

UNIVERSIDADE NOVE DE JULHO – UNINOVE
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA E GESTÃO DO
CONHECIMENTO

Nilson Salvetti

PROÁGIL-29110: Processo ágil aderente à norma ISO/IEC 29110
baseado em Scrum e princípios Lean

São Paulo

2019

NILSON SALVETTI

PROÁGIL-29110: Processo ágil aderente à norma ISO/IEC 29110
baseado em Scrum e princípios Lean

Tese de doutorado apresentada ao
Programa de Pós-Graduação em
Informática e Gestão do Conhecimento da
Universidade Nove de Julho – UNINOVE,
como requisito parcial à obtenção do título
de Doutor em Informática e Gestão do
Conhecimento.

Orientador: Prof. Dr. IVANIR COSTA

São Paulo

2019

Dedico este trabalho ao meu filho,
Daniel Tura Salvetti, a principal
inspiração e a razão da transpiração
para o seu término.

AGRADECIMENTOS

Meus agradecimentos a todos que ajudaram na elaboração deste trabalho, direta ou indiretamente:

À Maria Fernanda Maymone e Kátia Regina Tura Salvetti por me fazerem acreditar que esse sonho pudesse se realizar;

À Sarah Kohan pela ajuda, incentivo e revisões em parte do trabalho;

Aos meus companheiros de coordenação pela paciência e a costumeira ajuda.

Aos professores da Uninove, e em especial ao Prof. Daniel Ferreira de Barros Jr, à Profa. Débora Virgília Canne e ao Prof. Gabriel Lara Baptista por abrirem as portas para que parte dessa pesquisa fosse realizada.

Ao Prof. Dr. Marcos Alberto Bussab, Diretor dos cursos de Informática da Universidade Nove de Julho (UNINOVE), pelo apoio e incentivo;

Aos membros da banca, pela participação e contribuições;

Ao meu orientador, Prof. Dr. Ivanir Costa pelas ideias, trocas de experiências e apoio continuo durante toda a trajetória.

RESUMO

As empresas contemporâneas produtoras de software, para serem competitivas em um mercado cada vez mais globalizado, precisam garantir que seus produtos sejam desenvolvidos e atualizados de acordo com as suas necessidades de negócios. Nessa realidade as áreas de desenvolvimento de software vêm enfrentado desafios quanto aos seus processos de trabalho. Nesse contexto, o objetivo desta pesquisa é desenvolver um processo ágil de desenvolvimento de software para pequenas empresas brasileiras, no intuito de garantir a melhoria da qualidade de seus produtos e incorporando a agilidade e flexibilidade no atendimento das necessidades das áreas de negócio. Esse processo ágil (PROÁGIL-29110) utiliza as práticas da norma ISO/IEC 29110 e do método ágil Scrum, incorporando o conjunto dos princípios da filosofia Lean Software. A opção pela norma ISO/IEC 29110 deve-se ao fato da sua aplicabilidade estar direcionada para microempresas, com times de desenvolvimento menores que 25 colaboradores. A norma também permite a obtenção de um certificado de qualidade em seus processos que diferencia essas organizações junto a seus clientes, no mercado nacional e internacional. A validação do processo PROÁGIL-29110 se deu por meio de uma pesquisa-ação em duas empresas e os seus resultados são apresentados para a comunidade acadêmica e para as pequenas empresas de software do mercado brasileiro que buscam a competitividade e sobrevivência em seus negócios.

Palavras-chave: Desenvolvimento de software, Scrum, Lean Software, ISO/IEC 29110, métodos ágeis.

ABSTRACT

Contemporary software companies, to be competitive in an increasingly globalized market, need to ensure that their products are developed and up-to-date according to their business needs. In this reality the areas of software development have faced challenges regarding their work processes. In this context, the objective of this research is to develop an agile software development process for small Brazilian companies, in order to guarantee the improvement of the quality of their products and incorporating agility and flexibility in meeting the needs of the business areas. This agile process (PROAGIL-29110) uses the practices of ISO / IEC 29110 and the agile Scrum method, incorporating the principles of the Lean Software philosophy. The choice of ISO / IEC 29110 standard is due to the fact that its applicability is directed to micro-enterprises, with development teams smaller than 25 employees. The standard also allows obtaining a certificate of quality in its processes that differentiates these organizations from their clients in the national and international markets. The validation of the PROÁGIL-29110 process was done through an action research in two companies and its results are divulged to the academic community and to the small software companies of the Brazilian market that seek the competitiveness and survival in their businesses.

Keywords: Software development, Scrum, Lean Software, ISO/IEC 29110, agile methods.

LISTA DE SIGLAS

ABES	Associação Brasileira das Empresas de Software
ABNT	Associação Brasileira de Normas Técnicas
BRASSCOM	Associação Brasileira de Empresas de Tecnologia da Informação e Comunicação
CAPES	Coordenação de Aperfeiçoamento de Pessoal de Nível Superior
CE	Critério de Exclusão
CI	Critério de Inclusão
CMM	<i>Capability Maturity Model</i>
CMMI	<i>Capability Maturity Model Integration</i>
IDC	<i>International Data Corporation</i>
IEC	<i>International Electrotechnical Commission</i>
IEEE	<i>Institute of Electrical and Electronics Engineers</i>
INPI	Instituto Nacional da Propriedade Industrial
ISO	<i>International Organization for Standardization</i>
MPE	Micro e Pequenas Empresas
MPS.Br	Melhoria do Processo de Software Brasileiro
OECD	<i>Organization for Economic Cooperation and Development</i>
PIB	Produto Interno Bruto
PM	<i>Project Management</i>
PME	Pequena e Médias Empresas
SI	<i>Software Implementation)</i>
QB	Questões Bibliométricas
QP	Questões de Pesquisa
RUP	Rational Unified Process
SciELO	<i>Scientific Electronic Library Online</i>
SEI	<i>Software Engineering Institute</i>
SLR	<i>Systematic Literature Review</i>

SOFTEX	Associação para Promoção da Excelência do Software Brasileiro
SPI	Software Process Improvement
TI	Tecnologia da Informação
VSE	<i>Very Small Entities</i>
XP	<i>Extreme Programming</i>
WIP	<i>Work in Progress</i>

LISTA DE FIGURAS

Figura 1: Comparação entre avaliações no MPS.Br e em outros modelos....	17
Figura 2: Método científico de pesquisa SLR.....	23
Figura 3: Evolução temporal das publicações sobre ISO/IEC 29110, Scrum e Lean Software.	31
Figura 4: Fontes de pesquisas contemplando a ISO/IEC 29110, Scrum e Lean Software	31
Figura 5: Os sete níveis de maturidade do MPS.Br	47
Figura 6: Interação entre os processos do Perfil Básico	49
Figura 7: Diagrama do Processo de Gerência do Projeto (PM)	50
Figura 8: Diagrama do Processo de Implementação de Software	55
Figura 9: Etapas do Scrum	66
Figura 10: Sucessão de iterações no Scrum	68
Figura 11: Exemplo de Kanban.....	82
Figura 12: Processo simplificado da pesquisa-ação	89
Figura 13: Processos das ciências naturais e da pesquisa-ação.....	90
Figura 14: Fases do ciclo da Pesquisa-ação	91
Figura 15: Modelo Proposto.....	95
Figura 16: Gráfico burndown do primeiro Sprint	118
Figura 17: Criação do PROÁGIL-29110 no Azure DevOps	122
Figura 18: Plano do Projeto na ferramenta Azure DevOps.....	125
Figura 19: Work Item da Matriz de Risco no Azure DevOps.....	126
Figura 20: Requisitos do sistema na forma de "Features"	127
Figura 21: "Feature" x User Story x Sprint 1	128
Figura 22: Kanban e distribuição de tarefas	130
Figura 23: Burndown do Sprint 1	131
Figura 24: Repositório GIT.....	133

LISTA DE QUADROS

Quadro 1: Diferenças das características entre pequenas e grandes empresas	15
Quadro 2: Critérios de inclusão / exclusão de trabalhos	26
Quadro 3: Relação de Consultas realizadas x resultados obtidos	27
Quadro 4: Relação do número de estudos x base científica	28
Quadro 5: Relacionamento entre objetivos e atividades (PM)	53
Quadro 6: Comparativo entre objetivos e atividades (SI)	59
Quadro 7: Práticas Scrum do Guia ScrumSTUDY	72
Quadro 8: Comparação Scrum x Lean	79
Quadro 9: Ferramentas Lean	83
Quadro 10: Resumo sinóptico dos temas abordados no referencial teórico para aplicação na pesquisa	84
Quadro 11: Alterações no Processo PROÁGIL-29110	97
Quadro 12: Relacionamento entre ISO/IEC 29110 x Scrum X Lean	101
Quadro 13: Análise dos itens da cerimônia / artefatos Scrum e princípios Lean	120
Quadro 14: Análise dos itens da cerimônia / artefatos Scrum e princípios Lean	134
Quadro 15: Quadro comparativo antes e após a implantação do PROÁGIL-29110 nas duas empresas	137
Quadro 16: Papel x Responsabilidade	191
Quadro 17: Fase 1 – Gerência de Projetos	194
Quadro 18: Fase 2 – Implementação do Software	198

Sumário

1.	INTRODUÇÃO.....	11
1.1.	Contextualização.....	11
1.2.	Objetivos	14
1.1.1.	Objetivo Geral	14
1.1.2.	Objetivos Específicos.....	14
1.3.	Justificativa.....	14
1.4.	Delimitação do tema	19
1.5.	Estrutura da tese.....	20
2.	Revisão Bibliométrica e Sistemática sobre a ISO/IEC 29110, Scrum e Lean Software	21
2.1	Método Científico de Pesquisa Systematic Literature Review (SLR) .	21
2.2	Fase 1 – Plano da Revisão	23
2.3	Fase 2: Condução da Revisão	27
2.4	Fase 3: Revisão da documentação	30
3.	Fundamentação Teórica.....	38
3.1	Processo de Software	38
3.2	Modelos de Qualidade de Software.....	41
3.2.1	Modelo CMMI	43
3.2.2	Modelo MPS.Br para Software	46
3.3	A Norma ISO/IEC 29110	48
3.4	Método Ágil Scrum	61
3.5	Lean Software	72
3.6	Estruturação dos temas abordados no referencial teórico para aplicação na pesquisa	83
4.	PESQUISA-AÇÃO	87
4.1	Método Pesquisa–Ação.....	87

5.	Modelo e Processo PROÁGIL-29110 proposto	95
5.1	Validacao do PROÁGIL-29110.....	96
6.	ANALISE DE DADOS E DISCUSSÃO DOS RESULTADOS	114
7.	CONCLUSÕES.....	136
7.1	Contribuições.....	138
7.2	Limitações da pesquisa	139
7.3	Trabalhos futuros.....	139
	GLOSSÁRIO.....	140
	Referências.....	141
	APÊNDICE A – ESTUDOS IDENTIFICADOS	159
	APÊNDICE B – ESTUDOS X CANAL DE PUBLICAÇÃO.....	166
	APÊNDICE C – FORMATAÇÃO DE TEMPLATES DO PROÁGIL-29110 ...	168
	APÊNDICE D – TEMPLATE DE SPRINT BACKLOG	169
	APÊNDICE E – TEMPLATE DE PRODUCT BACKOLOG.....	170
	APÊNDICE F – TEMPLATE DE CRONOGRAMA	172
	APÊNDICE G – TEMPLATE DE VERIFICAÇÃO.....	173
	APÊNDICE H – TEMPLATE DE REGISTRO DE REUNIÃO	174
	APÊNDICE I – TEMPLATE DE MATRIZ DE RISCOS	175
	APÊNDICE I – TEMPLATE DE MATRIZ DE RISCOS (ABA 2)	176
	APÊNDICE J – TEMPLATE DE LIÇÕES APRENDIDAS.....	177
	APÊNDICE K – TEMPLATE DO PLANO DE PROJETO	178
	APÊNDICE L – TEMPLATE DO PLANO DE TESTES	180
	APÊNDICE M – TEMPLATE DO TERMO DE ACEITE.....	182
	APÊNDICE N – PROCESSO PROÁGIL-29110.....	183

1. INTRODUÇÃO

Neste capítulo é apresentada a contextualização, objetivos, justificativa, a delimitação do tema e estrutura do trabalho.

1.1.CONTEXTUALIZAÇÃO

As empresas contemporâneas que produzem software têm a possibilidade de usarem e adaptarem processos que apoiem o desenvolvimento de Sistemas de Informação. Ao longo dos últimos 40 anos foram propostas e utilizadas metodologias que dão suporte aos processos de desenvolvimento de software, desde os modelos dirigidos por planejamento, até o surgimento dos métodos ágeis, como o Scrum em 1986.

Os modelos de processo ou metodologias usadas em organizações de desenvolvimento de software evoluíram ao longo do tempo. Como consequência dessa evolução, nos últimos anos essas organizações têm considerado (com mais interesse) a adoção de práticas ágeis no desenvolvimento de software (DAHLEM,2014; DINGSOYR, 2012). Essa abordagem ágil promove uma forma fácil e rápida de desenvolvimento de software por meio da qual iterações curtas estão agendadas para satisfazer clientes com entregas parciais do produto (DAHLEM *et al.* 2014; COCKBURN, 2002; e BOEHM, 2003).

Os métodos ágeis caracterizam-se na divisão do desenvolvimento de um sistema em ciclos curtos denominados Sprints. Eles sugerem um planejamento forte para os Sprints e não para o projeto como um todo, características dos modelos clássicos. O desenvolvimento denominado de ágil se baseia na premissa de que o cliente e o time aprendem ao longo do desenvolvimento, à medida que são capazes de manipular/usar as partes do sistema que forem sendo entregues ao longo do tempo. Dessa forma, o cliente percebe os detalhes, compreende as dificuldades, visualiza novas possibilidades e, conseqüentemente, solicita novas alterações para que o software se aproxime, ao máximo, da solução que ele acredita que resolverá os seus problemas (TELES, 2006).

Mais recentemente, os métodos ágeis e por consequência o Scrum vêm incorporando os princípios Lean ao desenvolvimento de software. Em geral, o Lean é uma proposta de manufatura e das práticas de produção que considera o gasto de recursos para qualquer objetivo que não seja a criação de valor para os clientes finais, um desperdício e, portanto, um alvo para a eliminação. Segundo Cruz (2018), os métodos ágeis pregam o uso de pequenos times ou equipes autônomas com projetos com baixa formalidade e com documentação reduzida.

A utilização dos princípios Lean nos métodos ágeis se mostra como um fator competitivo, como visto em um estudo de caso elaborado por Parnell-Klabo (2006), que apresenta a “Capital One”, uma grande empresa do setor financeiro que, por necessidade de diminuir os seus custos e aumentar a competitividade no mercado, optou pelo Lean Ágil no desenvolvimento de seus produtos de software.

A adoção de métodos de garantia de qualidade baseados em modelos de melhoria de processos de software foi considerada como uma importante fonte de variabilidade na produtividade do processo de desenvolvimento de software. Algumas empresas percebem que sua implementação tem custos proibitivos, ao passo que identificam em seu uso uma maneira de se adequar aos padrões de desenvolvimento de software, produzem valor econômico e levam à melhoria do desempenho corporativo. As empresas desenvolvedoras de software precisam ser mais produtivas, aumentar a qualidade e diminuir o esforço e os custos, com o objetivo de manter o sucesso do negócio (DUARTE, 2017).

Algumas organizações do Brasil e no exterior se apoiam, quanto à qualidade de seu processo e produtos de desenvolvimento, em um modelo de qualidade concebido pelo *Software Engineering Institute* (SEI), chamado *Capability Maturity Model Integratrion* (CMMI). Esse modelo permite que uma empresa seja certificada em relação à sua maturidade no processo de desenvolvimento de software, sendo que nas bases de dados do Instituto estão cadastradas 16499 organizações certificadas no período de janeiro de 2007 a julho de 2017. Deste montante, 76 são brasileiras (SEI, 2017).

Já para as empresas brasileiras surge uma outra alternativa que é o modelo de qualidade MPS-Br, cuja concepção envolveu o Ministério da Ciência e Tecnologia, por meio de um órgão chamado Associação para Promoção da Excelência do Software

Brasileiro (SOFTEX) em 2005. Nas bases de dados do site da SOFTEX tem-se atualmente 697 empresas certificadas desde a criação do modelo (SOFTEX, 2018).

Ainda na área de qualidade de software, foi elaborada a norma ISO/IEC 29110 para micro organizações, chamadas de *Very Small Entities* (VSE). Uma VSE é uma entidade (empresa, organização, departamento ou projeto) com até 25 pessoas. Essa norma foi colocada à disposição da coletividade acadêmica e empresarial em 2012 (ABNT, 2012). A norma pode ser utilizada com qualquer processo, técnica ou método no sentido de aumento da produtividade, melhoria da qualidade e, conseqüentemente, maior satisfação do cliente. Também pode ser usada como um guia de engenharia e gestão. A norma é consistente com ciclos de vida de desenvolvimento de sistemas de informação consagrados da Engenharia de Software (ABNT, 2012).

A falta de um processo organizado acaba gerando problemas em planejamento e implementação de produtos de software, e que acabam apresentando defeitos quando são usados, afetando os usuários, os custos e, conseqüentemente, sua qualidade. Por outro lado, a pressão do mercado leva as empresas de pequeno porte a desenvolverem os sistemas de informação com rapidez, visto que a maioria dos projetos deve ser realizado em curto prazo. Esse é um desafio enfrentado pelas áreas de desenvolvimento de software, principalmente pelas pequenas empresas quanto aos seus processos de produção.

Por outro lado, a ISO/IEC 29110 se apresenta como uma norma aplicável à pequenas empresas desenvolvedoras de software. De acordo com Omran (2008), as práticas e o processo de adoção de um modelo ou norma representa uma mudança em pequenas organizações, razão pela qual a agilidade se torna um fator importante na escolha da implementação.

Nesse contexto, surge o problema fundamental e a questão de pesquisa que norteia esse trabalho:

Como é possível desenvolver, implementar e validar um processo ágil aderente à norma ISO/IEC 29110, incorporando as práticas ágeis do Scrum e princípios da filosofia Lean?

1.2.OBJETIVOS

Os objetivos dessa pesquisa são:

1.1.1. Objetivo Geral

O objetivo geral desta tese é desenvolver, implementar e validar um processo ágil (PROÁGIL-29110) aderente à norma ISO/IEC 29110, incorporando as práticas ágeis Scrum e os princípios da filosofia Lean adaptados ao desenvolvimento de software.

1.1.2. Objetivos Específicos

Em complemento, os seguintes objetivos específicos são estabelecidos:

- ✓ Desenvolver o processo PROÁGIL–29110, combinando as práticas ágeis do método Scrum e filosofia Lean com os processos da norma ISO/IEC 29110;
- ✓ Validar o processo PROÁGIL–29110 envolvendo especialistas em ISO/IEC 29110, Scrum e filosofia Lean;
- ✓ Aprimorar o processo PROÁGIL–29110, considerando a implementação em duas empresas do tipo VSE desenvolvedoras de software;

1.3.JUSTIFICATIVA

O mercado brasileiro de Tecnologia da Informação (TI), incluindo hardware, software, serviços e exportações de TI, movimentou 39,5 bilhões de dólares em 2017, representando 1,9% do Produto Interno Bruto (PIB) brasileiro e 1,8% do total de investimentos de TI no mundo, um resultado sensivelmente inferior às participações apontadas no ano anterior. Considerando-se apenas o mercado interno, sem considerar a exportação da ordem de 1 bilhão de dólares, o mercado total de TI foi da ordem de 38,5 bilhões de dólares. Deste valor, 8,183 bilhões vieram do mercado de

software e 10,426 bilhões do mercado de serviços, sendo que a soma destes dois segmentos representou 48,8% do mercado total de TI, consolidando a tendência de passagem do país para o grupo de economias com maior grau de maturidade no mundo, que privilegiam o desenvolvimento de soluções e sistemas. O setor de software teve um crescimento de 2,8% sobre 2016 (ABESSOFTWARE, 2018).

O estudo apontou cerca de 17.000 empresas dedicadas ao desenvolvimento, produção, distribuição de software e de prestação de serviços no mercado nacional, sendo que aproximadamente 61,7% delas possui atividade principal voltada para o desenvolvimento e produção de software ou a prestação de serviços. Considerando-se apenas as 5.138 empresas que atuam no desenvolvimento e produção de software, cerca de 95,5% podem ser classificadas como micro e pequenas empresas, segundo análise realizada pelo critério de número de funcionários (até 99 funcionários) (ABESSOFTWARE, 2018). Ainda considerando as 5.138 empresas, esse mesmo estudo mostra que 49,3% delas são consideradas microempresas (possuem menos do que 10 funcionários), enquanto 46,2% são consideradas pequenas (de 10 a 99 funcionários) (ABESSOFTWARE, 2018).

Como o foco dessa tese volta-se às pequenas empresas de software brasileiras, torna-se importante apresentar as características que diferenciam as micro/pequenas empresas e grandes empresas. Essas características são apresentadas no Quadro 1.

Quadro 1: Diferenças das características entre pequenas e grandes empresas

(Característica)	(Micros/Pequenas Empresas)	(Grandes empresas)
Orientação em relação ao planejamento	Não estruturadas / operacional	Estruturadas / estratégicas
Flexibilidade	Alta	Estruturadas / estratégicas
Processo gerenciado	Informal	Baixo
Susceptibilidade ao risco	Alta	Médio
Capacidade de absorver aprendizado e conhecimento	Limitado	Alto
Impacto de efeitos negativos em relação ao marketing	Muito profundo	Mais gerenciável
Vantagem competitiva	Centrado no capital humano	Centrado no capital organizacional

Fonte: Adaptado de MTIGWE (2005)

Como mostrado no Quadro 1, na coluna característica “Orientação em relação ao planejamento”, as micro e pequenas empresas (MPEs) não são estruturadas e na característica “processo gerenciado”, o mesmo é dito do tipo informal. Nesse sentido implementar um processo mais simples e objetivo em pequenas empresas pode ajudá-las em relação à qualidade dos seus produtos e, conseqüentemente no relacionamento com os clientes (THIRY, 2006).

