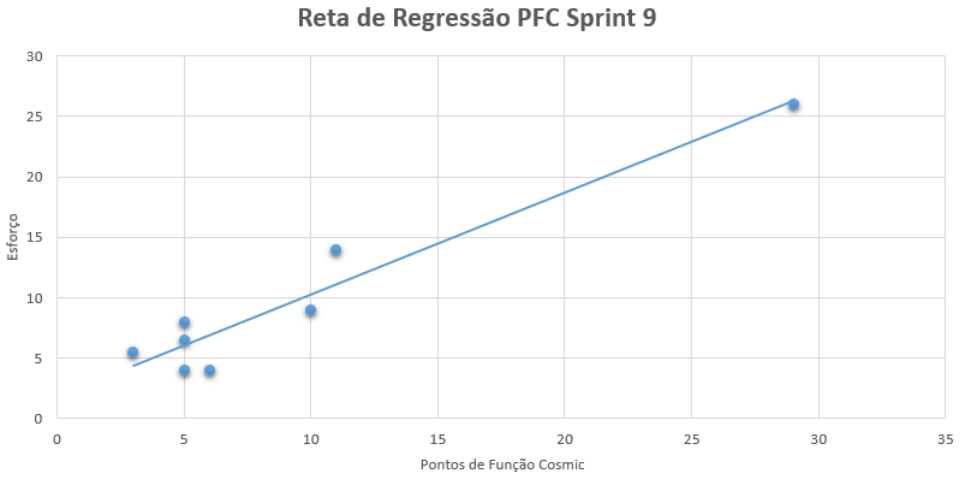

Figura 11 – Reta de regressão para PFC referente à Sprint 9. Fonte: Autor

A regressão linear dos Pontos de Função Cosmic da Sprint 9 obteve como resultados a Equação da Reta e o MMRE seguintes:

$$Y = 1.839 + 0.841X \tag{6.9}$$

$$MMRE = 26,39\% \tag{6.10}$$

6.5.2.2 Story Points

Com os dados finais obteve-se a seguinte reta de regressão para os Story Points:

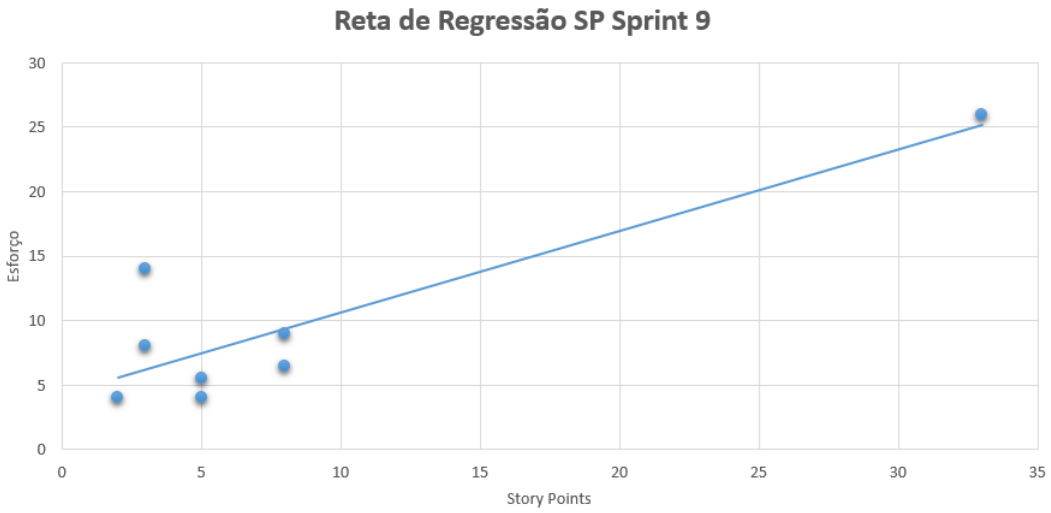

Figura 12 – Reta de regressão para SP referente à Sprint 9. Fonte: Autor

Foram calculados os seguintes valores para a equação da reta e MMRE:

$$Y = 4.348 + 0.63X \tag{6.11}$$

$$MMRE = 36,69\% \tag{6.12}$$

6.5.2.3 Resultado da Sprint

É possível observar que o erro médio das regressões tanto apartir de PFC, quanto apartir de SP teve uma menor variação entre as Sprints 7 e 9 quando comparadas com as variações entre as Sprints 6 e 7. Isso deve-se ao fato de que à cada ciclo os valores tendem à uma maior acurácia.

