UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
COORDENADORIA DO CURSO DE ENGENHARIA DE SOFTWARE

BRUNO RUSSI LAUTENSCHLAGER

APLICAÇÃO DE MELHORIA CONTÍNUA COMO AUXÍLIO NA GESTÃO DE PROJETOS DE DESENVOLVIMENTO DE SISTEMAS DE GRANDE PORTE: UM ESTUDO DE CASO

TRABALHO DE CONCLUSÃO DE CURSO

DOIS VIZINHOS

2018

BRUNO RUSSI LAUTENSCHLAGER

APLICAÇÃO DE MELHORIA CONTÍNUA COMO AUXÍLIO NA GESTÃO DE PROJETOS DE DESENVOLVIMENTO DE SISTEMAS DE GRANDE PORTE: UM ESTUDO DE CASO

Trabalho de Conclusão de Curso apresentado como requisito parcial para obtenção do grau de Bacharel em Engenharia de Software, da Universidade Tecnológica Federal do Paraná.

Orientador: Me. Pedro Henrique de Alencar Machado

Co-orientador: Dr. Rafael Alves Paes de Oliveira

DOIS VIZINHOS

2018

TERMO DE APROVAÇÃO

Aplicação de melhoria continua como auxílio na gestão de projetos de desenvolvimento de sistemas de grande porte: Um estudo de caso

por

Bruno Russi Lautenschlager

Este Trabalho de Conclusão de Curso foi apresentado em 25 de Junho de 2018 como requisito parcial para a obtenção do título de Bacharel em Engenharia de Software. O(a) candidato(a) foi arguido(a) pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Pedro Henrique de Alencar Machado
Presidente da Banca

Marisangela Pacheco Brittes
Membro Titular

Simone de Souza Borges
Membro Titular

* A Folha de Aprovação assinada encontra-se na Coordenação do Curso

"O que pode ser medido, pode ser melhorado."

(Peter Drucker)

RESUMO

LAUTENSCHLAGER, Bruno. APLICAÇÃO DE MELHORIA CONTÍNUA COMO AUXÍLIO NA GESTÃO DE PROJETOS DE DESENVOLVIMENTO DE SISTEMAS DE GRANDE PORTE: UM ESTUDO DE CASO . 99 f. Monografia – Coordenadoria do Curso de Engenharia de Software, Universidade Tecnológica Federal do Paraná. Dois Vizinhos, 2018.

Com a evolução da engenharia de software, os métodos e processos utilizados para a construção de sistemas, também sofreram adaptações junto com a evolução mercadológica. Um conceito já difundido na academia, e na indústria, são as metodologias ágeis para o desenvolvimento de software. Essa prática está sendo bem aceita pelo mercado, pois vem mostrando bons resultados comparados às práticas tradicionais. No entanto, estudos vêm apontando algumas dificuldades encontradas no desenvolvimento de sistemas utilizando de práticas ágeis. Trata-se do desenvolvimento de softwares complexos ou de grande porte. Visando contribuir com o alívio de tais dificuldades, o presente estudo aplicou conceitos de melhoria contínua *(kaizen)*. Este estudo apresenta um projeto de pesquisa acerca da melhoria contínua no gerenciamento de projetos ágeis em empresas grande porte. É importante destacar que as melhorias propostas serão validadas em uma empresa real.

Palavras-chave: Ágil, Melhoria contínua, Lean, Kaizen, Software de grande porte

ABSTRACT

LAUTENSCHLAGER, Bruno. CONTINUOUS IMPROVEMENT APPLICATION AS AN AID IN THE MANAGEMENT OF LARGE SYSTEMS DEVELOPMENT PROJECTS: A CASE STUDY. 99 f. Monografia – Coordenadoria do Curso de Engenharia de Software, Universidade Tecnológica Federal do Paraná. Dois Vizinhos, 2018.

Regarding the Software Engineering advances, the methods and processes used to implement solution have also undergone adaptations along with market evolution. A concept already widespread in academia and industry is agile methodologies for software development. This practice is being well accepted by practitioneers, since it has been showing acceptable results compared to traditional practices. However, studies have pointed out some difficulties encountered in the development of systems using agile practices when developing complex or large software. The application of continuous improvement in agile methodologies can help solving the difficulties encountered in this process. This study presents a research project about continuous improvement in the management of large agile projects. The proposed improvements will be valid in a real-world large company.

Keywords: Agile, Continuous improvement, Lean, Kaizen, Complex software

LISTA DE FIGURAS

FIGURA 1 – Ciclo de vida do *scrum* 23
FIGURA 2 – Diagrama condução das atividades 32
FIGURA 3 – Triangulação dos dados coletados 34
FIGURA 4 – Formulário para registro de fatos 36
FIGURA 5 – Planilha de dados 38
FIGURA 6 – Planilha de dados 41
FIGURA 7 – Gráfico de dados do time 1 43
FIGURA 8 – Gráfico de dados do time 2 51
FIGURA 9 – Gráfico de dados do time 3 55
FIGURA 10 – Gráfico de dados do time 4 62
FIGURA 11 – Gráfico de dados do time 5 71
FIGURA 12 – Gráfico de dados do time 6 78
FIGURA 13 – Gráfico de dados do time 7 84
FIGURA 14 – Gráfico de respostas a questão 1 88
FIGURA 15 – Gráfico de respostas a questão 2 89
FIGURA 16 – Gráfico de respostas a questão 3 90

LISTA DE TABELAS

TABELA 1 – Estrutura dos times de manutenção 33
TABELA 2 – Estrutura dos times de desenvolvimento 34
TABELA 3 – Exemplo de dados sumarizados 40
TABELA 4 – Tabela dos dados coletados do time 1 43
TABELA 5 – Tabela de Ações, Motivações e resultados do time 1 49
TABELA 6 – Tabela dos dados coletados do time 2 51
TABELA 7 – Tabela de Ações, Motivações e resultados do time 2 53
TABELA 8 – Tabela dos dados coletados do time 3 54
TABELA 9 – Tabela de Ações, Motivações e resultados do time 3 60
TABELA 10 – Tabela dos dados coletados do time 4 62
TABELA 11 – Tabela de Ações, Motivações e resultados do time 4 69
TABELA 12 – Tabela dos dados coletados do time 5 70
TABELA 13 – Tabela de Ações, Motivações e resultados do time 5 76
TABELA 14 – Tabela dos dados coletados do time 6 77
TABELA 15 – Tabela de Ações, Motivações e resultados do time 6 82
TABELA 16 – Tabela dos dados coletados do time 7 83
TABELA 17 – Tabela das principais categorias 86
TABELA 18 – Tabela de categorias e soluções propostas genéricas 87

LISTA DE SIGLAS

XP	Extreme Programming
PDCA	Plan, Do, Check, Act
DMAIC	Define, Measure, Analyse, Improve e Control

SUMÁRIO

1 INTRODUÇÃO 12
1.1 CONTEXTO 12
1.2 JUSTIFICATIVA 13
1.3 QUESTÕES DE PESQUISA 15
1.4 OBJETIVOS 16
1.4.1 Objetivo Geral 16
1.4.2 Objetivos Específicos 16
1.5 ORGANIZAÇÃO DO TRABALHO 17
2 REFERENCIAL TEÓRICO 18
2.1 ENGENHARIA DE *SOFTWARE* 18
2.2 METODOLOGIAS DE DESENVOLVIMENTO DE *SOFTWARE* 19
2.2.1 Metodologias e Processos de Desenvolvimento Tradicionais 19
2.2.2 Metodologias de Desenvolvimento Ágeis 20
2.2.2.1 Extreme Programming (XP) 21
2.2.2.2 Scrum 21
2.3 SOFTWARE DE GRANDE PORTE 23
2.4 LEAN 25
2.4.1 *Lean Software Development* 26
2.5 MELHORIA CONTÍNUA 28
2.5.1 Melhoria Contínua no Processo de Desenvolvimento de Produtos 29
3 APLICAÇÃO DA MELHORIA CONTÍNUA: UM ESTUDO DE CASO 31
3.1 CENÁRIO ATUAL 31
3.2 ESTUDO DE CASO 31
3.3 ESCOLHA DOS TIMES 33
3.4 PROCEDIMENTO DE COLETA DE DADOS 34
3.4.1 Observação 35
3.4.2 Entrevista 35
3.4.3 Dados Arquivados 35
3.5 PROCEDIMENTO DE REFINAMENTO E ANÁLISE DOS DADOS 38
3.6 PROCEDIMENTO DE VALIDAÇÃO DOS DADOS 40
4 RESULTADOS 42
4.1 TIMES DE DESENVOLVIMENTO 42
4.1.1 Time 1 42
4.1.1.1 Apresentação das informações obtidas 42
4.1.1.2 Análise das informações obtidas 43
4.1.1.2.1 Pontos Positivos: ..... p. 44
4.1.1.2.2 Pontos Negativos: ..... p. 44
4.1.1.2.3 Pontos a Melhorar: ..... p. 44

4.1.1.3 Ações Executadas 45
4.1.1.4 Resultado do Time 49
4.1.2 Time 2 50
4.1.2.1 Apresentação das informações obtidas 50
4.1.2.2 Análise das informações obtidas 51
4.1.2.2.1 Pontos Positivos: ..... p. 52
4.1.2.2.2 Pontos Negativos: ..... p. 52
4.1.2.2.3 Pontos a Melhorar: ..... p. 52
4.1.2.3 Ações Executadas 52
4.1.2.4 Resultado do Time 53
4.1.3 Time 3 54
4.1.3.1 Apresentação das informações obtidas 54
4.1.3.2 Análise das informações obtidas 55
4.1.3.2.1 Pontos Positivos ..... p. 55
4.1.3.2.2 Pontos Negativos ..... p. 55
4.1.3.2.3 Pontos a Melhorar: ..... p. 56
4.1.3.3 Ações Executadas 56
4.1.3.4 Resultado do Time 61
4.1.4 Time 4 61
4.1.4.1 Apresentação das informações obtidas 61
4.1.4.2 Análise das informações obtidas 63
4.1.4.2.1 Pontos Positivos: ..... p. 63
4.1.4.2.2 Pontos Negativos: ..... p. 63
4.1.4.2.3 Pontos a Melhorar: ..... p. 63
4.1.4.3 Ações Executadas 64
4.1.4.4 Resultado do Time 69
4.1.5 Time 5 70
4.1.5.1 Apresentação das informações obtidas 70
4.1.5.2 Análise das informações obtidas 71
4.1.5.2.1 Pontos Positivos: ..... p. 71
4.1.5.2.2 Pontos Negativos: ..... p. 72
4.1.5.2.3 Pontos a Melhorar: ..... p. 72
4.1.5.3 Ações Executadas 72
4.1.5.4 Resultado do Time 76
4.2 TIMES DE MANUTENÇÃO 77
4.2.1 Time 6 77
4.2.1.1 Apresentação das informações obtidas 77
4.2.1.2 Análise das informações obtidas 78

4.2.1.2.1 Pontos Positivos: . . . . . p. 78
4.2.1.2.2 Pontos Negativos: . . . . . p. 79
4.2.1.2.3 Pontos a Melhorar: . . . . . p. 79
4.2.1.3 Ações Executadas 79
4.2.1.4 Resultado do Time 82
4.2.2 Time 7 83
4.2.2.1 Apresentação das informações obtidas 83
4.2.2.2 Análise das informações obtidas 84
4.2.2.2.1 Pontos Positivos: . . . . . p. 84
4.2.2.2.2 Pontos Negativos: . . . . . p. 85
4.2.2.2.3 Pontos a Melhorar: . . . . . p. 85
4.2.2.3 Resultado do Time 85
4.3 RESULTADOS FINAIS 85
4.3.1 Principais categorias e soluções 86
4.3.2 Aceitação e avaliação de técnicas de melhoria contínua 87
4.4 DISCUSSÕES 90
4.4.1 Ameaças 91
4.4.1.1 Tempo dos líderes 92
4.4.1.2 Resultados demorados 92
4.4.1.3 Participação 92
5 CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS 93
REFERÊNCIAS 95

1 INTRODUÇÃO

1.1 CONTEXTO

Sistemas de *software* de grande porte envolvem diversas áreas de conhecimento, além de afetarem um grande número de usuários. Pelo fato de incorporar diferentes segmentos, sistemas de grande porte se tornam cada vez mais complexos com o passar do tempo e a evolução no seu ciclo de vida (MISHRA; MISHRA, 2011; RAUSCH et al., 2014; WINGO; TANIK, 2015).

Com a evolução da Engenharia de *Software*, os métodos e processos utilizados para a construção de sistemas, também sofreram adaptações junto com a evolução mercadológica. Um conceito já difundido na academia e na indústria, são as metodologias ágeis para o desenvolvimento de *software* (MISHRA; MISHRA, 2011). Essa prática está sendo bem aceita pelo mercado, pois vem mostrando bons resultados comparados às práticas tradicionais. No entanto, estudos vêm apontando algumas dificuldades encontradas no desenvolvimento de sistemas utilizando de práticas ágeis. Trata-se do desenvolvimento de *softwares* complexos ou de grande porte utilizando essas práticas (SUIKKI et al., 2006; WINGO; TANIK, 2015).

As metodologias ágeis começaram a surgir depois da criação do manifesto ágil, cujos princípios e valores são referencial teórico Beck et al. (2001). De acordo com Highsmith (2009), agilidade relacionado ao desenvolvimento de software é: "a habilidade de criar e responder a mudanças, buscando a obtenção de lucro em um ambiente de negócio turbulento". As metodologias ágeis buscam entregar um maior valor ao cliente de uma forma mais interativa e com mais qualidade no produto final. Para isso, algumas práticas são adotadas e incorporadas nos *frameworks* ágeis. É o caso da melhoria contínua (MISHRA; MISHRA, 2011; POPPENDIECK; POPPENDIECK, 2003). O conceito de melhoria contínua (*kaizen*) define que deve-se analisar o processo, eliminar as impurezas e maximizar o todo, o que proporciona uma maior produtividade. Imai (1994) descreve esta prática como:

> "... A essência do *kaizen* é simples e direta: *kaizen* significa melhoramento. Mais ainda, *kaizen* significa contínuo melhoramento, envolvendo todos, inclusive gerentes e operários. A filosofia do *kaizen* afirma que o nosso modo de vida – seja no trabalho, na sociedade ou em casa – merece ser constantemente melhorado ..." (IMAI, 1994).

De acordo com Mishra e Mishra (2011), existem nove grandes lacunas no processo de desenvolvimento de *software* ágil, quando aplicado na construção de sistemas de grande porte:

1. Desafios com a realização de testes contínuos;
2. O aumento do esforço de manutenção com a variação de versões;
3. Despesas de gerenciamento devido à necessidade de coordenação entre equipes;
4. Devido à falta de foco no *design* dependências detalhadas não são descobertas;
5. Longa duração da engenharia de requisitos, devido a processos de decisão complexos;
6. Listas de prioridades dos requisitos são difíceis de criar e manter;
7. Tempos de espera no processo, especificamente na concepção em espera dos requisitos;
8. Redução da cobertura de testes devido a falta de testes independentes;
9. Maior esforço no gerenciamento das configurações.

Com base nas dificuldades ilustradas por Mishra e Mishra (2011), a proposta deste trabalho é a aplicação de um processo de melhoria contínua como auxílio na gestão de equipes ágeis no desenvolvimento de software de grande porte. Por meio de estudos empíricos realizados em distintas equipes de uma organização desenvolvedora de sistemas de grande porte, pretende-se identificar a associação de melhoria contínua com métodos ágeis.

1.2 JUSTIFICATIVA

Com a evolução e a necessidade de informação a tecnologia em *software* vem se tornando cada vez mais presente no dia a dia das pessoas, independente da área de atuação. O setor de desenvolvimento de *software* está presente em várias áreas hoje, e com isso, vem ganhando maior visibilidade e complexidade. Por ser uma nova necessidade

na estrutura do mercado, a indústria de *software* procurou alguns conceitos já utilizados em outras áreas, aderindo inicialmente aos modelos baseados no processo cascata. Surgiu então a Engenharia de *software*, a partir de então houve o encontro com o setor industrial tradicional, em questão aquele baseado nas ideias Tayloristas (BOEHM, 2006; ROYCE, 1987).

Ainda baseado na indústria fabril, as metodologias que surgiram a partir do modelo de cascata tiveram um processo de desenvolvimento bastante rígido, como exemplo podemos citar: processos muito sequenciais e documentações excessivamente rigorosas, deixando as equipes de desenvolvimento presas a processos pré-definidos. Assim gerando um grande desperdício de trabalho ao decorrer do projeto (NEILL; LAPLANTE, 2003; RAJLICH, 2006).

A partir do século XXI, o mercado de software emergiu para uma fase de alta competição, na qual ocorreram algumas mudanças no processo de desenvolvimento de software, como: alterações constantes nos requisitos, prazos demasiadamente curtos e entregas mais rápidas e eficientes (geram valor ao produto) (HIGHSMITH; COCKBURN, 2001). De acordo com Takeuchi e Nonaka (1998) essas mudanças também já estavam ocorrendo na indústria tradicional.

Considerando o mercado atual de software, não há mais espaço para um *software* entregue com baixa qualidade, portanto a qualidade de um produto entregue ao cliente final é algo crucial para que se tenha um negócio de sucesso, e além disso a qualidade é um comprometimento que as empresas têm com seus clientes, ainda mais em um processo bastante complexo. Para isso surgiram várias metodologias e *frameworks* com o intuito de promover o aumento da qualidade do produto e do processo adotado em seu desenvolvimento. Além da qualidade as metodologias e *frameworks* ágeis buscam tratar alguns pontos enfrentados no desenvolvimento, como: atrasos em entregas, mudanças de requisitos, número exponencial de defeitos e bugs e até a saída de alguns membros do time.

De acordo com PENHA (1993): "A concorrência saudável é necessária ao equilíbrio da lei da oferta e da procura, sendo um ótimo remédio para evitar a acomodação". A concorrência só irá ocorrer se todas as empresas envolvidas tenham um objetivo em especifico, que é a qualidade.

A complexidade da aplicação de metodologias ágeis para o desenvolvimento de software de grande porte é um sério problema que vem sendo enfrentado pelo mercado. Como uma empresa de grande porte geralmente já está dentro do mercado de software

há algum tempo, sua estrutura organizacional não adota como regra o que os *frameworks* ágeis propõem para sua utilização, portanto sua estrutura organizacional é uma adaptação ao *framework* ágil.

O *frameworks Scrum* prega que uma equipe ágil deve ser auto gerenciável, sem a necessidade de um gerente de projetos dentro da equipe, porém nada impede que exista um escritório de projetos com esses papéis. Esse é um ponto em que uma empresa de software de grande porte pode transgredir. Outro ponto que os *frameworks* ágeis implantam, é que a equipe deve ser multidisciplinar, porém dentro de uma equipe na qual cada pessoa era responsável por apenas uma tarefa, isso é uma ação difícil de ser adotada, não impossível mas difícil, pois em algumas tarefas específicas não é necessário somente saber como fazer, mas é necessário ter a *skill* para a execução daquela atividade.

As metodologias ágeis tem-se mostrado uma escolha viável para o desenvolvimento de *software*, pois estão atendendo de uma forma satisfatória as necessidades da evolução mercadológica, para tanto, é necessário que as falhas sejam sanadas, para que se obtenha um melhor resultado. A priori uma medida adotada para sanar essas falhas, seria a prática da melhoria contínua no processo de desenvolvimento de *software*.

Dado o presente contexto, essa pesquisa é necessária para aproximar ambientes de produção de software que utilizam metodologias ágeis a conceitos de melhoria contínua. Além disso, poder contribuir com um cenário real de desenvolvimento de software, este estudo visa servir como base de conhecimento para equipes que tenham dificuldade na aplicação de métodos ágeis para desenvolvimento de software de grande porte.

1.3 QUESTÕES DE PESQUISA

É importante destacar que este estudo surgiu de um problema real de indústria por meio da cooperação com a academia. Sendo assim, uma Questão de Pesquisa foi explorada desde o início da cooperação. A partir das problemáticas destacadas por Mishra e Mishra (2011) no processo de desenvolvimento de *software* complexo ou de grande porte e das melhorias identificadas com a utilização de metodologias ágeis de desenvolvimento de *software*, surgiu a problemática deste estudo, a saber:

- De que forma a aplicação da técnica de melhoria contínua, pode auxiliar no desenvolvimento de sistema de *software* de grande porte?

Sendo assim, devido à natureza do problema, este estudo visa a levantar evidências de pesquisa que possam contribuir com a resposta de tal questão de pesquisa. É importante destacar que um descrição sucinta e genérica de tal problemática foi previamente abordado, inserido e contextualizo por Lautenschlager et al. (2017).

1.4 OBJETIVOS

A seguir tais objetivos são evidenciados com maiores detalhes.

1.4.1 OBJETIVO GERAL

O objetivo geral do presente trabalho é aplicar técnicas de melhoria contínua com o propósito de auxiliar no processo de desenvolvimento de *software* de grande porte. A partir da implementação da técnica de melhoria contínua junto a metodologias ágeis, vislumbram-se a proposição e avaliação de um método estruturado que possibilite a redução de desperdício na execução de atividades durante o desenvolvimento do software.

1.4.2 OBJETIVOS ESPECÍFICOS

A partir do objetivo geral apresentando, derivam-se os seguintes objetivos específicos:

- Entender e catalogar o contexto atual do processo de desenvolvimento de *software* de grande porte com metodologias ágeis em uma empresa;
- Contextualizar o desenvolvimento de *software* de grande porte em cenários reais de desenvolvimento;
- Propor uma técnica de melhoria contínua em times ágeis de desenvolvimento de *software* de grande porte;
- Acompanhar o processo da aplicação da técnica de melhoria continua; e
- Coletar e apresentar os resultados a partir da aplicação da técnica de melhoria contínua.