Ainda em relação ao Quadro 1 a característica “impactos de efeitos negativos em relação ao marketing”, mostra que as micro e pequenas empresas apresentam um impacto “muito profundo”. Dessa forma, pode-se entender, que produzir sistemas com falhas prejudica a imagem da organização, pois a característica “vantagem competitiva” indica que as micro e pequenas empresas são “centrado no capital humano”.

Um estudo comparativo entre avaliações MPS.Br (Melhoria de Processo de Software Brasileiro) e outros modelos é mostrado na Figura 1. A Figura 1 mostra que nos anos de 2013, 2014 e 2015, utilizando como parâmetro o grupo de países do G-19 (19 maiores economias do mundo), o MPS.BR¹ contribuiu para que o Brasil se destacasse por três anos consecutivos em 4º lugar em avaliações em qualidade de processos de software no mundo, sendo que a China, EUA e Índia, ocuparam respectivamente 1º, 2º e 3º lugares (SOFTEX, 2018).

Considerando somente avaliações CMMI, MoProSoft (Modelo de processos desenvolvido no México) e IT Mark (Esquema de certificação espanhol desenvolvido para pequenas e médias empresas de TI), o Brasil ocupou os 6º, 7º e 9º. Em relação aos países que não estão no G-19, Colômbia, Espanha e México, pode-se destacar México e Espanha, onde o México ocupou o 4º lugar nos três anos e a Espanha ocupou 5º, 6º e 5º, respectivamente (SOFTEX, 2018).

¹ MPS.Br – Modelo Brasileiro de qualidade de software desenvolvido pela SOFTEX

Figura 1: Comparação entre avaliações no MPS.Br e em outros modelos

1º	CHN	576	CHN	576	CHN	716	CHN	716	CHN	884	CHN	884
2º	EUA	287	EUA	287	EUA	289	EUA	289	EUA	336	EUA	366
3º	IND	170	IND	170	IND	158	IND	168	IND	177	IND	177
4º	MEX	90	BRA 146	MEX	99	BRA 150	MEX	100	BRA 105			
5º	ESP	40	MEX	90	COL	59	MEX	95	COL	66	MEX	100
6º	BRA 37		ESP	40	BRA 54		COL	64	ESP	44	COL	66
7º	KOR	32	KOR	32	KOR	47	KOR	47	KOR	42	ESP	44
8º	JPN	25	JPN	25	ESP	41	ESP	41	JPN	27	KOR	42
9º	FRA	19	FRA	19	JPN	25	JPN	25	BRA 24		JPN	27
10º	COL	15	COL	15	FRA	14	FRA	14	VGB	13	VGB	13
11º	TUR	12	TUR	12	TUR	12	TUR	12	CAN	13	CAN	13
12º	ARG	11	ARG	11	GER	10	GER	10	FRA	11	FRA	11
13º	GER	11	GER	11	ITA	10	ITA	10	TUR	10	TUR	10
14º	ITA	8	ITA	8	VGB	10	VGB	10	GER	7	GER	7
15º	CAN	7	CAN	7	ARG	7	ARG	8	ITA	7	ITA	7
16º	VGB	5	VGB	5	CAN	7	CAN	7	ARG	7	ARG	7
17º	AUS	3	AUS	3	AUS	4	AUS	4	SAL	3	SAL	3
18º	RUS	3	RUS	3	IDN	4	IDN	4	IDN	1	IDN	1
19º	ZAF	2	ZAF	2	ZAF	1	ZAF	1	ZAF	1	ZAF	1
20º	SAL	1	SAL	1	RUS	1	RU	1	RUS	1	RUS	1
21º	IDN	1	IDN	1	SAL	1	SAL	1	AUS	0	AUS	0
	2013				2014				2015			
	AVALIAÇÃO EM OUTROS MODELOS				AVALIAÇÃO EM OUTROS MODELOS + MPS.BR							

Fonte: SOFTEX (2018)

Por outro lado, há uma norma criada e denominada (ISO/IEC 29110), direcionada para implementação em pequenas empresas que possuem menos de 25 colaboradores, as chamadas VSE's (*Very Small Entities*).

Visando uma contribuição para a teoria e a prática, com foco nas micro e pequenas empresas de software brasileiras, esta pesquisa proporcionará uma opção do uso de um processo ágil, aderente à ISO/IEC 29110, baseado em métodos ágeis e princípios da filosofia Lean adaptados a software. Esse processo poderá agregar valor à área de desenvolvimento dessas pequenas empresas, tanto internamente, como para o mercado consumidor de soluções de software.

A opção pela norma ISO/IEC 29110, publicada em 2012, baseia-se no fato que ela é recente no mercado e em meios acadêmicos e sua aplicação no Brasil é praticamente desconhecida. Estudos fora do Brasil mostram uma tendência no uso dessa norma, conforme relata Gordon (2016) em um estudo realizado com 30 empresas (VSE's) no Equador.

Há mais de uma década Boehm (2003) afirmava que a utilização de métodos ágeis para o desenvolvimento de software despertava o interesse, tanto da Academia, quanto na indústria. Esses métodos foram desenvolvidos visando a entrega rápida do software, na contínua interação entre desenvolvedores e clientes, a tentativa de aumentar a possibilidade de que um software atenderá as necessidades e as mudanças de requisitos dos clientes. Prazos de entregas curtos, somando-se à complexidade dos processos de desenvolvimento existentes, foram a principal motivação para que surgissem os métodos ágeis (COCKBURN *et al.*, 2002).

Entre os métodos ágeis existentes, o Scrum incorporado com as práticas da filosofia Lean pode fornecer às pequenas empresas um diferencial competitivo, pois de acordo com os autores O'Connor e Coleman (2007), empresas de software precisam ser flexíveis, criativas, dinâmicas e capazes de entregar produtos de qualidade ao mercado rapidamente por motivos de sobrevivência.

Uma outra contribuição dessa tese é que o processo a ser concebido poderá fornecer à empresa a possibilidade de uma futura certificação, fato que a tornará mais competitiva no mercado. Como benefícios a organização teria a formalização de um conjunto de requisitos necessários para projetos e produtos, além de um processo de

gerenciamento ágil em se tratando de projetos e um processo de implementação de software sistematizado.

As pesquisas realizadas nas bases científicas Scopus, IEEE Xplore, Elsevier ScienceDirect, ACM Digital Library, cujos trabalhos estão apresentados no item de revisão bibliométrica, não encontraram nenhum trabalho que tratasse especificamente do tema envolvendo o desenvolvimento de um processo ágil, contemplando os constructos ISO/IEC 29110, Scrum e Lean Software. Embora existam trabalhos que apresentam todos os constructos, eles não são utilizados para nenhuma solução combinada, mas são aplicados de forma independente, ou em combinações do tipo Scrum e Lean Software.

Considerando-se o contexto apresentado, essa pesquisa abordando o uso combinado dos constructos ISO/IEC 29110, Scrum e Lean Software se mostra relevante, principalmente para que as pequenas empresas brasileiras consigam produzir produtos de alta qualidade e se tornarem competitivas em um mercado extremamente globalizado e competitivo. Justifica-se a tese uma vez que o processo proposto expande o conhecimento do tema e sua aplicabilidade, tanto no meio acadêmico, como para o mercado empresarial. Nesse sentido, a pesquisa científica vislumbrada nessa tese pode ser entendida como um procedimento reflexivo sistemático, controlado e crítico, que permite descobrir novos fatos ou dados, relações ou leis. (LAKATOS, 2006).

1.4. DELIMITAÇÃO DO TEMA

O tema proposto contempla empresas de pequeno porte, preconizadas pela norma ISO/IEC 29110, ou seja, empresas de TI que possuem até 25 colaboradores. Não serão abordados nesta pesquisa os ciclos de vida utilizados em desenvolvimento de sistemas tradicionais, tais como: Cascata, Incremental, Espiral e o Rational Unified Process (RUP). Também não serão abordadas outras normas e modelos de qualidade de desenvolvimento de software, tais como a ISO 12207 e a ISO 15504, pela falta de implementação dessas normas em empresas brasileiras. Em relação aos métodos ágeis, não estão contemplados o *Feature Driven Development* (FDD), *eXtreme*

Programming (XP), *Microsoft Solutions Framework (MSF)* e o *Dynamic System Development Model (DSDM)*, por conta da atual escolha das empresas, pelo método ágil Scrum no desenvolvimento de projetos.

1.5. ESTRUTURA DA TESE

Essa tese está organizada em sete capítulos.

No primeiro capítulo inicia-se com a introdução do tema, com a contextualização do problema, a apresentação da questão de pesquisa e os subsequentes objetivos, geral e específicos, seguido pela justificativa para a realização da pesquisa.

Segue-se o segundo capítulo dedicado a explicitar a revisão bibliométrica e sistemática desse estudo.

O terceiro capítulo apresenta a fundamentação teórica da tese.

O quarto capítulo trata dos aspectos metodológicos considerados na realização da pesquisa.

O quinto capítulo traz o modelo e processo proposto, bem como lista as hipóteses formuladas a partir da observação dos posicionamentos teóricos examinados na análise de discussões teóricas sobre o tema.

O sexto capítulo expressa os aspectos associados com a análise dos dados capturados na pesquisa e discussão dos resultados encontrados frente à literatura existente.

O sétimo capítulo expõe as conclusões do estudo e trabalhos futuros.

Finalmente a tese se encerra com as referências bibliográficas que consultadas para a sustentação teórica da pesquisa e os apêndices, contemplando o processo desenvolvido e os templates utilizados.

2. REVISÃO BIBLIOMÉTRICA E SISTEMÁTICA SOBRE A ISO/IEC 29110, SCRUM E LEAN SOFTWARE

Neste capítulo é apresentado o resultado da revisão da literatura em trabalhos e pesquisas encontradas sobre o assunto. Essa revisão tem como objetivo fornecer uma visão dos principais conceitos abordados na literatura relacionados aos tópicos de pesquisa. Essas informações contribuem para a localização do presente estudo no contexto do que vem sendo discutido na literatura referente à área e também como esses conteúdos contribuem para o desenvolvimento da pesquisa.

Segundo Santos e Kobashi (2009), a revisão bibliométrica reúne informações importantes sobre determinado assunto em publicações dentro das bases de conhecimento científico. Para realizar a revisão bibliométrica desta tese foi aplicado o método de pesquisa *Systematic Literature Review* (SLR) apresentado no item 2.1.

2.1 MÉTODO CIENTÍFICO DE PESQUISA SYSTEMATIC LITERATURE REVIEW (SLR)

A SLR é um meio de avaliar e interpretar toda a pesquisa disponível relevante para uma questão de pesquisa específica, área temática ou fenômeno de interesse. As revisões sistemáticas têm como objetivo apresentar uma avaliação justa de um tópico de pesquisa usando uma metodologia confiável, rigorosa e auditável (KITCHENHAM, CHARTERS, 2007).

As diretrizes que apoiam o método científico SLR foram estabelecidas por Kitchenham e Charters (2007) e essa é a versão corrente. O documento original foi publicado em 2004 e revisado por membros do *Evidence-Based Software Engineering* (EBSE) que foi financiado pelo Conselho de Pesquisa de Economia e Ciências Físicas do Reino Unido (Keele Universit).

O objetivo original para empregar o método SLR foi apoiar a medicina baseada em evidências. Foi estabelecido um “*Guideline*” para uma melhor compreensão da utilização do método que sejam apropriadas às necessidades dos pesquisadores de

Engenharia de Software. Ele discute várias questões em que a pesquisa de engenharia de software difere da pesquisa médica (EBSE Technical Report, 2007).

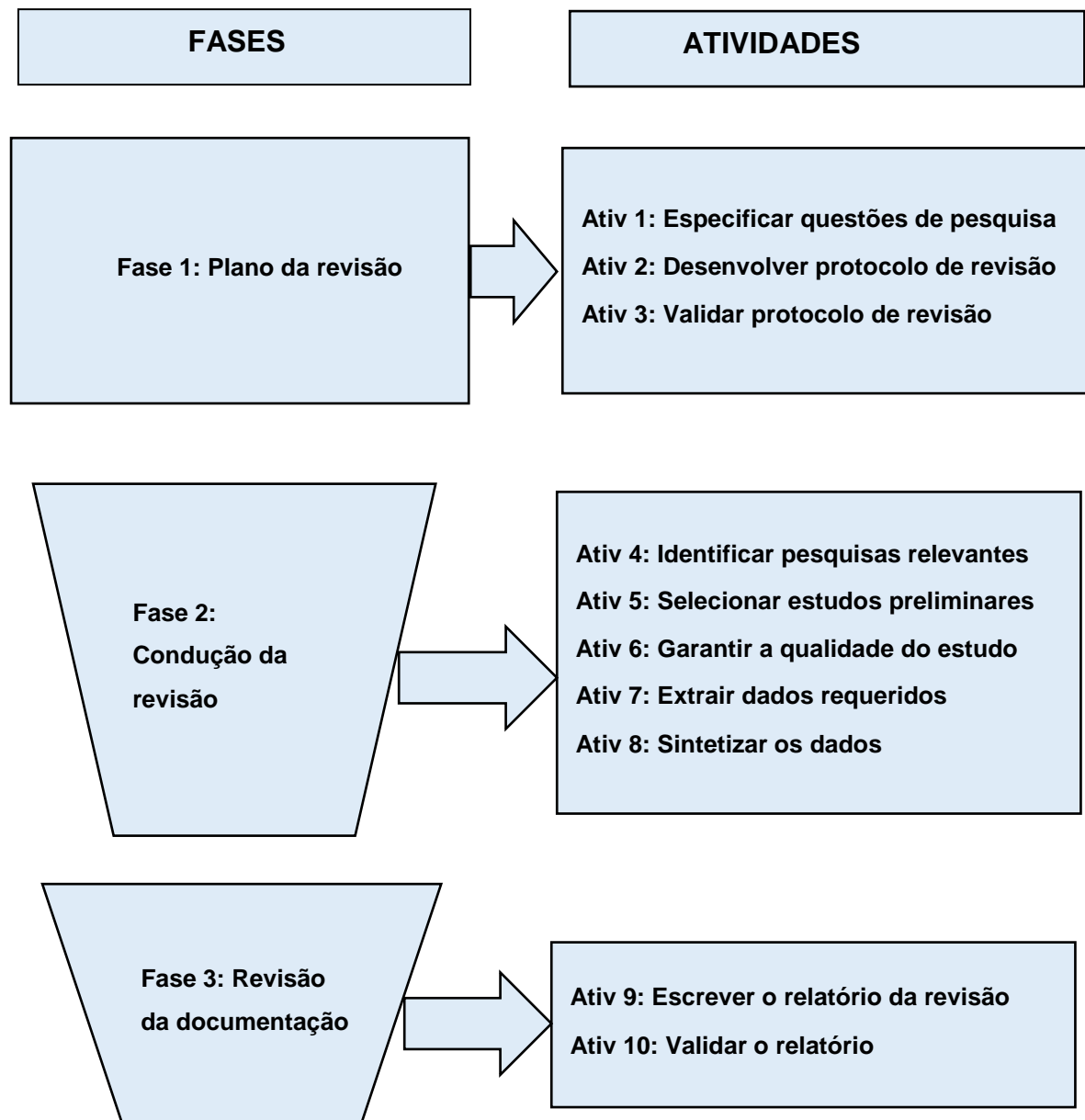
Segundo Kitchenham e Charters (2007), existem muitas razões para realizar uma revisão sistemática da literatura. As razões mais comuns são:

- Resumir as evidências existentes relativas a um tratamento ou tecnologia, por ex. resumir as evidências empíricas dos benefícios e limitações de um método ágil específico;
- Identificar quaisquer lacunas na pesquisa atual, a fim de sugerir áreas para investigação adicional;
- Fornecer um enquadramento / contexto para posicionar adequadamente novas atividades de investigação.

Para atender as necessidades de conhecimento sobre o assunto da Tese, a pesquisa é baseada em três constructos: ISO/IEC 29110, Scrum e Lean Software. Esses constructos foram utilizados como palavras-chave para a realização do método SLR ou por meio dos bancos de dados digitais (KITCHENHAM, CHARTERS, 2007).

O método de pesquisa SLR pode ser representado pela Figura 2.

Figura 2: Método científico de pesquisa SLR



Fonte: Adaptado de Brereton (2006)

2.2 FASE 1 – PLANO DA REVISÃO

Em relação a fase 1: “Plano de Revisão”, Kitchenham e Charters (2007) afirmam que a atividade 1: “Especificar questões de pesquisa” é a parte mais importante de qualquer revisão sistemática. As questões de pesquisa conduzem toda

a metodologia de revisão sistemática, ou seja, a busca deve identificar os estudos que abordam as questões de pesquisa; a extração de dados deve extrair os itens de dados necessários para responder às perguntas e finalmente a análise de dados deve sintetizar os dados de tal forma que as perguntas possam ser respondidas.

Segundo os mesmos autores, o importante em qualquer revisão sistemática é fazer a pergunta certa. Neste contexto, a pergunta certa é geralmente aquela que:

- a) É significativa e importante para os profissionais, bem como para os pesquisadores. Por exemplo, os pesquisadores podem estar interessados em saber se uma técnica de análise específica leva a uma estimativa significativamente mais precisa dos defeitos remanescentes após as inspeções de projeto. No entanto, um profissional pode querer saber se a adoção de uma técnica de análise específica para prever defeitos remanescentes é mais eficaz do que a opinião de especialistas na identificação de documentos de projeto que exigem inspeção.
- b) Levará a mudanças nas práticas atuais de engenharia de software ou aumentará a confiança no valor da prática atual. Por exemplo, pesquisadores e profissionais gostariam de saber em que condições um projeto pode adotar com segurança métodos ágeis e sob quais condições ele não deveria.

No entanto, existem revisões sistemáticas que fazem perguntas que são de interesse primordial para os pesquisadores. Tais revisões fazem perguntas que identificam e/ou determinam futuras atividades de pesquisa. Por exemplo, uma revisão sistemática em uma tese de doutorado deve identificar a base existente para o trabalho do aluno de pesquisa e deixar claro de que forma a pesquisa proposta se encaixa no corpo atual do conhecimento (KITCHENHAM, CHARTERS, 2007).

Na visão de Pritchard (1969), Sancho (2002) e Lopes *et. al.* (2012) a bibliometria possui as seguintes características:

- a) Apontar os estudos em crescimento;
- b) Identificar a redução de determinados temas científicos;
- c) Medir os impactos das revistas científicas;

- d) Verificar as instituições de ensino e autores que mais possuem produção acadêmica.

Em conformidade com o exposto, foram determinadas pelo pesquisador, a partir da literatura, três Questões Bibliométricas (QB) e três Questões de Pesquisa (QP), (BRERETON *et al.*, 2006; FELDERER, 2014; e WIKLUND, 2017).

As questões bibliométricas (QB) foram preparadas pelo pesquisador tendo como fonte os autores (Brereton *et al.*, 2006; Felderer, 2014 e Wiklund, 2017):

- I. QB-1: Como foi a evolução do número de artigos publicados/ano, relacionados com o tema da pesquisa (constructos)?
- II. QB-2: Quais as fontes de pesquisas utilizadas (Workshop, dissertação/tese, revista, biblioteca online, livro, jornal, anais e periódicos científicos)?
- III. QB-3: Quais são as bibliotecas digitais de conferências e periódicos científicos, com mais publicações relacionadas ao tema da pesquisa?

As questões de pesquisa (QP) formuladas para o tema da tese foram determinadas pelo pesquisador, que considerou os constructos da tese:

- I. QP-1: De que forma o Scrum pode ser usado em combinação com a ISO/IEC 29110?
- II. QP-2: Há vantagens ou desvantagens da utilização de práticas ágeis em conjunto com a ISO/IEC 29110?
- III. QP-3: Existem alguns estudos primários relacionados com o uso combinado de práticas ágeis do método Scrum, filosofia Lean Software e a norma ISO/IEC 29110?

Em relação à atividade 2: “Desenvolver protocolo de revisão”, Kitchenham e Charters (2007) afirmam que um protocolo de revisão deve conter os métodos que serão usados para realizar uma revisão sistemática específica. Um protocolo pré-definido é necessário para reduzir a possibilidade de viés do pesquisador.

Os componentes de um protocolo podem incluir os elementos da revisão, além de algumas informações adicionais de planejamento. A definição dos critérios de seleção é realizada para determinar quais estudos estão incluídos ou excluídos de

uma revisão sistemática. Geralmente é útil conduzir os critérios de seleção em um subconjunto de estudos primários.

Os critérios de inclusão (CI) e exclusão (CE) considerados nesta pesquisa estão listados no Quadro 2.

Quadro 2: Critérios de inclusão / exclusão de trabalhos

Critério		Descrição
Inclusão	CI 1	Possuir base metodológica: experiência, estudo de caso, revisões sistemáticas, mapeamentos sistemáticos
	CI 2	Ter no título ou palavras chaves ou resumo os constructos ISO/IEC 29110, ou Scrum ou Lean Software
Exclusão	CE 1	Estudo duplicado
	CE 2	Estudo não contempla a área de Qualidade de Software

Para a atividade 3: “Validar protocolo de revisão”, Kitchenham e Charters (2007) afirmam que as questões básicas de revisão SLR discutidas na atividade 1 “Especificar questões de pesquisa”, podem ser adaptadas para auxiliar a avaliação de um protocolo de revisão sistemática. Além disso, a consistência interna do protocolo pode ser verificada para confirmar que:

- Os resultados da pesquisa são apropriadamente derivados das questões de pesquisa.
- Os dados a serem extraídos abordarão adequadamente as questões de pesquisa.
- O procedimento de análise de dados é apropriado para responder às questões de pesquisa

2.3 FASE 2: CONDUÇÃO DA REVISÃO

Para a atividade 4: “Identificar pesquisas relevantes”, as fontes de dados para busca dos embasamentos acadêmicos e científicos utilizadas foram as bases digitais: IEEE Xplore, Elsevier (Scopus + ScienceDirect), ProQuest, ACM Digital Library.

Essas fontes foram complementadas por uma busca nos principais repositórios e conferências relacionadas com métodos ágeis e a norma ISO/IEC 29110. A razão dessa busca adicional se justifica devido a haver estudos e pesquisas que discutem e combinam o método ágil Scrum e a norma ISO/IEC 29110 que não foram publicados em base de dados científicas, mas que agregam valor para esta pesquisa. Em seguida, os artigos foram analisados considerando o título do artigo, resumo, introdução e conclusão. Finalmente foram considerados os artigos publicados e que passaram pela revisão por pares.

