Análise de estimativa de software utilizando Scrum com a técnica Planning Poker

RogerRitter.com.br

rogersmartlife@gmail.com

Resumo. *Há muitos anos que a tarefa de estimar software com assertividade tem se tornado um desejo de muitas empresas. Neste trabalho, realizou-se uma análise do atendimento de um cenário de desenvolvimento que utiliza Scrum e a técnica Planning Poker para estimar software. Através desta análise obteve-se a porcentagem de aderência e sugestões para uma melhoria na estimativa de software.*

1. Introdução

A atividade de determinar o esforço para o desenvolvimento de um projeto é um elemento fundamental para obter um bom resultado. O esforço abrange diversos fatores como o cronograma, o custo e os recursos humanos do projeto, desta forma, estimar e planejar estão correlacionados e são atividades críticas em todas as abordagens de desenvolvimento de software.

O Scrum, como uma destas abordagens, é um framework que Schwaber (2013) afirma empregar outros processos e técnicas de acordo com a necessidade de uma equipe. Dentre os processos e técnicas que se pode empregar no framework, Boeg (2012) destaca o Kanban, que tem se tornado um complemento dos métodos ágeis como o Scrum e XP. Destacando também, uma técnica de estimativa de tamanho voltada as metodologias ágeis de desenvolvimento de software, denominada de Planning Poker. Que conforme Grenning (2002) e Cohn (2013), é uma técnica utilizada para determinar o tamanho de um software permitindo a interação entre membros de uma equipe.

O artigo desenvolvido por Krauze (2014), doravante denominado “Objeto”, descreve um cenário de desenvolvimento de software que utiliza o framework Scrum com Kanban e Planning Poker. Krauze (2014) descreve um problema na estimativa de software, no qual, por meio da aplicação de métricas obteve maior assertividade nas estimativas.

A motivação da análise do objeto ocorreu pela ciência de quanto este adere aos frameworks e técnica que se propõem, de forma a sugerir melhorias sem a utilização de métricas, com o objetivo de aumentar o nível de assertividade da estimativa de software.

Com o objetivo determinado, realizou-se uma análise de quanto o objeto de Krauze (2014) é aderente as prescrições de estimativas do framework Scrum propostas por Schwaber (2013) e Kniberg (2009) e da técnica Planning Poker, propostas por Cohn (2013) e Grenning (2002).

A estrutura do presente artigo é composta por sete sessões organizadas da seguinte forma: A primeira sessão apresenta a motivação e o objetivo deste artigo, na

segunda sessão, apresenta as metodologias ágeis de desenvolvimento de software destacando o framework Scrum e o seu complemento Kanban.

Na terceira sessão, a estimativa de software e a estimativa de tamanho de software são apresentadas, havendo destaque a técnica Planning Poker, utilizada para medir o tamanho de um software. Na quarta sessão, apresenta-se a metodologia utilizada na análise e o objeto de Krauze (2014).

Realizou-se na quinta sessão, a análise de Krauze (2014) buscando a sua aderência em relação ao framework Scrum e a técnica Planning Poker. Para cada item do objeto não atendido foi proposta uma melhoria. Na sexta sessão, apresentam-se as conclusões do artigo, as principais contribuições e trabalhos futuros. Na sétima e última sessão, as referências bibliográficas são apresentadas.

2. Metodologias Ágeis de Desenvolvimento de Software

A abordagem ágil tem como principal conceito a colaboração e integração entre os membros da equipe. As metodologias ágeis se caracterizam pelo gerenciamento de projeto em que o líder da equipe seja organizado, apoiador, inspecionador e que garanta o bem estar da equipe de desenvolvimento, ao mesmo tempo em que os resultados do projeto vão atendendo as solicitações do cliente. Segundo Cockburn (2001), não existem novidades relacionadas a aplicação das práticas usadas pelos métodos ágeis. O diferencial está em considerar o fator humano como o principal responsável pelo sucesso do projeto, focando na eficácia e na capacidade de gerenciar. Esta ideologia produz uma nova combinação de valores e princípios que definem uma visão do mundo ágil.

Existem algumas metodologias ágeis utilizadas no mercado que seguem estas bases, dentre as mais conhecidas, segundo Cockburn (2001), podem ser citadas: XP (*Extreme Programming*), Scrum, FDD (*Feature Driven Development*), DSDM (*Dynamic Systems Development Method*), Lean Development e Crystal. Assim como existem versões híbridas de algumas metodologias ágeis, como destaca Boeg (2012), entre as mais utilizadas estão: Scrum e XP e Scrum e Kanban.

Diante do exposto, detalha-se a seguir os frameworks Scrum e Kanban.

2.1. Scrum e Kanban

O Scrum, segundo Schwaber (2013), é um framework que permite empregar outros processos ou técnicas de acordo com a necessidade de uma equipe e Boeg (2012) descreve Kanban como um framework que complementa os métodos ágeis como o Scrum e XP.