Referências

ABOUT Function Point Analysis - IFPUG. Acessado em 17/05/2016. Disponível em: <http://www.ifpug.org/about-ifpug/about-function-point-analysis/>. Citado na página 32.

ACCURACY (trueness and precision) of measurement methods and results-Part 6: Use in practice of accuracy values. [S.l.]: ISO, Geneva, Switzerland, 1994. Citado na página 36.

ALBRECHT, A. J. Measuring application development productivity. *Proc. of the IBM Applications Development Symposium*, 1979. Citado na página 32.

BASILI, V. R. Software modeling and measurement: the goal/question/metric paradigm. 1992. Citado na página 31.

COHN, M. *Agile estimating and planning*. [S.l.]: Pearson Education, 2005. Citado 5 vezes nas páginas 24, 33, 41, 42 e 44.

COMMEYNE, C.; ABRAN, A.; DJOUAB, R. Effort estimation with story points and cosmic function points-an industry case study. In: *2016 21st Software Measurement News*. [S.l.: s.n.], 2016. p. 25–36. Citado 3 vezes nas páginas 24, 62 e 63.

COSMIC Functional Size Measurement Method-version 4.0 Measurement Manual (The COSMIC Implementation Guide for ISO/IEC 19761: 2011). [S.l.]: September, 2014. Citado 10 vezes nas páginas 11, 47, 48, 49, 50, 51, 52, 53, 54 e 55.

COSMIC Method. Acessado em 17/05/2016. Disponível em: <http://cosmic-sizing.org/cosmic-fsm/>. Citado na página 33.

FEHLMANN, T.; SANTILLO, L. From story points to cosmic function points in agile software development–a six sigma perspective. In: *Metrikon-Software Metrik Kongress*. [S.l.: s.n.], 2010. Citado na página 24.

FENTON, N.; BIEMAN, J. *Software metrics: a rigorous and practical approach*. [S.l.]: CRC Press, 2014. Citado 3 vezes nas páginas 31, 32 e 37.

FIDEL, R. The case study method: a case study. *Library and Information Science Research*, v. 6, n. 3, p. 273–288, 1984. Citado na página 26.

GIL, A. C. Como elaborar projetos de pesquisa. *São Paulo*, v. 5, p. 61, 2002. Citado na página 25.

GOPAL, A. et al. Measurement programs in software development: determinants of success. *Software Engineering, IEEE Transactions on*, IEEE, v. 28, n. 9, p. 863–875, 2002. Citado na página 31.

GRENNING, J. Planning poker or how to avoid analysis paralysis while release planning. 2002. Citado 5 vezes nas páginas 11, 41, 42, 43 e 44.

HAMOUDA, A. E. D. Using agile story points as an estimation technique in CMMI organizations. In: *Agile Conference (AGILE), 2014*. [S.l.: s.n.], 2014. p. 16–23. Citado 2 vezes nas páginas 23 e 43.

IDRI, A.; AMAZAL, F. azzahra; ABRAN, A. Analogy-based software development effort estimation: A systematic mapping and review. *Information and Software Technology*, v. 58, p. 206 – 230, 2015. ISSN 0950-5849. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0950584914001815>. Citado na página 33.

JORGENSEN, M. What we do and don't know about software development effort estimation. *IEEE software*, v. 31, n. 2, 2014. Citado na página 33.

JøRGENSEN, M. Unit effects in software project effort estimation: Work-hours gives lower effort estimates than workdays. *Journal of Systems and Software*, v. 117, p. 274 – 281, 2016. ISSN 0164-1212. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0164121216300085>. Citado na página 33.