Em relação a atividade 5: “Selecionar os estudos preliminares”, esse processo começou com a busca em novembro de 2017. Em maio de 2018 foi realizada a última execução. Usando um editor de planilhas, os títulos, resumos e referências foram selecionados após a execução da pesquisa em fontes digitais. A pesquisa foi realizada com base nos constructos desta tese, ou seja:

- a) Constructo 1, C1: “ISO/IEC 29110” ou “ISO 29110”;
- b) Constructo 2, C2: “Scrum”;
- c) Constructo 3, C3: “Lean Software”.

Foram realizados quatro tipos de consultas, listadas no Quadro 3.

Quadro 3: Relação de Consultas realizadas x resultados obtidos

Consulta	String	Resultados
Consulta 1	C1	50
Consulta 2	C1 and C2	10
Consulta 3	C1 and C3	5
Consulta 4	C1 and C2 and C3	3
Total de resultados da pesquisa		68

Todas as consultas envolveram o Constructo C1: ISO/IEC 29110, considerado o pilar desta tese, razão pela qual não foi realizada a pesquisa envolvendo o Constructo 2 em par com o Constructo C3. A busca também foi realizada utilizando-se o termo ISO 29110, pois alguns trabalhos suprimem o termo “IEC”. Após esta etapa, foram identificados um total de 68 estudos potenciais.

Os resultados contendo os 68 estudos são exibidos no Quadro 4.

Quadro 4: Relação do número de estudos x base científica

Base de dados	Número de estudos	Data da pesquisa
IEEE Xplore	6	09/08/2018
ACM Digital Library	2	09/08/2018
Elsevier (Science Direct + Scopus)	10	09/08/2018
Journal / Online library	33	09/08/2018
ProQuest	17	15/08/2018
Total de estudos	68	

Aplicando-se os critérios de exclusão CE1 e CE2, obtêm-se um total de 44 estudos.

Por outro lado, a última busca foi realizada em agosto de 2018 a fim de obter estudos relevantes em outros repositórios, além dos 5 relacionados na “Base de dados” do Quadro 4. Na primeira inserção de dados (usando-se os constructos C1 + C2 + C3) obteve-se 56 estudos como retorno. Aplicando-se os critérios de avaliação da qualidade descritos na atividade 6: “Garantira a qualidade do estudo”, 28 estudos foram definidos como relevantes neste tipo de pesquisa.

Os 44 estudos selecionados da base científica somados aos 28 encontrados na busca em outros repositórios resultaram nos 72 estudos relacionados no Apêndice A. Para cada estudo selecionado foi atribuída uma identificação do ID 1 até ID 72, tendo como critério a ordem da descoberta do estudo.

Ainda em relação à atividade 6: “Garantir a qualidade do estudo”, utilizou-se 7 dos 11 critérios definidos por Dyba *et al.* (2008) e Shea *et al.* (2007), que se mostraram aplicáveis ao estudo. Os critérios são:

- I. O estudo é baseado em pesquisa?
- II. Há uma declaração clara dos objetivos da investigação?
- III. Existe uma descrição adequada do contexto?
- IV. Os dados foram coletados conforme os objetivos de estudo?
- V. A análise de dados foi suficientemente rigorosa?
- VI. Há uma declaração clara dos resultados?
- VII. O estudo é relevante considerando prática ou pesquisa?

Ainda de acordo com Shea *et al.* (2007) esses critérios podem ser acrescidos de três questões importantes relacionadas à qualidade científica:

- I. Rigor: uma abordagem completa e adequada foi aplicada a métodos de investigação no estudo?
- II. Credibilidade: os resultados são apresentados de forma significativa?
- III. Relevância: os resultados são úteis para a indústria de software e a comunidade científica?

Os 7 critérios descritos por Dyba (2008) e Shea (2007) e as três questões relacionadas com a qualidade científica foram aplicados na seleção dos trabalhos.

Em relação à atividade 7: “Extrair dados requeridos”, os guias de Petersen (2008) sugerem a exploração de algumas seções de documentos, caso o resumo não seja bem estruturado ou vago. Para este estudo e visando objetivar as respostas para as perguntas da pesquisa, todos os estudos selecionados após a última etapa do protocolo SLR foram lidos. Utilizou-se um editor de planilhas para a elaboração de um modelo para obter as informações pertinentes de todos os estudos. Essa informação foi útil para resumir e facilitar a síntese de dados.

Segundo Kitchenham e Charters (2007), a atividade 8: “Sintetizar os dados”, envolve a comparação e o resumo dos resultados dos estudos primários incluídos. A síntese pode ser descritiva (não quantitativa). No entanto, às vezes é possível complementar uma síntese descritiva com um resumo quantitativo. Nas respostas das questões bibliométricas e de pesquisa foi usada uma abordagem descritiva e nas

questões bibliométricas QB -1 e QB – 2 foi colocado um resumo quantitativo na forma de gráfico.

2.4 FASE 3: REVISÃO DA DOCUMENTAÇÃO

Segundo Kitchenham e Charters (2007), a atividade 9: “Escrever o relatório da revisão”, significa integrar estudos que incluam resultados e conclusões, em que diferentes pesquisadores podem ter usado termos e conceitos com significados sutis (ou grosseiramente) diferentes. Assuntos de importância são identificados e é realizada uma síntese para cada estudo contemplado, visando responder as questões bibliométricas e as questões de pesquisa. Essas respostas são documentadas e tabuladas.

Apresentam-se os resultados para as três questões bibliométricas.

Em relação a primeira questão bibliométrica têm-se:

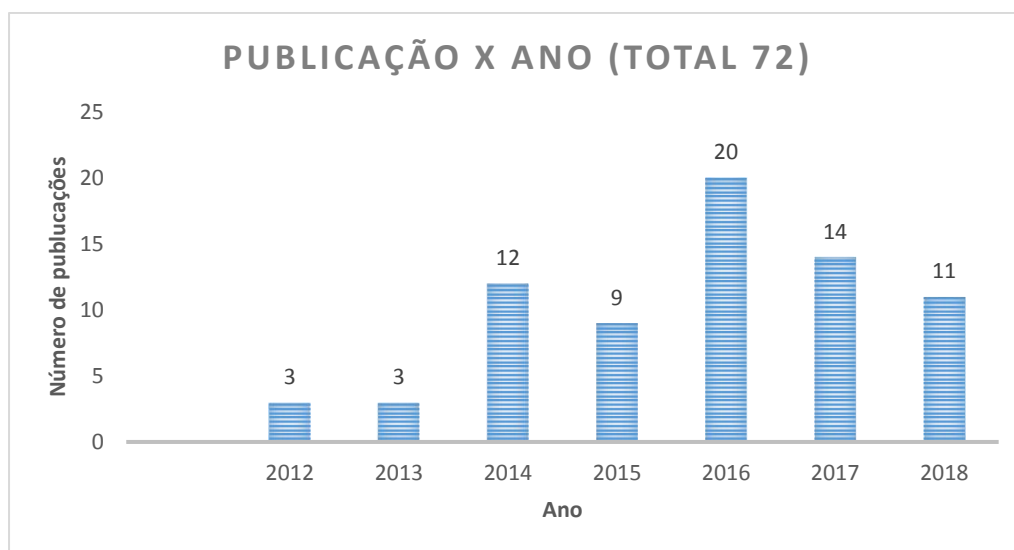
- I. QB-1: Como foi a evolução do número de artigos publicados, relacionados com o tema da pesquisa (constructos)?

O trabalho apresentado está pautado principalmente na norma ISO/IEC 29110. Pelo fato dessa norma ser publicada no ano 2012, esse ano passa a ser a referência para todas as pesquisas em bases científicas. Os métodos ágeis e mais propriamente o Scrum, bem como a filosofia Lean adaptada a software apresentam trabalhos em anos anteriores, porém não foram considerados, para manter a pesquisa no estado da arte.

Nesta análise foi possível verificar a evolução temporal do campo de pesquisa. O objetivo foi quantificar os trabalhos publicados que abarcam os termos ISO/IEC 29110, Scrum e Lean Software identificados ao longo do período de 2012 a 2017 e dar sentido ao interesse pelo assunto tratado nesta pesquisa. Essa distribuição temporal é apresentada na Figura 3 e mostra que o maior número de publicações foi encontrado em 2016.

A última busca por publicações foi realizada em agosto de 2018, e ressalta-se o número de publicações (11) já atingidas esse ano até o mês pesquisado.

Figura 3: Evolução temporal das publicações sobre ISO/IEC 29110, Scrum e Lean Software.

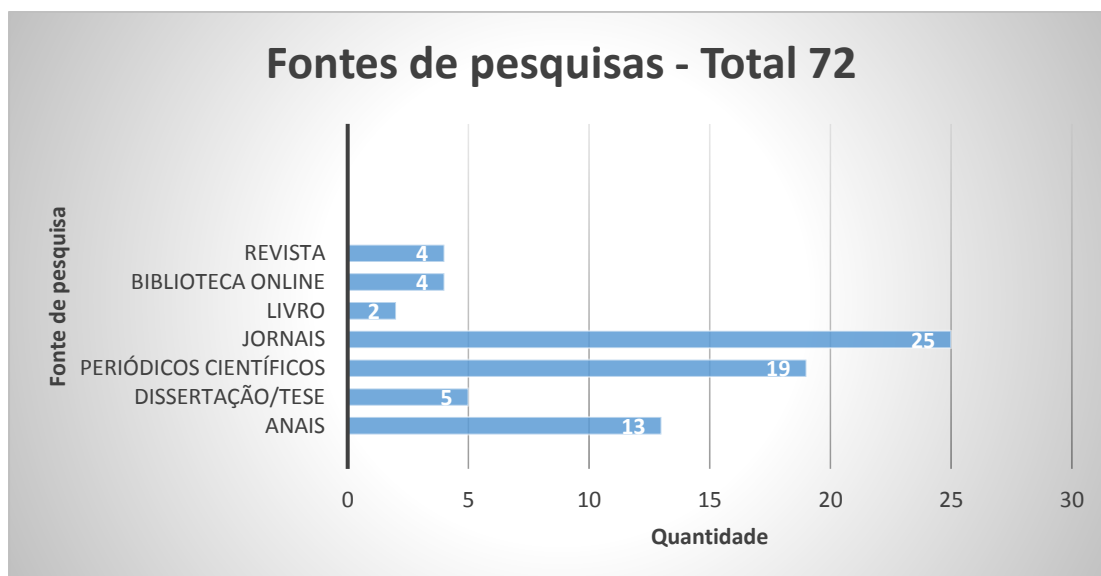


Em relação à segunda pergunta bibliométrica têm-se:

- II. QB-2: Quais as fontes de pesquisas utilizadas (dissertação/tese, revista, biblioteca online, livro, jornal, anais e periódicos científicos)?

Na Figura 4 há uma lista com o número de estudos agrupados indicando as fontes de pesquisas utilizadas. Pode-se notar que o maior número de ocorrências é encontrado em jornais e periódicos científicos (44 estudos) e representa 61% aproximadamente do total. É importante também apontar os estudos obtidos em anais de eventos contemplando Engenharia e Qualidade de Software (13 estudos) que representam 18% aproximadamente. Nota-se que a Academia em seus estudos de Stricto Sensu também está preocupada com o tema em questão, visto que se encontram entre os estudos, 4 dissertações de mestrado e uma tese de doutorado.

Figura 4: Fontes de pesquisas contemplando a ISO/IEC 29110, Scrum e Lean Software



A terceira questão bibliográfica é descrita da seguinte forma:

- III. QB-3: Quais são as bibliotecas digitais de jornais e periódicos, com mais publicações relacionadas ao tema da pesquisa?

Na consulta às principais bases de dados científicas no que tange a jornais e periódicos, obteve-se como retorno 44 estudos. A principal base de dados, no que se refere a quantidade de estudos, foi o ProQuest, com 17 estudos, representando 39% do total encontrado de estudos. Um outro fato que demonstra o interesse da comunidade científica são as bases principais de pesquisas (IEEE, Elsevier, ProQuest, ACM) colaboram com 43 publicações o que representa 60% do total dos estudos.

O Apêndice B mostra a lista de todos os veículos de publicações dos 72 estudos coletados. Esses estudos estão distribuídos em 41 veículos de publicações. Esse número demonstra um interesse elevado da comunidade científica pelo tema em questão.

Em relação as questões de pesquisa, tem-se:

- I. QP-1: De que forma o Scrum pode ser usado em combinação com a ISO/IEC 29110?

Para essa questão, 10 estudos apresentados no Apêndice A, a saber, [ID14], [ID33], [ID34], [ID35], [ID36], [ID37], [ID38], [ID39], [ID40], [ID41], foram encontrados, cujas palavras chaves ISO/IEC 29110 e Scrum aparecem combinadas.

O trabalho de Aleem (2016) em [ID41] tem como objetivo fornecer uma melhor compreensão da dimensão do desenvolvedor como um fator de sucesso na fabricação de “games”. Ele se concentra principalmente em uma investigação empírica do efeito dos principais facilitadores no desenvolvimento de “games” e, eventualmente, na qualidade do “game” a ser comercializado. A palavra ISO/IEC 29110 aparece apenas nas referências bibliográficas, não sendo citado no texto.

O artigo de Muñoz (2014) em [ID34] compara a teoria e a realidade focada na caracterização das necessidades que pequenas e médias empresas (PME) têm de enfrentar ao implementar uma melhoria do processo de software. O estudo está focado em mostrar os resultados encontrados nas PMEs da região de Zacatecas do México considerando uma revisão sistemática. Cita que as palavras-chaves ISO 29110 e Scrum são exemplos de termos utilizados no levantamento de dados realizados nas empresas mexicanas.

O trabalho de Felderer (2016) em [ID38] relata a inexistência da identificação de riscos nos processos de teste das PME e explora a importância e o benefício que essa identificação traz para as organizações. É feito um estudo de caso em cinco organizações, sendo que uma delas adota o Scrum como método de desenvolvimento. A ISO/IEC 29110 é citada como uma norma que pode ser utilizada em empresas com menos de 25 colaboradores.

O trabalho de Kuhrmann (2016) em [ID37] traz um estudo que mostra um crescente interesse em fatores de sucesso para auxiliar as empresas na condução da SPI (*Software Process Improvement*) e na adaptação de princípios e práticas ágeis para SPI. Afirma também a importância da combinação Agile/Lean nos novos modelos de SPI aplicados em empresas. Novamente relaciona esse estudo com o de (O’CONNOR, 2015).

O artigo de Sanchez-Gordon (2017) em [ID14] traz um estudo realizado no Equador envolvendo três empresas VSEs que objetiva conhecer práticas usadas no desenvolvimento de software e verificar a possibilidade de adoção da ISO/IEC 29110. Existe uma grande contribuição no entendimento das vantagens em implantar a norma que é exposta no item seguinte (QP-2). O mesmo artigo cita que o Scrum é adotado em algumas empresas na implantação do CMMI.

O trabalho de Pino *et al.* (2016) em [ID33] trata de uma pesquisa-ação que investigava a adoção de SPI em empresas desenvolvedoras de software da América Latina. O artigo cita o Scrum como sendo um processo leve para incorporar melhorias, e um método de apoio ao gerenciamento e execução das atividades da formulação e execução de melhorias. A relação com a ISO/IEC 29110 está no fato de que a pesquisa foi realizada em algumas empresas, sendo que uma delas era um piloto da adoção da norma.

O artigo de López-Lira (2014) em [ID35] propõe um modelo para classificar e melhor entender de que maneira diferentes conjuntos de práticas podem ser adotadas ou melhor se encaixem no desenvolvimento de software. Cita que esse modelo adotado é aplicável a empresas que adotam ISO/IEC 29110, outras normas de qualidade e empresas que adotam métodos ágeis, tais como o Scrum ou XP.

O trabalho de Berg *et al.* (2018) em [ID39] cita um artigo de Laporte *et al.* (2016) que apresenta os resultados dos testes iniciais da norma ISO/IEC 29110 para entidades muito pequenas e conclui que as certificações internacionais podem aumentar as chances de sucesso das pequenas empresas de software. Desenvolvedores em startups de software tipicamente priorizam práticas ágeis relacionadas à velocidade em vez de relacionadas à qualidade. Padrões como a ISO, adaptados ao contexto de inicialização, podem ajudar os desenvolvedores de software a combinar qualidade e velocidade, o que, por sua vez, pode aumentar as chances de sucesso. No entanto, como a norma ISO/IEC 29110 é destinada principalmente a empresas muito pequenas, é apenas parcialmente relevante para *startups*, diz o artigo. Um trabalho futuro deve ser realizado para desenvolver um padrão ISO adaptado ao contexto de inicialização, para apoiar os desenvolvedores na prática de engenharia de software profissional que contempla métodos ágeis. Nesse contexto a tese que se apresenta procura ajudar a resolver esse problema.

O artigo de Politowski *et al.* (2018) em [ID40] se mostra de interesse para esta tese. O autor faz um estudo de caso na indústria de jogos explorando o papel do processo de software no desenvolvimento do jogo. Os resultados mostraram que não há um modelo de boas práticas para o desenvolvimento nesse tipo de empresa e sugere criar um modelo, baseado em ISO/IEC 29110. Cita o Scrum com um possível método ágil a ser implementada.

Finalmente o artigo de Morales-Trujillo *et al.* (2015) em [ID36] cita a OMG (Object Management Group) que reúne algumas empresas, tais como Boeing, Microsoft, Oracle, estabelecida em 1989 e seus membros trabalham para transformar iniciativas em especificações tecnológicas. Apresenta-se um “*roadmap*” para implementação de processos. A ISO/IEC 29110 e Scrum são citados como “estado da arte”.

Dessa forma a combinação dos constructos ISO/IEC 29110 e Scrum presentes em 10 estudos mostram a ausência de um trabalho que contemple a elaboração de um processo definido e implementado em uma pequena empresa.

Para a segunda questão de pesquisa tem-se:

- II. QP-2: Há vantagens ou desvantagens da utilização de práticas ágeis e ISO/IEC 29110? Quais são?

Foram encontrados apenas 2 artigos: [ID14] e [ID38].

O artigo de Sanchez-Gordon, *et al.* (2017) em [ID14] traz um estudo realizado no Equador envolvendo três empresas VSE traz resultados de entrevistas com Diretores. Em uma das empresas o resultado foi que a implantação da norma poderia melhorar a imagem da empresa, melhorar o processo de trabalho, criando consistência no trabalho de desenvolvimento, tornando o negócio mais lucrativo. O estudo aponta que um entrevistado explicou que a norma, ou padrão é importante, porém tem que adaptá-lo à realidade da empresa. Também relata que um Diretor concluiu que "Se você pudesse alcançar o padrão, eventualmente você poderia diminuir os custos porque você teria um processo definido". Finalmente, um outro participante das entrevistas observou que "o grande benefício é um software mais controlado" pois seria produzido utilizando de um processo de desenvolvimento, o que inclusive levaria a um menor prazo para terminar um projeto.

Em contrapartida têm-se algumas barreiras. Foram listados os seguintes aspectos:

- Sobrecarga de recursos (tempo, pessoas e financeiro);
- Ausência de informações sobre os padrões, como diretrizes, pacotes de implantação e esquema de processo de certificação;

- Poucos artigos sobre estudos de caso contemplando prazo, custo, esforço e lições aprendidas.

O artigo de Felderer (2016) em [ID38] cita que o desenvolvimento de software é impulsionado pela manutenção e evolução contínuas do produto de software. Existe um grande volume de solicitações de curto prazo pelo suporte e gerenciamento de clientes que envolvem tarefas de desenvolvimento. A capacidade e a flexibilidade de implementar essas tarefas em curto prazo é uma vantagem competitiva para a empresa. Essa empresa possui 15 colaboradores.

Pelos estudos identificados existem vantagens e desvantagens na utilização de práticas ágeis e a ISO/IEC 29110, porém a quantidade de trabalhos encontrados não deixa clara a resposta para a questão. Uma pesquisa mais abrangente deve ser feita para que conclusões sejam realizadas.

Finalmente, para a terceira questão de pesquisa tem-se:

III. QP-3: Existem alguns estudos relacionados com o uso combinado de práticas ágeis (Scrum), Lean Software e ISO/IEC 29110?

Na pesquisa realizada encontrou-se apenas 3 trabalhos envolvendo os constructos que fazem o alicerce desta tese. São os trabalhos [ID36], [ID37] e [ID39].

O artigo de Morales-Trujillo *et al.* (2015) em [ID36] traz os constructos ao longo do trabalho, porém desconectados em pontos distintos da pesquisa. Cita, como já mencionado o Scrum e a ISO/IEC como “estado da arte” e Lean e Kanban como terminologias que devem ser consideradas para construção de metamodelo de desenvolvimento de software.

O artigo de Kuhrmann (2016) em [ID37] se mostra muito interessante no sentido de trazer uma coleção de metadados estudados na fase de análise da pesquisa cujo objetivo é realizar um estudo sistemático do estado da arte em melhoria de processo de software. O artigo separa os metadados em duas dimensões: Processo e Contexto. Na dimensão processo aparecem as palavras “Agile Lean”, ISO/IEC 29110. O constructo Scrum aparece como um exemplo de desenvolvimento híbrido chamado de “Water-Scrum-Fall”.

O artigo de Berg *et al.* (2018) em [ID39] cita a existência de estudos empíricos que demonstram que embora as *startups* cujo desenvolvimento de software tente

adotar os princípios do Lean Software e métodos ágeis, elas geralmente acham difícil aplicá-los. Conclui que mais pesquisas são necessárias para apoiar empreendedores e desenvolvedores de software a aumentar suas chances de sucesso. Traz também em relação a norma ISO/IEC 29110, a afirmação que a mesma é destinada principalmente a empresas muito pequenas, e é apenas parcialmente relevante para as *startups*. Ainda o artigo menciona que trabalhos futuros devem ser realizados para desenvolver um padrão ISO adaptado ao contexto de inicialização, para apoiar os desenvolvedores na prática de engenharia de software profissional.

Pode-se concluir por meio da revisão sistemática realizada que não existe, nas bases pesquisadas, um estudo que envolva a elaboração de um processo para apoiar pequenas empresas (VSE) envolvendo a norma ISO/IEC 29110, baseado no Scrum e Lean Software, o que mostra o ineditismo desta tese.