Scrum e Kanban são frameworks de uma metodologia ágil de desenvolvimento de software que auxiliam equipes, apresentando o que se deve fazer em determinadas circunstâncias, como destacam Schwaber (2013), Boeg (2012) e Kniberg (2009). Scrum e Kanban não são perfeitos e tampouco completos, pois não apresentam tudo o que se precisa fazer, e sim, apenas oferecem restrições e orientações para se desenvolver um software.

A importância de um framework é o limite de opções, portanto um framework auxiliará a obter um bom resultado, mas isto não garantirá o resultado de qualquer

projeto, destaca Kniberg (2009). Portanto, o Scrum concentra-se em “como” as equipes devem realizar suas tarefas de forma ordenada para produzir um sistema flexível num ambiente com constantes mudanças, descreve Zanatta (2004).

Scrum e Kanban são descritos através de prescrições, definindo que estas podem ser analisadas comparando a quantidade de “regras” que o framework possui. Quanto mais prescritivo é um framework, significa “mais regras a seguir”, enquanto mais adaptativo significa “menos regras a seguir”. Por meio de um gráfico, Kniberg (2009) define o Scrum utilizando nove prescrições, enquanto o Kanban possui apenas três.

O Scrum é composto basicamente por uma equipe que se baseia em um “dono” de produto denominado *Product Owner*, um líder da equipe, denominado *Scrum Master* e a equipe composta por pessoas com diferentes habilidades, mas que juntas são capazes de realizar uma tarefa proposta a aquela equipe. O Scrum é composto por eventos, como a Sprint, onde em um período de no máximo 30 (trinta) dias deve-se construir um incremento de valor para o cliente, a reunião de planejamento onde ocorre o planejamento da Sprint, uma reunião diária para acompanhamento, uma revisão ao final da Sprint onde se apresenta o incremento utilizável ao cliente e um evento de retrospectiva onde são discutidos os problemas que houveram, de modo a não se repetir na próxima iteração. Com artefatos o Scrum possui o *Product Backlog* e o *Backlog* da Sprint de forma a auxiliar no gerenciamento de produtos, descreve Schwaber (2013).

Para auxiliar no gerenciamento do desenvolvimento de software, o Scrum também pode utilizar outros frameworks de modo a complementar-se, como o Kanban, tornando-se uma versão híbrida. Porém, não se deve fazer comparações, pois cada framework possui características diferentes. O Kanban, por exemplo, apoia-se no uso dos princípios ágeis para otimizar processos existentes, de forma evolucionária, destaca Boeg (2012).

O Kanban é um quadro fixado na parede e visível por todos os interessados, dividido por colunas onde são inseridas as tarefas em forma de cartões, segundo Boeg (2012) e Kniberg (2009), a metodologia dá ênfase aos seguintes princípios:

- Visualizar o fluxo de trabalho dividindo em partes, descrevendo cada tarefa em um cartão e inserindo-o na parede de forma a utilizar as colunas denominadas para ilustrar onde cada tarefa está no fluxo de trabalho;
- Limitar o trabalho em progresso (WIP – *Work In Progress*), limitando explicitamente quantos cartões poderá estar em progresso em cada estado do fluxo de trabalho;
- Acompanhar o tempo de execução da tarefa (tempo médio para completar um cartão), otimizando o processo para tornar o tempo de execução o menor e mais previsível possível;

Na Figura 1, é apresentado um modelo de quadro Kanban, composto por cinco colunas. Sendo cada uma delas apresentadas da seguinte forma:

1) “A Fazer”, representando a coluna que contém os cartões a serem desenvolvidos.
2) “Desenv”, a coluna responsável pelos cartões que estão em fase de desenvolvimento.
3) A coluna de “Teste”, que deverá conter os cartões que estão em processo de teste.

4) “Release”, representando a coluna que contém os cartões que estão em processo de liberação para o ambiente onde o usuário poderá utilizar a aplicação.
5) E, “Feito!”, como a última coluna que deverá conter os cartões que foram concluídos.