1.5 ORGANIZAÇÃO DO TRABALHO

O trabalho em questão foi estruturado em o outras quatro seções. A segunda seção traz o referencial teórico onde são compilados os conceitos no qual o trabalho esta embasado. A terceira seção traz uma descrição do cenário onde foi feito o estudo e toda a metodologia utilizada para o desenvolvimento do trabalho. Na quarta seção são demonstrados e discutidos os resultados obtidos durante o estudo de caso. Em seguida são feitas as considerações finais do autor, e apresentados os trabalhos futuros.

2 REFERENCIAL TEÓRICO

Este capítulo aborda tópicos relacionados à temática que sustentam essa pesquisa. São eles: Engenharia de *software*, processo de desenvolvimento de *software*, melhoria continua e *Lean*.

2.1 ENGENHARIA DE *SOFTWARE*

Nos dias de atuais, quase todos os setores da sociedade dependem de um *software*, infraestrutura e serviços,que se tornaram complexos com a adoção dos circuitos integrados. A maioria dos produtos eletrônicos possui um computador ou um controlador e com isso um *software* de controle. Com base nisso, produzir *software* dentro de custos adequados e qualidade é essencial para o desenvolvimento da economia mundial. (SOMMERVILLE et al., 2011).

O conceito de engenharia de *software* se fortaleceu por conta de iniciativas para solucionar a "crise de *software*". A tal crise de *software* não era somente um problema de *software*, mas sim de *software* e *hardware*. A partir da crise foi-se implementado um computador baseado em circuitos integrados, a aplicação dos circuitos fez com que surgisse problemas mais complexos e maiores para o mundo do *software*. (SOMMERVILLE et al., 2011).

Diversas definições são atribuídas a engenharia de *software*. Todas elas são aceitas na literatura. Segundo Sommerville et al. (2011), a engenharia de *software* tem como objetivo é o desenvolvimento de *software* adequado e com uma boa qualidade. A engenharia de *software* é uma abordagem que tem varias camadas, que abrangem ferramentas, métodos, processo e qualidade. Todas as engenharias estão fundamentadas com um objetivo em especifico, que é a qualidade.

A gestão da qualidade total Seis Sigma (GYGI, 2008) e algumas filosofias parecidas presam a cultura do aperfeiçoamento contínuo de processos. E é esse pensamento leva a engenharia de *software* ser mais efetiva cada vez mais. O principal

foco da engenharia de software é a qualidade (PRESSMAN, 2016). Desse modo, surgiram os processos de desenvolvimento de *software*, que serão abordados nas próximas seções.

2.2 METODOLOGIAS DE DESENVOLVIMENTO DE *SOFTWARE*

Metodologias de desenvolvimento de *software* são um conjunto de práticas recomendadas para o desenvolvimento de aplicações. Essas práticas podem ser subdivididas em fases para ordenar e gerenciar o processo (SOMMERVILLE et al., 2011).

De acordo com Pressman (2016), a atividade de desenvolvimento de *software*, quando surgiu, era executada sem nenhum tipo de planejamento. Isso acarretava em produtos de má qualidade que muitas vezes não correspondiam às necessidades do cliente, desperdiçando recursos financeiros, esforço e tempo. Essa época ficou conhecida como o período da Crise do *Software*.

A partir dessa problemática, surgiram os primeiros processos de desenvolvimento de sistemas de *software*, que tendiam a planejar e padronizar cada uma das técnicas envolvidas em todo o procedimento a fim de, entre tantas outras, atender e corresponder às expectativas dos contratantes clientes. Esses processos se adequam às características organizacionais, ao ambiente de desenvolvimento implementado em cada corporação, ao paradigma de desenvolvimento, à plataforma do *software*, às características do projeto, entre tantas outras (NETO, 2004)

Mesmo a partir da criação de tantos paradigmas e conceitos, uma série de problemas enfrentadas no início, ainda existem atualmente, com projetos atrasados, estimativas de custo e tempo errôneas, entre outras problemáticas.

2.2.1 METODOLOGIAS E PROCESSOS DE DESENVOLVIMENTO TRADICIONAIS

Os modelos de processos foram criados para trazer uma ordem ao caos que existia naquela época no desenvolvimento de *software*. Os modelos tradicionais geraram um contribuição considerável para a estruturação da Engenharia de *Software*, e também foram muito eficazes para as equipes que desenvolvem software. (PRESSMAN, 2016).

As metodologias conhecidas como "tradicionais" têm como principal característica o fato de serem divididas em etapas ou fases. Essas etapas são muito bem definidas e englobam pequenas atividades como Análise de Requisitos, Modelagem, Desenvolvimento e Testes. (NETO, 2004).

A cada conclusão de uma etapa, geralmente um documento ou diagrama é concebido. A conclusão de tais etapas são conhecidos como marcos.

A principal preocupação das práticas tradicionais é a gestão dos requisitos do sistema, que traz a vantagem de tornar os projetos totalmente planejados, facilitando a sua gerência, mantendo sempre uma mesma linha. (ROCHA et al., 2004).

Para Pressman (2016), os modelos de processo de desenvolvimento de software mais comuns são:

- Modelo Cascata: O modelos cascata, também conhecido como ciclo de vida clássico, segue uma abordagem sequencial e sistemática de desenvolvimento, avançando de etapa por etapa, seguindo levantamento de requisitos, planejamento, modelagem, construção e posteriormente suporte ao cliente.
- Modelo Incremental: O modele incrementar tem como característica aplicar uma abordagem escalonada, um pouco de cada vez. Cada entrega do software gera um incremento, assim gerando uma versão do mesmo.

2.2.2 METODOLOGIAS DE DESENVOLVIMENTO ÁGEIS

Metodologias ágeis surgiram em meados de 2001 a partir de problemáticas representadas pelas metodologias "tradicionais": (1) Alta dependência de processos formais e (2) burocrática resposta quanto às mudanças de requisitos (TELES, 2004).

Depois de algum tempo de pesquisa, especialistas da indústria de software se uniram para definirem novos valores e princípios relacionados ao desenvolvimento e escreveram um manifesto que ficou conhecido, posteriormente, como *Manifest for Agile Software Development* (BECK et al., 2001). O referente manifesto, basicamente, destacava quatro valores:

- Indivíduos e interações ao invés de processos e ferramentas;
- *Software* funcional ao invés de documentação detalhada;
- Colaboração com o cliente ao invés de negociação de contrato; e
- Responder às mudanças ao invés de seguir um plano.

Como principais modelos de processos ágeis, destacam-se os seguintes: *Scrum, Extreme Programming, Feature Drivem Development, DSDM, Crystal*, entre outras.

Dente todas esses modelos de processos ágeis, dois deles são os mais adotados pelo mercado, *Scrum* e XP. A seguir é apresentado um detalhamento de como são as praticas de cada um desses processos.

2.2.2.1 EXTREME PROGRAMMING (XP)

O primeiro processo ágil para tornar-se popular foi o XP, do ingles *Extreme Programming* (BECK, 2000). XP se concentra em várias práticas técnicas, sendo o desenvolvimento mais notável em *test-driven*. O TDD, do ingles *Teste Driven Development* resulta em uma habilidade de teste unitário, que é executado com freqüência, permitindo detectar defeitos quase imediatamente após serem injetados na base do código, em outras palavras teste são processados e códigos são implementados parra contemplarem tais testes. Esta abordagem provou ser um primeiro passo efetivo no erro (POPPENDIECK; CUSUMANO, 2012). Porém como o foco da pesquisa está inserido em um ambiente *Scrum*, o XP não é muito relevante para este trabalho.

2.2.2.2 SCRUM

O *Scrum* é uma metodologia que surgiu nos anos 1990 para o gerenciamento de processos complexos. *Scrum* não é um processo, técnica ou método definitivo, mas sim um *framework* (SCHWABER; SUTHERLAND, 2013).

Para Prikladnicki et al. (2014), o *framework Scrum* auxilia no gerenciamento da produção de qualquer projeto ou produto. O *Scrum* é chamado de *famework*, pois conta com um conjunto de práticas leves e objetivas, as quais são muito utilizadas no desenvolvimento de *software*.

De acordo com Schwaber e Sutherland (2013), o *Scrum* é composto por uma abordagem empírica com interações de curta duração (*sprints*) com reuniões de acompanhamento diárias (*daily meeting*). Isso permite que sejam antecipados enventuais problemas e de uma maior visibilidade da interação. O *framework Scrum* aplica ideias do controle de processo industrial para o desenvolvimento de *software* com a introdução de algumas ideias de adaptabilidade, flexibilidade e produtividade. O objetivo do *Scrum* é encontrar um meio de desenvolver *software* de forma flexível em um ambiente com constantes mudanças. Três pilares mantêm todas as implementações do controle empírico dos processos: (1) transparência, (2) inspeção e (3) adaptação. (SCHWABER; SUTHERLAND, 2013).

- Transparência: Aspectos significativos do processo devem ser visíveis para os responsáveis pelo resultado. A transparência exige que esses aspectos sejam definidos por um padrão comum para que o time compartilhe um entendimento comum do que está sendo feito;

- Inspeção: O time *Scrum* devem frequentemente inspecionar artefatos e progredir em direção a um objetivo para detectar variações indesejáveis. Sua inspeção não deve ser tão frequente que a inspeção interfira no caminho do trabalho. As inspeções são mais benéficas quando realizadas diligentemente por inspetores qualificados no ponto de trabalho; e

- Adaptação: Se um inspetor determinar que um ou mais aspectos de um processo se desviam para fora de limites aceitáveis e que o produto resultante será inaceitável, o processo ou o material em processamento deve ser ajustado. Um ajuste deve ser feito o mais rápido possível para minimizar o desvio adicional.

De acordo com Sutherland (2010), no *Scrum* tem-se três papeis:

- *Scrum Master*: responsável por ensinar a como dever ser feita a utilização do *Scrum*, acompanhar o desenvolvimento e resolver os impedimentos do time;

- *Product owner*: responsável por garantir que o produto atenta todas as necessidades que foram solicitadas, priorizar as atividades a serem desenvolvidas de forma que gerem mais valor ao cliente; e

- Equipe de desenvolvimento: são formadas por arquitetos, analistas de qualidade e desenvolvedores.

No *Scrum*, os requisitos são representador por histórias de usuários, que em conjunto geram o *product backlog*. No início de cada iteração (*sprint*) é realizado uma cerimonia de planejamento, na qual são definidas e estimadas as histórias de usuários que que estão no *product backlog*, assim gerando o *sprint backlog* (tudo o que vai ser desenvolvido durante a *sprint*), assim garantindo a meta da *sprint*. Depois do planejamento realizado, o desenvolvimento da *sprint* é iniciado, podendo ter duração de duas a quatro semanas, dependendo do projeto. Durante a execução da *sprint* o *scrum master* deve assegurar que as historias de usuários não mudem, permitindo que o time fique concentrado em um objetivo. O *product owner* acompanha o desenvolvimento da *sprint* e debate algumas duvidas que podem surgir. Sendo assim, ele só pode realizar

alguma alteração na *sprint* se alguma regra altera o comportamento esperado do produto em desenvolvimento (SUTHERLAND, 2010).

Depois de finalizada a *sprint* tem-se uma reunião chamada de *sprint review*. Nessa etapa o *product owner* verifica se a meta da *sprint* foi atingida. Após a *sprint review*, tem-se outra reunião chamada de *sprint retrospective*, na qual o time discute quais foram os pontos bons e ruins da *sprint*. *Sprin reviews* evitam que os mesmo erros sejam cometidos nas próximas *sprints* e continuar com as boas praticas (SUTHERLAND, 2010).

Conforme a Figura 1 mostra, o *Scrum* tem um fluxo de trabalho bem definido.

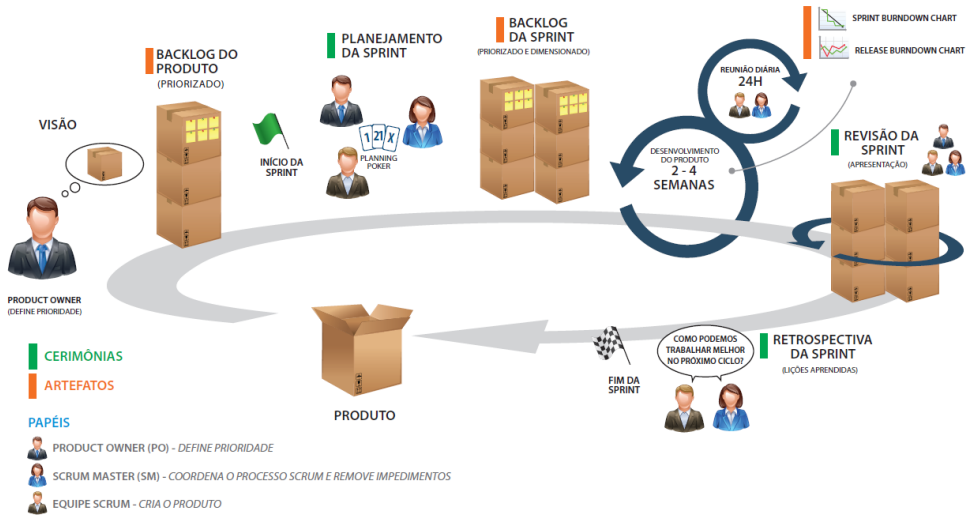


Figura 1: Ciclo de vida do *scrum*
Fonte: Adaptado de: (NEGOCIO, 2017)

O *Scrum* tem o intuito de simplificar a execução das atividades, permitindo interatividade e incrementos ágeis no processo de gestão do projeto.

Nos dias atuais diversas empresas estão optando pela utilização do *framework Scrum* como metodologia de desenvolvimento. Como o *Scrum* vem mostrando ser bem promissor com a qualidade e agilidade em que o produto é entregue, tanto empresas de pequeno porte, quanto empresas de grande porte vem se fazendo valor dos conceitos de *Scrum* em seus produtos.

2.3 SOFTWARE DE GRANDE PORTE

Na estrutura mercadológica atual, um conceito para que as empresas de *software* tenham uma carreira de sucesso é atender o que foi acordado com seus clientes

dentro do prazo definido e com o maior valor possível ao negócio. Com base nessa afirmação, o foco do desenvolvimento não deve ser somente na parte técnica, mas sim, na parte de gerenciamento e processos. Por tal razão, alguns pesquisadores afirmam que as metodologias ágeis popularmente conhecidas não são suficientes para garantir o desenvolvimento de um produto (COHEN et al., 2003 apud MACHADO, 2016).

Apesar da lacuna sugerida por Cohen et al. (2003), outros autores destacam algumas diferentes dificuldades na utilização dos princípios ágeis, sobretudo quando inserido em um meio de produção de *software*. Um dos fundamentos das praticas ágeis diz que o envolvimento do cliente durante o desenvolvimento do produto é fundamental, o que requer que eles temam total conhecimento sobre o problema em que o *software* está sendo incorporado. Considerando a complexidade que gira entorno de um *software* de grande porte, na maior parte dos dos casos não é somente um profissional que concentra o conhecimento necessário para o desenvolvimento de toda o sistema, levando a inserção de mais profissionais durante o processo de desenvolvimento. Com o envolvimento de varias pessoas para varias partes do sistema a chance de risco ao produto pode aumentar, podendo a chegar nada de valor ao cliente (COHEN et al., 2003 apud MACHADO, 2016), (MISHRA; MISHRA, 2011 apud MACHADO, 2016), (NERUR et al., 2005 apud MACHADO, 2016), (SILVA et al., 2011 apud MACHADO, 2016).

Segundo Pernstål et al. (2013), um *software* de grande porte nem sempre é homogéneo, ou seja, pode ser construido de forma separada a fim de se comunicar como um todo posteriormente. O desenvolvimento dessa partes não homogéneas pode ser feita por outra equipe ou empresa, e talvez nem estejam geograficamente juntos é o caso de muitas empresas multinacionais. Devido a isso, é necessária a efetivação de alguns processos de forma ágil, para que exista uma melhor comunicação e consenso, como a documentação e gerenciamento dos requisitos, visão geral do andamento do projeto, gerenciamento dos recursos e, além disso, alguns outros princípios ágeis (BIFFL et al., 2006 apud MACHADO, 2016).

As características peculiares e de difícil resolução fazem com que empresas de *software* de grande porte procurem por uma solução, por meio da investigação do seu processo de desenvolvimento de forma geral com o objetivo de entregar cada vez mais valor ao seu cliente, dessa forma a filosofia *Lean* tem sido mais procurada no mercado (MUSAT; RODRÍGUEZ, 2010 apud MACHADO, 2016).

2.4 LEAN

O *Lean* não é uma metodologia, *framework* ou processo, mas sim um modo de pensar como entregar mais valor ao cliente com a eliminação de desperdícios no processo de produção. O *Lean* teve origem do sistema Toyota de produção, do inglês (TPS – *Toyota Production System*) como *Lean manufacturing*. O *Lean* é um filosofia de gestão que possibilita algumas formas de gerar mais valor ao cliente, melhorar o fluxo de produção, melhorar a eficiência e eliminar os desperdícios da produção (POPPENDIECK; CUSUMANO, 2012).

O *Lean* surgiu para substituir o sistema de produção em massa que foi popularizado por Henry Ford. *Lean* tem o objetivo de gerar produção em larga escala de baixo custo, pois quebra o processo de chão de fábrica que era centralizado em pequenas partes. Assim é possível que o *Lean* seja desempenhado por um número maior de profissionais ainda que tais profissionais não tenham tanta habilidade. Para isso é utilizado um maquinário de alta precisão, gerando um trabalho padronizado. A produção em massa tem a característica de inflexibilidade, pois uma alteração na linha de produção pode ter um alto custo e o *Lean* vem para sanar isso, sendo mais econômico garantindo uma maior qualidade. (HIBBS et al., 2009).

De acordo com BRASIL (2010):

> "... *Lean* é uma estratégia de negócios para aumentar a satisfação dos clientes através da melhor utilização dos recursos. A gestão *Lean* procura fornecer consistentemente valor aos clientes com os custos mais baixos (Propósito) através da identificação de melhoria dos fluxos de valor primários e de suporte (Processos) por meio do envolvimento das pessoas qualificadas, motivadas e com iniciativa (Pessoas). O foco da implementação deve estar nas reais necessidades dos negócios e não na simples aplicação das ferramentas *Lean* ..." (BRASIL, 2010)

Nos anos 1990, a partir da publicação do livro "*Machine That Changed The World*", por Womack et al. (1991) o termo *Lean* ficou mais conhecido, então, uma nova maneira de pesar ficou conhecida, o "*Lean Thinking*" (WOMACK et al., 1991). Segundo Womack et al. (1991), o *Lean Thinking* é mapeado em 5 princípios:

1. Identificar valor para o cliente;
2. Identificar todas as etapas do fluxo de valor de cada produto, eliminado possíveis etapas que não agregam valor;

3. Criar um fluxo sequencial para cada produto;
4. Estabilizar e fazer com que o fluxo de valor seja puxado pelo cliente, a partir das suas reais necessidades; e
5. Garantir a contínua melhoria de todo o processo, a fim de atingir o estado da perfeição, sem nenhum tipo de desperdício.

Além dos princípios apresentados anteriormente, Poppendieck e Poppendieck (2003) destacam os seguintes princípios para o ambiente de desenvolvimento de *software*:

- Desenvolvimento de um componente que não está sendo utilizado, trata-se de um desperdício;
- Requisito coletado, documentado e não foi utilizado pela equipe e nem entregue ao cliente final, trata-se também de uma atividade desnecessária; e
- Desenvolvimento de rotinas de códigos que não estão e não serão executadas, é uma atividade que gerou desperdício para todo o ciclo de vida do software.

Com os conceitos de *Lean* e *Lean Thinking* surgiu o *Lean software development - LSD*, que é uma adaptação no *core* do *Lean* para o desenvolvimento de *software*.

2.4.1 *LEAN SOFTWARE DEVELOPMENT*

O *Lean software development* é uma adaptação do *Lean*, pois o desenvolvimento de *software* não é como o processo de fabricação. E, de fato, se *Lean* é considerado um conjunto de boas práticas pode-se ter a aplicação de conceitos básicos para desenvolvimento de produtos na engenharia de *software* podendo levar a melhoria de processo e qualidade (POPPENDIECK; POPPENDIECK, 2003).

Segundo Poppendieck e Cusumano (2012), existem sete princípios para o desenvolvimento de *software* utilizando *Lean*, que são:

- Eliminar o desperdício: Em termos simples, o "desperdício"é qualquer coisa que não adicione diretamente o valor do cliente ou adicione conhecimento sobre como entregar valor de forma mais eficaz. Algumas das maiores causas de desperdício no desenvolvimento de *software* são características desnecessárias, perda de conhecimento, trabalho parcialmente realizado e multi tarefas (POPPENDIECK; CUSUMANO, 2012).

- Integrar qualidade: O grande objetivo é construir um *Software* com qualidade desde o inicio do processo até o fim, para isso a organização da equipe é algo essencial e recomenda-se para que cada funcionalidade implementada haja uma inspeção do produto, para que assim o defeito seja encontrado logo após sua criação (POPPENDIECK; POPPENDIECK, 2003).

- Criar conhecimento: No final, o desenvolvimento consiste em criar conhecimento e incorporar esse conhecimento ao produto.

  Com relação à incorporação de conhecimento, o *Lean* recomenda abordar isso de duas maneiras diferentes, dependendo do contexto (POPPENDIECK; CUSUMANO, 2012).

  A primeiro é explorar múltiplas opções para decisões de preço, como arquitetura fundamental, escolha de linguagem, linguagem de *design* para interação do usuário, e assim por diante. Atrasar as decisões críticas para o último momento responsável e, em seguida, tomar decisões com base no melhor conhecimento disponível. Como várias opções foram exploradas, sempre haverá uma alternativa que funcionará, e a opção que melhor otimiza o sistema geral pode ser escolhida. Isso geralmente é chamado de uma abordagem "aprender primeiro"(POPPENDIECK; CUSUMANO, 2012).

  A segunda maneira é construir um conjunto mínimo de recursos para começar, seguido de entrega frequente, ao usar o *feedback* da experiência real ao cliente para tomar decisões para produto. Este processo de aprendizagem contínua minimizará o esforço gasto no desenvolvimento de recursos que os clientes não valem (POPPENDIECK; CUSUMANO, 2012).