Segundo Kitchenham e Charters (2007), a atividade 10: “Validar o relatório”, quando se tratar de artigos de periódicos, esses serão revisados por pares como uma questão natural. Especialistas revisam teses de doutorado como parte do processo de exame. Em contraste, os relatórios técnicos geralmente não são submetidos a nenhuma avaliação independente. Se um painel de especialistas fosse montado para revisar o protocolo do estudo, o mesmo painel seria apropriado para realizar a revisão por pares do relatório de revisão sistemática. Caso contrário, vários pesquisadores com experiência na área de tópicos e/ou metodologia de revisão sistemática deveriam ser abordados para revisar o relatório. Nesse contexto, os trabalhos mencionados foram revisados por pares.

3. FUNDAMENTAÇÃO TEÓRICA

Esse capítulo apresenta a fundamentação teórica que dá o suporte teórico para esta tese. Ela é composta pelos alicerces representados pelos constructos e outros aspectos que envolvem o trabalho.

3.1 PROCESSO DE SOFTWARE

Como o objetivo desta pesquisa é desenvolver um processo se faz necessário definir o conceito de processo e quais são os elementos principais que o compõem. Para isso foram usadas algumas definições encontradas na literatura. Segundo Hammer, Champy (1994) e Hammer (1997, p.42):

“Processo é qualquer atividade ou conjunto de atividades que toma um input, adiciona valor a ele e fornece um output a um cliente específico. Os processos utilizam os recursos da organização para oferecer resultados objetivos aos seus clientes. Mais normalmente, um processo é um grupo de atividades realizadas numa sequência lógica com o objetivo de produzir um bem ou um serviço que tem valor para um grupo específico de clientes”.

A literatura apresenta uma serie de definições no que se refere a processo. Processo é um conjunto de atividades inter-relacionadas que se interagem com o objetivo de transformar entradas em saídas (ISO 9000:2005).

Salviano (2006, p. 21) afirma que, em relação a processo de software:

“Processo de software é uma atividade, ou um conjunto de atividades, realizadas, com um determinado propósito, por pessoas que utilizam suas habilidades e seu conhecimento, com o apoio de artefatos, ferramentas e outros recursos, para produzir software e seus produtos associados”.

O processo de software é definido como o conjunto de ferramentas, métodos e práticas que as pessoas alocadas em um projeto usam para desenvolver um produto de software (HUMPHREY, 1989). Diferentes processos de software vêm sendo definidos e utilizados pela indústria. Esses processos definem etapas repetíveis para o desenvolvimento de software, utilizando muitas vezes técnicas para o gerenciamento e controle do projeto (PRESSMAN, 2010), autor clássico da Engenharia de Software.

Quando se abstrai o processo real de desenvolvimento a ponto de vê-lo como um padrão, têm-se um modelo desse processo de desenvolvimento. O propósito central de um modelo é reduzir a complexidade de um fenômeno e a interação com o mesmo pela eliminação de detalhes que não influenciam seu comportamento (CURTIS, 1992).

Um modelo de processo de software é caracterizado como uma representação suficientemente geral para modelar vários subprocessos específicos de software, e suficientemente específica para permitir o raciocínio sobre esses modelos. O modelo do processo de software é um elemento que permite organizar e descrever como o ciclo de vida, os métodos, as ferramentas, os produtos e os desenvolvedores se relacionam (DOWSON 1991; e SCACCHI, 1992).

As pesquisas na área de modelagem de processos de software se consolidam em torno de uma série de objetivos. Curtis (1992) agrupa esses objetivos em 5 categorias:

- a) Facilitar a compreensão e o entendimento humano
 - Representar o processo de forma compreensível ao grupo de trabalho.
 - Formalizar o processo para facilitar o trabalho em grupo.
 - Prover informações suficientes ao trabalho de um indivíduo ou do grupo.
 - Permitir a comunicação efetiva entre os membros da equipe.
- b) Dar suporte à evolução do processo
 - Identificar componentes necessários ao desenvolvimento de um software.
 - Reutilizar processos bem definidos.
 - Comparar alternativas de desenvolvimento para o processo.
 - Estimar o impacto de modificações potenciais.
 - Facilitar a incorporação de novas tecnologias ao processo
 - Dar suporte ao gerenciamento das modificações.
- c) Dar suporte ao gerenciamento do processo
 - Desenvolver um modelo de processo de software específico para o projeto.
 - Dar suporte ao desenvolvimento de planos para o projeto.

- Monitorar, gerenciar e coordenar o processo de desenvolvimento.
 - Fornecer uma base para medições do processo.
- d) Dar suporte à execução automática
- Automatizar partes do processo.
 - Dar suporte ao trabalho cooperativo.
 - Coletar dados que reflitam a experiência com o processo.
 - Aplicar regras que garantam a integridade do processo.
- e) Dar auxílio à execução do processo
- Definir um ambiente de trabalho.
 - Dar sugestões e material de referência a fim de facilitar o trabalho humano.
 - Armazenar representações reutilizáveis do processo em um repositório de dados.

A norma ISO 9001:2015 apresenta o conjunto de requisitos mínimos necessários no que se refere a modelagem do processo.

- a) Entradas e saídas: como o próprio nome sugere envolvem insumos e saídas. Podem ser de outros processos anteriores, fluxos de informação, recursos, produtos, serviços. É a maneira pela qual esses elementos nos alcançam e como eles passam por nossas mãos.
- b) Sequência e interação: que são as tarefas a serem executadas devem seguir uma sequência definida que em alguns casos apresentam um grau de dependência uma da outra.
- c) Ferramentas: são os recursos que a empresa precisará para realizar a atividade. Eles devem ser atribuídos corretamente para que o processo seja concluído corretamente.

A modelagem do processo PROÁGIL- 29110 contempla outros requisitos além daqueles propostos pela ISO 9001:2015 para uma melhor compreensão dos usuários nas empresas que adotarem o processo.

Esses outros requisitos são: Objetivo, descrição, responsável, envolvidos e *templates*² associados.

3.2 MODELOS DE QUALIDADE DE SOFTWARE

O século XX teve na melhoria da qualidade de produtos industriais um fator competitivo relevante. Estabeleceu-se novas regras e conceitos para o processo produtivo. Alguns estudos relevantes podem ser citados, tais como Crosby (1979) que definiu a qualidade de um produto como o resultado das qualidades das diversas atividades de uma empresa e que, para tanto, é necessário definir padrões desejados de desempenho para ela. Sendo assim, definindo-se conformidade aos padrões, o controle de produção pode ser precisamente avaliado, sendo que seu objetivo é produzir “zero defeitos” e “fazer certo na primeira vez”.

Deming (1982), definiu o Programa da Qualidade como a implementação da filosofia de que qualidade é fator de aumento da competitividade de uma organização. Por orientação desse programa a empresa deveria elaborar o controle estatístico de todos os processos técnicos e administrativos.

Feingenbaum (1991) estabeleceu que com a utilização de procedimentos de trabalho bem definidos, documentados e conhecidos por todos, uma empresa conseguirá a infraestrutura e a sistematização para a conquista da qualidade.

Juran (1988) introduziu três processos básicos para a conquista da qualidade. Esses processos ficaram conhecidos como Trilogia da Qualidade: Planejamento da Qualidade que tem como objetivo principal fazer com que a empresa atenda aos objetivos da qualidade estabelecidos para o produto usando-se um processo produtivo adequado; o Controle da Qualidade cuja finalidade é garantir que o processo produtivo seja o melhor com o máximo de redução de perdas que se pode alcançar e Aprimoramento da Qualidade cujos níveis aceitáveis de perdas sejam minimizados ao longo do tempo com a melhoria do processo produtivo.

² Um template é um modelo a ser seguido, com uma estrutura predefinida que facilita o desenvolvimento e criação do conteúdo a partir de algo construído a priori.

Ishikawa (1985) afirmou que para que um produto tenha qualidade é necessário que o produto esteja alinhado à satisfação do cliente. Neste sentido, cliente não representa apenas o consumidor final. Cada próxima etapa num processo produtivo também é o cliente, o que garantirá que seja estabelecida uma cadeia em que, cada fase produtiva, o produto esteja atingindo os requisitos necessários até chegar ao cliente final.

Aproveitando-se a difusão de todos esses conceitos e, também, da utilização destes na prática entre as indústrias de bem manufaturados em todo o mundo, surgiram várias normas e modelos para gestão e melhoria da qualidade. Entre eles, podem-se citar a série de Normas ISO 9000, ISO (1994) e Seis Sigma (PANDE, 2002).

As Normas ISO 9000, ISO (1994) foram desenvolvidas pela *International Organization for Standardization* (ISO) e utilizadas amplamente ao redor de todo o mundo. Compreende um conjunto de documentos sobre requisitos do sistema da qualidade para uso, quando, um contrato entre duas partes requer demonstração da capacidade do fornecedor para projetar e fornecer um produto.

Dessa forma, os dois aspectos do software, isto é, processo e produto, podem ser avaliados e existem na literatura vários modelos com esse objetivo. No que diz respeito à qualidade do processo de software vários modelos e normas de melhoria têm surgido e sido utilizados amplamente pela indústria de software global, e estudados por entidades de ensino e pesquisa. Entre eles podem-se destacar o SW-CMM, PAULK (1993), o CMMI da SEI (2002), a ISO 15504, a NBR ISO/IEC 12207, (ISO, 1997) e o MSP.BR da (SOFTEX, 2005).

A qualidade do software é o grau em que o software possui uma combinação desejada de atributos (IEEE STD. 1061, 1998). No entanto, a natureza complexa do software torna a conquista dessa propriedade um problema de difícil solução. Além disso, a delimitação do que define um nível adequado de qualidade em um sistema de software é uma questão altamente dependente do contexto (KITCHENHAM e PLEEGER, 1996). Este problema muda com o produto e a perspectiva das partes interessadas. Enquanto cada grupo tiver sua própria perspectiva sobre o que é importante, a qualidade do produto do software pode facilmente se tornar uma área de problemas e conflitos (BLAS *et al.*, 2018)

Rocha (2001) relata que a produção de software de qualidade é meta básica da Engenharia de Software, que disponibiliza métodos, técnicas e ferramentas para isso, resultando em um software confiável e eficaz. Pressman (2010) afirma que, “se o processo é fraco, o produto final irá, sem dúvidas, sofrer as consequências”. Essa assertiva se relaciona plenamente com o conceito de qualidade de software.

Os modelos e normas de qualidade de software foram criados de modo a atender plenamente os requisitos de qualidade auxiliando na melhoria dos processos internos e promovendo a normatização de produtos e serviços (MACIEL, 2011). Segundo Irama (2012), os modelos e padrões de qualidade podem ser usados para melhorarem os processos das empresas. Entre os mais conhecidos estão o MPS.Br e CMMI.

A seguir apresenta-se os modelos de qualidade CMMI e MPS.Br que oferecem certificações, são aceitos por empresas brasileiras e servem de referências para artigos publicados pela comunidade acadêmica mundial.

3.2.1 Modelo CMMI

O CMMI é uma estrutura para o avanço do processo de negócios. Os modelos CMMI oferecem uma direção que pode ser usado ao desenvolver processos. Existem atualmente três tipos de CMMI. O mais conhecido é CMMI para o desenvolvimento (CMMI-DEV). Os outros dois tipos são CMMI para aquisição (CMMI-AQU) e CMMI para serviços (CMMI- SVC). Cada tipo tem conteúdos diferentes que visam melhorias em áreas específicas (PANE, 2015).

O CMMI-DEV é um modelo que envolve atividades para o desenvolvimento de produtos e serviços. O CMMI-DEV é organizado com base em três conceitos: área de processo (PA), metas específicas (SG) e práticas específicas (SP). Existem 16 áreas de processo (PA) contidas no CMMI. Essas áreas de processos abrangem conceitos básicos que são fundamentais para a melhoria de processos em qualquer área de interesse. Uma área de processo contém uma série de práticas relacionadas que,

quando implementada simultaneamente, cumprem um conjunto de metas consideradas importantes para fazer melhorias nessa área (SEI, 2017).

De acordo com Sória (2006), após uma crise de qualidade e custos ocorrida no desenvolvimento de software para o Departamento de Defesa dos Estados Unidos, foi criado o Software Engineering Institute (SEI) na Carnegie Mellon University de Pittsburgh. Este instituto começa o desenvolvimento de um modelo de engenharia de software em 1988 e em 1991 publica a primeira versão do modelo chamado SW-CMM.

Com o passar do tempo, outras organizações foram se integrando ao projeto do SEI, agregando novas disciplinas ao CMM e com as necessidades do mercado as organizações foram criando modelos derivados que fossem adaptados à realidade individual de sua própria empresa. Observando essas mudanças, o SEI reuniu vários requisitos destas organizações e em dezembro de 2001 propôs a criação do CMMI que deu ênfase tanto para a engenharia de sistemas quanto para a engenharia de software.

Chrissis *et al.* (2004) conceituam CMMI como “as melhores práticas direcionadas ao desenvolvimento e à manutenção de produtos e dos serviços, abrangendo todo o ciclo de vida do produto, desde sua concepção até a sua entrega e manutenção”. Muitas organizações em todo o mundo têm adotado este modelo com o objetivo de possibilitar a elevação da maturidade da capacidade de suas equipes nas atividades relacionadas ao software.

Segundo Morgado *et al.* (2007) o modelo de maturidade CMMI descreve um caminho evolutivo, que começa com processos imaturos (inicial) e segue até um processo maduro e disciplinado (otimizado), sendo possível o controle do processo de produção de software por meio de métricas e modelos estatísticos.

Couto (2007) destaca que além dos benefícios naturais, como produtividade e qualidade, comercialmente acredita-se que, em curto prazo, a certificação dos processos fabris será um pré-requisito básico para as contratações de produtos de software. Por essa razão o SW-CMM tornou-se o modelo de qualidade mais conhecido, usado e respeitado pela comunidade de engenharia de software no mundo.

Ainda Couto (2007) informa que uma organização pode ser aferida ou avaliada comparando-se suas práticas reais com aquelas que o modelo de maturidade de capacitação prescreve ou recomenda, resultando em um diagnóstico da organização quanto aos seus processos. Este diagnóstico serve de base para recomendações de melhoria de processos, e as recomendações podem ser consolidadas em um plano de melhoria. A autora observa que tanto no CMMI como no CMM, não se aplica o conceito de certificação, o termo correto é avaliação (“*appraisal*”). Nesse sentido, não existe o formalismo encontrado nos procedimentos de certificação, como os utilizados, por exemplo, para a certificação ISO 9001, por isso mesmo não é baseado em procedimentos de certificação, não há exigência de reavaliações e nem há validade para o laudo, ou seja, o prazo é indeterminado.

O principal objetivo do CMMI, segundo AHERN (2008), é a redução do custo de implementação de melhoria de processo multidisciplinar por meio das seguintes ações:

- a) Eliminação de inconsistências e redução de duplicidades;
- b) Melhoria da clareza e entendimento;
- c) Utilização de terminologia comum e estilo consistente;
- d) Estabelecimento de regras uniformes de construção;
- e) Manutenção de componentes comuns;
- f) Consistência com a futura norma ISO/IEC 15504;
- g) Sensibilidade às implicações dos esforços legados.

Alguns dados mostram que o Brasil saltou de 17 empresas certificadas em 2004 para 63 em 2018 (CMMI Institute, 2018).

O interesse da Academia em relacionar modelos de qualidade com métodos ágeis, tal qual objetivo desta tese, pode ser observado em pesquisa realizada por (Palomino *et al.*, 2017) que aponta, na realização de uma revisão na literatura, 75 estudos contemplando CMMI e métodos ágeis.

3.2.2 Modelo MPS.Br para Software

O programa que abarca o modelo MPS.Br, coordenado pela Associação para Promoção do Software Brasileiro (SOFTEX), começou a ser desenvolvido em 2003, como uma forma de auxiliar às empresas brasileiras para alcançar a qualidade no desenvolvimento de software.

Segundo Koscianski (2006) o MPS.Br é um modelo de maturidade desenvolvido por consórcio de empresas, com foco principal de atender as micros, pequenas e médias empresas de software brasileiras, que necessitam melhorar seu processo de software.

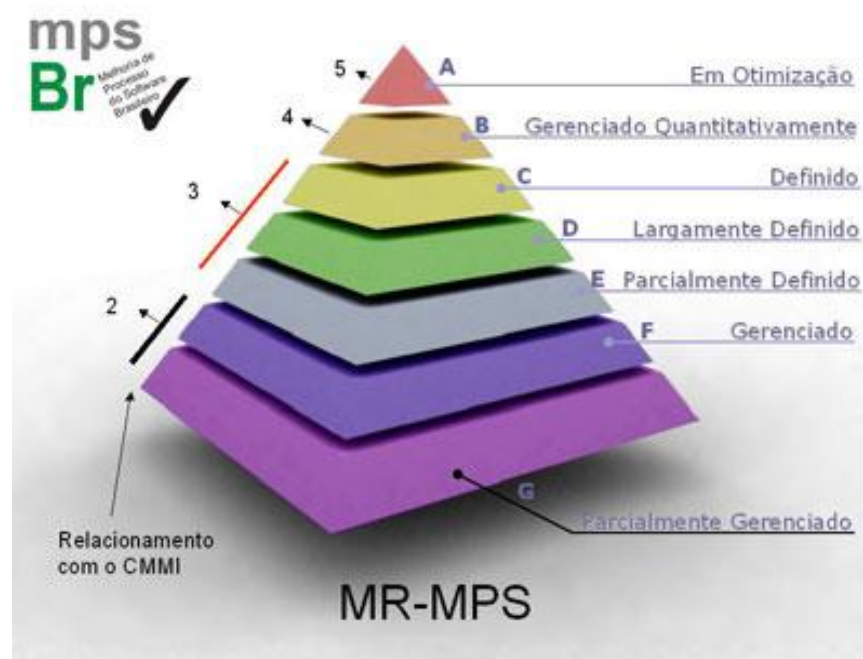
O programa MPS.Br surgiu a partir da necessidade de tornar as empresas brasileiras mais competitivas na área de desenvolvimento de software, ressaltando que a percepção que a demanda, tanto doméstica quanto internacional, tem com um dos principais fatores de sucesso a qualidade dos produtos e processos. Esse programa tem como seu principal objetivo desenvolver e disseminar modelos de melhoria de processos que atendam às necessidades da Indústria Brasileira de Software e Serviços de TI (atualmente a família de modelos é composta pelos modelos de referência MPS-SW para Software e MPS-SV para Serviços de TI), visando estabelecer um caminho economicamente viável para que organizações, incluindo as pequenas e médias empresas, alcancem os benefícios da melhoria de processos e da utilização de boas práticas da engenharia de software e da prestação de serviços de TI em um intervalo de tempo razoável (SANTOS *et al.*, 2012).

O modelo MPS-SW foi desenvolvido levando em consideração normas internacionais, modelos internacionalmente reconhecidos, boas práticas da engenharia de software e as necessidades de negócio da indústria de software brasileira (KALINOWSKI *et al.*, 2010).

O MPS.Br para software define, conforme visualizado na Figura 5, sete níveis de maturidade de processos para organizações que produzem software: A (Em Otimização), B (Gerenciado Quantitativamente), C (Definido), D (Largamente Definido), E (Parcialmente Definido), F (Gerenciado) e G (Parcialmente Gerenciado). O nível G é o primeiro estágio de maturidade e o nível A é o mais maduro. Cada um

dos níveis de maturidade possui um conjunto de processos e atributos de processos que indicam onde a unidade organizacional tem que colocar esforço para melhoria, de forma a atender aos seus objetivos de negócio e ao MPS.Br. Assim, os níveis de maturidade são definidos em duas dimensões: a dimensão de capacidade de processos e a dimensão de processos (KIVAL, 2018). A Figura 5 traz o relacionamento entre os níveis de maturidade do MPS,Br e do CMMI.

Figura 5: Os sete níveis de maturidade do MPS.Br



Fonte: OLIVEIRA (2008).

Os fatores que influenciam as empresas brasileiras na continuidade da execução dos processos de software referentes ao modelo MPS.Br foram investigados por Almeida (2011) que identificou que problemas técnicos no modelo e na sua forma de implantação, problemas culturais das empresas e relacionados aos seus colaboradores, problemas relacionados ao comprometimento dos principais envolvidos e à disponibilização de recursos influenciam de maneira negativa na continuidade no modelo. Montoni (2010) afirma que dentre os fatores críticos de sucesso relacionados à melhoria no processo de software em empresas que adotaram

o MPS.Br estão a influência de fatores organizacionais, técnicos e relacionados ao modelo de melhoria do processo de software.

Segundo a SOFTEX, (2018), 690 empresas foram avaliadas desde o nascimento do modelo no Brasil e 7 avaliações ocorreram no exterior

3.3 A NORMA ISO/IEC 29110

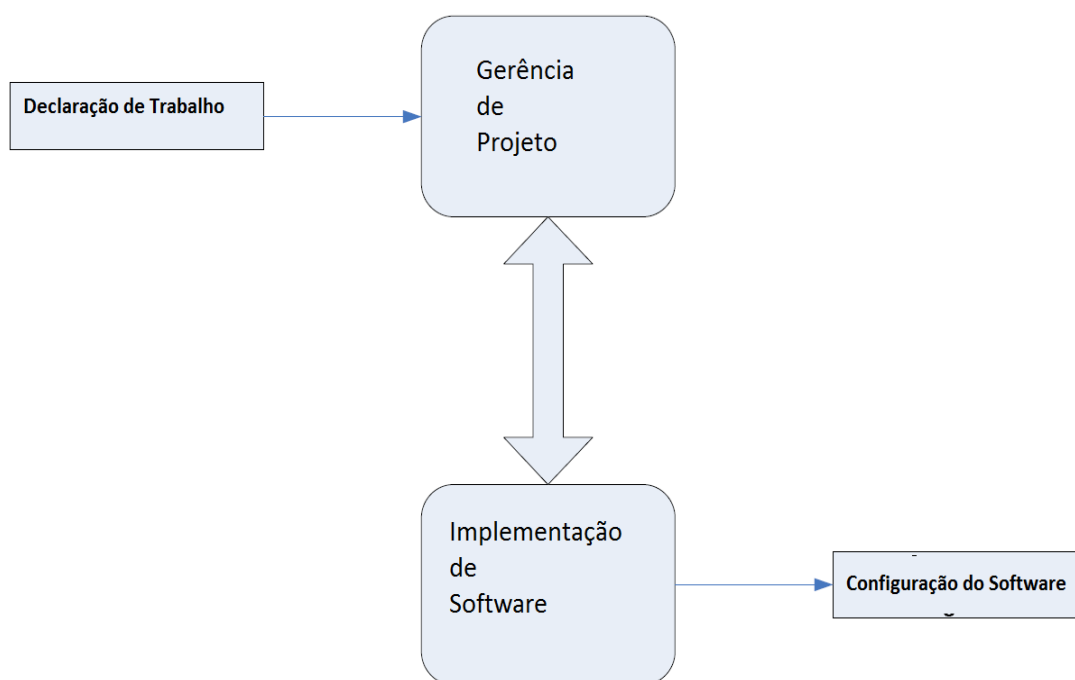
A norma ISO/IEC 29110 segundo o relatório técnico ISO/IEC TR 29110-5-1-2 define três perfis: Básico, Intermediário e Avançado. O primeiro dos perfis, foi publicado em 2012 sob o nome ISO/IEC 29110 Perfil Básico da Associação Brasileira de Normas Técnicas (ABNT, 2012).