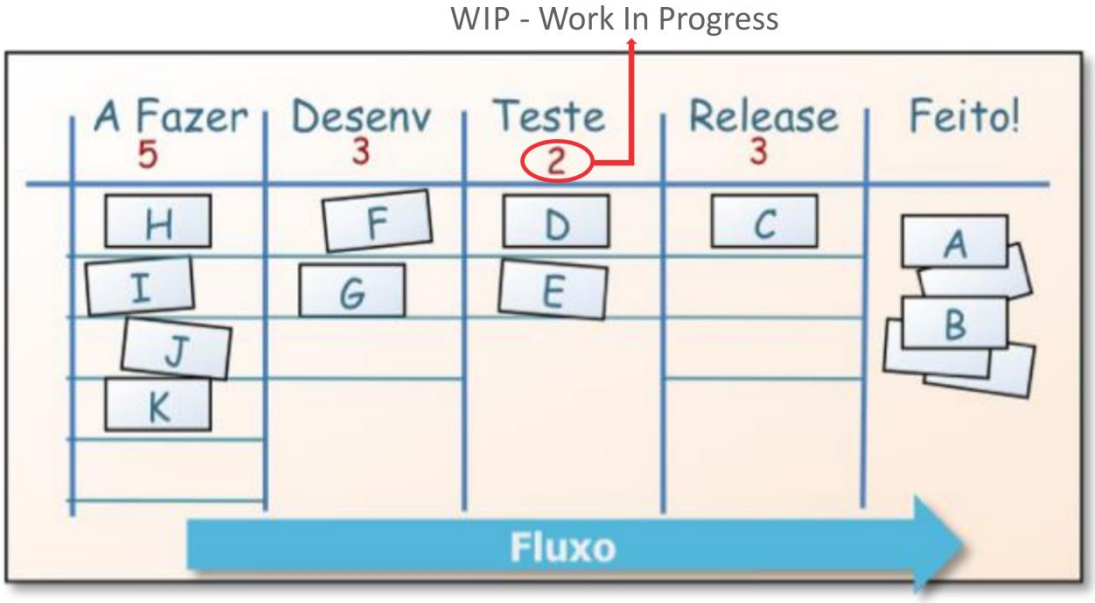

Figura 1. Quadro Kanban. Fonte: Kniberg (2009)

Em destaque circular na Figura 1, o número 2 representa o limite de número de cartões que se é permitido alocar na coluna “Teste”, este limite é denominado de WIP ou Limitador de Cartões, segundo Boeg (2012) e Kniberg (2009). Quando houver um próximo cartão, os membros da equipe deverão ajudar a desenvolver os cartões que estão na coluna, move-los para a próxima etapa do fluxo de trabalho e liberar espaço para a entrada de novos cartões. Este limite, segundo Kniberg (2009), é o principal diferenciador entre o Quadro Scrum e o Quadro Kanban.

Com o foco no fluxo e no contexto, o Kanban oferece como um de seus princípios acompanhar o tempo de execução de cada tarefa, auxiliando o framework a realizar uma estimativa de software mais assertiva, destaca Boeg (2012).

O Scrum, segundo Kniberg (2009), prescreve estimativas de software e velocidade das equipes, no qual devem realizar estimativas utilizando o tamanho de um software. Quando Scrum utiliza um complemento híbrido que também estima, como o Kanban, por exemplo, então a estimativa pode ser considerada mais assertiva ou tende-se a escolher uma única forma de se estimar o tamanho de um software.

3. Estimativa de Software

A estimativa de software é uma atividade contínua que pode ser aplicada em todas as etapas do ciclo de vida do projeto. Com a estimativa, é possível coletar métricas que permitem prever a quantidade de pessoas, tempo e custos necessários para o desenvolvimento de um projeto de software, destaca Hazan (2008).

Roger S. Pressman (2011) ressalta que se você não sabe para onde você quer ir, qualquer caminho você pode seguir.

Se você não sabe onde você está um mapa não irá lhe ajudar.

Estimativas de software em métodos ágeis, segundo Bassi (2010), são compostas por uma dupla de valores < v, p >, onde "v" é um palpite sobre determinado evento ou atividade e "p" é a probabilidade de "v" acontecer. Equipes ágeis se baseiam na comunicação e na transparência, ao invés de tratar suas estimativas com fatos admitem que existe uma incerteza ao valor estimado e evidenciam isso para que o cliente e outros envolvidos no processo também tomem ciência do grau de dificuldade de cada tarefa.

Nas estimativas de longo prazo a incerteza e o conhecimento sobre o assunto aumentam, quando isto ocorre, Bassi (2010) descreve que as estimativas devem ser refeitas considerando um nível maior de detalhamento e assim associando em probabilidades maiores de acerto.

As estimativas de custo e esforço em projetos de software são normalmente baseadas na predição do tamanho do sistema que será desenvolvido. A atividade de estimar o tamanho do software, segundo Abrahao (2008), é uma das mais críticas dentro do ciclo de vida de um projeto, pois será a partir dela que o projeto será contratualmente regido e outras estimativas serão derivadas.

3.1. Estimativa de Tamanho

A estimativa de tamanho de software, segundo o Hazan (2008), é um processo pelo qual uma pessoa ou um grupo de pessoas estima o tamanho de um software. As estimativas de tamanho constituem a base para a derivação das estimativas de esforço, prazo e custos, descreve o CMMI (2012). Normalmente, o tamanho têm impacto na solução técnica e na gestão do projeto já que, estimativas imprecisas podem levar ao fracasso de qualquer projeto.

Em frameworks como o Scrum e o Kanban, as equipes devem estimar o tamanho que é igual a quantidade de trabalho aos quais a equipe se comprometeu a fazer. Somando o tamanho de cada cartão concluído ao final de cada Sprint, o resultado será a estimativa de velocidade, frisa Kniberg (2009).

Estimar o tamanho sobre um objeto permitindo a interação entre os membros de uma equipe é uma boa prática que promove maior assertividade no tamanho de um software, pois cada membro poderá possuir uma visão diferenciada deste objeto, destaca Cohn (2013). Técnicas como Triangulação, Analogia, Planning Poker e Desagregação possuem estas características, mas segundo Grenning (2002) e Cohn (2013) a mais conhecida é a técnica Planning Poker.