- Entregar rápido: Em muitos ambientes de desenvolvimento *Lean*, os lançamentos para ocorrem de uma forma contínua, mensalmente, semanalmente, diariamente, etc. Quando se fala em entrega continua, implica-se que sejam encontrados mais problemas em clientes, implicando em uma melhoria no modelo de desenvolvimento assim aumentando a qualidade (POPPENDIECK; CUSUMANO, 2012).

- Respeitar as pessoas: Esse principio concerne que não deve-se impedir uma pessoa que não faça um trabalha só porque outra pessoa acha que não é o melhor meio de fazer. Todos os membros da equipe devem se respeitar e trabalharem juntos para no final chegarem melhor solução, além disso todos os membro vão melhorando com o tempo, devido a melhoria contínua (POPPENDIECK; CUSUMANO, 2012).

- Otimizar o todo: O *Lean* prega que deve-se compreender e descobrir quais a as principais preocupações dos clientes, assim podendo desenvolver um *software* com o real valor que os clientes necessitam, otimizando o seu trabalho de forma que não fique um processo engessado, mas sim um processo autónomo, da mesma forma que não houvesse *software*, esse é a característica ligada ao valor do produto que está ligado com qualidade, tempo de esforço e capacidade de modificação (POPPENDIECK; CUSUMANO, 2012).

- Otimização constante: O *Lean* prega que independente do processo utilizado para o desenvolvimento, talvez até uma combinação entre eles, o processo de desenvolvimento deve estar em constante otimização (melhoria contínua), para que a cada entrega o valor do produto seja maior (POPPENDIECK; CUSUMANO, 2012).

2.5 MELHORIA CONTÍNUA

Melhoria contínua pode ser definida como um conjunto de boas praticas que melhoram a performance de um processo. Melhoria contínua é a criação de uma cultura sustentável de que todos os integrantes da organização eliminem todos os resquícios de desperdício dos processos de desenvolvimento (JHA et al., 1996). O objetivo é criar uma cultura de melhoria sustentável através do envolvimento de todos os participantes da organização na eliminação dos resíduos dentro de sistemas e processos organizacionais (CAFFYN, 1997; BHUIYAN et al., 2006).

Para a aplicação da melhoria contínua, vários programas foram utilizados como TQM, *Lean* e *Six Sigma*. Os programas de melhoria têm objetivos, ferramentas e métodos específicos. O *Six Sigma* concentra-se na redução da variabilidade, o *Lean* tem como objetivo melhorar os processos e o TQM aumenta o nível de satisfação dos clientes. *Lean-Sigma* combina filosofia *Lean* e ferramentas *Six Sigma*, visando uma rápida redução de resíduos e variações fornecendo valor ao cliente (HELLSTEN; KLEFSJÖ, 2000; ANDERSSON et al., 2006; BHUIYAN et al., 2006; BHUIYAN; BAGHEL, 2005).

Os programas são desenvolvidos por meio de atividades da melhoria contínua, que são ações, projetos e iniciativas que permitem implementar os objetivos e princípios dos programas (BHUIYAN; BAGHEL, 2005; SINGH; SINGH, 2015). As atividades seguem métodos como o PDCA e o DMAIC, que visam sistematizar a identificação e mensuração de problemas, identificar as causas, propor planos de ação, analisar e medir os resultados

gerados e padronizar as ações realizadas (SAVOLAINEN; HAIKONEN, 2007; NÄSLUND, 2008; SINGH; SINGH, 2015). Os métodos utilizam ferramentas para que as decisões sejam tomadas com base em informações, fatos e dados (ANDERSSON et al., 2006; BHUIYAN; BAGHEL, 2005).

Mesmo as empresas com resultados bem satisfatórios nos seus processo de produção devem melhorar continuamente seu desempenho (LAGER, 2000). Com base nisso é necessário que os programas de melhoria contínua sejam expandidos para incluir iniciativas que melhorem o desempenho do processo de desenvolvimento de produtos (PDP) (SPIVEY et al., 1997; NILSSON-WITELL et al., 2005; SUN et al., 2009; YAN; MAKINDE, 2009).

2.5.1 MELHORIA CONTÍNUA NO PROCESSO DE DESENVOLVIMENTO DE PRODUTOS

O primeiro autor a utilizar a expressão melhoria em processo de *software*, foi Humphrey (1989), estabelecendo alguns processos para o desenvolvimento de *software* ser mais adaptável aos requisitos, assim garantindo uma maior qualidade na entrega do produto.

O modelo proposto por Humphrey (1989) tinha como objetivo atingir a melhoria contínua com base na mensuração e controle dos processos. Esse *framework* foi o início de tudo conhecido como melhoria contínua dentro do setor de *software*, como por exemplo temos o CMM/CMMI (SEI, 2010) e o *Software Process Improvement and Capability Determination* (SPICE) que deu origem à ISO/IEC 15504 (ISO/IEC, 2003), que foram criados com base no modelo proposto por Humphrey (1989) e assim se segui os modelos de melhoria contínua em constante evolução.

A aplicação da melhoria contínua dentro de empresas que realizam o desenvolvimento de produtos pode proporcionar uma melhor a qualidade. A melhora em questões de tempo de desenvolvimento e o custo do produto alinhado com essas características, reduzindo os problemas, o desperdício e o retrabalho (SUN et al., 2009; HARTER et al., 2000).

Ferramentas e métodos, como PDCA podem ser utilizados para reduzir o tempo do desenvolvimento do produto. Contudo não é possível saber quais resultados da melhoria vão ser obtidos com o uso e quais iniciativas melhoria contínua podem ser usadas de forma mais incisiva no processo de desenvolvimento de um produto (GRIFFIN, 1993; LODGAARD; AASLAND, 2011).

O melhor processo de desenvolvimento de um produto é somente obtido quando se tem uma equipe focada na resolução de problemas e melhoria contínua (KOWANG; RASLI, 2011). Adicionalmente, é necessário um processo que esteja muito bem definido e descrito para que seja aplicado no desenvolvimento de forma efetiva (LAGER, 2000) e uma padronização dos processos para um subsequente otimização (CHAUDHURI, 2013).

Contudo quando o processo de aplicação da melhoria contínua é efetivamente aplicado em um processo de produção, podem existir vários pontos que podem impactar na melhora, tais como: (1) a complexidade das atividades, (2) volume de informações envolvido nas pessoas que compõe o processo e a (3) resistência a padronização de procedimentos (RINGEN; HOLTSKOG, 2013). Também é sabido que para cada projeto é concebido um processo novo e único, relacionado a seus objetivos singularidades e especificidades (RINGEN; HOLTSKOG, 2013), levando a acarretar em um dificuldade de padronização e melhoria no processo. De acordo com Ringen e Holtskog (2013), a padronização das atividades de forma excessiva gerada pela melhoria contínua pode dificultar a alteração dos processos, pois um processo deveria ser algo que não tivesse uma rotina especifica, não fosse algo instável e nem experimental.

3 APLICAÇÃO DA MELHORIA CONTÍNUA: UM ESTUDO DE CASO

3.1 CENÁRIO ATUAL

Antes da aplicação do estudo, as equipes não possuíam uma metodologia estruturada que incentivasse o registro dos fatos ocorridos durante as sprints, de forma que não era possível a aplicação de conceitos de melhoria contínua. Também tinham a percepção de que haviam dificuldades durante as sprints, mas não conseguiam identificar de forma clara onde os problemas que ocorriam.

Os fatos que ocorriam durante as sprints eram somente comentados durante as cerimônias de retrospectiva da sprint e o próprio time acaba esquecendo destes fatos com o passar do tempo. Em razão de não terem uma metodologia estruturada alguns problemas já evidenciados, voltavam a acontecer, isso causando um desperdício no processo de desenvolvimento.

3.2 ESTUDO DE CASO

A empresa cujo estudo de caso foi conduzido conta com aproximadamente 400 funcionários. Suas principais soluções estão focadas em software para gestão de supermercados e materiais de construção, mas existem outras soluções que a empresa também desenvolve, como soluções para restaurantes. Como a empresa já está consolidada no mercado há mais de 25 anos, possui abrangência em todo o território nacional, contabilizando aproximadamente 4.000 lojas que utilizam suas soluções e mais de 100.000 usuários.

A empresa está dividida nos seguintes setores: fábrica de software, suporte, implantação, administrativo, comercial, pós-vendas, marketing, talentos humanos e universidade corporativa.

O estudo de caso foi estruturado em várias atividades que podem ser visualizadas na Figura 2 a seguir. Cada etapa do processo será explicada de forma detalhada nas

próximas sessões.

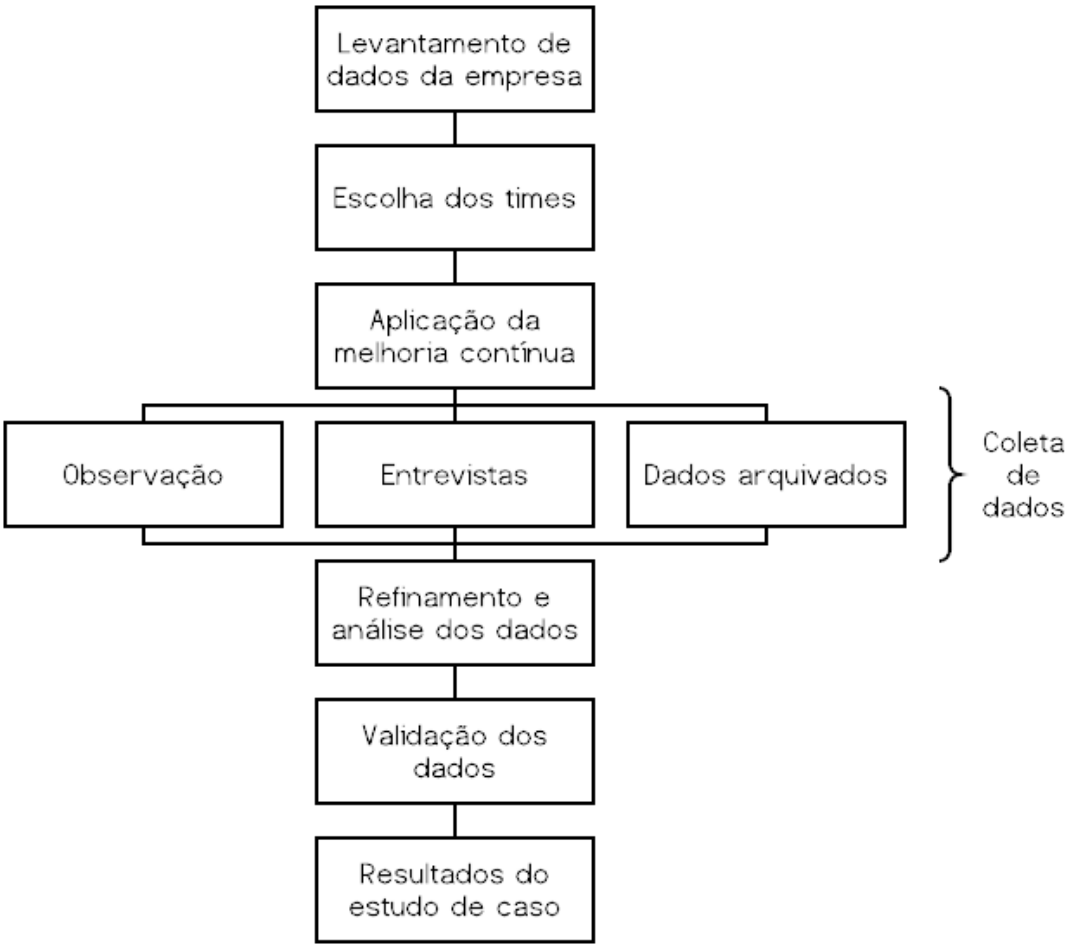


Figura 2: Diagrama condução das atividades
Fonte: Autoria própria

O estudo de caso foi realizado junto à fábrica de software. Tal departamento é dividido em dois subdepartamentos: inovação e manutenção. Ambos utilizam como metodologia de desenvolvimento o framework Scrum. Optou-se por escolher os subdepartamentos de inovação e manutenção por se tratarem de equipes de tamanho equivalente e demandas regulares de projetos, facilitando assim a padronização na coleta e análise dos dados da pesquisa.

3.3 ESCOLHA DOS TIMES

A partir da escolha do departamento a ser aplicado o estudo de caso, foram selecionados sete times da fábrica para a pesquisa, sendo dois times de manutenção e cinco de inovação.

Os times do departamento de manutenção trabalham com desenvolvimento e manutenção de softwares legados e o time de inovação trabalha com o desenvolvimento de novos produtos.

Quatro times trabalham no desenvolvimento de uma plataforma web com linguagem Java, outra trabalha no desenvolvimento de um software embarcado com linguagem C++ e dois times trabalham em manutenção de produtos com linguagem Power Builder.

Em um estudo prévio para levantamento da caracterização dos times foram identificadas as seguintes composições:

- Times de Manutenção: Os times de manutenção são responsáveis por manter e criar novas funcionalidades requisitadas por clientes em softwares legado. Os times possuem a seguinte configuração, demonstrada na Tabela 1:

Cargo	**Quantidade**
Analista de requisitos	1
Desenvolvedor	3 - 4
Líder técnico	1
Testador	2 - 4

Tabela 1: Estrutura dos times de manutenção

Fonte: Autoria própria

- Times de Desenvolvimento Os times de desenvolvimento têm como responsabilidade analisar e desenvolver softwares e soluções inovadoras, com o objetivo de manter a empresa alinhada em relação aos seus concorrentes e tendências tecnologias. Sua estrutura pode ser visualizada na Tabela 2:

Cargo	Quantidade
Analista de requisitos	1
Desenvolvedor	4 - 5
Líder técnico	1
Testador	1 - 2

Tabela 2: Estrutura dos times de desenvolvimento
Fonte: Autoria própria

3.4 PROCEDIMENTO DE COLETA DE DADOS

Os dados foram coletados durante quatro (4) Sprints, cada uma com duração de duas semanas, totalizando um período de coleta de dados de oito (8) semanas. Para a coleta dos dados foram utilizadas três fontes de informação, também conhecida como triangulação de dados, que podem ser melhor visualizados na Figura 3:

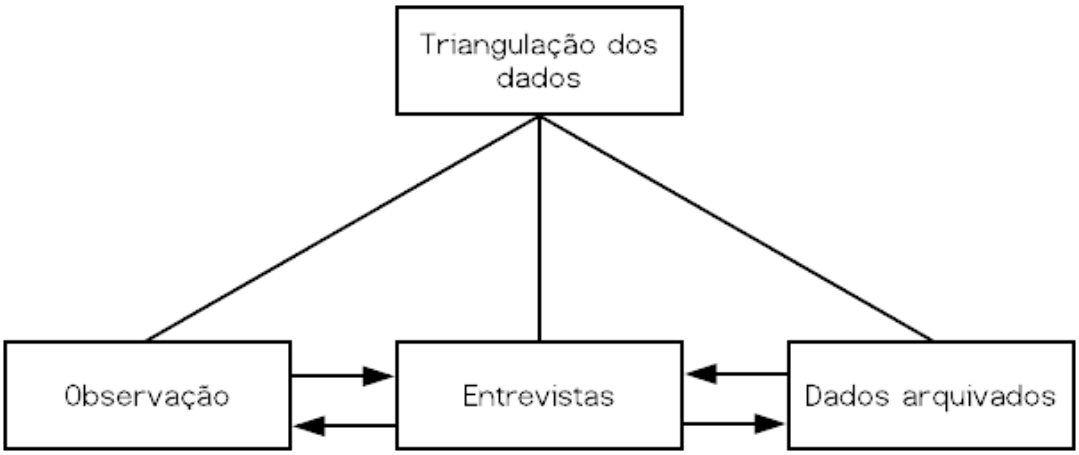


Figura 3: Triangulação dos dados coletados
Fonte: Adaptado de: Godoi et al. (2006)

A primeira etapa da coleta de dados foi realizada pelas observações em seguida pelos dados arquivados e entrevista. A triangulação foi realizada com intuito de analisar se os dados vindos das três (3) fontes estão de acordo. Cada uma das fontes de dados será explicada com maiores detalhes nas seguintes sessões.

3.4.1 OBSERVAÇÃO

A observação dos times ocorreu de duas maneiras: direta e indireta.

1. Direta: Participando da cerimônia de retrospectiva das Sprints; e
2. Indireta: Foram realizadas observações no processo seguido pelo time no dia a dia.

3.4.2 ENTREVISTA

Foram realizadas entrevistas com sete (7) líderes técnicos e um (1) gerente de projeto, com objetivo de descobrir qual era o cenário que estavam enfrentando no momento, e validar se as técnicas de melhoria contínua estavam auxiliando no processo de desenvolvimento.

Uma parte da entrevista foi realizada de maneira estruturada, para posterior extensão dela para todos os times, as questões realizadas de maneira estruturada foram as seguintes:

1. Foi difícil de realizar a aplicação de melhoria contínua?
2. A aplicação da melhoria contínua interferiu no processo de trabalho do time?
3. A melhoria contínua é um processo demorado?
4. A melhoria contínua trouxe um resultado positivo ou negativo?

A outra parte da entrevista foi realizada de maneira semiestruturada, em uma breve conversa sobre qual era o status atual do time em relação a aplicação da melhoria contínua, dificuldades enfrentadas pelos times e quando disponíveis os dados coletados eram discutidos.

3.4.3 DADOS ARQUIVADOS

Os dados arquivados foram gerados a partir de informações sobre os fatos ocorridos durante a Sprint. Estas poderiam ser obtidas de duas maneiras: através de um formulário disponibilizado para as equipes registrarem um fato a qualquer momento da Sprint e uma planilha cuja utilização foi indicada somente na reunião de retrospectiva da Sprint.

O formulário disponibilizado foi criado com o objetivo de facilitar e agilizar o processo de coleta dos dados, sendo necessário somente duas informações, a primeira era a descrição do fato e a segunda era o seu tipo, que poderiam ser classificados de três (3) formas diferentes: pontos positivos (que bom), pontos negativos (que pena) e pontos a melhorar (que tal).

Pode-se visualizar o modelo do formulário utilizado conforme a Figura 4:

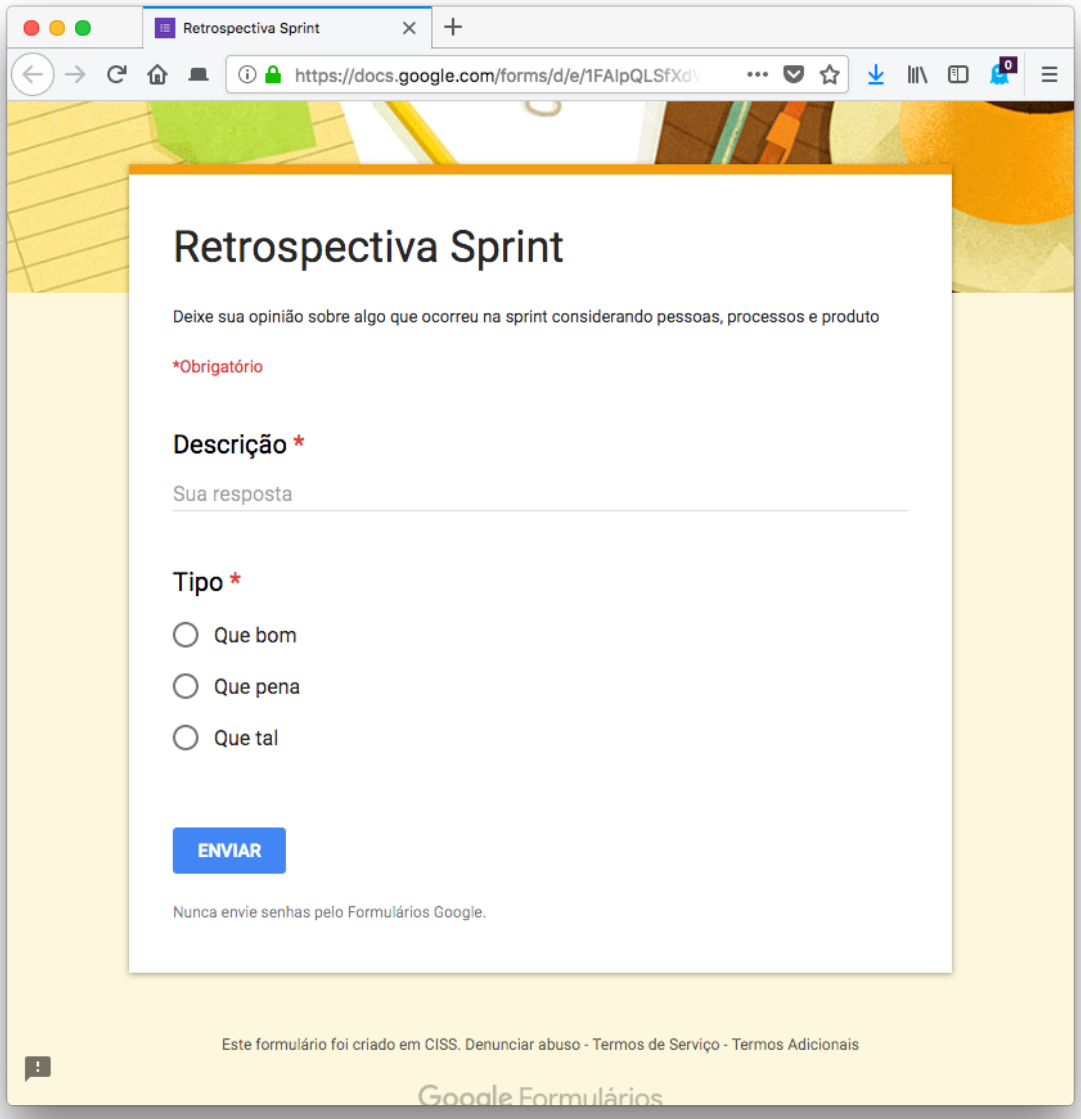


Figura 4: Formulário para registro de fatos
Fonte: Autoria própria

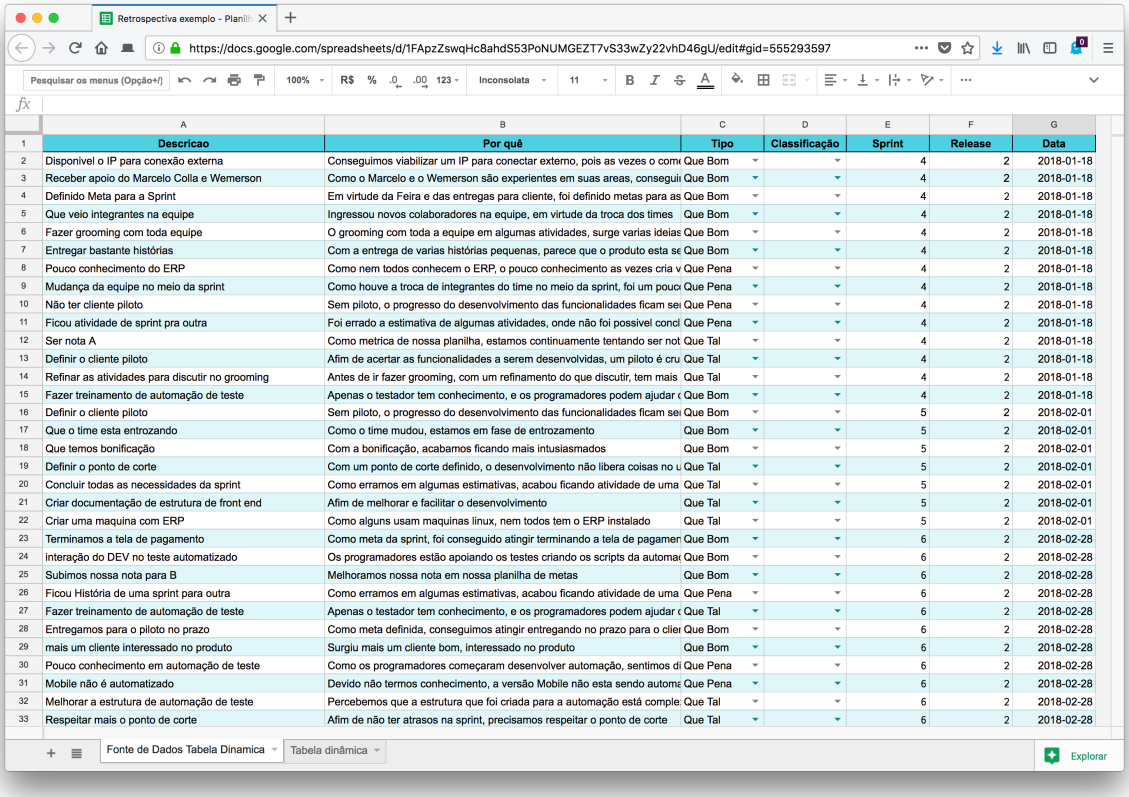
Os fatos registrados através do formulário eram transpostos para a planilha na

cerimônia de retrospectiva da Sprint e em seguida categorizados (quando possível).