O Perfil Básico é definido com dois processos: Gerência de Projetos (PM – *Project Management*) e Implementação do Software (SI – *Software Implementation*).

No Perfil Básico, espera-se que, para iniciar o ciclo de vida de desenvolvimento de um projeto de software, a organização tenha como entrada uma declaração de trabalho definida. Ao final do seu ciclo, o projeto terá como saída o software “configurado” para ser entregue ao cliente. O diagrama representado na Figura 6 ilustra a interação entre esses dois processos descritos na ISO/IEC 29110.

Na Figura 6 aparecem dois processos para o Perfil Básico: a Gerência de Projeto e o processo Implementação de Software

Figura 6: Interação entre os processos do Perfil Básico

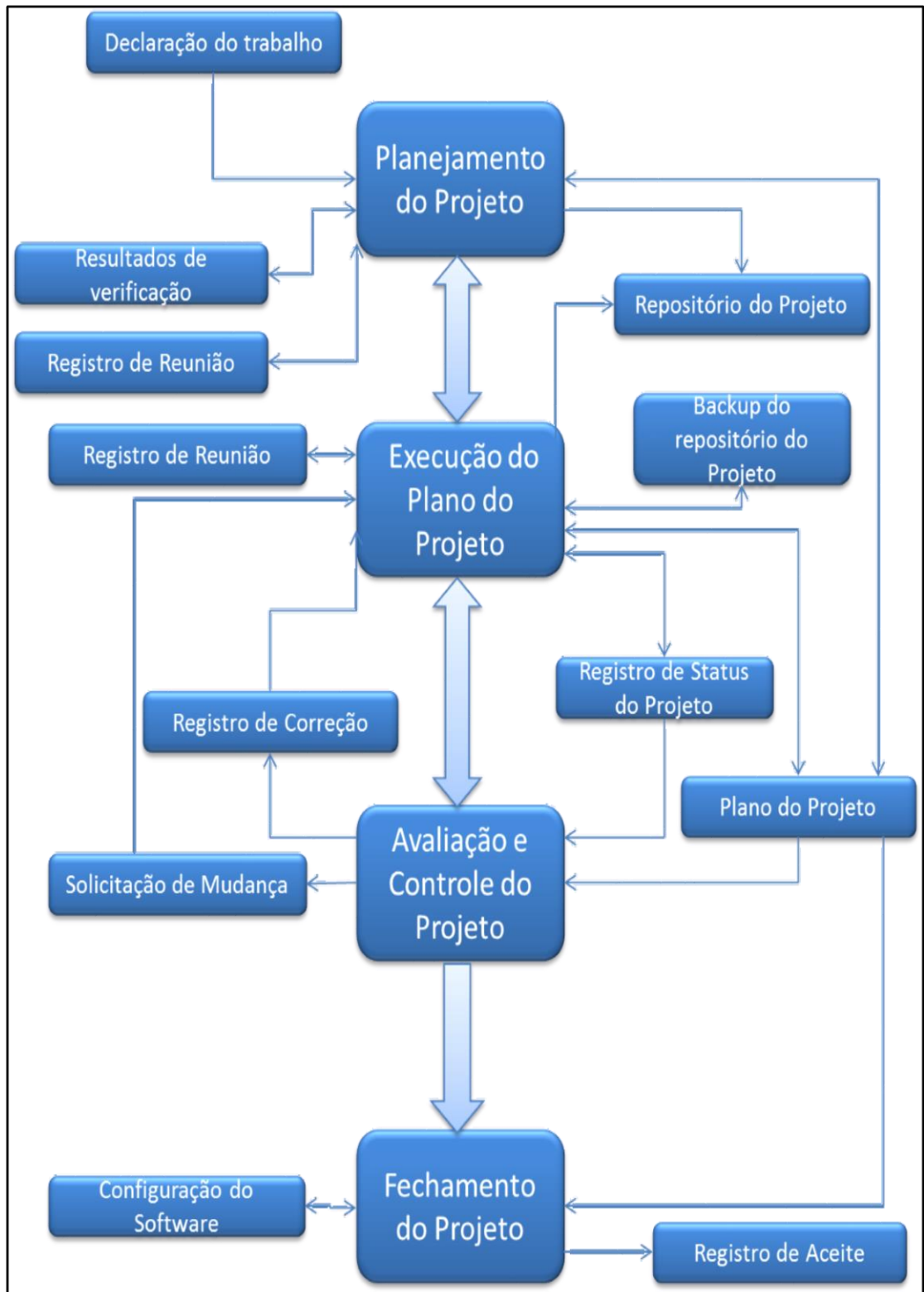


Fonte: Adaptada de ABNT (2012)

O propósito do processo de Gerência de Projetos (PM) é estabelecer e manter sistematicamente as tarefas de implementação, visando os objetivos de qualidade esperada, tempo e custo.

A Figura 7 apresenta o processo PM de forma detalhada.

Figura 7: Diagrama do Processo de Gerência do Projeto (PM)



Fonte: ABNT (2012)

O Processo PM é constituído de um conjunto de objetivos que são organizados sequencialmente de 01 a 07 e cobrem todo o processo que são:

PM.01 - O Plano de Projeto para a execução do projeto é desenvolvido de acordo com a Declaração de Trabalho e revisto e aceito pelo Cliente (Registro de Reunião). As tarefas e os recursos necessários para completar o trabalho são dimensionados e estimados.

PM.02 - O progresso do projeto é monitorado contra o Plano de Projeto e registrado no Registro de Status de Progresso. Ações corretivas para corrigir os problemas e desvios do plano são tomadas quando as metas do projeto não forem alcançadas (Registro de Correção). O encerramento do projeto é formalizado para obter o aceite do cliente, documentado no Registro de Aceite.

PM.03 - As Solicitações de Mudança são tratadas através de sua recepção e análise. Alterações nos requisitos de software são avaliadas quanto ao custo, cronograma e impacto técnico.

PM.04 - São mantidas reuniões de revisão com a equipe de trabalho e os clientes. As decisões são registradas e monitoradas (Registro de Reunião).

PM.05 - Os riscos são identificados inicialmente e durante a condução do projeto (Repositório do Projeto).

PM.06 - Uma Estratégia de Controle de Versão do software é desenvolvida. Itens de Configuração de Software são identificados, definidos e postos em baseline. As modificações e liberações dos itens são controladas e disponibilizadas ao Cliente e à Equipe de Trabalho. O armazenamento, manuseio e entrega dos itens são controlados (Configuração do Software).

PM.07 - A Garantia de Qualidade de Software é realizada para assegurar que produtos e processos de trabalho cumpram o Plano de Projeto e a Especificação de Requisitos (Resultados de Verificação).

As atividades do processo de Gerência de Projeto são as seguintes:

- PM.1 – Planejamento de Projeto

A atividade Planejamento de Projeto documenta os detalhes do planejamento necessários para gerenciar o projeto.

A atividade fornece:

- ✓ *Declaração do Trabalho* revisada e as *Tarefas* necessárias para prover os *Entregáveis* do contrato e para satisfazer os requisitos do Cliente.
- ✓ Ciclo de vida do projeto, incluindo dependência e duração das tarefas.
- ✓ Estratégia da Garantia da Qualidade do Projeto através da verificação e validação dos produtos de trabalho/*Entregáveis*, revisões do Cliente e da Equipe de Trabalho.
- ✓ Funções e responsabilidades do Cliente e da Equipe de Trabalho.
- ✓ *Recursos* do projeto e competências necessárias.
- ✓ Estimativas de esforço, custo e tempo.
- ✓ Riscos de projeto identificados.
- ✓ Estratégia de controle de versão e *baseline* do projeto.
- ✓ *Repositório de Projeto* para armazenar, manusear e entregar versões de produtos e documentos controladas e *baselines*.

- PM.2 – Execução do Plano de Projeto

A atividade de Execução do Plano do Projeto implementa o plano documentado no projeto.

A atividade fornece:

- ✓ *Registro de Status do Progresso* atualizado.
- ✓ Solicitações de mudança analisadas e avaliadas impactando o plano em custo, cronograma e requisitos técnicos.
- ✓ As mudanças aprovadas para o plano.
- ✓ Revisões e aceitações com o Equipe de Trabalho (WT) e Cliente (CUS).
- ✓ Backup do *Repositório do Projeto*, e sua recuperação, se necessária.

- PM.3 – Avaliação e Controle de Projeto

Avaliação e controle do projeto avalia o desempenho do plano de acordo com os compromissos documentados.

A atividade fornece:

- ✓ Avaliação da realização do plano e progresso reais frente aos objetivos.

- ✓ Desvios significativos e problemas identificados e avaliados do custo, do cronograma e do desempenho técnico.
- ✓ Revisão dos riscos do projeto e identificação de novos riscos.
- ✓ Solicitações de mudança documentadas, ações corretivas apropriadas definidas e ajustes acompanhados até o encerramento.

- PM.4 – Fechamento de Projeto

Essa atividade provê os produtos e documentação do projeto de acordo com os requisitos do contrato.

A atividade fornece:

- ✓ Liberação dos produtos como especificado nas *Instruções de Entrega*.
- ✓ Suporte ao produto de acordo com as *Instruções de Entrega*.
- ✓ Conclusão do projeto e assinatura do *Registro de Aceitação*.

O mapeamento entre as atividades e os objetivos do processo de Gerência de Projetos está representado no Quadro 5.

Quadro 5: Relacionamento entre objetivos e atividades (PM)

		ATIVIDADES			
		PM.1	PM.2	PM.3	PM.4
OBJETIVOS	PMO.01	x			
	PMO.02		x	x	x
	PMO.03		x		
	PMO.04		x		
	PMO.05	x	x		
	PMO.06	x			
	PMO.07	x	x		

Além dos objetivos e atividades, o processo de Gerência de Projetos (PM) possui as seguintes entradas:

- ✓ Declaração de Trabalho
- ✓ Solicitações de Mudança
- ✓ Configuração de Software

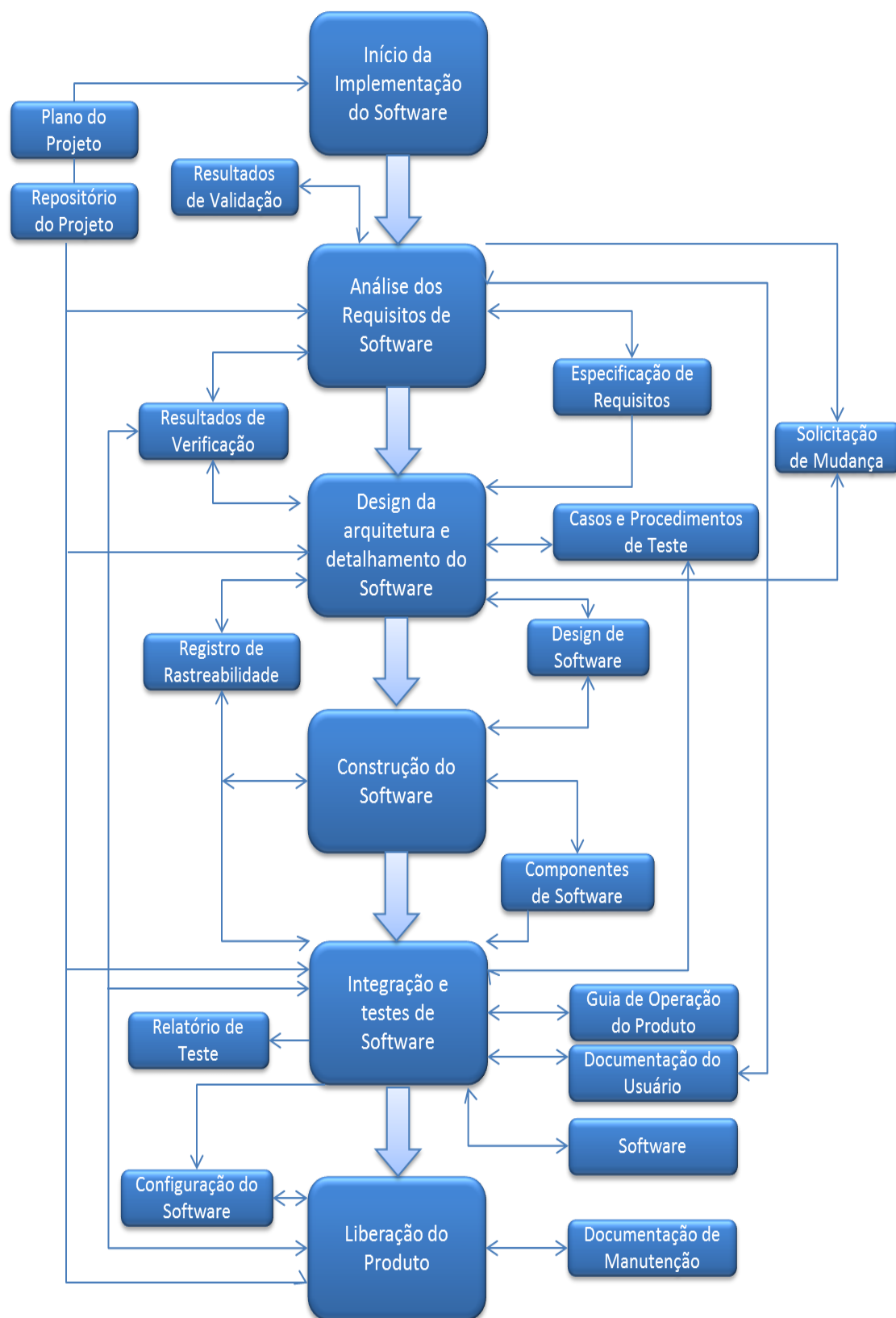
Os produtos de saída para o processo de Gerência de Projetos (PM) são:

- ✓ Plano de Projeto
- ✓ Registro de Aceitação
- ✓ Repositório de Projeto
- ✓ Registro de Reunião
- ✓ Configuração de Software

Por outro lado, o propósito do processo de Implementação de Software (SI), da norma ISO/IEC 29110 é realizar sistematicamente as atividades de análise, projeto, construção, integração e testes, para um novo software ou uma modificação, de acordo com os requisitos especificados.

Esse processo possui o diagrama apresentado na Figura 8.

Figura 8: Diagrama do Processo de Implementação de Software



Fonte: ABNT (2012)

O Processo SI é constituído de um conjunto de objetivos que são organizados sequencialmente de 01 a 07 e cobrem todo o processo apresentado na Figura 8, que são:

SI.01. - Tarefas das atividades são realizadas em cumprimento do Plano de Projeto.

SI.02. - Requisitos de Software são definidos, analisados quanto à correção e testabilidade, aprovados pelo Cliente, colocados em baseline e comunicados (Especificação de Requisitos).

SI.03. - Um projeto de arquitetura e detalhamento é desenvolvido e posto em baseline. Ele descreve os itens de software e suas interfaces internas e externas (Design de Software). É estabelecida consistência e rastreabilidade aos requisitos de software (Registro de Rastreabilidade).

SI.04. - Os componentes de software definidos pelo projeto são produzidos. Testes unitários são definidos e realizados para verificar a consistência com os requisitos e com o projeto (Casos e Procedimentos de Testes). É estabelecida rastreabilidade para os requisitos e para o projeto (Registro de Rastreabilidade).

SI.05. - Software é produzido fazendo a integração dos componentes de software e é verificado usando Casos de Teste e Procedimentos Teste. Os resultados são registrados no Relatório de Teste. Os defeitos são corrigidos e é estabelecida consistência e rastreabilidade ao Projeto Software (Registro de Rastreabilidade).

SI.06. - Uma Configuração de Software, que atende à Especificação de Requisitos conforme acordado com o Cliente, a qual inclui documentações do usuário, de operação e de manutenção é integrada, colocada em baseline e armazenada no Repositório do Projeto. Necessidades de alterações na Configuração do Software são detectadas e as devidas Solicitações de Mudança são iniciadas.

SI.07. - Tarefas de Verificação e Validação de todos os produtos de trabalho necessários são realizadas usando critérios definidos para assegurar a consistência entre produtos de saída e entrada em cada atividade. Defeitos são identificados e corrigidos, Registros são armazenados em Resultados de Verificação/Validação.

As atividades do processo de implementação (SI) são as seguintes:

- SI.1 – Início da Implementação do *Software*

Essa atividade garante que o *Plano do Projeto* estabelecido na atividade de Planejamento do Projeto tem o comprometimento da equipe do trabalho.

A atividade fornece:

- ✓ Revisão do *Plano do Projeto* pela Equipe de Trabalho para determinar a atribuição de tarefas.
- ✓ Compromisso da Equipe de Trabalho e do Gerente do Projeto com o *Plano do Projeto*.
- ✓ Um ambiente de implementação estabelecido.

- SI.2 – Análise dos Requisitos do *Software*

A atividade de análise dos requisitos de *software* analisa os requisitos acordados do Cliente e estabelece os requisitos validados do projeto.

A atividade fornece:

- ✓ Revisão do *Plano do Projeto* pela equipe de trabalho para determinar atribuição de tarefas.
- ✓ Levantamento, análise e especificação dos requisitos do Cliente.
- ✓ Acordo sobre os requisitos do Cliente.
- ✓ Verificação e validação dos Requisitos.
- ✓ Controle de versão dos requisitos dos produtos de *software*.

- SI.3 – Projeto de Arquitetura e Detalhamento do *Software*

A atividade de Projeto de Arquitetura e Detalhamento do *Software* transforma os requisitos na arquitetura do sistema e no projeto detalhamento do *software*.

A atividade fornece:

- ✓ Revisão da equipe de trabalho do *Plano do Projeto* para determinar atribuição de tarefas.
- ✓ Projeto da arquitetura, *Componentes do Software* e interfaces associadas.
- ✓ Projeto detalhado dos *Componentes do Software* e interfaces.
- ✓ Revisão das *Especificações dos Requisitos* pela equipe de trabalho.
- ✓ *Projeto do Software* verificado e defeitos corrigidos.

- ✓ *Casos de Teste e Procedimentos de Teste* verificados para teste de integração.
- ✓ Rastreabilidade dos requisitos do *software* com o *Projeto do Software*, *Casos de Teste e Procedimentos de Teste*.
- ✓ Produtos e documentos do projeto sob controle de versão.

NOTA: O Projeto de Arquitetura e Detalhamento do Software pode ser estabelecido separadamente, de acordo com o cronograma do projeto.

- **SI.4 – Construção do Software**

A atividade de Construção de *Software* desenvolve o código e os dados do *software* a partir do *Projeto de Software*.

A atividade provê:

- ✓ Revisão da Equipe de Trabalho do *Projeto de Software* para determinar a sequência de construção do *software*.
- ✓ *Componentes do Software* codificados e testes unitários aplicados.
- ✓ Rastreabilidade entre *Componentes do Software* e o *Projeto de Software*.

- **SI.5 – Integração e Testes do Software**

A atividade de Integração e Testes de *Software* garante que os *Componentes de Softwares* integrados satisfazem os requisitos do *software*.

A atividade fornece:

- ✓ Revisão do *Plano do Projeto* pela Equipe de Trabalho para determinar a atribuição de tarefas.
- ✓ Entendimento dos *Casos de Teste e Procedimentos de Teste* e ambiente de integração.
- ✓ *Componentes do Software* integrados, defeitos corrigidos e resultados documentados.
- ✓ Rastreabilidade dos requisitos e do projeto do produto de *software* integrado.

- ✓ Documentações do usuário do *software* e operacional documentados e verificados.
- ✓ *Baseline do Software* verificada.

- SI.6 – Entrega do Produto

A atividade de Entrega do Produto fornece o produto de *software* integrado para o Cliente.

A atividade fornece:

- ✓ Revisão do *Plano do Projeto* pela Equipe de Trabalho para determinar a atribuição de tarefas.
- ✓ *Documentação de Manutenção* verificada.
- ✓ Entrega do produto de software e documentação aplicável em concordância com as *Instruções de Entrega*.

O mapeamento entre as atividades e os objetivos do processo de Implementação de Software está representado no Quadro 6.

Quadro 6: Comparativo entre objetivos e atividades (SI)

		ATIVIDADES					
		SI.1	SI.2	SI.3	SI.4	SI.5	SI.6
OBJETIVOS	SI.01	x					
	SI.02		x				
	SI.03			x			
	SI.04				x		
	SI.05					x	
	SI.06		x	x	x	x	x
	SI.07		x	x	x	x	x

Como pode ser observado, nas descrições dos processos PM e SI, para cada processo existe um conjunto de objetivos. Para alcançar os objetivos do processo, são definidas atividades obrigatórias que recebem produtos de entrada e geram produtos de saída.

Os produtos de entrada são gerados por atividades que podem ser intrínsecas ou extrínsecas ao processo e são, portanto, opcionais. Os produtos de saída são gerados pelas atividades realizadas ou pelas tarefas detalhadas de cada uma delas.