MAHNIČ, V.; HOVELJA, T. On using planning poker for estimating user stories. *Journal of Systems and Software*, Elsevier, v. 85, n. 9, p. 2086–2095, 2012. Citado na página 42.

MATOS, M. A. Manual operacional para a regressão linear. *Faculdade de Engenharia da Universidade do Porto*, 1995. Citado na página 63.

MORESI, E. et al. Metodologia da pesquisa. *Universidade Católica de Brasília*, 2003. Citado na página 25.

OMANOVIC, S.; BUZA, E. Importance of stable velocity in agile maintenance. In: IEEE. *Information, Communication and Automation Technologies (ICAT), 2013 XXIV International Symposium on*. [S.l.], 2013. p. 1–8. Citado na página 44.

PEIXOTO, C. E. L.; AUDY, J. L. N.; PRIKLADNICKI, R. The importance of the use of an estimation process. In: *Proceedings of the 2010 ICSE Workshop on Software Development Governance*. ACM, 2010. (SDG '10), p. 13–17. ISBN 978-1-60558-979-4. Disponível em: <http://doi.acm.org/10.1145/1808981.1808983>. Citado na página 23.

POPLI, R.; CHAUAHN, N. Managing uncertainty of story-points in agile software. In: *2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*. [S.l.: s.n.], 2015. p. 1357–1361. Citado na página 23.

POPLI, R.; CHAUHAN, N. Cost and effort estimation in agile software development. In: *2014 International Conference on Optimization, Reliabilty, and Information Technology (ICROIT)*. [S.l.: s.n.], 2014. p. 57–61. Citado na página 23.

REIS, E. *Estatística Descritiva*. [S.l.]: Edições Sílabo, 2008. Citado na página 62.

RUNESON, P. et al. *Case study research in software engineering: Guidelines and examples*. [S.l.]: John Wiley & Sons, 2012. Citado na página 26.

SYSTEMS and software engineering — Measurement process. [S.l.]: ISO, Geneva, Switzerland, 2008. Citado 2 vezes nas páginas 11 e 59.

SZALVAY, V. *Glossary of Scrum Terms*. 2007. Acessado em 22/05/2016. Disponível em: <https://www.scrumalliance.org/community/articles/2007/march/glossary-of-scrum-terms>. Citado na página 44.

TARIQ, S. et al. On learning software effort estimation. In: IEEE. *Computational and Business Intelligence (ISCBI), 2015 3rd International Symposium on*. [S.l.], 2015. p. 79–84. Citado na página 33.

TRAN-CAO, D.; LEVESQUE, G.; ABRAN, A. Measuring software functional size: towards an effective measurement of complexity. In: IEEE. *Software Maintenance, 2002. Proceedings. International Conference on*. [S.l.], 2002. p. 370–376. Citado na página 32.

TRUDEL, S.; BUGLIONE, L. Guideline for sizing agile projects with cosmic. In: *Proceedings of International Workshop on Software Measurement*. [S.l.: s.n.], 2010. Citado na página 47.

UNGAN, E.; ÇIZMELI, N.; DEMIRöRS, O. Comparison of functional size based estimation and story points, based on effort estimation effectiveness in SCRUM projects. In: *2014 40th EUROMICRO Conference on Software Engineering and*

Advanced Applications (SEAA). [S.l.: s.n.], 2014. p. 77–80. Citado 3 vezes nas páginas 24, 32 e 62.

USMAN, M. et al. Effort estimation in agile software development: A systematic literature review. In: ACM. *Proceedings of the 10th International Conference on Predictive Models in Software Engineering*. [S.l.], 2014. p. 82–91. Citado 3 vezes nas páginas 13, 37 e 38.

WEN, J. et al. Systematic literature review of machine learning based software development effort estimation models. *Information and Software Technology*, Elsevier, v. 54, n. 1, p. 41–59, 2012. Citado na página 33.