A planilha foi estruturada já pensando na categorização dos dados coletados, para isso eram necessários mais alguns campos, sendo esses:

- Descrição: descrição do fato ocorrido durante a Sprint (vindo dos resultados do formulário);
- Por que: qual o motivo de tal fato ter ocorrido, qual sua origem. Informação necessário para a classificação;
- Tipo: qual era o tipo do fato ocorrido (vindo dos resultados do formulário);
- Classificação: essa informação é gerada a partir da análise dos dados, cuja será explicada na seção seguinte;
- Sprint: qual a Sprint de trabalho do time para saber o contexto de desenvolvimento do time;
- Release: qual a release de do time trabalho para saber o contexto de desenvolvimento do time;
- Data: data em que foi realizada a cerimônia de retrospectiva.

A estrutura da planilha pode ser visualizada na Figura 5, a seguir:



	A	B	C	D	E	F	G
1	Descricao	Por quê	Tipo	Classificação	Sprint	Release	Data
2	Disponivel o IP para conexão externa	Conseguimos viabilizar um IP para conectar externo, pois as vezes o com	Que Bom		4	2	2018-01-18
3	Receber apoio do Marcelo Colla e Wemerson	Como o Marcelo e o Wemerson são experientes em suas areas, conseguii	Que Bom		4	2	2018-01-18
4	Definido Meta para a Sprint	Em virtude da Feira e das entregas para cliente, foi definido metas para as	Que Bom		4	2	2018-01-18
5	Que veio integrantes na equipe	Ingressou novos colaboradores na equipe, em virtude da troca dos times	Que Bom		4	2	2018-01-18
6	Fazer grooming com toda equipe	O grooming com toda a equipe em algumas atividades, surge varias ideias	Que Bom		4	2	2018-01-18
7	Entregar bastante histórias	Com a entrega de varias histórias pequenas, parece que o produto esta se	Que Bom		4	2	2018-01-18
8	Pouco conhecimento do ERP	Como nem todos conhecem o ERP, o pouco conhecimento as vezes cria v	Que Pena		4	2	2018-01-18
9	Mudança da equipe no meio da sprint	Como houve a troca de integrantes do time no meio da sprint, foi um pouc	Que Pena		4	2	2018-01-18
10	Não ter cliente piloto	Sem piloto, o progresso do desenvolvimento das funcionalidades ficam se	Que Pena		4	2	2018-01-18
11	Ficou atividade de sprint pra outra	Foi errado a estimativa de algumas atividades, onde não foi possivel concl	Que Pena		4	2	2018-01-18
12	Ser nota A	Como metrica de nossa planilha, estamos continuamente tentando ser not	Que Tal		4	2	2018-01-18
13	Definir o cliente piloto	Afim de acertar as funcionalidades a serem desenvolvidas, um piloto é cru	Que Tal		4	2	2018-01-18
14	Refinar as atividades para discutir no grooming	Antes de ir fazer grooming, com um refinamento do que discutir, tem mais	Que Tal		4	2	2018-01-18
15	Fazer treinamento de automação de teste	Apenas o testador tem conhecimento, e os programadores podem ajudar	Que Bom		4	2	2018-01-18
16	Definir o cliente piloto	Sem piloto, o progresso do desenvolvimento das funcionalidades ficam se	Que Bom		5	2	2018-02-01
17	Que o time esta entrozando	Como o time mudou, estamos em fase de entrozamento	Que Bom		5	2	2018-02-01
18	Que temos bonificação	Com a bonificação, acabamos ficando mais intusiasmados	Que Bom		5	2	2018-02-01
19	Definir o ponto de corte	Com um ponto de corte definido, o desenvolvimento não libera coisas no u	Que Tal		5	2	2018-02-01
20	Concluir todas as necessidades da sprint	Como erramos em algumas estimativas, acabou ficando atividade de uma	Que Tal		5	2	2018-02-01
21	Criar documentação de estrutura de front end	Afim de melhorar e facilitar o desenvolvimento	Que Tal		5	2	2018-02-01
22	Criar uma maquina com ERP	Como alguns usam maquinas linux, nem todos tem o ERP instalado	Que Tal		5	2	2018-02-01
23	Terminamos a tela de pagamento	Como meta da sprint, foi conseguido atingir terminando a tela de pagamen	Que Bom		6	2	2018-02-28
24	interação do DEV no teste automatizado	Os programadores estão apoiando os testes criando os scripts da automa	Que Bom		6	2	2018-02-28
25	Subimos nossa nota para B	Melhoramos nossa nota em nossa planilha de metas	Que Bom		6	2	2018-02-28
26	Ficou História de uma sprint para outra	Como erramos em algumas estimativas, acabou ficando atividade de uma	Que Pena		6	2	2018-02-28
27	Fazer treinamento de automação de teste	Apenas o testador tem conhecimento, e os programadores podem ajudar	Que Tal		6	2	2018-02-28
28	Entregamos para o piloto no prazo	Como meta definida, conseguimos atingir entregando no prazo para o clie	Que Bom		6	2	2018-02-28
29	mais um cliente interessado no produto	Surgiu mais um cliente bom, interessado no produto	Que Bom		6	2	2018-02-28
30	Pouco conhecimento em automação de teste	Como os programadores começaram desenvolver automação, sentimos di	Que Pena		6	2	2018-02-28
31	Mobile não é automatizado	Devido não termos conhecimento, a versão Mobile não esta sendo automa	Que Pena		6	2	2018-02-28
32	Melhorar a estrutura de automação de teste	Percebemos que a estrutura que foi criada para a automação está comple	Que Tal		6	2	2018-02-28
33	Respeitar mais o ponto de corte	Afim de não ter atrasos na sprint, precisamos respeitar o ponto de corte	Que Tal		6	2	2018-02-28

Figura 5: Planilha de dados

Fonte: Autoria própria

Por meio dessa coleta de dados foi possível gerar insumos para prosseguir com o refinamento e análise dos dados, abordados na próxima sessão.

3.5 PROCEDIMENTO DE REFINAMENTO E ANÁLISE DOS DADOS

O procedimento de refinamento e análise dos dados consiste em uma análise meticulosa dos dados obtidos pela etapa de coleta. A partir dos dados vindos do formulário e da planilha, foram realizadas as classificações dos dados obtidos em categorias - a classificação dos dados foi realizada em conjunto com um membro da equipe para que seja mais assertiva e permita o entendimento do real contexto enfrentado pelo time - cujas podem ser visualizadas a seguir, bem como sua origem:

- Ambiente: tudo que estivesse relacionado ao ambiente de desenvolvimento, hardware e T.I;

- Gestão de Conhecimento: tudo que estivesse relacionado ao conhecimento necessário para a realização das atividades e também ao compartilhamento de conhecimento;
- Planejamento: tudo que estivesse relacionado ao planejamento, entrega e meta da Sprint;
- Comportamento: tudo que estivesse relacionado ao comportamento dos membros das equipes;
- Incentivo Organizacional: tudo que estivesse relacionado ao incentivo em que os membros das equipes estavam sentindo para serem mais comprometido com o produto;
- Multidisciplinaridade: todas as ações que demonstrarem multidisciplinaridade entre as atividades desenvolvidas;
- Qualidade: tudo que estivesse relacionado a qualidade do produto entregue;
- Processo: tudo que estivesse relacionado com o processo seguido pelo time;
- Experiências Passadas: tudo que os membros viessem apresentar em discussões técnicas e de negócio relacionadas a experiências vindas de outro ambiente; e
- Rotatividade: tudo que estivesse relacionado a rotatividade dos membros entre os time.

Após a categorização de todos os dados obtidos, os mesmo foram agrupados em uma tabela contendo o seu tipo e categoria, assim podendo visualizar de forma sumarizada quais as categorias com maior incidência em cada time. Para melhor entendimento de como foi realizada pode-se visualizar a Tabela 3.

Categorias	Que Bom	Que Pena	Que Tal
Ambiente	0	0	0
Gestão de Conhecimento	0	0	0
Planejamento	0	0	0
Comportamento	0	0	0
Incentivo Organizacional	0	0	0
Multidisciplinaridade	0	0	0
Qualidade	0	0	0
Processo	0	0	0
Experiências Passadas	0	0	0
Rotatividade	0	0	0

Tabela 3: Exemplo de dados sumarizados

Com a formulação da tabela foi mais fácil de visualizar o cenário do time. A partir dele foram criadas ações para as categorias que estivessem com maior ocorrência.

As ações para cada tipo tem resultados diferentes, cujas serão explicadas a seguir:

- Caso a categoria cujo tipo seja positivo (que bom), a ação é realizada para que a categoria continue assim;
- Caso a categoria cujo tipo seja negativo (que pena), a ação realizada é realizada para aliviar as dificuldades; e
- Caso a categoria cujo tipo seja a melhorar (que tal), o ação é realizada para que aquele ponto a melhorar realmente aconteça.

Após a criação de todas as ações os responsáveis a repassar cada ação ao time de desenvolvimento são os líderes técnicos, fazendo assim com que o time entenda o porquê de cada ação e colocando em prática cada uma delas. Após a execução das ações foi realizado o procedimento de validação, cujo será melhor explicado na seguinte seção.

3.6 PROCEDIMENTO DE VALIDAÇÃO DOS DADOS

O procedimento de validação dos dados consiste em verificar quais foram os resultados obtidos pelas ações criadas anteriormente. Para a validação dos dados foram

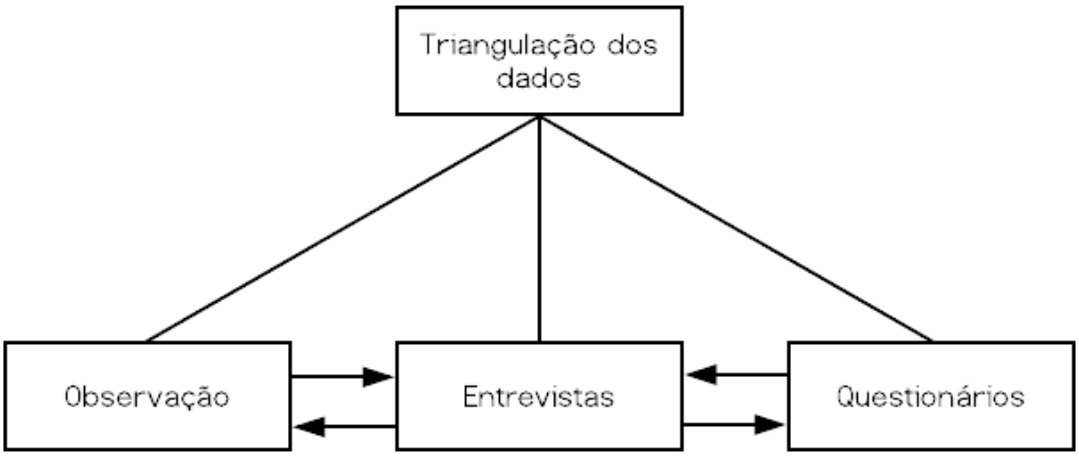


Figura 6: Planilha de dados
Fonte: Adaptado de: Godoi et al. (2006)

utilizadas três fontes, também conhecida como triangulação de dados, que podem ser melhor visualizadas pela Figura 6.

A triangulação dos dados foi realizada da seguinte maneira:

- Observações: ocorreram de maneira indireta, observando o processo de trabalho dos times;
- Entrevistas: foram realizadas entrevistas semi estruturadas com os líderes técnicos e gestores a fim de verificar se as mudanças causaram algum efeito nas equipes;
- Questionário: foi aplicado um questionário para todos os membros dos times de desenvolvimento, com intuito de avaliar qual o efeito da aplicação da melhoria contínua.

Utilizando todo o processo detalhado nesta sessão foi desenvolvido o estudo de caso e se obteve os resultados que serão demonstrados na próxima sessão.

4 RESULTADOS

Esta sessão apresenta os resultados obtidos por meio da coleta de dados durante o estudo de caso. Para facilitar a análise dos resultados, os dados foram agrupados em duas categorias sendo times de manutenção e times de desenvolvimento como citado na sessão três (3), e ao final, agrupadas em um único resultado para uma analise geral de resultados.

4.1 TIMES DE DESENVOLVIMENTO

4.1.1 TIME 1

O time um (1) trabalha com o desenvolvimento do projeto P, utilizando tecnologias como Qt, C++ e python. O projeto P é um sistema de ponto de vendas (PDV), a característica diferencial do projeto P é seu desenvolvimento embarcado em um hardware específico.

4.1.1.1 APRESENTAÇÃO DAS INFORMAÇÕES OBTIDAS

Com o intuito de apresentar e classificar as informações obtidas durante o período de coleta de dados do time um (1), foi desenvolvida a Tabela 4.

Categorias	Que Bom	Que Pena	Que Tal
Ambiente	0	0	4
Gestão de Conhecimento	1	1	0
Planejamento	6	9	8
Comportamento	1	0	0
Incentivo Organizacional	0	0	0
Multidisciplinaridade	0	0	0
Qualidade	6	4	11
Processo	0	1	2
Experiências Passadas	0	0	0
Rotatividade	0	0	0

Tabela 4: Tabela dos dados coletados do time 1

Fonte: Autoria própria

Para uma efetiva análise dos dados acima apresentados, é conveniente a apresentação da Figura 7.

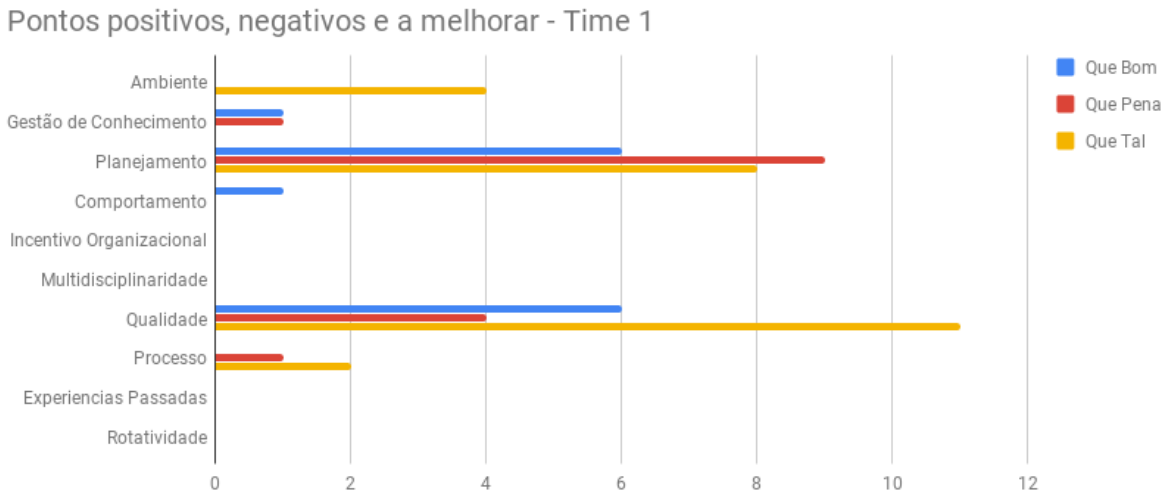


Figura 7: Gráfico de dados do time 1

Fonte: Autoria própria

4.1.1.2 ANÁLISE DAS INFORMAÇÕES OBTIDAS

Analisando as informações apresentaras pela Figura 7, é possível constatar que existe uma grande incidência de pontos negativos em uma (1) das categorias

classificadas, sendo ela a a categoria de planejamento, porem outro categorias também apresentaram pontos negativos. Três (3) das categorias apresentaram pontos a melhorar que chamaram atenção também sendo elas: ambiente, qualidade e planejamento (possivelmente relacionado a quantidade de pontos negativos).

Também é possível observar que o time tem quatro (4) categorias pontuadas como positivas, sendo elas gestão de conhecimento, planejamento (possivelmente relacionado aos pontos a melhorar e negativos já apresentados para a categoria), comportamento e qualidade.

Para um maior entendimento do cenário atual das categorias foi necessária uma analise mais detalhada de cada categoria, sendo esse vista nos próximos parágrafos.

4.1.1.2.1 PONTOS POSITIVOS: Após uma verificação minuciosa dos dados obtidos, foi possível identificar que a origem dos pontos positivos relacionados a categoria de gestão conhecimento esta ligada ao grande compartilhamento de conhecimento entre os membros do time, os pontos positivos na categoria de comportamento estavam relacionados ao grande comprometimento do time com o produto desenvolvido e os pontos pontos positivos relacionados as categorias de planejamento e qualidade tem ligação com os pontos a melhorar.

4.1.1.2.2 PONTOS NEGATIVOS: Devido a verificação de uma forte incidência de pontos negativos no planejamento do time, fez se necessária a análise pontual das informações para empregar soluções frente a alguns problemas evidenciados no planejamento.

Após a análise dos dados de maneira pontual, obteve-se a conclusão, de que a dificuldade do time se encontra em seu planejamento, por encontrar dificuldades em: entrega da meta da sprint, mal planejamento das histórias e requisitos e estimativas não assertivas.

4.1.1.2.3 PONTOS A MELHORAR: Além dos pontos negativos foram analisados os pontos que o time estava sugerindo como pontos a melhorar, pois ainda não eram satisfatórios nas categorias de processo, qualidade e ambiente.

Na categoria de processo foi pode-se perceber que o time não estava realizando a cerimônia do grooming corretamente. No quesito qualidade o time 1 era muito rigoroso,

eles possuíam uma qualidade boa, porém, sempre queriam melhorar, já na questão do ambiente o time buscava uma melhora do DevOps.

4.1.1.3 AÇÕES EXECUTADAS

Para cada categoria analisada foi definida uma ação que o time de desenvolvimento executou para alavancar seus resultados, pode se observar na Tabela 5 as ações tomadas para cada categoria, bem como as motivações e os seus resultados:

Categoria	**Ações**	**Motivo**	**Resultado**
Planejamento	Definir uma meta única por sprint	O time estava se comprometendo a entregar muitas demandas por sprint, por isso não conseguiam entregar alguns recursos de maneira completa.	Definição de uma meta única por sprint para que o time tenha uma visão de trabalho no decorrer toda aquela iteração.
Planejamento	Criar um DOR (Definition of ready)	Algumas histórias ingressaram na sprint sem ter uma definição de requisitos, e em alguns casos mais complexos também era necessária uma arquitetura de desenvolvimento das histórias.	Para que a história possa ser incluída na demanda da sprint é necessário que a mesma tenha uma definição dos requisitos bem elaborada e para atividades mais complexas a arquitetura de software.
Continua na próxima pagina			

Tabela 5 – continuação da pagina anterior

Categoria	Ações	Motivo	Resultado
Planejamento	Grooming mais técnico	As estimativas das histórias não estavam muito assertivas, pois quando o desenvolvedor iniciava aquela atividade, muitas vezes era uma atividade muito complexa.	Para melhor estimativa das atividades foi definido que todas as atividades mais complexas devem ter um grooming realizado de forma mais técnica e não somente a nível de negócio, assim melhorando a assertividade da estimativa.
Planejamento	Roadmap unificado	As histórias planejadas pelo time de desenvolvimento não coincidem com o planejamento de história da parte comercial da empresa. Causando dessa forma mudanças drásticas no roadmap do produto.	Para que todos os setores da empresa tivessem uma mesma visão do roadmap do produto foi criado um roadmap unificado no qual todas as partes utilizem e alteram o mesmo.