Diante do exposto, detalharemos a seguir a técnica Planning Poker.

3.2. Técnica de Estimativa de Software Planning Poker

A técnica Planning Poker foi definida pela primeira vez por James Grenning em 2002, e popularizada por Mike Cohn no livro *Agile Estimating and Planning* no qual registrou o termo Planning Poker (PP). Cohn (2013) e Grenning (2002) descrevem o PP como uma técnica de estimativa de tamanho voltada para as metodologias ágeis de desenvolvimento de software.

O Planning Poker consiste-se em obter a estimativa por meio de um jogo de cartas, que, segundo Alves (2012), deve permitir que todos os membros da equipe de desenvolvimento (programadores, testadores, design e analistas) participem colocando a sua visão de complexidade, levando em consideração o fator tempo e esforço para pontuar um cartão e após juntos chegar a um denominador comum na equipe através de consenso.

No Planning Poker cada integrante tem a sua disposição um baralho de 13 cartas, numeradas em uma sequência similar a encontrada nos números de *Fibonacci*[1]. As cartas contém os tamanhos de 0, ½, 1, 2, 3, 5, 8, 13, 20, 40 e 100 que serão atribuídos a um cartão, havendo ainda uma carta com o símbolo de interrogação no qual representa que o estimador não está apto a estimar e outra carta com a imagem de uma xícara de café no qual representa a sugestão de uma pausa.

Durante o Planning Poker devem ser realizadas rodadas para obter a estimativa de um cartão que possui uma estória ou tarefa a desenvolver. As diferenças que surgirem durante as rodadas deverão ser mediadas por um coordenador, que no Scrum este papel é de responsabilidade do *Scrum Master*. O *Product Owner*, é o responsável por explicar o que deverá ser desenvolvido, sendo um importante papel para retirar possíveis dúvidas a respeito do cartão, evitando assim, o retrabalho.

[1] - Uma sequência de números inteiros, começando normalmente por 0 e 1, na qual, cada termo subsequente corresponde a soma dos dois números anteriores.

4. Metodologia

Devido a significativa importância da estimativa de software no gerenciamento de projetos, este trabalho analisa Krauze (2014), através de prescrições de estimativa de software do framework Scrum e da técnica Planning Poker.

Para auxiliar na análise e verificação no atendimento do objeto de Krauze (2014), fez-se uma revisão bibliográfica de: Schwaber (2013), Kniberg (2009), Cohn (2013) e Grenning (2002). Estas referências auxiliaram no processo de mapeamento das prescrições que interferem diretamente ou indiretamente na estimativa de um software.

No objeto de Krauze (2014), foram mapeados itens que atendiam, não atendiam ou atendiam parcialmente determinadas prescrições de estimativas. Caso o item apresentado por Krauze (2014) não possuísse nenhuma ou pouca evidência é categorizado como "Não Atendido", se ocorresse evidências de uma prática sistemática semelhante a prescrição correspondente é categorizado como "Parcialmente Atendido". E, por fim, caso existisse a prática de satisfação da prescrição e evidências significativas, este é posicionado na categoria de "Atendido".

Itens de Krauze (2014) que apresentavam informações difusas ou poucas informações, foram descritos através de um questionário e enviado ao autor. No qual, através das respostas os itens receberam informações adicionais que permitiram uma categorização mais ampla.

Itens categorizados como "Não Atendido", são descritos através de uma análise individual e uma sugestão de melhoria para que este seja aprimorado podendo promover uma maior assertividade na estimativa de software.

4.1. Objeto

O objeto de análise é um artigo desenvolvido por Krauze (2014). O artigo descreve um problema na estimativa de software, no qual, através da aplicação de métricas obteve maior assertividade nas estimativas.

Krauze (2014), desenvolveu um conjunto de métricas com o objetivo de obter uma média histórica de impedimentos e tarefas não previstas na definição das Sprints de forma a somar esta média no tempo total de uma estimativa de software. O resultado foi uma melhora significativa da assertividade das estimativas no decorrer das iterações (Sprints).

O objeto baseia-se na utilização do framework Scrum aplicado juntamente com a técnica de estimativa de tamanho Planning Poker. Modelando o Scrum com as seguintes técnicas adicionais que são relevantes para a estimativa de um software:

- O papel de *Product Owner* é concedido a um colaborador que não participa ativamente em todos os projetos da equipe, e sim, permanece na equipe por um determinado tempo.
- Uma reunião semanal com o cliente, que tem como objetivo alinhar as metas e integráveis na presença do *Scrum Master* e do *Product Owner*.
- Uma reunião ao meio da Sprint de forma a apresentar ao cliente um protótipo do que será entregue como incremento, com o objetivo de validar se o entendido é o solicitado.