Os produtos de saída para o Processo de Implementação de Software são:

- ✓ Configuração do Software
- ✓ Especificação de Requisitos
- ✓ Projeto do Software
- ✓ Registro de Rastreabilidade
- ✓ Componentes do Software
- ✓ Software
- ✓ Casos de Teste e Procedimentos de Teste
- ✓ Relatório de Teste
- ✓ Guia de Operação do Produto
- ✓ Documentação de Usuário do Software
- ✓ Documentação de Manutenção
- ✓ Solicitação de Mudança

Há ainda produtos internos que servem de apoio à realização das atividades e que são, também, opcionais. As atividades definidas como obrigatórias para cada processo são descritas no nível de macro atividades e devem ser executadas por meio de um conjunto de tarefas mais detalhadas.

A norma ISO/IEC 29110 segundo o relatório técnico ISO/IEC TR 29110-5-1-2 define três perfis: Básico, Intermediário e Avançado. O primeiro dos perfis, foi publicado em 2012 sob o nome ISO/IEC 29110 Perfil Básico.

Criar um processo ágil, tendo-se como base os métodos ágeis e ao mesmo tempo estar aderente ao que preconiza a ISO/IEC 29110 será o foco dessa tese.

3.4 MÉTODO ÁGIL SCRUM

A Engenharia de Software contempla na sua essência uma série de tipos de ciclos de vidas que podem ser adotados por uma organização com o objetivo de ajudá-la no desenvolvimento de software ou sistemas. Nesse contexto se encaixam os métodos ágeis, tais como o Scrum, *Extreme Programming* (XP), *Feature Driven Development* (FDD), e outros (PRESSMAN, 2010)

Larman (2004) afirma que no ano de 2001, um grupo interessado em métodos ágeis e iterativos formaram a “Aliança Ágil”³, trazendo um manifesto e uma declaração de princípios. O manifesto reuniu representantes de diferentes métodos ágeis que vinham sendo propostas, com *Extreming Programming* (XP), Scrum, Crystal, entre outros, além de pessoas interessadas em nova abordagens diferentes das anteriores, orientadas a extensa documentação e muitas vezes morosas.

O manifesto promove as melhores formas de desenvolver softwares, por meio dos seus valores (Cockburn *et al.*, 2001):

- I. Indivíduos e interações mais que processos e ferramentas;
- II. Software em funcionamento mais que documentação abrangente;
- III. Colaboração com o cliente mais que negociação de contratos;
- IV. Responder a mudanças mais que seguir um plano.

Os valores podem ser traduzidos nos doze princípios dos métodos ágeis também listados por (Cockburn *et al.*, 2001 e Sabbbagh, 2013):

- a) Nossa maior prioridade é satisfazer o cliente considerando-se a entrega continua e adiantada de software com valor agregado: Gera-se, desde cedo e frequentemente, retorno ao investimento dos clientes no projeto a partir da entrega de partes do produto que atendam às suas necessidades;
- b) Mudanças nos requisitos são bem-vindas, mesmo tardiamente no desenvolvimento. Processos ágeis tiram vantagens das mudanças visando vantagem competitiva para o cliente: ao se trabalhar com ciclos curtos de *feedback*, permite-se aos clientes evoluírem o produto à medida que melhor

³ www.agilealliance.com

entendem suas necessidades e adaptarem às mudanças de mercado, tornando-se mais competitivos;

- c) Entregar frequentemente software funcionando, de poucas semanas a pouco meses, com preferência à menor escala de tempo: fornecer aos clientes e usuários, com frequência, partes do produto, gera, a cada entrega, retorno ao investimento dos clientes obter-se *feedback* sobre o que foi produzido. Assim pode-se adaptar o produto às necessidades dos clientes de forma incremental, reduzindo-se os riscos do projeto;
- d) Pessoas de negócio e desenvolvedores devem trabalhar diariamente em conjunto por todo o projeto: pessoas de negócio e desenvolvedores do produto possuem o objetivo comum de garantir a geração de valor para os clientes e, para realizar esse objetivo, cooperam continuamente durante todo o projeto, interagindo com frequência.
- e) Construa projetos em torno de indivíduos motivados. Dê a eles o ambiente e o suporte necessário e confie neles para fazer o trabalho: o produto é construído por pessoas. O ambiente, o suporte e a confiança necessários para realizar seu trabalho são fatores fundamentais para sua motivação.
- f) O método mais eficiente e eficaz de transmitir informações para e entre uma equipe de desenvolvimento é através da conversa face a face: a comunicação face a face é direta, síncrona e enriquecida pela entonação de voz, olhar e linguagem corporal, entre outros fatores. Quando a comunicação presencial não for viável, uma boa prática é fazer uso da tecnologia.
- g) Software funcionando é a medida primária de progresso: o progresso do projeto ocorre à medida que partes do produto que agreguem valor são entregues aos clientes.
- h) Os processos ágeis promovem desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente: busca-se promover um ritmo constante e sustentável para o trabalho do time que desenvolve o produto, o que se torna possível quando o ritmo é apoiado por toda a cadeia, incluindo usuários e patrocinadores;

- i) A continua atenção a excelência técnica e bom projeto aumenta a agilidade: o produto projetado com qualidade e produzido com excelência técnica permite que possa ser modificado e, assim, que aceite a mudança como natural do processo de seu desenvolvimento;
- j) Simplicidade – a arte de examinar a quantidade de trabalho não realizado – é essencial: evita-se o desperdício no desenvolvimento do produto ao não se realizar trabalho que não é necessário. Exemplos comuns de desperdícios incluem desenvolvimento de funcionalidades de que os clientes não precisam;
- k) As melhores arquiteturas, requisitos, e projetos emergem de equipes auto organizáveis: equipes com maior autonomia são mais eficientes. Essas equipes trabalham em direção a metas acordadas, mas tem autonomia para decidirem quanto trabalho serão capazes de realizar, planejarem qual a melhor forma de realiza-lo e monitorarem seu progresso;
- l) Em intervalos regulares, a equipe reflete sobre como se tornar mais eficaz e então refina e ajusta seu comportamento de acordo: para de tornar cada vez mais efetiva, a equipe regularmente inspeciona suas formas de trabalho, identifica pontos de melhoria e se adapta de acordo, promovendo a melhoria incremental continua. É a inspeção e adaptação que o time realiza em seus processos de trabalho.

Segundo Munoz (2011), métodos ágeis representam uma alternativa para o desenvolvimento de sistemas e de software, com foco no fator humano e do produto de software e aumentando o relacionamento com clientes. Estes métodos fornecem entregas frequentes de software funcionando, permitindo alterações de requisitos e envolvimento direto do cliente.

Takeuchi e Nonaka (1986) foram os primeiros a utilizar o termo Scrum no contexto de desenvolvimento para descrever uma nova forma de trabalhar dentro de um processo acelerado, respondendo a necessidade de ser cada vez mais competitivo. A comparação com o jogo de *rugby* se faz presente pela similaridade entre a forma de jogar desse esporte e da necessidade de que se modifique a maneira de trabalhar entre as equipes de desenvolvimento, visto que o *rugby* é altamente coordenado e colaborativo (GÓMEZ *et al.*, 2017).

O Scrum foi criado quando Schwaber e Beedle (2001) perceberam que o desenvolvimento de software é uma atividade imprevisível. Esse método ágil define um *framework* de gestão, conduzido por um profissional chamado de Scrum Master. No Scrum, a iteração dura 30 dias e é chamada de *Sprint*. Toda a equipe deve trabalhar nos requisitos definidos no início de cada *Sprint* (ABRAHAMSSON *et al.*, 2002; e SCHWABER e BEEDLE, 2001).

Segundo Pressman (2010) “O Scrum incorpora um conjunto de padrões de processo (valores) que enfatiza prioridades do projeto, unidades de trabalho compartimentalizadas, comunicação e *feedback* constante com o cliente”.

O Scrum é definido como um processo gradual e iterativo para gerenciar e controlar o trabalho de desenvolvimento, pois se inicia com uma visão que é a base para o planejamento. Schwaber e Sutherland (2013) afirmam que o Scrum é fundamentado nas teorias empíricas de controle de processo e que três pilares apoiam a implementação de um processo empírico: transparência, inspeção e adaptação.

Ainda Schwaber e Sutherland (2013) definem transparência de aspectos significativos do processo que devem estar visíveis aos responsáveis pelos resultados. Esta transparência requer aspectos definidos por um padrão comum para que os observadores compartilhem um mesmo entendimento do que está sendo visto. Os mesmos autores dizem que os usuários Scrum devem, frequentemente, inspecionar os artefatos Scrum e o progresso em direção a detectar variações. Esta inspeção não deve, no entanto, ser tão frequente que atrapalhe a própria execução das tarefas. As inspeções são mais benéficas quando realizadas de forma diligente por inspetores especializados no trabalho a se verificar.

Em relação a adaptação Schwaber e Sutherland (2013) afirmam que um ou mais aspectos de um processo desviou para fora dos limites aceitáveis, e que o produto resultado será inaceitável, o processo ou o material sendo produzido deve ser ajustado. O ajuste deve ser realizado o mais breve possível para minimizar mais desvios.

Segundo Cruz (2018) o *Backlog*⁴ é o principal artefato do Scrum. Ele reúne os requisitos do produto a ser entregue, bem como todo o entendimento necessário para atender aso requisitos, produzir funcionalidades, e por fim, entregar o produto.

O mesmo autor afirma que o *backlog* pode ser dividido em conjuntos menores:

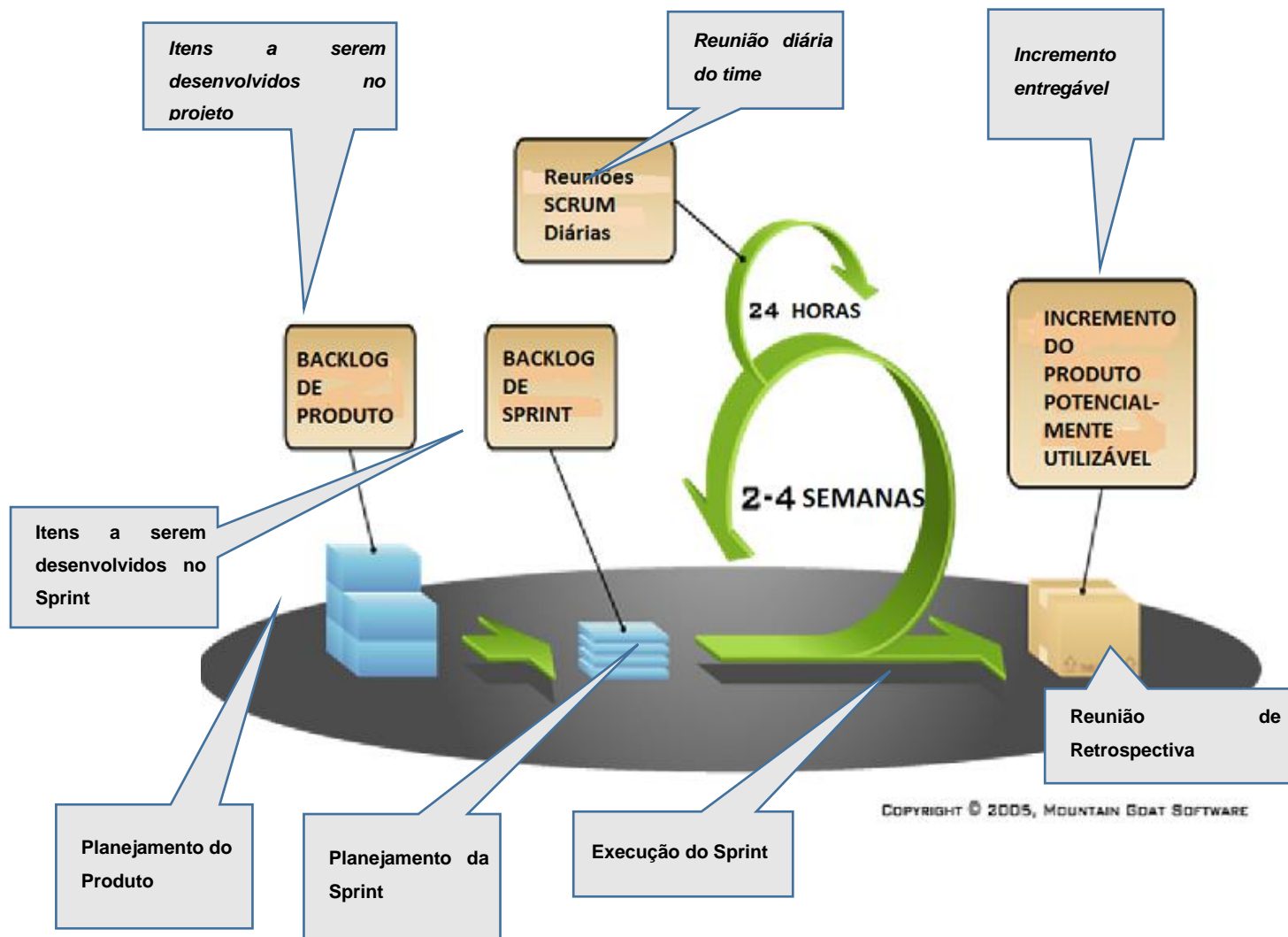
- a) *Backlog* de produto (*Product Backlog*) que é todo *Backlog* que será trabalhado ao longo do projeto. É o lugar que contém os requisitos do cliente priorizados e estimados.
- b) *Backlog* da versão de entrega que é a parte do *Backlog* que será trabalhado na versão de entrega definida entre o Time Scrum e o cliente;
- c) *Backlog* de *Sprint* que é somente a parte do *backlog* considerada preparada e selecionada para ser trabalhada na versão do *Sprint*.

As prioridades dos itens do *Product Backlog* determinam o quanto de valor cada item gera para o negócio. Depois de priorizados os itens, antes de cada iteração (*Sprint*), a equipe se reúne para informar quantos itens é possível entregar em um *Sprint*, que segundo Schwaber e Beedle (2001) deve durar cerca de 30 dias, como boa prática. Sabbag (2013) afirma que a duração das *Sprints*, que era originalmente de um mês, passou a poder ser tão curta quanto duas semanas.

Na Figura 9 pode-se observar as etapas do Scrum com algumas atividades associadas.

⁴ Backlog refere-se a um log (resumo histórico) de acumulação de trabalho num determinado período de tempo

Figura 9: Etapas do Scrum



Fonte: Adaptado de Costa (2013) “*Agile Project Management with Scrum*”

Segundo Cruz (2018) o Time Scrum é a união de todos os papéis do Scrum delimitando-se aos três papéis e às suas responsabilidades conhecidas. O *Product Owner* define o que é preciso para ser construído e a ordem de construção. O Time, ou desenvolvedores, ou time de desenvolvimento entram no processo entendendo as necessidades trazidas pelo *Product Owner* e definindo como realizarão os trabalhos necessários para atingir a meta do projeto. Também é responsável por determinar a sua própria capacidade e velocidade do trabalho. O autor sugere que o Time seja formado por no mínimo três e no máximo nove integrantes.

Ainda Cruz (2018) afirma que o *Scrum Master* assume a responsabilidade de garantir que os desenvolvedores sigam as regras do Scrum durante os trabalhos, além

de ser a pessoa mais indicada para remover obstáculos, orientar resolução de problemas que atrapalhem a evolução do trabalho.

Finalmente o autor afirma que diversos outros papéis podem contribuir para um melhor rendimento do Time de desenvolvimento e um maior aproveitamento de habilidades e competências. Papéis como arquitetos de software, especialistas em linguagens de programação específicas, administradores de banco de dados, especialistas em infraestruturas, testes, analistas de qualidade, entre outros, podem ser benéficos, fazendo parte do Time.

Segundo Sabbagh (2013), no Scrum, os ciclos são chamados de *Sprints*. Eles têm tamanho fixo e acontecem um depois do outro, sem intervalos entre eles, ou seja, tudo em um projeto que utiliza Scrum acontece dentro dos *Sprints*. Cada *Sprint* tem um formato bem estabelecido, que é composto pelos seguintes eventos:

- a) Reunião de Planejamento do Sprint ou *Sprint Planning*, momento que se realiza o planejamento da própria *Sprint*. Essa reunião deve ser finalizada com um objetivo claro do que deve ser realizado na iteração que se segue. A equipe seleciona os itens do *Product Backlog* e assume o compromisso a realizar durante o *Sprint*. Esses itens serão divididos em tarefas de uma forma colaborativa;
- b) Trabalho de desenvolvimento do produto, que inclui tudo que é necessário para se entregar uma parte do produto pronta ao final do Sprint;
- c) Reunião Scrum Diárias ou *Daily Scrum*, contida no dia a dia de desenvolvimento, que é uma reunião diária para gerar visibilidade e planejar o próximo dia de desenvolvimento. Esse é o momento de sincronização da equipe em que cada membro comenta com o restante em que estado se encontra o trabalho que está realizando e de que forma pensa em continuar. É um momento também de expor se tem algum impedimento ou dificuldade em continuar seu trabalho. Essas reuniões não devem demorar mais do que 15 minutos;
- d) Reunião de Sprint e Revisão de Sprint ou *Sprint Review*, momento que se obtém *feedback* dos clientes e demais partes interessadas sobre os resultados do trabalho realizado no Sprint;

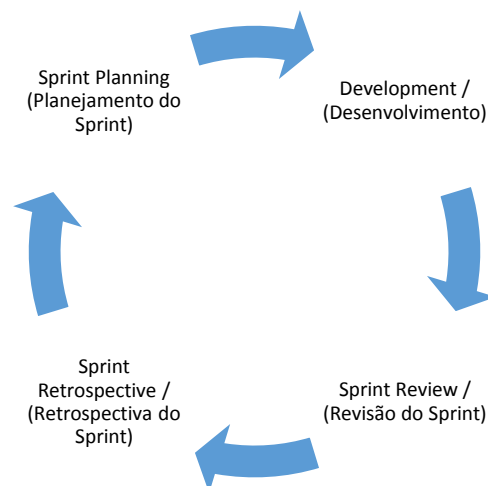
- e) Reunião de Retrospectiva ou *Sprint Retrospective*, momento que o time avalia o que foi bem e o que precisa ser melhorado na forma de trabalho;
- f) Quaisquer reuniões ou sessões programadas adicionais que se fizerem necessárias, como por exemplo, reunião de *Release Planning*, sessões de Refinamento do *Product Backlog*, etc.

Todas essas reuniões devem seguir o chamado “*Time-boxing*”, ou seja, fixar uma duração temporal e obedecê-la com fidelidade no sentido de alcançar maior produtividade (GÓMEZ *et al.*, 2017).

Segundo Taniguchi e Correa (2009) o Scrum possui também a definição de “pronto”, que é um acordo entre a equipe e *Product Owner* no sentido de fixar as atividades a serem realizadas em um Sprint. Na prática, se uma história de usuário for considerada terminada, ela deve obrigatoriamente encaixar-se no conceito de “pronto” que foi acordado. A definição de pronto inicial deve ser realizada antes do primeiro *Sprint*.

A Figura 10 representa a iteração do Scrum, indicando a sucessão de elementos presentes em um Sprint. A reunião de *Daily Scrum* e quaisquer reuniões adicionais acontecem no dia a dia do desenvolvimento.

Figura 10: Sucessão de iterações no Scrum



Fonte: SABBAGH (2013)

Segundo Schwaber e Sutherland (2013) afirmam que o Scrum nos times de Scrum associadas a papéis, eventos, artefatos e regras. Os mesmos Schwaber e Sutherland (2013), antes do Sprint é realizada uma reunião (*Sprint Planning Meeting*)

entre o time e o PO para que o PO possa apresentar as prioridades e que o time possa estima-las. A execução do Sprint é monitorada pelo time nas reuniões diárias (*daily meeting*), com duração de 15 minutos para verificação do progresso das atividades utilizando um gráfico chamado *burndown chart*⁵. No final do Sprint realiza-se uma reunião de revisão (*sprint review*), momento que o PO valida as atividades desenvolvidas pelo time. Após, é realizada a reunião de retrospectiva (*sprint retrospective*), ocasião em que são levantados os pontos positivos e negativos do Sprint e são criadas ações de melhoria para as próximas *Sprints*.

Liubchenko (2016) define que um processo com método Scrum deve conter as seguintes práticas (Pn) envolvidas:

- a) Pequenas Equipes (P1), e recomenda para tanto o tamanho da equipe até sete pessoas. Isso significa que o Gerente de Projeto (PM) tem que gerenciar e distribuir o trabalho entre as equipes e os líderes de equipe. As pequenas equipes ajudam a quebrar barreiras de comunicação, permitindo uma comunicação fácil e informal entre todas as partes nas equipes (Desenvolvedores, PM e Cliente). Pequenas Equipes facilitam todos os níveis de tomada de decisão, envolvendo um número menor de indivíduos em decisões de nível mais baixo.
- b) Compilações Frequentes (P2), e recomenda fornecer a construção de executáveis que podem ser frequentemente inspecionados, ajustados, testados, documentados e compilados. Os Builds permitem que o PM veja o estado do sistema a qualquer momento para acompanhar o progresso. Essa prática permite que os Desenvolvedores comuniquem o status do sistema com outros Desenvolvedores e PM a qualquer momento, além de ajudar o PM a planejar e monitorar Liberações e Desenvolvimento.
- c) Pacotes de Baixo Acoplamento (P3), e recomenda o particionamento de tarefas de trabalho e de equipe em partições ou pacotes limpos e de baixo acoplamento. Os Pacotes de Baixo Acoplamento influenciam

⁵ Burndown Chart é um gráfico que representa o trabalho pendente da equipe. Serve para acompanhamento da produtividade da equipe durante uma iteração.

como o PO a distribuir o trabalho, reduz a necessidade de comunicações técnicas entre os Desenvolvedores e ajuda a orientar as decisões de Design.

- d) Testes Constantes (P4), pela razão de fornecer ao PO uma visão geral do sistema e permite que os desenvolvedores conheçam o estado atual do sistema a qualquer momento.
- e) Documentação Constante (P5), pela razão de fornecer uma “foto” atualizada do sistema e seu progresso. O PO pode usá-las para rastreamento ou para trazer novas pessoas para aumentar velocidade do projeto. Ao fornecer uma documentação abrangente e atualizada, qualquer parte interessada pode aprender sobre seus respectivos interesses no sistema. A prática determina que a documentação seja produzida e mantida atualizada não apenas para o código, mas também para os requisitos e o planejamento da Liberação. A documentação fornecida ajuda a guiar o PO.
- f) Controles Iterativos (P6) por conta de ajudar a calibrar os requisitos, a funcionalidade, o risco e permitir ao PO planejar as iterações. A prática fornece um meio pelo qual Clientes, PO e Desenvolvedores colaboram para planejar iterações.
- g) Capacidade de declarar o projeto feito a qualquer momento (P7) que representa uma filosofia de gerenciamento que enfatiza a usabilidade e a correção do sistema, em vez do crescimento estrito de recursos. Essa capacidade afeta os tipos de recursos ou correções que são incorporados na próxima versão e também altera a mentalidade com a qual o gerente gerencia o projeto.