Continua na próxima pagina

Tabela 5 – continuação da pagina anterior

Categoria	Ações	Motivo	Resultado
Processo	Definir uma data específica para realização do grooming	Em muitos casos o grooming não era realizado, pois o time de desenvolvimento não havia reservado tempo para a cerimônia, e se realizado acabava resultando no atraso da sprint.	Para que o grooming possa ser realizado de forma que não interfira no andamento da sprint, foi definido uma data fixa (segundo dia da segunda semana da sprint) para a realização da mesma.
Qualidade	Criar um DOD (Definition of done)	Eram encontrados alguns bugs no ambiente de produção e algumas histórias nem sempre estavam conforme os requisitos.	Para garantir a qualidade das histórias em produção e o que foi solicitado fosse correspondente com o desenvolvido, foi sugerido que o time realizasse a verificação do DOD na cerimônia de encerramento, verificando os itens: Testes unitários, testes automatizados, build sem falhas, critérios de aceite e aceite do P.O.

Continua na próxima pagina

Tabela 5 – continuação da pagina anterior

Categoria	Ações	Motivo	Resultado
Qualidade	Resolução de uma dívida técnica por sprint	Como o time estava entregando um produto já e deveria realizar as histórias solicitadas conforme a necessidade dos clientes, não restava muito tempo para a resolução das dívidas técnicas.	Em toda a sprint pelo menos uma dividida técnica deve ser resolvida pelo time de desenvolvimento.
Qualidade	Ponto de corte de desenvolvimento	Muitas vezes os desenvolvedores estavam produzindo novas funcionalidades até o final da sprint o que causava um gargalo nas atividades de teste.	Para que os testadores possam testar todas as funcionalidades desenvolvidas foi estabelecido um ponto de corte, esse localizado no terceiro dia da segunda semana onde os desenvolvedores não criam mais nenhuma funcionalidade, somente há a correção de bugs e dívidas técnicas.

Continua na próxima pagina

Tabela 5 – continuação da pagina anterior

Categoria	Ações	Motivo	Resultado
Ambiente	Estudar e aplicar DevOps no projeto	Para o ambiente de desenvolvimento e deploy do produto eram necessárias muitas configurações que causavam uma lentidão no processo.	Foram estudadas algumas técnicas de DevOps para auxiliar na automatização e facilitação de algumas atividades relacionadas ao deploy e gerenciamento de branchs de desenvolvimento.

Tabela 5: Tabela de Ações, Motivações e resultados do time 1

4.1.1.4 RESULTADO DO TIME

Após a aplicação de todas as ações propostas, foi realizada uma entrevista com o time de desenvolvimento, na qual os mesmos relataram que houve uma grande melhora em todas as categorias acima evidenciadas.

As histórias planejadas estão sendo desenvolvidas com maior facilidade pelo time, pelo motivo de passarem pelo DOR antes de entrarem no processo de desenvolvimento. Além disso as histórias planejadas para um sprint são muito mais assertivas agora, não tendo que passar uma história de uma sprint para a outra através da realização de um grooming mais técnico, o que alavancou isso também foi a definição de um dia específico para a realização do mesmo. A equipe notou que fica muita mais orientada quando tem uma meta para cumprir ao final na sprint, que é a meta única, assim garantido o sucesso da entrega.

O sucesso de uma entrega também está relacionado com a qualidade entregue pelo time durante a sprint, e com isso remetemos as ações relacionadas à qualidade. Os testadores relatam que com a definição do ponto de corte não sentem mais aquela grande

pressão em saber se o que foi desenvolvido vai dar tempo de passar pela etapa do teste, pois após o ponto de corte alguns desenvolvedores auxiliam na etapa do teste já que não podem desenvolver novas funcionalidades, somente resolução de bugs e dívidas técnicas.

Após a entrega da sprint todo o processo de deploy também foi amenizado com a aplicação de um gerenciamento de branchs mais efetivo, e a geração automatizada do mesmo.

A aplicação da melhoria contínua auxiliou na resolução de muitos problemas e alavancou algumas mudanças, as quais o time gostaria de executar, portanto a melhoria contínua funcionou para o time um (1) e o mesmo vai continuar aplicando.

4.1.2 TIME 2

O time dois (2) está trabalhando no desenvolvimento do projeto G, cujo possui 5 membros, sendo 2 programadores, 1 líder técnico, 1 testador e 1 analista de negócio. Este produto vem sendo desenvolvido em Java, ext.js, html e javascript, utilizando a metodologia Scrum para a criação e organização. O projeto G está sendo desenvolvido para os clientes que utilizam o módulo de vendas ou qualquer outro software de força de vendas integrados ao sistema ERP da empresa C.

4.1.2.1 APRESENTAÇÃO DAS INFORMAÇÕES OBTIDAS

Com o intuito de apresentar e classificar as informações obtidas durante o período de coleta de dados do time dois (2), foi desenvolvida a Tabela 6.

Categorias	Que Bom	Que Pena	Que Tal
Ambiente	3	0	2
Gestão de Conhecimento	4	2	3
Planejamento	16	4	12
Comportamento	2	2	0
Incentivo Organizacional	2	0	1
Multidisciplinaridade	3	0	0
Qualidade	7	3	5
Processo	0	0	2
Experiências Passadas	0	0	0
Rotatividade	0	0	0

Tabela 6: Tabela dos dados coletados do time 2

Fonte: Autoria própria

Para uma efetiva análise dos dados acima apresentados, é conveniente a apresentação da Figura 8.

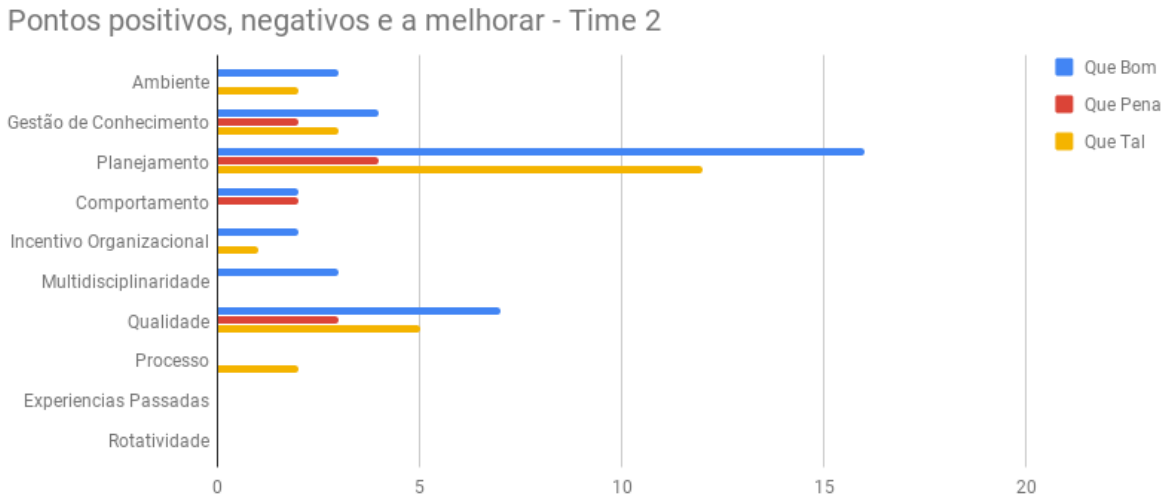


Figura 8: Gráfico de dados do time 2

Fonte: Autoria própria

4.1.2.2 ANÁLISE DAS INFORMAÇÕES OBTIDAS

A partir de uma análise macro é possível constatar em quase todas as categorias pontuadas pelo time uma pontuação com tendências a serem positivas. Porém alguns

pontos negativos estão presentes em quatro das categorias avaliadas pelo time, contudo nenhuma pontuação com grande incidência. Depois da análise os pontos positivos e negativos ainda existem, e os pontos a melhorar que estão presentes em quase todas as categorias do time dois (2), isso representando que os membros da equipe buscam uma forma de melhoria contínua.

Como a apresentação do gráfico ilustra as categorias e pontos relacionados a tais de forma genérica é conveniente realizarmos uma análise minuciosa das informações obtidas para entender melhor o contexto do time, portanto nas próximas seções serão apresentados os pontos e suas respectivas categorias de forma mais detalhada.

4.1.2.2.1 PONTOS POSITIVOS: Devido a uma forte incidência de pontos positivos em todas as categorias, fez se necessária uma entrevista com o time de desenvolvimento para entender o motivo de estarem com tal resultado.

Após a entrevista com o time, obteve se o resultado de que eles já vinham aplicando conceitos de melhoria contínua. Todos os pontos levantados pelo time que poderiam melhorar já estavam sendo absorvidos durante a sprint, esses pontos eram designados a uma pessoa e ela ficava encarregada da resolução do mesmo.

4.1.2.2.2 PONTOS NEGATIVOS: Como o time apresentava poucos pontos negativos nenhum resultado foi obtido nesta categoria através da análise dos dados coletados. Contudo após uma conversa com o time, foi possível extrair que o time estava enfrentando algumas dificuldades relacionadas ao planejamento do produto, mais especificamente em relação a visão que o produto tem para diferentes públicos. Para cada público o produto desenvolvido pelo time tinha uma visão, isso dificultando a visualização dos benefícios propostos pelo produto do time.

4.1.2.2.3 PONTOS A MELHORAR: O time evidenciou vários pontos a melhorar, porém como já estavam aplicando conceitos de melhoria contínua eles não necessitavam de nenhum apoio para realizar as mudanças.

4.1.2.3 AÇÕES EXECUTADAS

Em seguida a análise dos dados obtidos, pode-se definir a realização de ações que o time de desenvolvimento iria executar para alavancar seus resultados, pode se observar na Tabela 7 as ações tomadas, bem como suas motivações e resultados:

Categoria	Ações	Motivo	Resultado
Planejamento	Definição de uma visão	O produto desenvolvido pelo time apresentada visões diferentes para cada público que era apresentada, contudo a verdadeira visão é a que o cliente tem sobre o produto.	Com intuito de obter uma visão mais clara do produto, foi proposto que o time criasse um documento de visão, para que todos os interessados tivessem a mesma visão a partir de então.

Tabela 7: Tabela de Ações, Motivações e resultados do time 2

4.1.2.4 RESULTADO DO TIME

Após a aplicação da ação proposta para o time dois (2), foi realizada uma entrevista com intuído de identificar a percepção do time das técnicas de melhoria contínua. Como o time dois (2) já estava aplicando conceitos de melhoria contínua, eles não tiveram dificuldades na aplicação da ação proposta.

A ação proposta para o time foi a criação de um documento de visão do produto desenvolvido, isso pelo motivo de se manter uma visão centralizada e única do produto, mesmo que os membros do time sejam alterados, e foi o que aconteceu com esse time. No decorrer do projeto o time dois (2) sofreu uma alteração do P.O da equipe, afetando a visão do produto, pois não tinham um documento de visão.

Com a criação do documento de visão o time relata que houve uma melhora na comunicação e requisição de histórias mais assertivas entre os diferentes setores da empresa. Assim proporcionando um desenvolvimento de software mais fluido.

4.1.3 TIME 3

O time três (3) é um dos times que está trabalhando no desenvolvimento do projeto XL, no total são quatro (4)times. O projeto XL é um ERP Cloud que vem sendo desenvolvido em Java, ext.js, html e javascript, utilizando a metodologia Scrum para a criação e organização.

4.1.3.1 APRESENTAÇÃO DAS INFORMAÇÕES OBTIDAS

Com o intuito de apresentar e classificar as informações obtidas durante o período de coleta de dados do time três (3), foi desenvolvida a Tabela 8.

Categorias	**Que Bom**	**Que Pena**	**Que Tal**
Ambiente	2	3	0
Gestão de Conhecimento	0	0	1
Planejamento	5	9	4
Comportamento	0	0	0
Incentivo Organizacional	0	0	0
Multidisciplinaridade	0	0	0
Qualidade	0	0	0
Processo	0	0	0
Experiências Passadas	2	0	0
Rotatividade	0	0	1

Tabela 8: Tabela dos dados coletados do time 3
Fonte: Autoria própria

Para uma efetiva análise dos dados acima apresentados, é conveniente a apresentação da Figura 9.

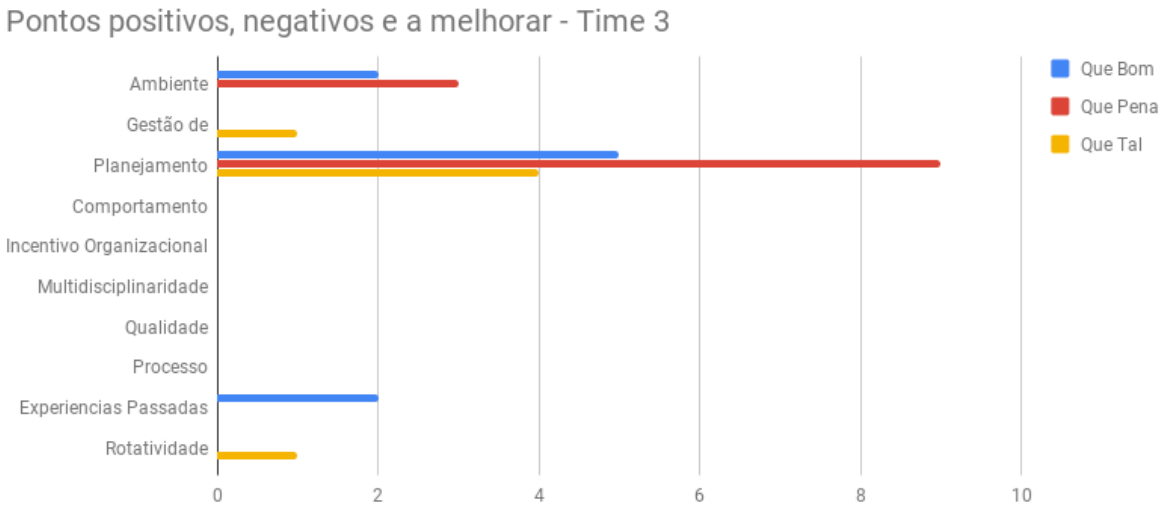


Figura 9: Gráfico de dados do time 3

Fonte: Autoria própria

4.1.3.2 ANÁLISE DAS INFORMAÇÕES OBTIDAS

É possível analisar que o time tem uma grande incidência em pontos negativos - como por exemplo podemos mencionar o planejamento e o ambiente - em uma das categorias e um comportamento mais nivelado nas outras categorias, apresentando pontos positivos e a melhorar.

4.1.3.2.1 PONTOS POSITIVOS Dentre os pontos positivos apresentados, uma categoria que foi bem vista e deveria acontecer com mais frequência está relacionada com as experiências externas do time de desenvolvimento. No caso apresentado foi um novo colaborador que teve uma abordagem diferente da proposta do P.O para a resolução de uma história.

4.1.3.2.2 PONTOS NEGATIVOS O índice que fica mais evidente no gráfico do time três (3) são os pontos negativos relacionados ao seu planejamento. O time três (3) estava falhando muito com o planejamento por motivos de terem muitos problemas técnicos para a realização da entrega do produto, dentre eles problemas com: deploy da aplicação, ambiente de Q.A instável, sistema de branchs, P.O compartilhado com outras equipes, falta de DevOps, dificuldades com a ferramenta de gerenciamento da sprint, dificuldades nas entregas das sprint's .

Além de problemas no planejamento o ambiente do time também estava apresentando alguns pontos negativos, muitos deles relacionados ao sistema de branchs que no fim também acabavam causando um problema no planejamento, pois se o ambiente não estava funcionando corretamente, acabava impedindo a entrega das demandas, ocasionando falhas no planejamento.

Em conversa com o time de desenvolvimento também foi descoberto que os mesmos não estavam com o processo muito bem organizado, faltando a realização do grooming.

4.1.3.2.3 PONTOS A MELHORAR: As categorias mais evidenciadas a melhorar foram as de rotatividade, gestão de conhecimento e planejamento. Planejamento pois já estavam tendo muitos pontos negativos relacionados e estavam tentando ajustá los, já na gestão de conhecimento e rotatividade foram novas ideias que surgiram do time de desenvolvimento.

Tanto na parte de gestão de conhecimento como na parte de rotatividade foram evidenciadas algumas ideias relacionadas a pessoas. Na categoria de gestão de conhecimentos foi proposto que houvesse um compartilhamento maior dos conhecimentos relacionados tanto a área técnica como a área de negócios. Na categoria de rotatividade foi sugerido que para os times de desenvolvimento no projeto web tivesse a rotatividade dos desenvolvedores front-end para assim poderem estar auxiliando em todos os projetos.

4.1.3.3 AÇÕES EXECUTADAS

Em cada categoria a ser investigada, o time de desenvolvimento estipulava uma ação com o intuito de alavancar seus resultados, podemos observar na Tabela 9 as ações efetivadas em cada categoria, bem como os resultados gerados através dessas ações e suas motivações.

Categoria	Ações	Motivo	Resultado
Planejamento	Definir uma meta única por sprint	Como o time de desenvolvimento trabalham com muitas histórias em uma sprint, nem sempre se mantinham focados, assim fazendo com que algumas histórias importantes não fossem entregues.	Definido uma meta na sprint, a qual o time deve se esforçar o máximo para entregar, se essa meta não é entregue a sprint é dada como falha.
Planejamento	Criar um DOR (Definition of ready)	Em consequência ao P.O ser compartilhado com demais equipes, algumas atividades ingressaram na sprint sem ter uma definição de requisitos bem definida.	Para que a história possa ser adicionada ao backlog da sprint é necessário que a mesma passe por uma definição de pronto (pronto para desenvolvimento).
Continua na próxima pagina			

Tabela 9 – continuação da pagina anterior

Categoria	Ações	Motivo	Resultado
Planejamento	Gestão visual	Como o time estava enfrentando alguns problemas com a ferramenta de gerenciamento das sprint's foi proposto que utilizassem a gestão visual e somente a descrição das histórias ficasse na ferramenta.	Houve a criação de um quadro kanban para o time, no qual os mesmos conseguiram realizar o monitoramento de progresso da sprint de movimentação das histórias de maneira simplificada, assim economizando esforço e tempo.
Planejamento / Processo	Definir uma data específica para realização do grooming	Muitas das vezes o time não planejava a realização da cerimônia de grooming na sprint, assim tornando a sprint de trabalho atual morosa.	Com intuito da cerimônia de grooming não interferir no andamento do time, foi definido que um dia específico da sprint teria uma sessão de grooming, assim todos já se planejando para tal.

Continua na próxima pagina

Tabela 9 – continuação da pagina anterior

Categoria	Ações	Motivo	Resultado
Ambiente	Gerente de configurações no time.	Muitos problemas com builds e ambientes de teste estavam ocorrendo e assim não entregavam o que estava planejado, gerava desperdício de trabalho pois os testadores não tinham um deploy para testar.	Houve o remanejamento de um colaborador para atuar como gerente de configurações dentro do projeto K, assim amenizando os problemas relacionados ao deploy, ambiente e gerenciamento de branch's.
Gestão de conhecimento	Criação de conteúdos em blog.	Nem todas as pessoas que estão no time trabalham ou conhecem sobre o processo do cliente (negócio). Conhecimento vindo de colaboradores novos não estariam sendo compartilhados entre os próprios membros do time ou outros times.	Para que haja uma maior troca de informações entre os colaboradores foi sugerido que quando tivessem o conhecimento em alguma área em específico escrevessem um post em blog interno, para que esse conhecimento seja difundido entre todos.

Continua na próxima pagina

Tabela 9 – continuação da pagina anterior

Categoria	Ações	Motivo	Resultado
Gestão de conhecimento	Realização de workshops e treinamentos internos.	Com o mesmo problema apresentado para a criação de postagens no blog, também houve a iniciativa de criação de workshops e treinamentos para assuntos que fossem de grande relevância para a empresa.	Com a necessidade de disseminar conhecimentos relevantes para a empresa, foi sugerida a realização de workshops, visando a disseminação de conhecimentos importantes para garantir o aprendizado dos interessados no assunto.
Planejamento / Qualidade	DOD (definition of done)	Em razão do time trabalhar com um compartilhamento de P.O, ele nem sempre estava presente para avaliar todas as alterações produzidas pelo time.	Para que todas as alterações possam ser produzidas com maior assertividade e menos dúvidas durante o processo de produção foi criada uma definição de "feito", para que o time possa se orientar.

Tabela 9: Tabela de Ações, Motivações e resultados do time 3

4.1.3.4 RESULTADO DO TIME

Após a aplicação de todas as ações no time houve uma entrevista com o time de desenvolvimento e o gestor, com o intuito de saber se a aplicação da melhoria contínua teve algum efeito no time. Tanto o relato do time, quanto o do gestor foram positivos.

Os resultados relacionados a gestão visual para auxílio no planejamento foram demasiadamente positivos, pois o time teve uma maior produtividade e o gestor conseguia acompanhar como estava o progresso da sprint somente por passar em frente ao quadro kanban do time. Para as dificuldades com entregas das sprints também houve um avanço, com a definição de uma meta na sprint o time estava mais orientado e conseguindo realizar o desenvolvimento das histórias de forma mais assertiva com a criação do DOR.

Na categoria de ambiente pode-se dizer que grande parte dos problemas foram solucionados com a vinda de um gerente de configurações, esse que está tomando conta de todo o gerenciamento de branch's, deploy da aplicação e criação/manutenção de ambientes de produção e Q.A.

Já na categoria de gestão de conhecimento foram ministrados alguns workshops e treinamentos internos, mas os membros do time ainda não se demonstraram muito encorajados a ministrar esses conteúdos. O próximo passo é idealizar uma forma em que todas as pessoas se sintam engajadas a ministrar uma palestra, ou até mesmo um treinamento.