Krauze (2014) também apresentou os seguintes itens não previstos de alteração que inferem diretamente ou indiretamente nas estimativas de um software:

- O quadro Kanban não possui WIP (*Work In Progress*);
- Na técnica Planning Poker, os pontos (tamanho) são convertidos em períodos de turnos (tempo);

5. Análise

A análise da Estimativa de Software foi concedida como resultado da comparação entre o objeto de Krauze (2014) e a descrição bibliográfica de Schwaber (2013), Kniberg (2009), Cohn (2013) e Grenning (2002) sendo dividida em duas subseções, a comparação entre os itens correspondentes ao framework Scrum e os itens correspondentes à técnica Planning Poker.

Na comparação com o Scrum os itens foram intitulados e identificados com a sigla SCR, na comparação da técnica Planning Poker foram identificados pela sigla PP.

5.1. Comparando Scrum ao Objeto

A comparação descreve a categoria de atendimento do objeto de Krauze (2014) em relação às prescrições de estimativas do framework Scrum, a análise individual dos itens não atendidos e, por fim, um gráfico apresentando o quanto o objeto é aderente ao framework Scrum.

Para realizar a análise dos resultados, baseada na verificação do grau de atendimento, propõe-se uma escala ordenada de três categorias: *Não Atendido*, *Parcialmente Atendido* e *Atendido*. De forma a auxiliar na definição do quão aderente é o objeto de Krauze (2014) às prescrições propostas por Schwaber (2013) e Kniberg (2009).

As três categorias são:

a) Não Atendido: Há pouca evidência de que o item foi satisfeito, ou a prescrição propõe diferentes propostas ao framework. Sendo os itens desta categoria são destacados com análise individual através de tópicos intitulados com a sigla SCR, referenciado itens Scrum que não foram atendidos.
b) Parcialmente Atendido: Existem evidências de uma prática sistemática no item na satisfação da prescrição.
c) Atendido: Existe uma prática na satisfação da prescrição, existindo evidências significativas.

A Tabela 1 apresenta as prescrições definidas por Schwaber (2013) e Kniberg (2009), com a respectiva categoria de atendimento de Krauze (2014).

Tabela 1. Atendimento do Objeto as Prescrições de Estimativas Scrum

Fonte: *Primária*

Prescrição	Categoria
Quando o horizonte da Sprint é muito longo, a definição do que será construído pode mudar, a complexidade pode aumentar e o risco pode crescer.	Não Atendido

A equipe de desenvolvimento consiste de profissionais que realizam o trabalho de entregar uma versão usável que potencialmente incrementa o produto "Pronto" ao final de cada Sprint. Somente integrantes do time de desenvolvimento criam incrementos.	Atendido
Em Scrum, equipes devem estimar o tamanho (quantidade de trabalho) de cada item aos quais estes se comprometeram.	Atendido Parcialmente
A função do *Scrum Master* é trabalhar com a equipe Scrum e organizar o aumento da transparência dos artefatos.	Atendido
Um dos propósitos da Reunião de Retrospectiva é de inspecionar como a última Sprint foi em relação às pessoas, aos relacionamentos, aos processos e às ferramentas.	Atendido
Todos os eventos prescritos são usados para criar uma rotina e minimizar a necessidade de reuniões não definidas no Scrum.	Não Atendido
Com base no que foi construído na Sprint e a qualquer mudança no *Product Backlog*, a equipe colabora nos próximos incrementos que podem aperfeiçoar o valor.	Atendido
O evento de revisão de uma Sprint trata-se de uma reunião informal e não uma reunião de status. A apresentação do incremento destina-se a motivar e obter comentários promovendo a colaboração.	Atendido Parcialmente
O resultado do evento de revisão da Sprint deve ser um *Product Backlog* revisado que define o provável *Product Backlog* para a próxima Sprint.	Não Atendido
No evento de revisão o *Product Owner* esclarece quais itens do *Product Backlog* foram "Prontos" e quais não foram "Prontos".	Atendido
O time de desenvolvimento discute no evento de revisão o que foi bem durante a Sprint, quais os problemas ocorreram e como estes problemas foram resolvidos.	Atendido
O *Product Owner* deve discutir o *Product Backlog* tal como está e projetam-se as prováveis datas de conclusão baseado no progresso até a presente data, quando necessário.	Atendido
O papel de *Product Owner* assim como todos os papéis Scrum, promovem o empirismo sobre uma equipe. Através disto, é possível fazer adaptações com segurança à ferramenta.	Não Atendido

A comparação das prescrições de estimativas do Scrum ao objeto resultou na análise de quatro itens com a categoria de *Não Atendido*. Sendo cada item composto pela prescrição, que consta a consideração de Schwaber (2013) ou Kniberg (2009), o comportamento do objeto, que destaca o atual comportamento do objeto perante a prescrição, a análise, que determina a análise entre a prescrição e o comportamento do objeto e a melhoria, que faz a sugestão de como o item pode obter uma maior aderência a prescrição.

SCR_01 – Sprints muito longas tendem a obter uma estimativa errônea

- Prescrição: Quando o horizonte da Sprint é muito longo, a definição do que será construído pode mudar, a complexidade pode aumentar e o risco pode crescer.