Segundo Massari (2017) o Scrum pode trazer as seguintes vantagens em um projeto:

- a) Maior produtividade e menores custos. Gerar entregas com valor implícito ou explícito e o foco em obter rápidos *feedbacks* do cliente e trabalhar construindo qualidade eliminam os chamados custos de não conformidade, que são gerados por retrabalho ou defeitos encontrados pelo cliente.

- b) Aumento do engajamento da equipe. O trabalho em equipe é um dos fatores cruciais para o sucesso ou insucesso da transição para o Scrum. O conceito de equipes auto organizadas e a filosofia do trabalho colaborativo com o cliente criam um senso de engajamento dentro da equipe.
- c) *Time-to-market*⁶ mais rápido. São duas as razões para os projetos que utilizam Scrum terem uma maior rapidez. A primeira é que as equipes Scrum possuem alta produtividade que permite desenvolver funcionalidades mais rapidamente. A outra é que cada *Sprint* gera um incremento no produto, o que possibilita que o mesmo seja lançado no mercado mesmo que todas as funcionalidades ainda não estejam concluídas.
- d) Maior qualidade. É fortemente incentivado que as equipes Scrum utilizem alguma técnica para a garantia de qualidade, tais como desenvolvimento orientado a testes, programação em par, entre outras.
- e) Aumento da satisfação das partes interessadas. O Scrum e suas entregas iterativas e incrementais constantemente, suas reuniões de revisão ou retrospectiva, dão visibilidade ao progresso do projeto. As pessoas se sentem participando do processo como todo e não apenas na tradicional reunião de *Kick-off*.

As práticas específicas de diferentes padrões internacionais seguido pela indústria de software e as identificadas no Scrum nos levam a um novo quadro que ajuda a minimizar os problemas de integração nos procedimentos de desenvolvimento. Vale a pena notar que essas práticas adotadas na indústria melhoram a capacidade dos processos específicos da indústria e como resulta que aumenta o Retorno sobre o Investimento (ROI) (HAYAT, 2016).

Entre os métodos ágeis existentes, o Scrum incorporado com as práticas da filosofia Lean pode fornecer às pequenas empresas um outro diferencial competitivo. Empresas de software, precisam ser flexíveis, criativas, dinâmicas e capazes de entregar produtos rapidamente por motivos de sobrevivência (O'CONNOR e COLEMAN, 2007).

⁶ Tempo entre a análise de um produto e sua disponibilização para a venda.

Termina-se essa seção com o Quadro 7 que apresenta algumas práticas Scrum extraídas do livro: *A guide to the Scrum Body of Knowledge* (ScrumSTUDY, 2016):

Quadro 7: Práticas Scrum do Guia ScrumSTUDY

Nome da prática	Paginas
Scrum Core Team (Time principal do Scrum)	41, 103
Product Owner (Dono do Produto)	41, 43-45, 300
Scrum Master (Orientador Scrum)	41, 45-47, 309
Scrum Team (Time Scrum)	156, 309
Sprint (Ciclo de Desenvolvimento)	33, 36-37, 178, 295, 310
Sprint Planning (Planejamento do Sprint)	34, 208, 311
Daily Scrum (Daily Stand-up Meeting)	223-224, 286, 289
Sprint Review (Revisão do Sprint)	34, 247, 311
Sprint Retrospect (Retrospectiva do Sprint)	236, 249-253
Product Backlog (Fila de Produto)	111, 266, 300
Prioritized Product Backlog (Fila do Produto Priorizada)	86, 90, 100, 110-111, 167-173, 190, 300
Sprint Backlog (Fila do Sprint)	205-209, 311
Release Planning Schedule (Cronograma do Planejamento da Release)	174-179, 304

Fonte: *A guide to the Scrum Body of Knowledge* (ScrumSTUDY, 2016)

Todas essas práticas Scrum foram utilizadas no desenvolvimento do processo PROÁGIL-29110.

3.5 LEAN SOFTWARE

A filosofia “*Lean Thinking*” ou Pensamento Enxuto nasceu em meados dos anos 1990 com o lançamento do *best seller* “*The Machine That Changed the World: The Story of Lean Production*”. Os princípios de produção puxada (*pull systems*), *just-in-time*, qualidade total, teoria das restrições, melhoria contínua e flexibilidade aplicados na indústria japonesa, mais precisamente na Toyota e conhecidos como Toyota

Way inspiraram também a indústria de software e fez surgir a abordagem do *Lean Software Development* (RAZZAK, 2016).

Poppendieck (2003), diz que o desenvolvimento de software utilizando práticas “Lean”, também chamada de “Lean Software”, oferece às organizações um conjunto de princípios a partir dos quais pode englobar processos de engenharia de software que funcionem melhor no contexto de seus clientes, seu domínio e sua capacidade de desenvolvimento. O autor afirma, no que se refere a interpretações do pensamento enxuto no desenvolvimento de software, que os princípios perfazem o total de sete (7), como segue:

I. Eliminar os desperdícios

O desperdício é algo que não agrega valor ao produto, e isso é percebido pelo cliente. No pensamento Lean, desperdício é sinônimo de obstáculo. São exemplos de desperdícios: requisitos para um projeto foram coletados de uma maneira difusa; uma planta de fabricação ter mais equipamentos do que o necessário; os programadores codificarem mais funcionalidades do que é necessário. O ideal é encontrar o que o cliente quer, e desenvolver e entregar exatamente o que corresponde as expectativas do mesmo. Qualquer coisa que seja feita fora desse contexto é desperdício. Algumas práticas podem ajudar na identificação de desperdícios:

- ✓ *Seeing Waste* ou enxergando desperdícios;
- ✓ *Value Stream Mapping* ou Mapeamento do Fluxo de Valor

II. Amplificar o conhecimento

O desenvolvimento é um exercício de descobertas. A abordagem Lean para desenvolvimento resulta em práticas muito diferentes do que práticas para produção. Desenvolvimento é como criar uma receita, enquanto a produção significa fazer um prato. Receitas são feitas por “*chefs*” experientes, que aprendem com o tempo, o que significa que uma receita ideal não é criada na primeira tentativa. Para o desenvolvimento de software, o processo é semelhante ao descrito, acrescentando-se que as equipes são maiores e os resultados mais complexos. A melhor abordagem para melhorar um ambiente de desenvolvimento de software é amplificar a

aprendizagem. Lições podem ser extraídas das experiências vividas pelas equipes e incorporada ao processo, fazendo com que as dificuldades passadas sejam fonte de conhecimento e contribuam para o amadurecimento da equipe e do processo. São exemplos de promoção do conhecimento:

- ✓ Ciclos de *feedback*, inspeções e adaptações;
- ✓ Desenvolvimento interativo;
- ✓ Equipes pequenas;
- ✓ Treinamentos e *mentoring*;
- ✓ Criação e utilização de padrões, *guidelines*, *templates* e artefatos;
- ✓ Revisões de Código ou *Code reviews*;
- ✓ Meios de compartilhamento de informações, tais como blog ou wiki.

III. Adiar decisões

As práticas de desenvolvimento que permitem uma criação tardia são eficazes em domínios onde ocorrem incerteza, porque fornecem uma abordagem baseada em opções. Adiar decisões permite que as escolhas sejam apoiadas por mais experiência e conhecimento adquiridos no decorrer do processo. Para retardar decisões durante a construção de sistemas é importante que a equipe crie a capacidade de absorver mudanças tratando os planejamentos como estratégias para atingir um objetivo e não como comprometimentos. Assim, mudanças serão vistas como oportunidades para aprender e atingir as metas. Algumas práticas para adiar decisões:

- ✓ Iterações;
- ✓ Reuniões planejadas ou *Planning meetings*;
- ✓ Desenvolvimento orientado por recursos.

IV. Entregas rápidas

Até recentemente, a rápida entrega de software não era valorizada. A estratégia de não cometer erros era vista como mais importante. Porém, o desenvolvimento rápido do software tem diversas vantagens, já que a velocidade de desenvolvimento também ajuda atender às necessidades atuais do cliente, além de permitir adiar a tomada de decisões para quando for acumulado conhecimento suficiente para tal.

Através de ciclos rápidos, o desenvolvimento caminha em um processo iterativo no qual o cliente refina suas necessidades e as obtém implementadas já no próximo ciclo. Iterações curtas trazem mais experiência para a equipe e aumentam a sua segurança para tomar decisões. São exemplos de práticas que apoiam as entregas rápidas:

- ✓ Kanban;
- ✓ Iterações.

V. Fortalecer a equipe

O time quando equipados com a experiência necessária e orientados por um líder, tomarão melhores decisões técnicas. Como as decisões são tomadas com atrasos e a execução é rápida, não é possível aguardar uma decisão de uma autoridade central. Assim, as práticas enxutas usam técnicas de extração para agendar as tarefas e mecanismos de sinalização locais para que os colaboradores possam se comunicar do que deve ser feito. Para um desenvolvimento de software “Lean”, a estratégia é fornecer ao cliente versões que vão se refinando em intervalos regulares. A sinalização é realizada por meio de gráficos visíveis, reuniões diárias, integração constante e testes abrangentes. Algumas práticas associadas para fortalecimento da equipe são:

- ✓ Auto-gestão;
- ✓ Trabalho em equipe;
- ✓ Feedback.

VI. Construir a Qualidade

Qualidade é uma questão fundamental. Entregar ao cliente qualidade intrínseca e explícita é sinônimo de satisfação. Existem duas dimensões de integridade: integridade percebida e integridade conceitual. A integridade percebida significa que a totalidade do produto alcança um equilíbrio entre funções, usabilidade, confiabilidade, economia e isso encanta o cliente. A integridade conceitual significa que os conceitos centrais do sistema desenvolvido são facilitados e coesos. É normalmente esperado que o software evolua com correção à medida que se adapta as funcionalidades. O software com integridade tem uma arquitetura coerente,

pontuação alta no que diz respeito a usabilidade e adequação, e é sustentável, adaptável e extensível. Isso pode ser conseguido levando-se em consideração a liderança sensata, experiência relevante, comunicação eficaz, e disciplina saudável. A promoção para a qualidade pode ser feita considerando práticas, tais como:

- ✓ Desenvolvimento Dirigido por Testes ou *Test Driven Development* (TDD);
- ✓ Refatoração ou *Refactoring*;
- ✓ Integração contínua;
- ✓ Revisão de Código / Inspeção de Código ou *Code Review / Code Inspection*;
- ✓ Padrões;
- ✓ Testes contínuos e automatizados.

VII. Otimizar o todo

A integridade em sistemas complexos exige uma profunda especialização em diferentes áreas. Os pontos de vista dos clientes e dos usuários equivalem a visões de alto nível do sistema. Otimizações macro canalizam os esforços para aumentar a satisfação dos usuários finais através de um produto consistente. Este princípio ainda recomenda a escolha de métricas de alto nível que sejam representativas para identificar a evolução. Essas métricas devem levar em consideração também a qualidade e a satisfação do cliente, pois a partir delas será possível avaliar quais trocas são vantajosas. Algumas práticas associadas à otimização do todo:

- ✓ Utilização de métricas que valorize o trabalho em equipe;
- ✓ Medida do ROI;
- ✓ Medida da satisfação do cliente.

Paralelamente aos sete princípios expostos, Kleim (2016) contribui com os principais valores que rodeiam o método de gerenciamento de projetos Lean:

- concentre-se no cliente;
- elimine os resíduos;
- valor acrescentado separado e não-valor acrescentado;

- o produto deve ser puxado pelo mercado em vez de empurrado para o mercado;
- adicione padronização;
- use a tecnologia disponível para sua vantagem;
- busque informações sobre dados;
- enfatize um fluxo de entrega contínua;
- aponte para a simplicidade e flexibilidade;
- veja a floresta através das árvores;
- tenha confiança no núcleo da cultura da organização;
- busque melhorias contínuas educando as pessoas.

Para Kniberg e Skarin (2010), o Scrum é dito por algumas pesquisas não ser apenas gerenciamento “Ágil” de projetos, mas também perto de gerenciamento “Lean” de projetos. Suetin *et al.* (2016), dizem que o gerenciamento de projetos Lean não difere muito do Scrum no que tange a nível de valor. Foco no cliente, entrega contínua, simplicidade, flexibilidade, confiança e transparência são valores semelhantes ao Scrum. O valor acrescentado da filosofia Lean é principalmente sobre a concentração nos resíduos, a fim de reduzir os custos do projeto, que foi uma das principais limitações do Scrum.

Redução de custo é uma desvantagem do Scrum, que é capaz de fornecer flexibilidade e qualidade, mas não tem a capacidade de gerenciar o orçamento e o cronograma (KUMAR e SHANKAR, 2016). Essa afirmação é corroborada por Lei *et al.* (2017), que afirma que Lean, mostra melhores resultados na gestão do cronograma do projeto em relação ao método Scrum e resultados ligeiramente melhores na gestão do orçamento do projeto.

Os dois principais argumentos que sustentam a abordagem de desenvolvimento de software Lean são que, quando aplicados corretamente, os projetos são concluídos com o menor custo possível e os projetos são concluídos rapidamente (POPPEDIECK, 2007).

Para Al-Baik (2015) a filosofia Lean é atualmente o método de desenvolvimento de produtos que mais cresce. Os estudos a respeito dessa filosofia estão, na maioria das vezes, fornecendo resultados semelhantes aos obtidos com o Scrum. Ambas os

métodos auxiliam na condução de projetos de sucesso. O desenvolvimento de software Lean está adquirindo uma identidade própria de como escalar o Agile. Mas Wang e Conboy (2012), perguntam se Agile e Lean são apenas dois nomes diferentes para a mesma coisa, ou se eles são realmente diferentes e, portanto, os desafios e problemas enfrentados pelos processos ágeis poderia ser abordado por abordagens Lean.

SjøBerg, Jphnsen e Solberg (2012) realizaram um estudo de caso, em uma empresa que trabalha com *Software Innovation* e substituiu o método Scrum por uma versão Lean e obteve resultados benéficos no tempo de entrega do produto, com redução de “bugs”⁷ em 10% e melhor produtividade. Outro estudo realizado por Cocco *et al.* (2011) que simulou os efeitos do uso do Scrum, Lean e *Waterfall*, concluiu que o Scrum é melhor que o *Waterfall*, mas o Lean é melhor que o Scrum porque usa *releases*⁸ frequentes, como o Scrum, mas também o WIP⁹ (*Work In Progress*) é mantido no mínimo.

Massari (2017) associa a filosofia Lean ao método ágil Scrum da seguinte maneira:

- a) Eliminar os desperdícios: como os *Sprints* são iterações curtas, a Equipe Scrum deve trabalhar de forma mais otimizada possível, evitando processos e documentação desnecessários, burocracia de comunicação, falta de testes na execução do *Sprint*, e principalmente defeitos ou bugs embutidos na entrega final do *Sprint*.
- b) Amplificar o conhecimento: o conhecimento compartilhado é uma característica das Equipes Scrum. Elas devem interagir o tempo todo.
- c) Adiar decisões: totalmente aderente ao conceito do Product Backlog emergente. Requisitos prioritários do Backlog devem estar bem detalhados, requisitos menos prioritários devem estar menos detalhados

⁷ Bug - defeito, falha ou erro no código de um programa que provoca seu mau funcionamento

⁸ Release - nova liberação de uma nova versão do software que está sendo desenvolvido.

⁹ WIP - abreviação de trabalho em andamento. É a quantidade de trabalho iniciada em um ativo ou item, mas que não foi entregue a um cliente.

- d) Entregas rápidas: totalmente aderente como conceito de *Sprints* de duração entre uma e quatro semanas.
- e) Fortalecer a equipe: equipe fortalecida é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção.
- f) Construir qualidade: efetuar testes automatizados, mostrar protótipos ao Product Owner e demais partes interessadas e eliminar defeitos e bugs identificados durante o Sprint.
- g) Otimizar o todo: o sucesso de um Sprint não é gerado pelas ações individuais de cada membro da Equipe Scrum e sim pelo trabalho colaborativo em equipe.

Um estudo de (Dharmapal e Sikamani, 2014), apresentado no Quadro 8 colocou os desafios enfrentados nos projetos Scrum e ofereceu soluções de práticas Lean.

Quadro 8: Comparação Scrum x Lean

Desafio Scrum	Solução Lean
Histórias de usuários indesejadas no backlog	Identifique resíduos em um estágio inicial
Integração de histórias em partes independentes de software é difícil	Olhe para o sistema como um todo, o tempo todo
Muitas histórias no backlog	Decida sobre histórias quando necessário
Ferramentas não capturam informações necessárias	Use a tecnologia como vantagem e adquira essa ferramenta
Recursos não disponíveis a tempo	Elimine o tempo de espera e planeje os recursos
Proprietário do produto que não compreende totalmente os requisitos e as metas	Veja o sistema como um todo e comece o desenvolvimento somente quando os requisitos estiverem claros
Indisponibilidade de colaboradores com habilidades adequadas	Enfatize que o trabalho correto deve ir para o recurso certo
Reuniões levam muito tempo	Encurte as reuniões, tanto quanto possível; traga mentalidade magra

Desafio Scrum	Solução Lean
O escopo não fica congelado, mas continua mudando	Congele o escopo o mais cedo possível para ajudar a obter o produto entregue mais rapidamente
Muita flexibilidade no prazo	Congele o prazo o mais rápido possível para uma entrega mais rápida
Nenhum critério de aceitação	Torne os critérios de aceitação claros para que a equipe não desperdice o trabalho em excesso
Estimativas impróprias de trabalho	Capacite a equipe para entender a importância das estimativas de trabalho
Histórias de usuários incorretamente priorizadas	De importância a priorização, isso ajuda a eliminar o desperdício
Não há tempo marcado para revisões	Capacite a equipe a fazer revisões
Nenhuma demonstração para o cliente	Demonstre artefatos quando possível para ajudar a criar honestidade e confiança para com o cliente

Fonte: DHARMAPAL e SIKAMANI, 2014

A filosofia Lean usa várias ferramentas para apoiar o gerenciamento de sua operação. Uma dessas ferramentas é chamada Kanban (baseada no Toyota Production System). De acordo com Anderson (2010), Kanban é uma abordagem para visualizar o fluxo de trabalho de um sistema de produção. Utiliza a teoria das filas para controlar e melhorar o fluxo de valor, chamando a atenção para o fluxo de produção.

Para representar as características do Kanban, equipes ágeis de desenvolvimento de software usam quadros radiadores de informações, no qual itens de trabalho são expostos em raias, que representam o andamento da produção destes itens tornando visível o fluxo da produção.

Na execução de um processo, diversos aspectos devem ser observados e mensurados. Neste sentido, o Kanban também possui um sistema de métricas, que possibilita a mensuração do tempo que uma história de usuário leva para ser concluída, e a eficiência em atingir um valor de referência. Essa mensuração pode ser feita nos dados do processo, por exemplo por meio de cartões, identificando o trabalho em progresso ou “*work in progress*” (WIP).

Estudos sobre o Kanban usando técnicas de simulação foram conduzidos para analisar a aplicabilidade e eficácia do Kanban no desenvolvimento de software (Anderson et al., 2011; Cocco *et al.*, 2011; e Concas et al., 2013). Por exemplo, Cocco *et al.* (2011) analisa o comportamento dinâmico da adoção de Kanban e Scrum em comparação com um processo de software tradicional. O resultado relatado foi que Kanban ajudou a controlar e gerenciar o fluxo de trabalho de forma eficaz, minimizando o tempo de entrega.

Segundo (Anderson, 2010) o foco principal do Kanban é afirmar com precisão o trabalho que precisa ser feito e quando isso precisa ser feito. Isso é feito priorizando as tarefas e definindo o fluxo de trabalho, bem como o prazo de entrega. O processo Kanban apresenta explicitamente as tarefas mais importantes que precisam de mais atenção para reduzir o risco de sua incompletude e também para aumentar a flexibilidade entre outras tarefas do projeto.

Segundo Lei (2017) no que se refere a desenvolvimento de software, o Kanban tem os seguintes princípios:

- a) Limitar o trabalho em andamento (WIP);
- b) Elevar valor através do processo de desenvolvimento;
- c) Tornar o processo de desenvolvimento visível;
- d) Aumentar o rendimento;
- e) Usar um backlog fixo;
- f) Incorporar qualidade.

Na Figura 11 apresenta-se um exemplo de kanban.

Figura 11: Exemplo de Kanban



Kniberg (2010) afirma que o Kanban e o Scrum são semelhantes por serem Lean e Agile, tendo a capacidade de dividir o trabalho em partes menores, equipes auto-organizadas, concentrando-se em entregar software mais cedo e frequentemente, adaptando as mudanças rapidamente, limitando o WIP, e usando transparência. Além disso, Keogh (2011) observa que ambos os métodos são mais eficientes que o método *Waterfall*. Eles contêm mecanismos de feedback e melhoria e explicam claramente o escopo do projeto. Eles também dão importância ao valor do projeto e à obtenção de resultados finais.

Puonti (2017), afirma que a estrutura de gerenciamento de projetos Lean, desenvolvida por Mary e Tom Poppendieck, abrange uma quantidade suficiente de práticas Lean e algumas práticas ágeis, mas há pesquisas que usam o método ágil Scrum como base e adicionam ferramentas Lean apresentadas por Poppendieck (POPPENDIECK e POPPENDIECK, 2003).