Em um contexto mais genérico a aplicação da melhoria contínua foi positiva nesse time, pois houveram grandes impactos no ambiente de trabalho. O time estava sentindo se mais engajado e comprometido após essas mudanças, principalmente as relacionadas ao ambiente e gestão visual.

4.1.4 TIME 4

O time quatro (4) também é um dos times que está trabalhando no desenvolvimento do projeto XL.

4.1.4.1 APRESENTAÇÃO DAS INFORMAÇÕES OBTIDAS

Com o intuito de apresentar e classificar as informações obtidas durante o período de coleta de dados do time quatro (4), foi desenvolvida a Tabela 10.

Categorias	Que Bom	Que Pena	Que Tal
Ambiente	0	2	0
Gestão de Conhecimento	1	0	0
Planejamento	3	4	0
Comportamento	1	1	0
Incentivo Organizacional	0	0	0
Multidisciplinaridade	0	0	1
Qualidade	1	1	1
Processo	1	0	2
Experiências Passadas	0	0	0
Rotatividade	1	0	0

Tabela 10: Tabela dos dados coletados do time 4

Fonte: Autoria própria

Para uma efetiva análise dos dados acima apresentados, é conveniente a apresentação da Figura 10.

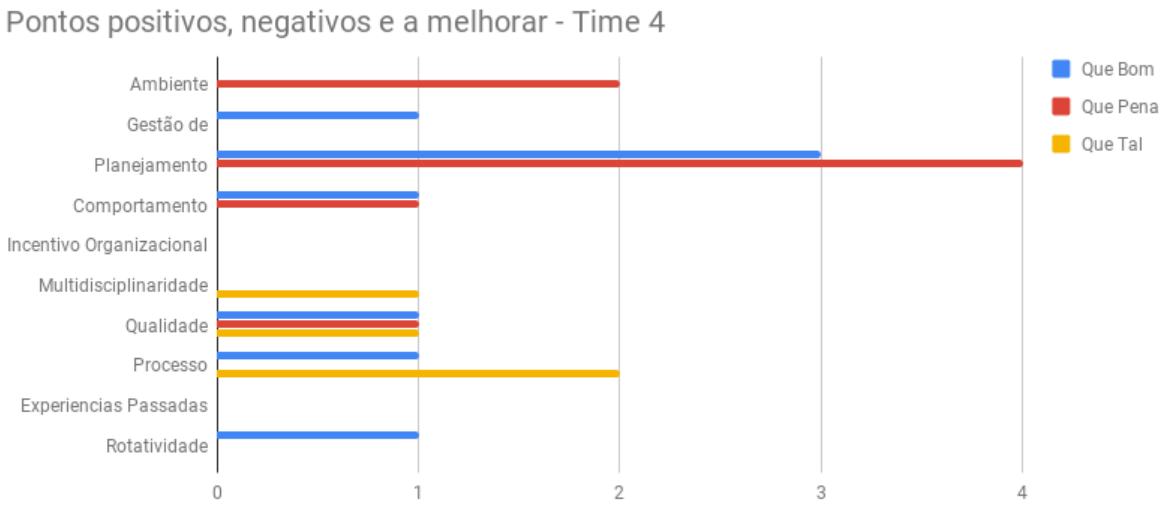


Figura 10: Gráfico de dados do time 4

Fonte: Autoria própria

4.1.4.2 ANÁLISE DAS INFORMAÇÕES OBTIDAS

É possível constatar que em quatro categorias há uma incidência de pontos negativos e pontos a melhorar e em duas categorias houve a incidência de pontos positivos. Para entender melhor o ocorrido cada um dos pontos mais relevantes vai ser compreendido nos próximos capítulos.

4.1.4.2.1 PONTOS POSITIVOS: Para os pontos positivos relacionados a categoria de rotatividade e gestão de conhecimento foram evidenciados que a rotatividades dos desenvolvedores front-end estavam causando um efeito positivos - em consequência de ter um time especializado em front-end, gerando a rotatividade quando necessário pelos demais times - compartilhando conhecimento com a equipe e assim tornando o trabalho mais fluido.

4.1.4.2.2 PONTOS NEGATIVOS: Devido a verificação de uma forte incidência nas categorias de planejamento, ambiente e comportamento, fez se necessária a análise pontual das informações para realizar ações que realmente fossem acarretar mudanças.

Na categoria relacionada ao planejamento do time foram evidenciados que os pontos negativos estavam ocorrendo devido ao fato do time enfrentar algumas dificuldades como por exemplo: idealizar a visão do produto como um todo, realização de estimativas não assertivas, má concepção de sprints. As mesmas dificuldades que estavam impactando no planejamento, também refletiam na qualidade do produto entregue pelo time.

Na categoria relacionada a comportamento, pode-se observar que era um time ainda muito novo não possuindo a maturidade adequada, os membros da equipe vinham enfrentando algumas dificuldades relacionadas a convivência e a comunicação, isso afetava todas as outras categorias.

4.1.4.2.3 PONTOS A MELHORAR: As principais categorias identificadas com pontos a melhorar foram as categorias relacionadas ao: processo e a multidisciplinaridade. Na categoria de processo o time tentava efetuar a aplicação de melhoria contínua e propor um mecanismo de registro para os impedimentos encontrados durante as sprints, a propósito de não manter nenhum recurso impedido por um longo período.

Pertinente à categoria de multidisciplinaridade, foi identificado que com a rotatividade dos desenvolvedores front-end que se concentravam em outro time, alguns

membros do próprio time quando trabalharam próximos a eles absorveram esse conhecimento e puderam exercer algumas atividades pertencentes a eles.

4.1.4.3 AÇÕES EXECUTADAS

Através da análise das categorias, é possível definir uma estratégia que vise buscar a melhoria dos resultados, podemos encontrar na Tabela 11 as condutas aplicadas em cada categoria, bem como suas motivações e possíveis resultados.

Categoria	**Ações**	**Motivo**	**Resultado**
Planejamento	Visão clara e exposta para todos do roadmap	O roadmap do projeto era mantido apenas em uma ferramenta web e nem sempre todos os membros do time estavam com uma visão clara de o que será feito e qual o objetivo geral do projeto.	Para melhor visualização do que estava acontecendo com o roadmap do projeto e objetivo a ser cumprido com uma determinada entrega, foi proposta a criação de um roadmap físico em uma parede para que todos possam ver e analisar o roadmap de forma mais fácil, rápida e frequente.
Continua na próxima pagina			

Tabela 11 – continuação da pagina anterior

Categoria	Ações	Motivo	Resultado
Planejamento	Toda equipe visitar clientes	Alguns membros da equipe enfrentaram algumas dificuldades para visualizar o cenário real do cliente.	Para melhor compreensão do cenário real do cliente, o processo do cliente e como o software funcionará após sua implantação, foi proposto que todos os membros da equipe realizassem pelo menos uma visita ao mesmo assim, se aproximando do cenário real.
Planejamento	Abrir o XL para outras fábrica e testes	Existem muitas pessoas que não estão no projeto, no entanto detém muito conhecimento de regras de negócio, cujas poderiam ser úteis na validação do projeto.	Foi proposto a abertura do projeto para todas as áreas da empresa, almejando a melhorar o processo de amadurecimento do produto com aumento de conhecimento de negócio e processo.

Continua na próxima pagina

Tabela 11 – continuação da pagina anterior

Categoria	Ações	Motivo	Resultado
Planejamento	Reforçar com o gerente a implantação do cliente BRZ	Pouca frequência de feedback com dos clientes que já estavam em produção.	Foi proposta a implantação de uma postura mais próxima ao cliente, visando uma melhor aproximação na relação e no feedback do cliente.
Planejamento / Comportamento	Uso consciente da liberdade	O uso demasiado de equipamentos como celulares, até mesmo utilização irregular de internet durante o horário de trabalho causavam uma dispersão causando uma baixa produtividade.	Em razão da baixa produtividade causada por diversos fatores, foi realizado uma conversa com os membros da equipe para que usufruam da liberdade oferecida de forma controlada, sem que afete o planejamento da sprint.

Continua na próxima pagina

Tabela 11 – continuação da pagina anterior

Categoria	Ações	Motivo	Resultado
Ambiente	Gerente de configurações no time.	Muitos problemas com builds e esferas de Q.A estavam decorrendo, assim não entregavam o que estava contemplado, gerava desperdício de desempenho pois os testadores não tinham um deploy testável.	Foi necessário a realocação de um recurso para atuar como gerente de configurações dentro do traço K, assim amenizando os pontos relacionados ao deploy, condição e gerenciamento de branch.
Comportamento	Maior socialização	Nível baixo de comunicação entre os membros da equipe e problemas comportamentais.	Para que o próprio time possa se conhecer melhor foi definido que uma parte dos bônus que eles ganhassem durante o projeto seria destinado a confraternização da equipe.
Processo	Aplicação de melhoria contínua	O time estava tentando realizar a aplicação de conceitos de melhoria contínua, porém ainda não estava trazendo os efeitos de forma satisfatória.	A aplicação da melhoria contínua dentro da equipe foi proposta pela presente pesquisa, já sanando o que era necessitado.

Continua na próxima pagina

Tabela 11 – continuação da pagina anterior

Categoria	Ações	Motivo	Resultado
Processo	Mecanismo para registro de impedimentos	O processo de identificar um impedimento durante a sprint era muito demorado, assim causando a pausa de um ou mais colaboradores.	Para que o gestor possa ver que existe algum impedimento durante a sprint, foi sugerido que o time criar uma atividade do tipo impedimento no kanban, assim ficando mais visual e orientado.
Multidisciplinaridade	Definição de uma data para o grooming	Em consequência ao compartilhamento de P.O o time necessita de mais conhecimento atrelado ao negócio do produto, a fim de atuar com mais facilidade nas sprints.	Foi sugerido que o time tivesse uma data fixa para a realização de grooming, assim dispersando o conhecimento de negócio detido pelo P.O para todos os membros da equipe.

Continua na próxima pagina

Tabela 11 – continuação da pagina anterior

Categoria	Ações	Motivo	Resultado
Planejamento / Qualidade	DOD (Definition of done)	Em razão de não tem um recurso de P.O dedicado, o time não sabe o que era para estar pronto muitas das vezes.	Para que haja a facilitação de o que deve estar desenvolvido para que haja o aceite do P.O na história foi proposto que o time criasse um DOD.

Tabela 11: Tabela de Ações, Motivações e resultados do time 4

4.1.4.4 RESULTADO DO TIME

Posteriormente a aplicação das ações proposta fez-se necessário a condução de uma entrevista com o time de desenvolvimento, cujos os melhores relataram que a melhoria contínua está acontecendo de maneira lenta, porém satisfatória. Nas categorias de planejamento e ambiente onde se encontravam os maiores gargalos enfrentados pelo time houve uma evolução a ser explanada melhor por cada categoria.

Na categoria de planejamento com a definição de um cliente mais próximo e com isso um feedback mais constante houve a evolução do produto de maneira muito mais acelerada do que vinha acontecendo, podendo também com essa ação se tornar mais acessível para que outros colaboradores do time visitassem o cliente. Além disso outras ações importantes para o time foram a aproximação do time de desenvolvimento para as regras de negócio com o grooming realizado corretamente, além disso também houve a criação do DOR e DOD, que também auxiliaram no planejamento atrelado a negócio.

Na categoria de ambiente quase todas as dificuldades enfrentadas pelo time foram satisfeitas com o papel de GCO (gerente de configurações). Todas aquelas dificuldades com o deploy não eram mais encargo do time de desenvolvimento, assim tornando o processo de desenvolvimento mais límpido.

O time gostou da aplicação de conceitos de melhoria contínua em seu processo de desenvolvimento. Em relato dizem que isso é o começo para a mudança do time.

4.1.5 TIME 5

O time cinco (5) é um dos times que compõe o desenvolvimento do projeto XL.

4.1.5.1 APRESENTAÇÃO DAS INFORMAÇÕES OBTIDAS

Com o intuito de apresentar e classificar as informações obtidas durante o período de coleta de dados do time cinco (5), foi desenvolvida a Tabela 12

Categorias	**Que Bom**	**Que Pena**	**Que Tal**
Ambiente	4	5	2
Gestão de Conhecimento	2	1	2
Planejamento	5	13	4
Comportamento	0	0	0
Incentivo Organizacional	1	0	0
Multidisciplinaridade	0	0	0
Qualidade	0	1	1
Processo	0	1	0
Experiências Passadas	0	0	0
Rotatividade	0	0	0

Tabela 12: Tabela dos dados coletados do time 5
Fonte: Autoria própria

Para uma efetiva análise dos dados acima apresentados, é conveniente a apresentação da Figura 11.

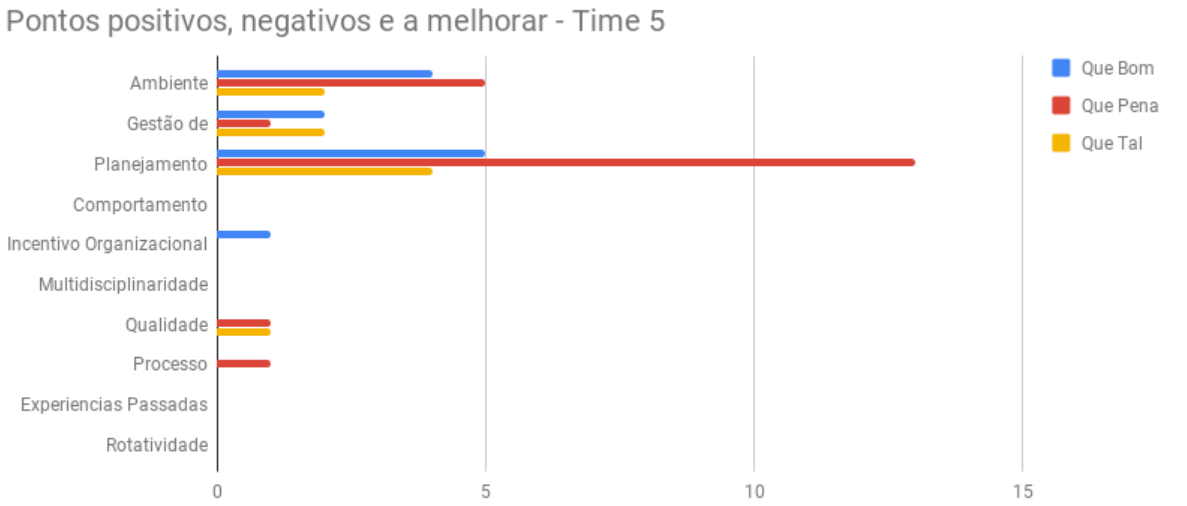


Figura 11: Gráfico de dados do time 5

Fonte: Autoria própria

4.1.5.2 ANÁLISE DAS INFORMAÇÕES OBTIDAS

É possível analisar que o time apresenta uma grande incidência de pontos negativos em duas das categorias pontuadas, que são: planejamento e ambiente. Além dos pontos negativos mais incidentes existem outras categorias que que são apresentados alguns pontos negativos, porém com uma incidência menor que são as categorias de: gestão de conhecimento, qualidade e processo.

Já nos pontos positivos o time apresenta em três categorias uma incidência, mas em nenhuma delas houve uma muita incidência, que são as categorias de: ambiente, gestão de conhecimento e incentivo organizacional.

Além dos pontos negativos e positivos temos ainda os pontos a melhorar que apareceram com uma menor ocorrência, isso mostrando que o time não vinha aplicando muitos conceitos de melhoria contínua. Os pontos a melhorar foram contatados nas categorias de: ambiente, gestão de conhecimento, planejamento e qualidade.

Devido a uma generalização das categorias faz-se necessário uma análise minuciosa para o entender o real motivo para tais pontos positivos, negativos e a melhorar em cada categoria, que serão apresentados nas seguintes sessões.

4.1.5.2.1 PONTOS POSITIVOS: Após essa análise minuciosa obteve-se o resultado em que os pontos positivos relacionados a categoria de incentivo organizacional

estavam ligados com a grande motivação do time cumprir uma meta para o recebimento de um bônus.

Já relacionado a categoria de gestão de conhecimento o time estava com um grande compartilhamento de conhecimento entre os membros da equipe, o que indica também que o time tem uma boa integração entre seus membros.

4.1.5.2.2 PONTOS NEGATIVOS: Após essa análise minuciosa obteve-se o resultado em que os pontos negativos apresentados anteriormente na categoria de planejamento estão relacionados ao time estar com uma dificuldade em cumprir a entrega da meta da sprint, estimativas não assertivas e falta de grooming (que está relacionado ao ponto negativo evidenciado na categoria de processo).

Para a categoria de ambiente o time estava enfrentando algumas dificuldades relacionadas ao deploy da aplicação, ambiente de Q.A instável e sistema de brachs mal organizado.

4.1.5.2.3 PONTOS A MELHORAR: Após essa análise minuciosa obteve-se o resultado que os pontos a melhorar para a categoria de ambiente estavam relacionados ao processo mal organizado de gerenciamento dos branchs e deploy da aplicação.

Para a categoria de gestão de conhecimento o time busca melhorar o meio de propagação de conhecimento dentro da própria equipe, assim podendo realizar a multidisciplinaridade.

Para a categoria de planejamento o time estava sofrendo algumas dificuldades com o tempo de aceite dos merges e falta de conhecimento para merges mais técnicos, assim propondo que existisse mais alguém responsável por essa atividade além do líder técnico e arquiteto do projeto. Investir mais tempo em design e arquitetura também era uma necessidade, cuja nem sempre era considerada no planejamento.

4.1.5.3 AÇÕES EXECUTADAS

Para cada categoria analisada foi definida uma ação que o time de desenvolvimento executou para alavancar seus resultados, pode se observar na Tabela 13 as ações tomadas para cada categoria, bem como motivações e seus resultados.

Categoria	Ações	Motivo	Resultado
Planejamento	Definir uma meta única por sprint	Muitas necessidades eram alocadas em uma iteração, assim causando uma dificuldade em entregar a história substancial da sprint.	A partir da definição de somente uma meta para a sprint o time ficou mais orientado a necessidade que deve ser tratado como essencial para a sprint.
Planejamento	Criar um DOR (Definition of ready)	Histórias que ingressaram na sprint sem ter uma análise de negócio, definição de requisitos bem elaborada ou arquitetura de software adequada muitas das vezes causavam atraso na sprint.	Para que uma história de usuário possa ser dada como uma demanda da sprint ela deve passar pelo DOR, cujo vai requisitar que a história tenha uma definição de requisitos bem elaborada e se necessário uma arquitetura de software.
			Continua na próxima pagina

Tabela 13 – continuação da pagina anterior

Categoria	Ações	Motivo	Resultado
Planejamento	Grooming mais técnico	Estimativas das histórias não estavam sendo eficientes, pois quando se iniciava o desenvolvimento da atividade, muitas das vezes era mais complexo que o suposto.	A fim de melhorar a assertividade da estimativa com a complexidade da história foi proposto que o time realizasse um grooming mais técnico para tais atividades, assim não analisando somente a nível de negócio, mas também a complexidade de desenvolvimento.
Planejamento	Alocação de mais um recurso para verificação de merges	Os merges abertos pelo time estavam levando muito tempo para ser avaliados.	Foi alocado um recurso do próprio time para auxiliar no aceite dos merges, assim os tornando mais rápidos e de maior entendimento como era um membro alocado full-time no time, já que o arquiteto é compartilhado.

Continua na próxima pagina

Tabela 13 – continuação da pagina anterior

Categoria	Ações	Motivo	Resultado
Processo	Definir uma data específica para realização do grooming	Rotineiramente o grooming não era efetivamente realizado pelo time, isso causando um impacto na próxima sprint ou se realizado de forma não planejada impactando na atual sprint de trabalho.	De forma que o grooming possa ser realizado que não atrapalhe o processo da sprint atual e nem que a próxima sprint falhe pela falta dele, foi definido que o grooming será realizado em uma data fixa, assim fazendo com que o time se planeje para ele e o processo funcione.
Ambiente	Gerente de configurações no time.	Frequentemente estavam ocorrendo muitos problemas relacionados ao deploy da aplicação, controle de branch's e relacionados ao ambiente de trabalho no geral, causando um desperdício.	Foi necessário a comutação de um membro do time de gerenciamento de configurações de outra fábrica para atuar como gerente de configurações no projeto K, assim tornando o ambiente muito mais estável.

Continua na próxima pagina

Tabela 13 – continuação da pagina anterior

Categoria	Ações	Motivo	Resultado
Gestão de conhecimento	Definido um membro do time de desenvolvimento para apoio	Nem todos os membros do time de desenvolvimento tinham total conhecimento sobre a tecnologia utilizada.	Para auxiliar os membros da equipe foi definido que um membro do time iria prestar apoio ao demais membros do time de desenvolvimento, assim espalhando o conhecimento e evitando demoras em buscas na internet.

Tabela 13: Tabela de Ações, Motivações e resultados do time 5

4.1.5.4 RESULTADO DO TIME

A seguir da aplicação das ações propostas para o time se sucedeu uma entrevista, a fim de verificar se houve alguma transição no seu processo de desenvolvimento com a aplicação dos conceitos de melhoria contínua.

Foram abordados os principais problemas enfrentados pelo time na entrevista, o relato do ocorrido procedeu-se que nas categorias de ambiente, planejamento e gestão de conhecimento houve uma melhora significativa. Na categoria de planejamento muitos das dificuldades que estavam sendo enfrentadas foram aliviadas com a alocação de mais um recurso para estar efetuando a validação das alterações do time, cujo o mesmo era o ponto de referência para o time (relacionado a categoria de gestão de conhecimento), atuando como um facilitador no time.

Na categoria de ambiente a maior parte das dificuldades enfrentadas pelo time foram atenuadas com um GCO (gerente de configurações) responsável, passando a não

ser mais encargo do time de desenvolvimento atual com grande parte dos pipelines, configurações e disponibilização de ambientes.