- Comportamento do Objeto: A média de dias das Sprints acompanhadas no objeto resultou em 27 dias.
- Análise: Quando se trata de uma Sprint muito longa esta vem acompanhada de riscos eminentes tanto na estimativa de um software quanto do incremento não ser o desejado pelo cliente.
- Melhoria: Sugere-se que a duração de uma Sprint inicie em 15 (quinze) dias e com maior ocorrência de assertividade nas estimativas seja gradualmente expandida, quando necessário.

SCR_02 – Product Owner não promove empirismo

- Prescrição: O papel de *Product Owner* e todos os papéis do Scrum promovem o empirismo sobre uma equipe. Com isso, é possível fazer adaptações com segurança ao framework.
- Comportamento do Objeto: O papel de *Product Owner* permanece por tempo determinado a equipe.
- Análise: A falta de experiência de como é o comportamento de determinada equipe pode resultar em atrasos significativos na entrega de incrementos, falta de confiança a equipe e falhas no planejamento.
- Melhoria: Sugere-se o papel de *Product Owner* fixo a equipe ou as equipes.

SCR_03 – Reunião de Revisão deve fornecer informações para a próxima Reunião de Planejamento

- Prescrição: O resultado da Reunião de Revisão da Sprint é um *Product Backlog* revisado que define o provável *Product Backlog* para a próxima Sprint.
- Comportamento do Objeto: Em análise ao objeto, a reunião de revisão não apresentou nenhum auxílio ou ligação entre a reunião de planejamento. Entretanto, localizou-se um tempo significativo entre uma Sprint e outra, de média de dez dias.
- Análise: Um tempo significativo entre as Sprints, assim como um *Product Owner* que não realiza a ligação entre as duas cerimônias, promovem a falha de planejamento das tarefas.
- Melhoria: Sugere-se que o objeto de resultado da reunião de revisão seja o *Product Backlog* revisado para a próxima Sprint.

SCR_04 – Reunião de meio da Sprint e Reunião Semanal promovem o erro na estimativa

- Prescrição: Eventos prescritos são usados para criar uma rotina e minimizar a necessidade de reuniões não definidas no Scrum.
- Comportamento do Objeto: Realiza uma reunião denominada “meio da Sprint” e uma reunião semanal, esta última juntamente com o cliente.
- Análise: Alteração no incremento anteriormente estimado no qual a estimativa não é refeita, promovem falha na estimativa.
- Melhoria: Sugere-se a busca da remoção do evento de reunião semanal e da reunião de “meio da Sprint”.

A Figura 2 apresenta o gráfico de aderência do objeto às prescrições de estimativa Scrum, sendo 54% atendido no objeto de Krauze (2014), 15% parcialmente atendido e 31% não atendido.

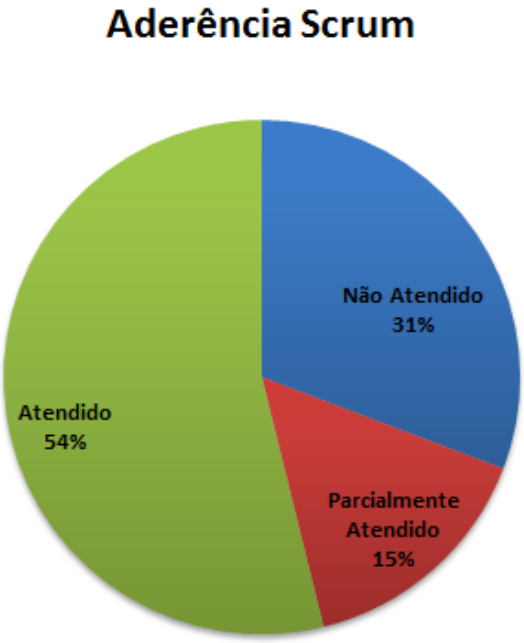

Figura 2. Gráfico de Aderência do Objeto ao Scrum.

Fonte: ***Primária***

Os resultados foram concedidos através das categorias da Tabela 1, considerando que foram mapeadas 13 prescrições de estimativa do framework Scrum. Sendo que 46% é a porcentagem faltante, considerando uma completa aderência em 100% para que o objeto de Krauze (2014) seja completamente fiel as prescrições de estimativa mapeadas no Scrum.

A aderência as prescrições Scrum certifica que as estimativas propostas pela literatura de Schwaber (2013) e Kniberg (2009) estão sendo executadas, com isso a promoção de uma maior assertividade no framework resultará também em um maior grau de acertos nas estimativas de um software.

5.2. Comparando Planning Poker ao Objeto

A comparação descreve a categoria de atendimento de Krauze (2014) em relação às prescrições de estimativas da técnica Planning Poker, a análise dos itens não atendidos e, por fim, um gráfico apresentando o quanto o objeto é aderente a técnica Planning Poker.

Para realizar a análise dos resultados, baseada na verificação do grau de atendimento, propõe-se uma escala ordenada de duas categorias: *Não Atendido* e *Atendido*. De forma a auxiliar na definição do quão aderente é o objeto de Krauze (2014) às prescrições definidas por Cohn (2013) e Grenning (2002).