Considerando o cenário antes e depois da aplicação da melhoria contínua e o relato do time, a melhoria contínua suavizou algumas dificuldades enfrentadas pelo time.

4.2 TIMES DE MANUTENÇÃO

4.2.1 TIME 6

O time seis (6), é um time de desenvolvimento e manutenção do projeto CP, o qual é um software legado com vinte (20) anos de existência. Este time especificamente realiza alterações corretivas em um dos módulos mais complexos do software, que é o de venda e logística de entrega de mercadorias. A principal tecnologia utilizada no projeto é PowerBuilder.

4.2.1.1 APRESENTAÇÃO DAS INFORMAÇÕES OBTIDAS

Com o intuito de apresentar e classificar as informações obtidas durante o período de coleta de dados do time seis (6), foi desenvolvida a Tabela 14.

Categorias	**Que Bom**	**Que Pena**	**Que Tal**
Ambiente	1	10	0
Gestão de Conhecimento	0	0	0
Planejamento	8	14	2
Comportamento	2	0	0
Incentivo Organizacional	0	0	0
Multidisciplinaridade	0	0	0
Qualidade	8	3	4
Processo	2	2	0
Experiências Passadas	0	0	0
Rotatividade	0	0	0

Tabela 14: Tabela dos dados coletados do time 6

Fonte: Autoria própria

Para uma efetiva análise dos dados acima apresentados, é conveniente a apresentação da Figura 12.

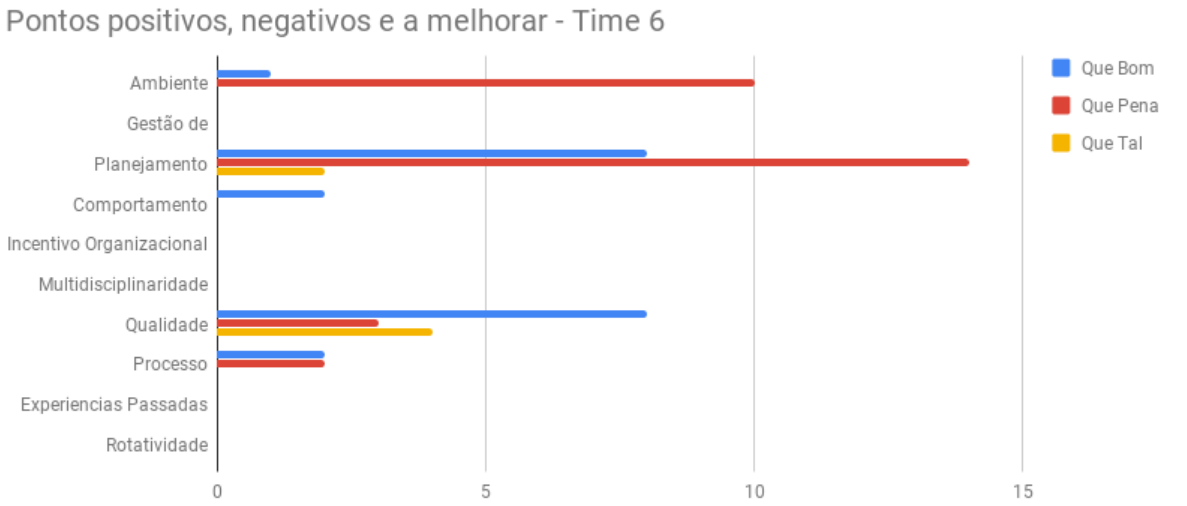


Figura 12: Gráfico de dados do time 6

Fonte: Autoria própria

4.2.1.2 ANÁLISE DAS INFORMAÇÕES OBTIDAS

É possível analisar que o time apresenta uma maior incidência de pontos negativos em duas categorias pontuadas que são: planejamento e ambiente. Além dos pontos negativos apresentados nessas categorias também existe uma menor incidência de pontos negativos nas categorias de qualidade e processo. Porém essas categorias não estão com sobressaliência de pontos negativos, o que representa que houve uma mudança. Também pode-se observar que o time apresenta uma incidência de pontos positivos em duas categorias sendo elas: qualidade e comportamento. Além dos pontos positivos e negativos ainda detemos dos pontos a melhorar nas categorias de qualidade e planejamento. Os pontos a melhorar não se apresentaram com uma grande incidência, demonstrando que a equipe não aplica muitos conceitos de melhoria contínua.

Devido a uma generalização das categorias faz-se necessário uma análise minuciosa para o entender o real motivo para tais pontos positivos, negativos e a melhorar em cada categoria, que serão apresentados nos seguintes subcapítulos.

4.2.1.2.1 PONTOS POSITIVOS: Após uma análise minuciosa observou-se que a categoria de qualidade apresentou pontos positivos pela participação de um inspetor de qualidade inserido dentro do time de desenvolvimento.

Na categoria de comportamento foi possível analisar que todo o time estava

comprometido com a entrega do produto de forma muito forte e boa integração com outras equipes.

4.2.1.2.2 PONTOS NEGATIVOS: Após a execução da análise minuciosa pode-se observar que os pontos negativos relacionados ao planejamento eram consequências de dificuldades como: realização de grooming, entrega de histórias planejadas na sprint e estimativas não assertivas.

Na categoria de ambiente todos os pontos negativos estavam relacionados com o problema em infraestrutura de banco de dados, como: indisponibilidade de serviço muito frequente e lentidão dos serviços.

4.2.1.2.3 PONTOS A MELHORAR: Após a realização de uma análise minuciosa observou-se que o time não está aplicando muitos conceitos de melhoria contínua pela baixa incidência de pontos a melhorar no geral.

Os pontos a melhorar apresentados pelo time na categoria de qualidade estão relacionados a melhorar a qualidade do desenvolvimento, assim garantindo uma maior integridade do código entregue, ainda nos pontos a melhorar tem-se a existência de alguns pontos a melhorar na categoria de planejamento que surgiram devido a uma grande incidência de pontos negativos e o time tomou algumas iniciativas, porém não executadas.

4.2.1.3 AÇÕES EXECUTADAS

Para cada categoria analisada foi estipulada uma ação que o time de desenvolvimento executou para estimular seus resultados, pode se examinar na Tabela 15 as ações tomadas para cada categoria, bem como motivações e seus resultados.

Categoria	Ações	Motivo	Resultado
Planejamento / Processo	Definir uma data específica para realização do grooming	Em várias ocasiões o time não estava realizando o grooming corretamente, isso acarretando em estimativas não assertivas.	Definido uma data fixa da sprint para a realização do grooming, assim o time podendo se organizar com suas atividades para a realização do grooming, não interferindo no processo.
Planejamento	Criar um DOR (Definition of ready)	O time muitas das vezes estava tendo alguns problemas em estimar as atividades.	Para que uma atividade possa ser incluída na sprint ela necessariamente deve conter uma definição bem elaborada de seus requisitos e para história mais complexas uma arquitetura de software, assim garantindo uma maior integridade das entregas da sprint.
Continua na próxima pagina			

Tabela 15 – continuação da pagina anterior

Categoria	Ações	Motivo	Resultado
Planejamento	Gestão visual	As entregas das sprints não estavam ocorrendo como o esperado.	Com a visualização das tarefas pendentes de forma mais visual o time consegue ter uma maior visão do andamento da sprint atual apenas por observação das atividades em um painel.
Ambiente	Ambiente específico para a equipe.	Como a indisponibilidade de serviços dos bancos de dados estava ocorrendo de maneira muito frequente impedia o progresso do time	Para que o time consiga trabalhar de forma esperada, é necessário que o ambiente esteja ok, para isso foi proposto que o time tivesse um ambiente específico ao invés de compartilhado.

Continua na próxima pagina

Tabela 15 – continuação da pagina anterior

Categoria	Ações	Motivo	Resultado
Planejamento / Qualidade	DOD (definition of done)	Algumas histórias não estavam sendo conferidas pelo P.O antes de serem liberadas para produção.	Em razão que o P.O realize a conferência de todas as histórias desenvolvidas durante a sprint um dos critérios do DOD, foi a aceitação pelo P.O da história, assim gerando mais qualidade.

Tabela 15: Tabela de Ações, Motivações e resultados do time 6

4.2.1.4 RESULTADO DO TIME

Posteriormente a aplicação das ações propostas ao time fez-se necessário uma entrevista com o time a fim de verificar o resultado obtido pela aplicação de conceitos de melhoria contínua em seu processo de trabalho.

Para a categoria de planejamento que era uma das mais fortes em pontos negativos no time, foram amenizados alguns problemas com a realização de um grooming, DOR e gestão visual. Para a categoria de ambiente os problemas relacionados ao ambiente não foram solucionados, pois não tinham recursos de máquina suficientes naquele momento, porém foi tomado nota e levado para a direção.

Obteve-se um bom feed back do time em relação a aplicação de conceitos de melhoria contínua, o time pretende continuar realizando a aplicação da melhoria contínua em seu processo.

4.2.2 TIME 7

O time sete (7), é um time de desenvolvimento e manutenção do projeto CF, o qual também é um software legado e sua principal tecnologia é PowerBuilder. O projeto CF sistema de ponto de vendas (PDV) e tem dez (10) anos de existência.

4.2.2.1 APRESENTAÇÃO DAS INFORMAÇÕES OBTIDAS

Com o intuito de apresentar e classificar as informações obtidas durante o período de coleta de dados do time sete (7), foi desenvolvida a Tabela 16.

Categorias	**Que Bom**	**Que Pena**	**Que Tal**
Ambiente	0	0	1
Gestão de Conhecimento	1	0	1
Planejamento	1	1	1
Comportamento	2	1	0
Incentivo Organizacional	1	0	0
Multidisciplinaridade	1	0	0
Qualidade	7	4	2
Processo	0	0	0
Experiências Passadas	0	0	0
Rotatividade	0	0	0

Tabela 16: Tabela dos dados coletados do time 7
Fonte: Autoria própria

Para uma efetiva análise dos dados acima apresentados, é conveniente a apresentação da Figura 13.

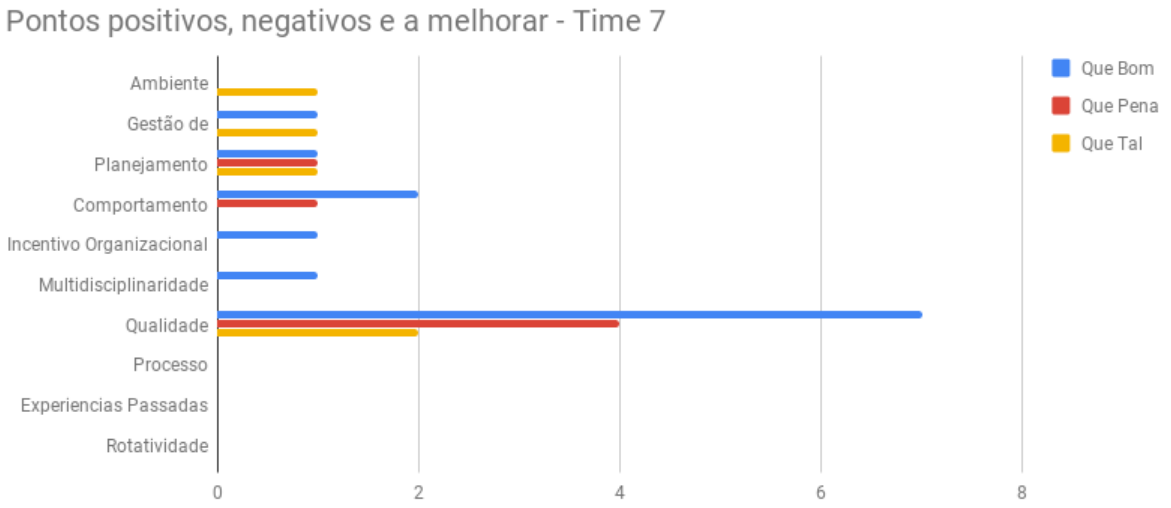


Figura 13: Gráfico de dados do time 7

Fonte: Autoria própria

4.2.2.2 ANÁLISE DAS INFORMAÇÕES OBTIDAS

É possível observar que o time apresenta uma grande incidência de pontos positivos na categoria relacionada a qualidade e um comportamento mais equilibrado nas demais categorias. Em relação aos pontos negativos pode se observar que o time não apresenta nenhuma categoria que chame atenção e em relação aos pontos a melhorar observa se que os mesmos também não são muito aparentes, mas estão presentes em várias categorias, demonstrando que há uma tendência a melhoria contínua.

Devido a uma análise muito macro das categorias por observação do gráfico, faz-se necessário uma investigação das categorias de maneira mais detalhada para entender a classificação dos pontos positivos, negativos e a melhorar.

4.2.2.2.1 PONTOS POSITIVOS: Após a investigação dos pontos positivos observou-se que o time tem uma grande ocorrência de pontos positivos na categoria de qualidade, que está relacionado ao time estar executando algumas ações para que haja a melhora da qualidade há algum tempo, como: aproximação da equipe de homologação da de desenvolvimento, inspetor de testes, análises de impacto das alterações e automação de teste com grande cobertura.

4.2.2.2.2 PONTOS NEGATIVOS: Após a investigação dos pontos negativos relacionadas ao time não foi possível encontrar nenhum ponto que estivesse realmente com um problema efetivo.

Porém após uma entrevista realizada com os membros do time os mesmos relataram que o time poderia ser mais participativo podendo influenciar no processo do time e comportamento.

4.2.2.2.3 PONTOS A MELHORAR: Após a análise dos pontos a melhorar evidenciados pelo time não foi possível encontrar nenhuma ação para mudança evidenciando que o time está aplicando muito pouco ou nenhum conceito de melhoria contínua em seu processo de desenvolvimento. O mesmo citado acima replica-se para os pontos a melhorar, se o time fosse mais participativo poderia ter um resultado mais aparente.

4.2.2.3 RESULTADO DO TIME

A aplicação de técnicas de melhoria continua para esse time não obteve sucesso através da coleta de dados, pois o time não estava participando do técnica como fora proposta. Apesar disso foi evidenciado em entrevista de maneira semi estruturado com os membros do time que os mesmos não tinham uma participação muito grande, realizando somente o trabalho necessário. Essa dificuldade foi descoberta muito tarde e não foi possível realizar nenhuma ação. Outro fato relacionado a essa dificuldade é que o time possivelmente não quis expor os fatos ocorridos com total transparência.

Uma ação que pode ser executada para auxiliar o time com a participação são os atividades energizantes que o time realize junto, para assim se soltar mais e contribuir mais com a coleta dos dados.

4.3 RESULTADOS FINAIS

Nesta sessão serão apresentados os resultados de maneira agrupada, quais as dificuldades e soluções mais comuns e também qual a satisfação e aceitação das técnicas de melhoria contínua, que servirão de insumo para a sessão de discussões.

4.3.1 PRINCIPAIS CATEGORIAS E SOLUÇÕES

Para melhor analisar o resultado sumarizado de todos os times de desenvolvimento criou-se a Tabela 17 mostrando as principais dificuldades enfrentadas (representadas pela letra X) e os principais pontos a melhorar (representados pela letra Y) evidenciados no estudo de caso.

Categoria	Time 1	Time 2	Time 3	Time 4	Time 5	Time 6	Time 7
Ambiente	Y		X	X	X	X	
Gestão de Conhecimento			Y		Y		
Planejamento	X	Y	X	X	X	X	
Comportamento				X			X
Incentivo Organizacional							
Multidisciplinaridade				Y			
Qualidade	Y				Y		
Processo	Y			Y			
Experiências Passadas							
Rotatividade			Y				

Tabela 17: Tabela das principais categorias
Fonte: Autoria própria

Pode-se observar que as categorias de planejamento, processo, qualidade e ambiente têm uma maior incidência, para cada uma dessas, foram criadas algumas ações que os times executaram. Com intuito de generalizar o resultado de tal pesquisa serão classificadas algumas soluções para possíveis dificuldades enfrentadas no dia a dia.

Para melhor relacionar as categorias e possíveis soluções, bem como as referências literárias, foi criada a Tabela 18.

Categorias	Ações	Referência
Planejamento	Definir uma meta única por sprint;	SCHWABER, Ken; SUTHERLAND, Jeff; BEEDLE, Mike. The definitive guide to scrum: The rules of the game.
	Criar um DOR (definition of ready);	
	Grooming mais técnico; e	
	Roadmap mais visível.	
Processo	Definir uma data específica para realização do grooming.	POPPENDIECK, Mary; POPPENDIECK,Tom. Lean software development: an agile toolkit.
Qualidade	Criar um DOD (definition of done);	
	Ponto de corte de desenvolvimento.	EBERT, Christof et al. DevOps.
Ambiente	Estudar e aplicar DevOps no projeto;	GUIDE, Student Leadership Team Building.
	Ambiente dedicado para equipe;	
	Gerente de configurações.	
Comportamento	Atividades em equipe.	

Tabela 18: Tabela de categorias e soluções propostas genéricas
Fonte: Autoria própria

4.3.2 ACEITAÇÃO E AVALIAÇÃO DE TÉCNICAS DE MELHORIA CONTÍNUA

Para coletar informações sobre a aplicação dos conceitos de melhoria contínua, foi aplicado um questionário com os membros das equipes participantes. No total foram coletados quarenta e duas (42) respostas.

As perguntas realizadas no questionário foram as mesmas da entrevista com os líderes técnicos dos times. Para cada pergunta foram compiladas as respostas e apresentados em formato de gráfico a seguir:

1. Foi difícil de realizar a aplicação de melhoria contínua?

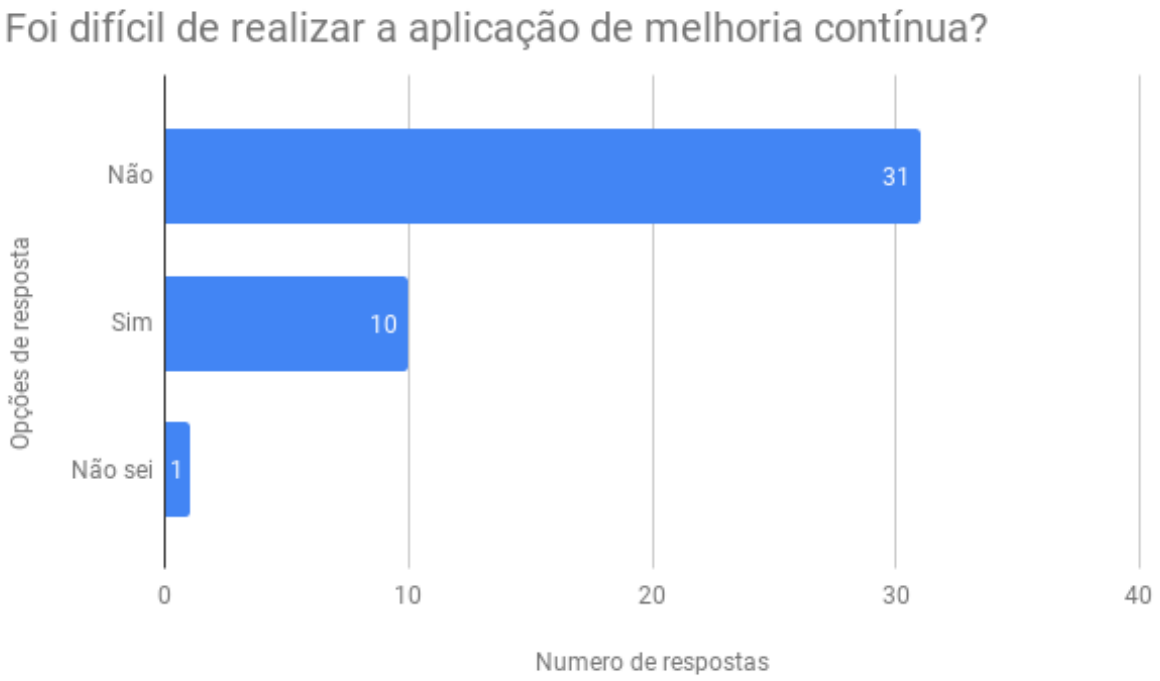


Figura 14: Gráfico de respostas a questão 1

Fonte: Autoria própria

Conforme a representação gráfica da questão um (1) na figura 14, pode-se verificar que a maioria das pessoas que responderam ao questionário, não acharam difícil a a aplicação dos conceitos de melhoria contínua.

2. A melhoria contínua é um processo demorado?

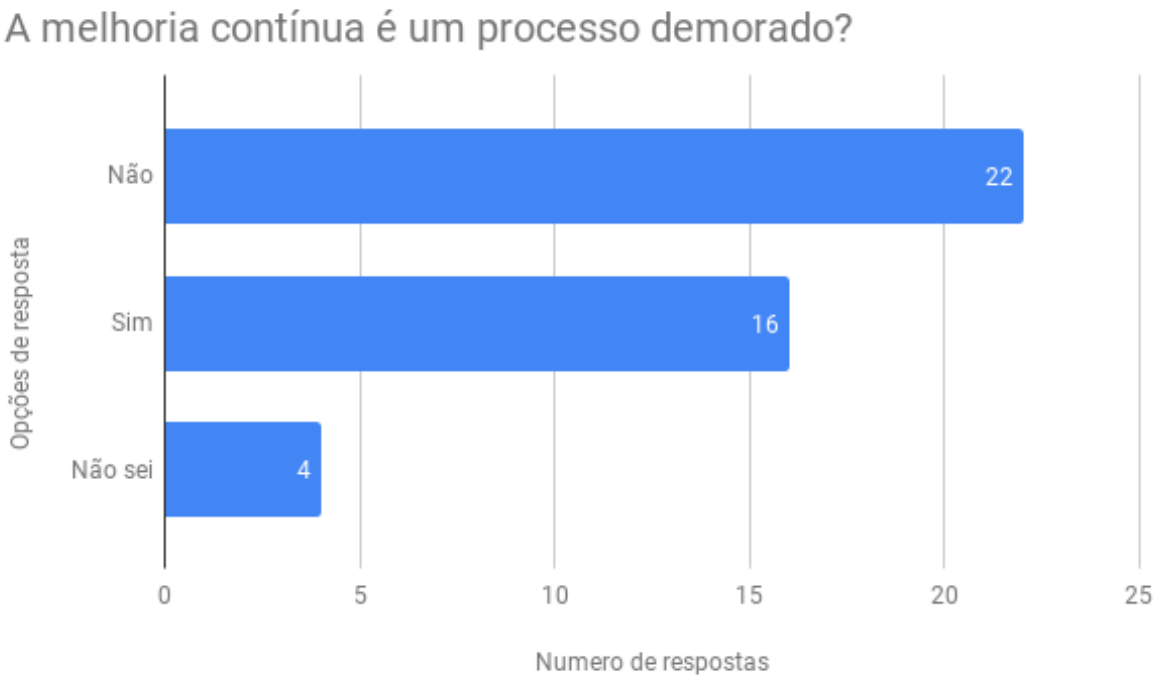


Figura 15: Gráfico de respostas a questão 2

Fonte: Autoria própria

Na figura 15, etá representado o gráfico referente a segunda questão, nesta, pode-se notar que o numero das pessoas que não acharam o processo demorado é maioria, demonstrando que a melhoria continua é um processo rápido.