As duas categorias são:

a) Não Atendido: Há pouca evidência de que o item foi satisfeito, ou a prescrição propõe diferentes propostas ao framework. Sendo os itens desta

categoria são destacados através de análise individual intitulados com a sigla PP, referenciado itens Planning Poker que não foram atendidos.

b) Atendido: Existe uma prática na satisfação da prescrição, existindo evidências significativas.

A Tabela 2 apresenta as prescrições definidas por Cohn (2013) e Grenning (2002) e a respectiva categoria de atendimento de Krauze (2014).

Tabela 2. Atendimento do Objeto às Prescrições de Estimativas a técnica Planning Poker.

Fonte: *Primária*

Prescrição	Categoria
Modelo de cartas da técnica Planning Poker.	Não Atendido
Product Owner lê os cartões e apresenta para a equipe.	Atendido
Cada membro da equipe deve pensar a respeito do tempo e esforço necessário para se implementar o cartão lido.	Atendido
Membros da equipe devem estimar considerando todas as tarefas envolvidas pelos responsáveis em desenvolver, testar e analisar.	Não Atendido
Caso a pontuação seja divergente o *Scrum Master* solicita aos membros que mostraram o menor e o maior valor estimado, para que expliquem o motivo que os levaram a tal estimativa.	Atendido
Deve-se escolher uma carta no baralho correspondente ao valor desta estimativa e coloca-a virada para baixo.	Atendido

A comparação das prescrições de estimativas da técnica Planning Poker ao objeto resultou na análise de dois itens com a categoria de *Não Atendido*. Sendo cada item composto pela prescrição, que consta a consideração de Cohn (2013) ou Grenning (2002), o comportamento do objeto, que destaca o atual comportamento do objeto perante a prescrição, a análise, que determina a análise entre a prescrição e o comportamento do objeto e a melhoria, que faz a sugestão de como o item pode obter uma maior aderência a prescrição.

PP_01 – A técnica Planning Poker fornece estimativa de tamanho

- Prescrição: Deve-se escolher uma carta no baralho correspondente ao valor da estimativa e coloca-la virada para baixo.
- Comportamento do Objeto: Após a pontuação de tamanho a estimativa é convertida em turnos.
- Análise: Ao converter pontos de tamanho para turnos de duração, altera-se a característica de uma estimativa de tamanho para uma estimativa de tempo, no qual perde-se o conjunto de características da estimativa de tamanho.
- Melhoria: Sugere-se manter as características de uma estimativa de tamanho utilizando tamanho ao invés de tempo.

PP_02 – Pontuação deve ser por cartão e não por processo

- Prescrição: Os membros da equipe devem estimar considerando todos os cartões envolvidos pelos responsáveis em desenvolver, analisar, testar e desenhar.

- Comportamento do Objeto: Os cartões são divididos em três (análise, desenvolvimento e testes) e estimados individualmente de acordo com cada processo.
- Análise: A pontuação por processo infere nas prescrições do Planning Poker promovendo uma estimativa errônea.
- Melhoria: Sugere-se estimar cartões considerando todas as tarefas.

A Figura 3 apresenta o gráfico de aderência do objeto as prescrições de estimativa da técnica Planning Poker, sendo 67% atendido no objeto de Krauze (2014), e 33% não atendido.

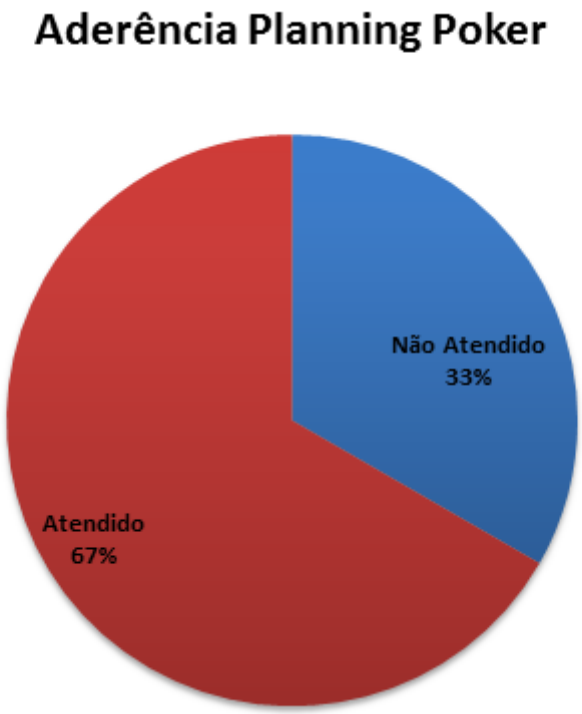

Figura 2. Gráfico de Aderência do Objeto a técnica Planning Poker.

Fonte: ***Primária***

Os resultados foram concedidos através das categorias da Tabela 2, considerando que foram mapeadas 6 prescrições de estimativa da técnica Planning Poker. Sendo que 33% é a porcentagem faltante, considerando uma completa aderência em 100% para que o objeto de Krauze (2014) seja completamente fiel as prescrições de estimativa mapeadas na técnica Planning Poker.

A aderência as prescrições da técnica certifica que as estimativas propostas pela literatura de Cohn (2013) e Grenning (2002) estão sendo executadas, com isso a promoção de uma maior assertividade na técnica resultará também em um maior grau de acertos nas estimativas de um software.

6. Considerações Finais

O presente analisou o grau de aderência do objeto de Krauze (2014) com relação aos frameworks Scrum e Kanban e técnica Planning Poker que se propõem. Para isto, foram amplamente estudadas as prescrições de Schwaber (2013), Kniberg (2009), Cohn (2013) e Grenning (2002).

Buscando uma análise de atendimento por parte do objeto as prescrições foram verificados 19 itens, sendo 13 do framework Scrum e 6 da técnica Planning Poker. Com o objetivo atendido e de forma a contribuir com o objeto de Krauze (2014), localizou-se 6 itens categorizados como *Não Atendido*, no qual foram analisados individualmente e propostos de melhoria.

Os itens Kanban não foram analisados, pois um dos princípios que dão ênfase ao framework, segundo Boeg (2012) e Kniberg (2009), o princípio de limitação de cartões nas colunas, o WIP, não foi atendido pelo objeto de Krauze (2014).

O objeto analisado obteve aderência superior a 50% em ambas as análises, porém existe um grau de aderência que pode ser perseguido a fim de promover maior assertividade nas estimativas de software.

O presente artigo pode ser continuado de várias formas visando aumentar a contribuição para as áreas de engenharia de software que pretendem aperfeiçoar sua estimativa de software. Alguns trabalhos sugeridos são: A aplicação das melhorias sugeridas de forma a analisar o comportamento do objeto de Krauze (2014), Analisar uma aderência da metodologia denominada ScrumBan ao objeto com o objetivo de localizar possíveis gargalos na estimativa de software e Mapear itens para a implantação do framework Kanban ao objeto de Krauze (2014).

7. Referências Bibliográficas

ABRAHAO, S., & Insfran, E., (2008) “A Metamodeling Approach to Estimate Software Size from Requirements Specifications”, Software Engineering and Advanced Applications, XXXIV , pp. 465-475, Washington (DC): IEEE Computer Society.

ALVES, da S. W. D., (2012) “Ferramenta para apoio à estimativa baseada em Planning Poker utilizando a metodologia Scrum”, UFPE – Recife, http://www.cin.ufpe.br/~tg/2012-1/dwas.pdf, Acesso em: 05 jun. 2014.

BASSI, F. D. L., (2010) “Experiências com desenvolvimento Ágil”, São Paulo, http://www.teses.usp.br/Dissertacao_Metodos_Ageis_Dairton_Bassi.pdf, Acesso em: 05 jun. 2014.

BOEG, J., (2012) “Kanban em 10 passos”, InfoQ, http://www.infoq.com/br/minibooks/priming-kanban-jesper-boeg, Acesso em: 05 jun. 2014.

CMMI, (2012) “Capability Maturity Model Integration”, CMMI for development: CMMI-DEV, Pittsburgh: Software Engineering Institute, v. 1.3, 573 p.

COCKBURN, A.; HIGHSMITH, J., (2001) “Agile Software Development: The Business of Innovation”. Pgs. 120-122.

COHN, M., (2013) “Agile Estimating and Planning”, http://www.mountaingoatsoftware.com/books/agile-estimating-and-planning e http://www.mountaingoatsoftware.com/blog/estimating-with-tee-shirt-sizes, Acesso em 05 jun. 2014.

GRENNING, J., (2002) “Planning Poker or How to avoid analysis paralysis while release planning”, http://renaissancesoftware.net/files/articles/PlanningPoker-v1.1.pdf, Acesso em 05 jun. 2014.

HAZAN, C., (2008) “Análise de Pontos de Função: Uma aplicação nas estimativas de tamanho de Projetos de Software”, Engenharia de Software, Rio de Janeiro, p.25-30.

KNIBERG, H.; SKARIN, M., (2009) “Kanban and Scrum – Making the most of both” Estados Unidos da América. ISBN: 978-0-557-13832-6. http://www.infoq.com/br/minibooks/kanban-scrum-minibook. Acesso em 05 jun. 2014.

KRAUZE, F. D., (2014) “Melhorando o Processo de Estimativa de Software em Equipe de Desenvolvimento Ágil”, Universidade de Passo Fundo, Passo Fundo.

SCHWABER, K. SUTHERLAND, J. (2013) “The Scrum Guide of Scrum.org”, https://www.scrum.org/Portals/0/Documents/Scrum%20Guides/2013/Scrum-Guide.pdf, Acesso em: 05 de jun. 2014.

ZANATTA, Alexandre. (2004) "xScrum: uma proposta de extensão de um Método Ágil para Gerência e Desenvolvimento de Requisitos visando adequação ao CMMI Florianópolis", Dissertação (Mestrado em Ciência da Computação) - Curso de Pós-Graduação em Ciência da Computação, Universidade Federal de Santa Catarina, www.tede.ufsc.br/teses/PGCC0651.pdf, Acesso em: 05 jun. 2014.