3. A melhoria contínua trouxe um resultado positivo?

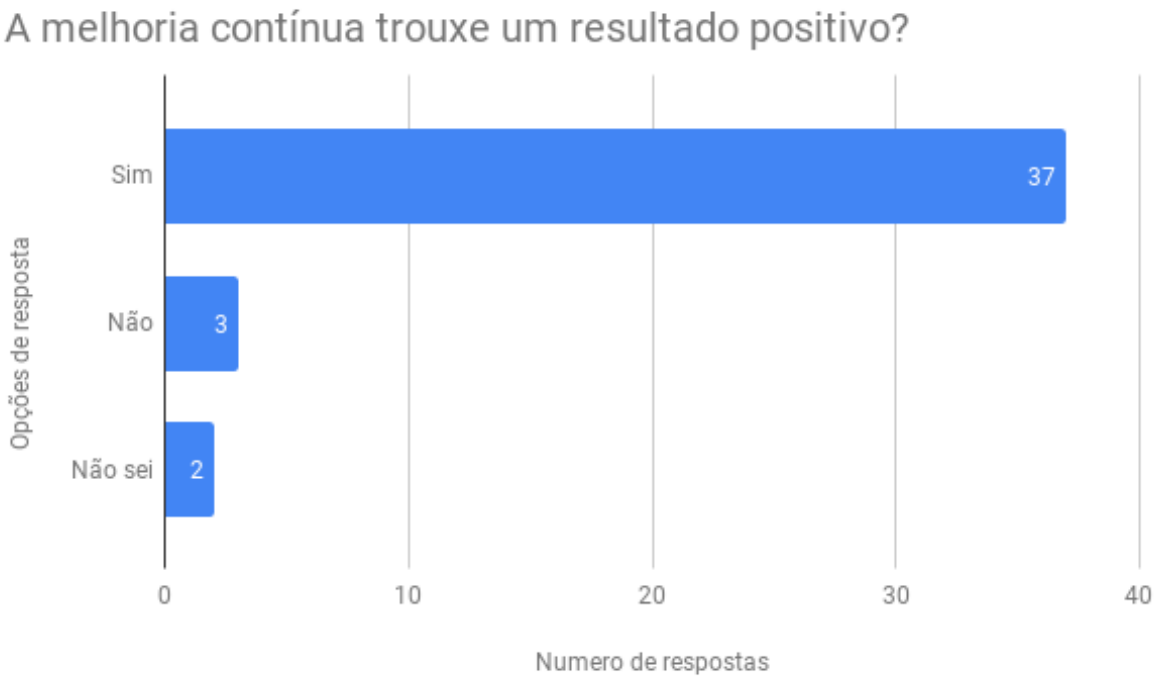


Figura 16: Gráfico de respostas a questão 3
Fonte: Autoria própria

Nota-se que grande parte das respostas obtidas para a pergunta três (3), foram respostas positivas, o que indica que a aplicação de conceitos de melhoria contínua trouxe um resultado positivo.

4.4 DISCUSSÕES

Diante das observações realizadas, a aplicação da melhoria contínua foi bem sucedida, devido ao resultados positivos alcançados apos a implementação das ações corretivas. A aplicação da metodologia na empresa em questão inicialmente foi dificultosa, pela resistência dos colaboradores, porém depois de algum tempo ela tornou-se mais natural, assim facilitando sua aplicação.

O estudo de caso começou com a proposição de uma avaliação comportamental dos fatos ocorridos durante as sprints, para assim poder elencar ações para a mudança de comportamentos que afetassem negativamente os times. Um fato que auxiliou na implantação da melhoria contínua, foi que a empresa já trabalhava com um framework ágil, assim não afetando tanto o processo auxiliando na implantação do estudo de caso.

O estudo de caso teve duração de cinco (5) meses, sendo realizadas coletada de dados por meio de planilhas, observações e entrevistas durante todo esse tempo. A primeira instância a coletada de dados fora feita através da inserção das informações manualmente na planilha, após uma sugestão vinda dos usuários foi elaborada a inserção dos itens por meio de um formulário, assim facilitando a inserção de itens durante a sprint.

A cada quatro (4) sprints eram realizadas as análises dos dados obtidos em conjunto com os líderes técnicos dos times. Uma dificuldade enfrentada nesta etapa foi conseguir tempo dos líderes técnicos, que muitas vezes estavam ocupados com a demanda da sprint.

Após a análise dos dados eram elaboradas as ações para aliviar as dificuldades enfrentadas pelos times - posteriormente a análise de todos os times também foi possível analisar que em vários times estavam ocorrendo problemas da mesma natureza, caracterizando assim as maiores dificuldades no desenvolvimento de software ágil - e apresentadas ao time, para validação se estavam de acordo. Os responsáveis pela aplicação das ações dentro dos times foram os líderes técnicos. Uma facilidade para aplicação das ações nos times foi total apoio da gerência, assim fazendo com que as ações fossem realizadas de uma maneira top down, ou seja de cima para baixo - em uma corporação muitas vezes uma abordagem acontece de forma down top chega na gerência e é barrada, com a abordagem anterior isso não acontece.

Almejando a verificação da assertividade proposta pela metodologia aplicada, fora realizada um questionário, cujas respostas evidenciaram que a aplicação da melhoria contínua não foi difícil nem demorada e trouxeram resultados positivos. Com isso em mente a pesquisa desenvolvida conseguiu alcançar a proposta inicial - aplicação da melhoria contínua como auxílio ao processo de desenvolvimento de software de grande porte. No entanto alguma ameaças foram evidenciadas durante a trajetória de tal pesquisa - segundo Zahran (1998), a implantação de uma melhora no processo de desenvolvimento de software muitas vezes é uma atividade complexa e que exige demasiado conhecimento - as quais serão melhores esplanadas na seguinte sessão.

4.4.1 AMEAÇAS

Nesta sessão apresentam-se as principais ameaças ocorridas com relação à viabilidade do trabalho, bem como seus contrapontos para a viabilização da pesquisa.

4.4.1.1 TEMPO DOS LÍDERES

Os líderes técnicos muitas vezes tinham varias tarefas para resolver durante o dia, ocasionando dificuldades na realização das entrevistas com os mesmos.

Como medida de contingência adotou-se agendamento prévio para realização das entrevistas com flexibilidade de mudança conforme a demanda de tempo dos líderes.

4.4.1.2 RESULTADOS DEMORADOS

As equipes que participaram da coleta de dados bem como líderes e suas chefias criaram expectativa de que os resultados da aplicação do estudo seriam imediatos, o que poderia ocasionar insatisfação pois os resultados são gradativos.

Adotou-se a estratégia de explicar detalhadamente o procedimento do estudo de caso, exemplificando outros casos de empresas que obtiveram sucesso com casos similares, visando dar ciência desse tipo de processo de mudança.

4.4.1.3 PARTICIPAÇÃO

Uma das principais ameaças à viabilidade do presente trabalho foi relacionado a participação das pessoas, sem uma participação efetiva das pessoas a aplicação dos conceitos de melhoria contínua não seriam efetivados, pois quem faz os processos são as pessoas e sem elas o trabalho seria dispensável.

Em contraponto a essa ameaça para aplicação de tal trabalho, foi concedido aval efetivo da diretoria para trabalhar com a maior parte dos times.

5 CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

A partir da realização deste trabalho observou-se que é possível realizar a aplicação de técnicas de melhoria contínua por meio de um estudo de caso.

A partir do projeto de estudo de caso foi possível definir de forma objetiva as questões específicas de pesquisa, que posteriormente responderam a questão de pesquisa e ao problema definido para tal pesquisa. Com base na resposta dessas questões através da coleta e análise de dados adiante da aplicação de técnicas de melhoria contínua, revelaram informações relevantes a nível organizacional.

A empresa passou a ter mais controle sobre os processos que estavam executando, por meio de monitoramento dos mesmos. A aplicação da melhoria contínua tornou possível a identificação das dificuldades, que passavam de maneira despercebida e a partir dela foi facilitada a tomada de ações com intuito de amenizar o impacto dessas dificuldades no processo. Tudo isso impacta na produtividade dos times, pois reduziu-se o tempo em que atividades que não geram valor ao produto fossem executadas e em razão disso o produto entregue passou a ter mais valor.

Conclui-se que a aplicação da melhoria contínua foi bem sucedida, pois foi possível estabelecer por meio de uma maneira simples e estruturada a identificação e alivio de dificuldades encontradas no processo de desenvolvimento de software de grande porte.

A partir de então, será possível replicar este tipo de estudo para outras empresas do mesmo porte que desejem avaliar seus processos e utilizar melhoria contínua para otimizar o desenvolvimento de projetos.

Por se tratar de um estudo de caso com aplicação em uma única empresa, a validação externa da pesquisa não foi realizada. Como um trabalho futuro decorrente do presente estudo, objetiva-se a aplicação dessa pesquisa para outras empresas.

Além disso, poderá ser proposta a automatização do processo de coleta de dados, assim como do acompanhamento do progresso das ações corretivas, realizados durante a

aplicação da melhoria contínua.

REFERÊNCIAS

ANDERSSON, R.; ERIKSSON, H.; TORSTENSSON, H. Similarities and differences between tqm, six sigma and lean. **The TQM magazine**, Emerald Group Publishing Limited, v. 18, n. 3, p. 282–296, 2006.

BECK, K. **Extreme programming explained: embrace change**. [S.l.]: addison-wesley professional, 2000.

BECK, K.; BEEDLE, M.; BENNEKUM, A. V.; COCKBURN, A.; CUNNINGHAM, W.; FOWLER, M.; GRENNING, J.; HIGHSMITH, J.; HUNT, A.; JEFFRIES, R. et al. Manifesto for agile software development. 2001.

BHUIYAN, N.; BAGHEL, A. An overview of continuous improvement: from the past to the present. **Management decision**, Emerald Group Publishing Limited, v. 43, n. 5, p. 761–771, 2005.

BHUIYAN, N.; BAGHEL, A.; WILSON, J. A sustainable continuous improvement methodology at an aerospace company. **International Journal of Productivity and Performance Management**, Emerald Group Publishing Limited, v. 55, n. 8, p. 671–687, 2006.

BIFFL, S.; AURUM, A.; BOEHM, B.; ERDOGMUS, H.; GRÜNBACHER, P. **Value-based software engineering**. [S.l.]: Springer Science & Business Media, 2006.

BOEHM, B. A view of 20th and 21st century software engineering. In: ACM. **Proceedings of the 28th international conference on Software engineering**. [S.l.], 2006. p. 12–29.

BRASIL, L. I. **Desenvolvimento Lean de produtos.** 2010. Disponível em: <http://www.lean.org.br>.

CAFFYN, S. Extending continuous improvement to the new product development process. **R&D Management**, Wiley Online Library, v. 27, n. 3, p. 253–267, 1997.

CHAUDHURI, A. Simultaneous improvement in development time, cost and quality: a practical framework for generic pharmaceuticals industry. **R&D Management**, Wiley Online Library, v. 43, n. 3, p. 227–241, 2013.

COHEN, D.; LINDVALL, M.; COSTA, P. Agile software development. **DACS SOAR Report**, v. 11, 2003.

GODOI, C. K.; MELLO, R. Bandeira-de; SILVA, A. d. Pesquisa qualitativa em estudos organizacionais: paradigmas, estratégias e métodos. **São Paulo: Saraiva**, v. 2, 2006.

GRIFFIN, A. Metrics for measuring product development cycle time. **Journal of product innovation management**, Wiley Online Library, v. 10, n. 2, p. 112–125, 1993.

GYGI, C. **Seis Sigma para leigos**. [S.l.]: Alta Books Editora, 2008.

HARTER, D. E.; KRISHNAN, M. S.; SLAUGHTER, S. A. Effects of process maturity on quality, cycle time, and effort in software product development. **Management Science**, INFORMS, v. 46, n. 4, p. 451–466, 2000.

HELLSTEN, U.; KLEFSJÖ, B. Tqm as a management system consisting of values, techniques and tools. **The TQM magazine**, MCB UP Ltd, v. 12, n. 4, p. 238–244, 2000.

HIBBS, C.; JEWETT, S.; SULLIVAN, M. **The art of lean software development: a practical and incremental approach**. [S.l.]: "O'Reilly Media, Inc.", 2009.

HIGHSMITH, J. **Agile project management: creating innovative products**. [S.l.]: Pearson Education, 2009.

HIGHSMITH, J.; COCKBURN, A. Agile software development: The business of innovation. **Computer**, IEEE, v. 34, n. 9, p. 120–127, 2001.

HUMPHREY, W. S. **Managing the software process**. [S.l.]: Addison-Wesley, 1989.

IMAI, M. **Kaizen: a estratégia para o sucesso competitivo**. [S.l.]: Imam, 1994.

JHA, S.; NOORI, H.; MICHELA, J. L. The dynamics of continuous improvement: aligning organizational attributes and activities for quality and productivity. **The International Journal of Quality Science**, Emerald Group Publishing, Limited, v. 1, n. 1, p. 19, 1996.

KOWANG, T. O.; RASLI, A. New product development in multi-location r&d organization: a concurrent engineering approach. **African Journal of Business Management**, Academic Journals, v. 5, n. 6, p. 2264, 2011.

LAGER, T. A new conceptual model for the development of process technology in process industry: A point of departure for the transformation of the"process development process"into a formal work process? **International Journal of Innovation Management**, World Scientific, v. 4, n. 03, p. 319–346, 2000.

LAUTENSCHLAGER, B. R.; OLIVEIRA, R.; MACHADO, P. H. de A. Melhoria contínua no gerenciamento de projetos ágeis em software de grande porte: um projeto de pesquisa. **SimES 2017: I SimES - Simpósio de Engenharia de Software da UTFPR-DV**, 2017.

LODGAARD, E.; AASLAND, K. E. An examination of the application of plan-do-check-act cycle in product development. In: **DS 68-10: Proceedings of the 18th International Conference on Engineering Design (ICED 11), Impacting Society through Engineering Design, Vol. 10: Design Methods and Tools pt. 2, Lyngby/Copenhagen, Denmark, 15.-19.08. 2011**. [S.l.: s.n.], 2011.

MACHADO, P. H. de A. IntegraÇÃo dos conceitos lean Às metodologias Ágeis como inovaÇÃo em fÁbricas de software de grande porte. **VI Congresso de Sistemas LEAN**, 2016.

MISHRA, D.; MISHRA, A. Complex software project development: agile methods adoption. **Journal of Software: Evolution and Process**, Wiley Online Library, v. 23, n. 8, p. 549–564, 2011.

MUSAT, D.; RODRÍGUEZ, P. Value stream mapping integration in software product lines. In: ACM. **Proceedings of the 11th International Conference on Product Focused Software**. [S.l.], 2010. p. 110–111.

NÄSLUND, D. Lean, six sigma and lean sigma: fads or real process improvement methods? **Business Process Management Journal**, Emerald Group Publishing Limited, v. 14, n. 3, p. 269–287, 2008.

NEGOCIO, V. e. 2017. Disponível em: <http://valorecompetencia.com.br>.

NEILL, C. J.; LAPLANTE, P. A. Requirements engineering: the state of the practice. **IEEE software**, IEEE, v. 20, n. 6, p. 40–45, 2003.

NERUR, S.; MAHAPATRA, R.; MANGALARAJ, G. Challenges of migrating to agile methodologies. **Communications of the ACM**, ACM, v. 48, n. 5, p. 72–78, 2005.

NETO, O. N. d. S. Análise comparativa das metodologias de desenvolvimento de softwares tradicionais e ágeis. **Trabalho de Conclusão de Curso. Universidade da Amazônia. Belém**, 2004.

NILSSON-WITELL, L.; ANTONI, M.; DAHLGAARD, J. J. Continuous improvement in product development: Improvement programs and quality principles. **International Journal of Quality & Reliability Management**, Emerald Group Publishing Limited, v. 22, n. 8, p. 753–768, 2005.

PENHA, C. D. Empresa rede: uma nova forma de gestão–algar s/a–empreendimentos e participações. **Produção, diagramação e impressão: ABC Sabe**, 1993.

PERNSTÅL, J.; FELDT, R.; GORSCHEK, T. The lean gap: A review of lean approaches to large-scale software systems development. **Journal of Systems and Software**, Elsevier, v. 86, n. 11, p. 2797–2821, 2013.

POPPENDIECK, M.; CUSUMANO, M. A. Lean software development: A tutorial. **IEEE software**, IEEE, v. 29, n. 5, p. 26–32, 2012.

POPPENDIECK, M.; POPPENDIECK, T. **Lean Software Development: An Agile Toolkit: An Agile Toolkit**. [S.l.]: Addison-Wesley, 2003.

PRESSMAN, R. S. **Engenharia de software**. [S.l.]: Makron books Sao Paulo, 2016.

PRIKLADNICKI, R.; WILLI, R.; MILANI, F. **Métodos ágeis para desenvolvimento de software**. [S.l.]: Bookman Editora, 2014.

RAJLICH, V. Changing the paradigm of software engineering. **Communications of the ACM**, ACM, v. 49, n. 8, p. 67–70, 2006.

RAUSCH, A.; BERGNER, S.; WANG, D. Model-and constraint-based engineering of complex software ecosystems. In: IEEE. **Information Technology and Electronic Commerce (ICITEC), 2014 2nd International Conference on**. [S.l.], 2014. p. 77–82.

RINGEN, G.; HOLTSKOG, H. How enablers for lean product development motivate engineers. **International Journal of Computer Integrated Manufacturing**, Taylor & Francis, v. 26, n. 12, p. 1117–1127, 2013.

ROCHA, T. Á. da; OLIVEIRA, S. R. B.; VASCONCELOS, A. M. L. de. Adequação de processos para fábricas de software. **Anais do Simpósio Internacional de Melhoria de Processo de Software (SIMPROS)**, 2004.

ROYCE, W. W. Managing the development of large software systems: concepts and techniques. In: IEEE COMPUTER SOCIETY PRESS. **Proceedings of the 9th international conference on Software Engineering**. [S.l.], 1987. p. 328–338.

SAVOLAINEN, T.; HAIKONEN, A. Dynamics of organizational learning and continuous improvement in six sigma implementation. **The TQM Magazine**, Emerald Group Publishing Limited, v. 19, n. 1, p. 6–17, 2007.

SCHWABER, K.; SUTHERLAND, J. The scrum guide (2013). **http://www. scrum. org/Scrum-Guides>. Acessado em**, v. 16, p. 18, 2013.

SILVA, I. F. da; NETO, P. A. da M. S.; O'LEARY, P.; ALMEIDA, E. S. de; MEIRA, S. R. de L. Agile software product lines: a systematic mapping study. **Software: Practice and Experience**, Wiley Online Library, v. 41, n. 8, p. 899–920, 2011.

SINGH, J.; SINGH, H. Continuous improvement philosophy–literature review and directions. **Benchmarking: An International Journal**, Emerald Group Publishing Limited, v. 22, n. 1, p. 75–119, 2015.

SOMMERVILLE, I. et al. **Engenharia de software**. [S.l.]: Addison Wesley São Paulo, 2011.

SPIVEY, W. A.; MUNSON, J. M.; WOLCOTT, J. H. Improving the new product development process: a fractal paradigm for high-technology products. **Journal of Product Innovation Management**, Elsevier, v. 14, n. 3, p. 203–218, 1997.

SUIKKI, R.; TROMSTEDT, R.; HAAPASALO, H. Project management competence development framework in turbulent business environment. **Technovation**, Elsevier, v. 26, n. 5, p. 723–738, 2006.

SUN, H.; ZHAO, Y.; YAU, H. K. The relationship between quality management and the speed of new product development. **The TQM Journal**, Emerald Group Publishing Limited, v. 21, n. 6, p. 576–588, 2009.

SUTHERLAND, J. Scrum handbook. **Somerville. USA: Scrum Training Institute Press. a**, 2010.

TAKEUCHI, H.; NONAKA, I. 16 the new new product development game. **Japanese Business: Part 1, Classics Part 2, Japanese management Vol. 2: Part 1, Manufacturing and production Part 2, Automotive industry Vol. 3: Part 1, Banking and finance Part 2, Corporate strategy and inter-organizational relationships Vol. 4: Part 1, Japanese management overseas Part 2, Innovation and learning**, Routledge, v. 64, n. 1, p. 321, 1998.

TELES, V. M. Extreme programming. **São Paulo: Novatec**, 2004.

WINGO, R. S.; TANIK, M. M. Using an agile software development methodology for a complex problem domain. In: IEEE. **SoutheastCon 2015**. [S.l.], 2015. p. 1–8.

WOMACK, J. P.; JONES, D. T.; ROOS, D. The machine that changed the world: The story of lean production. 1st harper perennial ed. **New York**, 1991.

YAN, B.; MAKINDE, O. D. Modelling the long term impact of existing products on perceived value of new products. In: IEEE. **Industrial Engineering and Engineering Management, 2009. IEEM 2009. IEEE International Conference on**. [S.l.], 2009. p. 1136–1140.