Para terminar esta seção, lista-se no Quadro 9, as Ferramentas de Desenvolvimento de Software Lean do livro "Desenvolvimento de Software Lean: Um Kit de Ferramentas Ágil" (POPPENDIECK e POPPENDIECK, 2003):

Quadro 9: Ferramentas Lean

Nome da Ferramenta	Páginas	Descrição
Melhoria continua	136-140	Método de trabalho que melhora a produtividade através de atividades baseadas no trabalho em equipe e na colaboração constante entre produção e manutenção
Mapeamento de Fluxo de Valor (VSM)	22-27	Método de fluxograma para ilustrar, analisar e melhorar os passos necessários para entregar um produto ou serviço.
Kanban	76-81	Método relacionado com a utilização de cartões (<i>post-it</i> e outros) para indicar o andamento dos fluxos de produção em empresas de fabricação em série.
Just-on-time	65-67	É um modelo japonês que procura eliminar estoques e agilizar a produção.
5S	18-22	Método que ajuda a criar a cultura da disciplina, identificar os problemas, e gerar oportunidades de melhoria.

Fonte: Práticas Lean (POPPENDIECK e POPPENDIECK, 2003), adaptada pelo autor

Dentre essas ferramentas o método Kanban foi utilizado no desenvolvimento do processo PROÁGIL-29110.

3.6 ESTRUTURAÇÃO DOS TEMAS ABORDADOS NO REFERENCIAL TEÓRICO PARA APLICACAO NA PESQUISA

Após a elaboração do referencial teórico exposto nos tópicos anteriores, o Quadro 10 apresenta um resumo dos principais temas, conceitos e autores considerados nesta pesquisa.

Quadro 10: Resumo sinóptico dos temas abordados no referencial teórico para aplicação na pesquisa

Tema	Conceito	Principais Autores	Aplicação na pesquisa
✓ Processo de software	✓ Definição: Hammer (1997) e Salviano (2006).	✓ Humphrey (1989); Hammer (1997); Salviano (2006); Pressman (2010).	✓ Desenvolvimento do PROÁGIL-29110
Qualidade de software	✓ Definição: Kitchenham, Pleeger (1996); Pressman (2010). ✓ Normas, modelos e a relação com melhoria de processos: Maciel (2011); Irama (2012).	✓ Pleeger (1996); Pressman (2010); Kitchenham e Maciel (2011); Irama (2012).	✓ Definição do PROÁGIL-29110 ✓ Objetivo do PROÁGIL – 29110 (promover a melhoria da qualidade nas empresas)
Modelo CMMI	✓ Definição: Chrissis <i>et al.</i> (2004). ✓ Histórico: Sória (2006). ✓ Relacionamento de modelos de qualidade com métodos ágeis: Palomino <i>et al</i> (2017).	✓ Chrissis <i>et al.</i> (2004); Sória (2006); Couto (2007); Pane (2015); Palomino <i>et al</i> (2017).	✓ Comparação com a ISO 29110 na escolha da norma para ser o pilar do PROÁGIL - 29110
Modelo MPS.Br	✓ Definição: Koscianski (2006); Kalinowski (2010); Kival (2018). ✓ Aplicação: Santos <i>et al</i> (2012).	✓ Koscianski (2006); Kalinowski (2010); Montoni (2010); Almeida (2011); Santos <i>et al</i> (2012); Kival (2018).	✓ Comparação com a ISO 29110 na escolha da norma para ser o pilar do PROÁGIL - 29110
Norma ISO/IEC 29110	✓ Definição: Laporte (2016). ✓ Aplicação em pequenas empresas: Dias (2016); Larrucea (2016); Gonzales (2016); Laporte (2018).	✓ Graziotin (2015); Dias (2016); Laporte (2016); Larrucea (2016); Gonzales (2016); Kuhrmann (2016); Laporte (2018);	✓ Definição do PROÁGIL-29110

Método ágil Scrum	<ul style="list-style-type: none"> ✓ Definição: Cockburn et al. (2001); Schwaber (2001); Sabaggh (2013); Costa (2013). ✓ Relação entre Scrum e processo de software: Takeuchi e Nonata (1986); Schwaber (2013); Gomez et al (2017). ✓ Eventos e práticas do Scrum: Sabbagh (2013); Liubchenko (2016); Gomez (2017); ✓ Relacionamento entre Scrum e a ISO/IEC 29110: Lopez-Lira (2014); Munoz (2014); Morales-Trujillo (2015); Allen (2016); Felderer (2016); Kuhrmann (2016); Pino (2016); Sanchez (2017); Berg et al. (2018), Politowski (2018). 	<ul style="list-style-type: none"> ✓ Takeuchi e Nonata (1986); Cockburn et al. (2001); Schwaber (2001); Costa (2013); Sabaggh (2013); Schwaber (2013); Lopez-Lira (2014); Munoz (2014); Morales-Trujillo (2015); Allen (2016); Felderer (2016); Pino (2016); Kuhrmann (2016); Gomez et al (2017); Sanchez (2017); Berg et al. (2018); Politowski (2018). 	<ul style="list-style-type: none"> ✓ Definição e incorporação dos artefatos, eventos, métodos e técnicas no processo PROÁGIL – 29110
Lean Software	<ul style="list-style-type: none"> ✓ Definição: Poppendieck (2003); Kleim (2016). ✓ Relacionamento entre Scrum e Lean: Kniberg (2010); Cocco et al. (2011); Kumar e Shankar (2016); Suetin et al. (2016), Lei et al. (2017), Massari (2017), Dharmapal e Sikamani (2014), 	<ul style="list-style-type: none"> ✓ Poppendieck (2003); Dharmapal e Anderson (2010); Kniberg (2010); Cocco et al. (2011); Concas et al. (2013); Sikamani (2014); Kleim (2016); Suetin et al. (2016); Kumar e Shankar (2016); Lei et al. (2017); Massari (2017). 	<ul style="list-style-type: none"> ✓ Incorporação da filosofia Lean Software ao processo PROÁGIL-29110

	✓ Lean e aplicação do Kanban: Anderson (2010), Concas <i>et al.</i> (2013); Lei <i>et al.</i> (2017).		
--	---	--	--

Esse quadro sinótico foi concebido considerando a ordem histórica de publicações dos autores citados.

4. PESQUISA-AÇÃO

Em relação à métodos científicos de pesquisa, além do método Systematic Literature Review (SLR), apresentado e utilizado no capítulo 2 para a revisão bibliométrica e sistemática, essa tese com intuito de validar o processo PROÁGIL-29110 aplicou o método de pesquisa-ação.

4.1 MÉTODO PESQUISA-AÇÃO

A pesquisa-ação é um método científico de pesquisa qualitativo, o que é uma característica adequada para os objetivos desta tese. Nessa visão aprofundada há interesse em “como”s e “por que”s, demandando uma pesquisa qualitativa e um ambiente real para desenvolvê-la (Yin, 2003), ambiente este disponível no contexto desta pesquisa.

Segundo Kock *et al.* (1997), a pesquisa-ação é um método de pesquisa que tem duplo objetivo:

- I. Pesquisa para ampliar o conhecimento científico;
- II. Ação para promover uma melhoria na organização ou comunidade onde a pesquisa está sendo realizada.

Ainda segundo Kock *et al.* (1997), a expressão “organização” aponta para a pesquisa-ação realizada com um grupo de profissionais numa instituição, como uma empresa ou uma escola; enquanto a expressão “comunidade” é usada em áreas que investigam grupos sociais, como moradores de um município ou idosos de uma determinada faixa etária. Nesta pesquisa, cujo objeto é a implementação de um processo de software em uma empresa, adota-se o termo “organização”.

A pesquisa-ação é uma abordagem à pesquisa social aplicada na qual o pesquisador e os atores envolvidos colaboram para o desenvolvimento de um diagnóstico e para a solução de um problema. (BRYMAN, 1989).

Em relação ao ponto de vista científico, de acordo com Thiollent (2018), a pesquisa-ação é uma proposta metodológica e técnica que oferece subsídios para organizar a pesquisa social aplicada sem os excessos da postura convencional ano

nível da observação, processamento de dados, experimentação, etc. Com ela se introduz uma maior flexibilidade na concepção e na aplicação dos meios de investigação concreta.

Freitas (2010) e Thiollent (2018) concordam que a pesquisa-ação tem sido usada de maneira restrita em pesquisas acadêmicas, porém Thiollent (2018) afirma que a pesquisa-ação possui relativa tradição em ambientes organizacionais e tecnológicos como uma forma de obtenção de informações, negociação de soluções para problemas de ordem técnico-organizativa, possibilitando uma maior participação de atores sociais em processos de tomada de decisão.

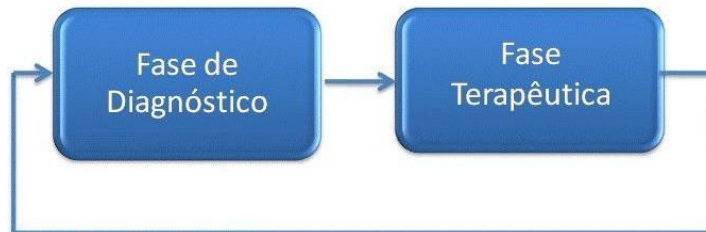
Conforme Martins e Theóphilo (2009) as principais características da pesquisa-ação são:

- Há uma ampla e conhecida interação entre as pessoas explícitas na pesquisa e o pesquisador;
- A interação entre os envolvidos e o pesquisador resulta a ordem de prioridade dos problemas a serem pesquisados e as respectivas ações a serem encaminhadas;
- O objeto da investigação emerge da situação social e pelos problemas de distintas naturezas encontrados e não pelas pessoas;
- O objetivo de uma pesquisa-ação é o de resolver ou ainda pelo menos esclarecer os problemas da situação observada;
- Durante o processo de pesquisa um acompanhamento das decisões, das ações, e de todas as atividades intencionais dos atores da situação é necessário;
- A pesquisa não se limita a uma única forma de ação, objetiva o aumento do conhecimento do pesquisador sobre o nível de consciência das pessoas e dos grupos considerados.

Em se tratando de processo, Thiollent (2008) afirma que a essência da definição de pesquisa-ação pode ser organizada em duas fases, conforme mostra a Figura 12. A primeira é a fase de diagnóstico, que envolve uma análise colaborativa da situação pelos pesquisadores e participantes da pesquisa. Como decorrência desse diagnóstico, são formuladas teorias em relação ao domínio da pesquisa. Em

uma segunda etapa, a fase terapêutica envolve experimentos e tentativas, realizados de forma colaborativa. Nessa fase, são introduzidas alterações e seus efeitos são estudados. Iteram-se as duas fases até que o problema seja resolvido.

Figura 12: Processo simplificado da pesquisa-ação



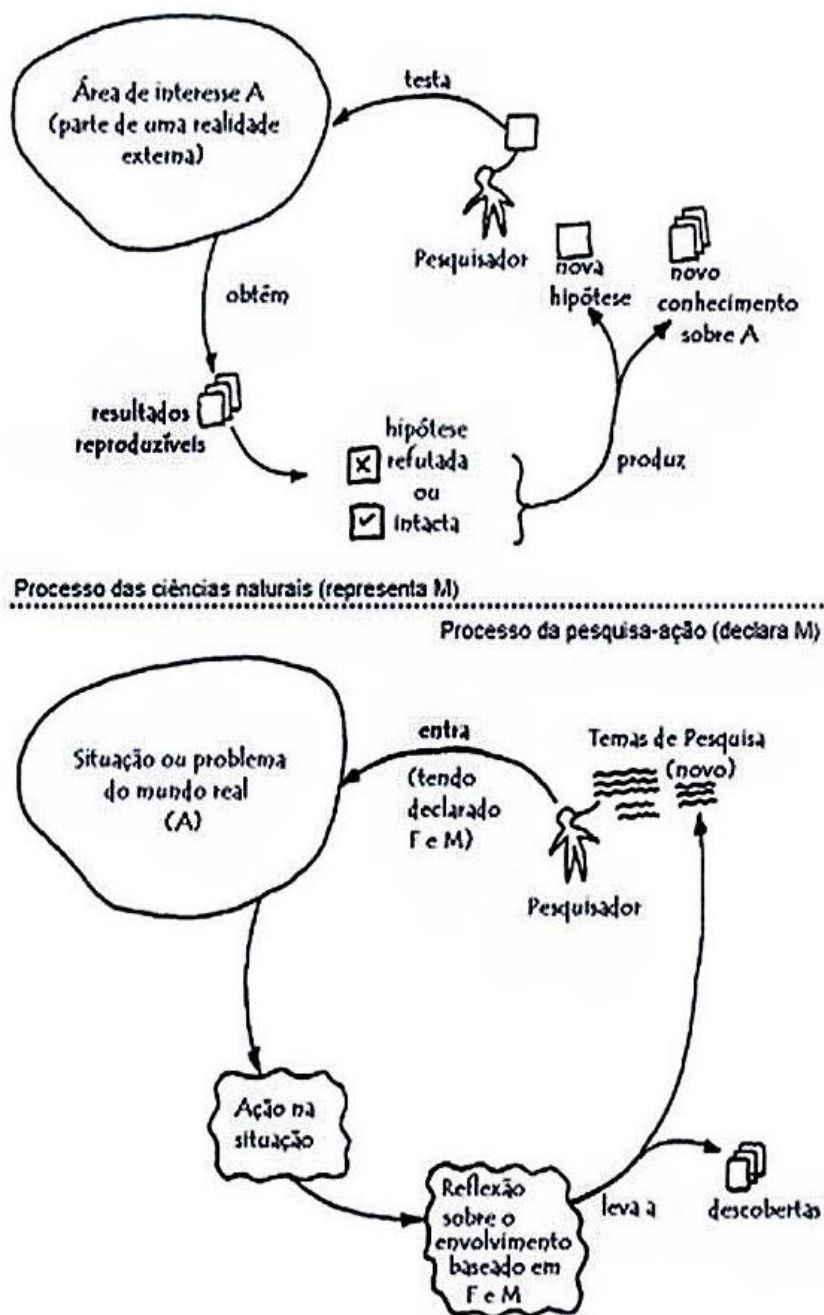
Fonte: Thiollent (2008)

Segundo McKay e Marshall (2001), a representação usual do processo de pesquisa-ação é feita por meio de um ciclo. Na sua origem, as duas etapas essenciais da pesquisa-ação são a de refletir sobre o problema (estágio de diagnóstico) e a de agir sobre o problema (estágio terapêutico).

Segundo Checkland e Holwell (1988), qualquer forma de pesquisa pode ser pensada como vinculada aos seguintes fatores: um conjunto coeso de ideias formando um arcabouço conceitual F o qual é utilizado em uma metodologia M para investigar uma área de interesse A, conforme Figura 13.

Por exemplo, em Qualidade de *Software* poderia se pensar na pesquisa desta tese, segundo a visão positivista: investigar se as pequenas empresas produtoras de software (VSE,s) (área de interesse A) podem trabalhar com um processo ágil aderente à norma ISO/IEC 29110 baseado em Scrum e princípios da filosofia Lean adaptadas a software (arcabouço conceitual F), a partir da aplicação da pesquisa-ação (metodologia M). Essa proposição é ilustrada na Figura 13.

Figura 13: Processos das ciências naturais e da pesquisa-ação



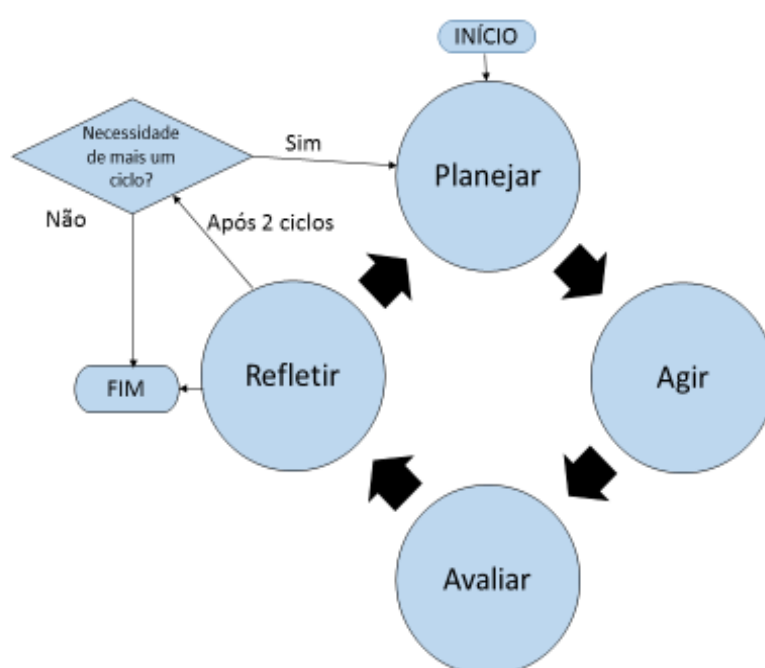
Fonte: Checkland e Holwell (1988)

Checkland e Holwell (1998) afirmam que a pesquisa-ação modifica os papéis de F, de M e até mesmo de A, na medida em que o pesquisador passa a estar envolvido no fluxo de transformações que ocorrem em uma determinada situação.

Na literatura que trata acerca de pesquisa-ação são encontradas diferentes formas de apresentar os ciclos, conforme observado em Kemmis e McTaggart (2005), Johnson (2004) e Baskerville (1999).

Esta tese adota para a pesquisa-ação ciclos com as quatro fases - Planejar, Agir, Avaliar e Refletir, adaptado de Kemmis e McTaggart (2005), esquematizadas na Figura 14.

Figura 14: Fases do ciclo da Pesquisa-ação



Fonte: Adaptado de Kemmis e McTaggart (2005)

A fase de “Planejar” deve considerar as ações alternativas para resolver o problema. Nessa fase são definidas as ações para o quadro diagnosticado. As definições são guiadas pelas hipóteses que representam as suposições formuladas pelo pesquisador a respeito de possíveis soluções e resultados. As hipóteses, por sua vez, devem ser acompanhadas de uma formulação teórica científica. A situação é pesquisada por meio de diversos tipos de instrumentos de coleta de dados, que são

discutidos e progressivamente interpretados pelos grupos que participam da pesquisa. (DAVISON *et al.*, 2004)

Conforme Thiollent (2008), a fase de “Planejar” pode ser formada por um conjunto de entrevistas individuais e coletivas ou questionários aplicados a pessoas chaves da organização, que irão expor suas queixas, constatações e sugestões a respeito do tema em pauta. Todas essas informações coletadas entre os entrevistados servem como base para posterior debate.

Na fase “Planejar”, planeja-se o ciclo da pesquisa-ação, definindo-se a solução (Implementação do PROÁGIL-29110) a ser investigada, as ações a serem realizadas, os dados a serem coletados e a forma como será feita a análise desses dados. Nessa fase também é definido o ambiente da pesquisa, que no caso desta pesquisa, são duas organizações que serão objetos da implementação e descritas no Capítulo de Análise de Dados e Discussões.

A fase “Agir” corresponde à implantação das ações planejadas. Um elemento central nessa etapa, segundo Thiollent (2008), é a técnica de seminário, utilizada para examinar, discutir e tomar decisões acerca do processo de investigação. Essa fase incide em, com base nas investigações em curso, expandir os resultados, definir objetivos alcançáveis por meio de ações concretas e apresentar sugestões que poderão ser negociadas entre as partes interessadas. Ainda segundo o mesmo autor, essa fase engloba medidas práticas baseadas nas etapas anteriores: difusão de resultados, definição de objetivos alcançáveis por meio de ações concretas, apresentação de propostas a serem negociadas entre as partes interessadas e implementação de ações-piloto que posteriormente, após avaliação, podem ser assumidas pelos atores sem atuação dos pesquisadores. Na fase “Agir”, a solução é aplicada. No caso desta pesquisa, a fase Agir consistiu na implementação do PROÁGIL-29110, descrito no Anexo A, e os apêndices de D a M, em conjunto com a equipe de projeto/pesquisadores, e na disseminação do aprendizado do processo junto às equipes das empresas participantes do estudo, conforme definido na etapa de “Planejar”.

Davison *et al.* (2004) afirma que na fase “Avaliar” realiza-se a análise dos efeitos das ações frente ao apoio teórico utilizado como ponto de partida para a definição das ações e que tem por objetivos: observar, redirecionar o que realmente

acontece e resgatar o conhecimento produzido no decorrer do processo. Essa etapa do processo de pesquisa-ação apresenta dois objetivos principais. O primeiro é verificar os resultados das ações no contexto organizacional da pesquisa e suas consequências a curto e médio prazos. O segundo é extrair ensinamentos que serão úteis para continuar a experiência e aplicá-la em estudos futuros. Na fase “Avaliar”, é realizada a coleta dos dados quantitativos e/ou qualitativos durante o período previamente definido na fase “Planejar”. A coleta desses dados se deu tanto nos resultados da implementação do PROÁGIL-29110 quanto por meio de observações, entrevistas e outras formas.

Finalmente a fase “Refletir”, o mesmo Davison *et al.* (2004) diz que se trata da aprendizagem específica e da identificação dos ensinamentos da experiência, com retorno ao ponto de partida para evidenciar o conhecimento generalizável adquirido sobre o problema. Ela envolve a circulação de informação entre os participantes e outros setores da organização. A aprendizagem é facilitada pela colaboração temporária com especialistas em assuntos técnicos cujo conhecimento tenha sido útil ao grupo.

Na fase “Refletir”, é realizada a análise dos dados resultantes da aplicação do PROÁGIL-29110 e, se necessário, de outras formas complementares, e uma reflexão sobre os resultados obtidos. Nessa reflexão, busca-se identificar os efeitos decorrentes da aplicação do processo, verificar se o mesmo foi apropriado por todos os envolvidos e, também, a necessidade de ajustes e refinamentos para a aplicação em um próximo “*Sprint*”. Para cada organização foram realizados dois *Sprints*. Ao término do segundo *Sprint*, se as observações não fossem suficientes para a análise de dados, planejar-se-ia um novo *Sprint*. Outros tantos seriam replanejados, conforme a necessidade do pesquisador.

Thiollent (2008) afirma que a apresentação dos resultados é feita aos respondentes das entrevistas e os demais atores envolvidos na pesquisa. Uma vez divulgadas as informações, é iniciada a etapa de apresentação de propostas visando melhorar os aspectos estudados. Essas propostas são analisadas por membros da gestão e implementadas de forma a colocar em prática as sugestões apresentadas.

Esta pesquisa-ação, que abrangeu a implementação do processo PROÁGIL-29110 em duas organizações do tipo (VSE), permitindo a coleta de dados, a

elaboração das conclusões acerca dos benefícios, dificuldades e lições aprendidas que permitiram retroalimentar o processo concebido no sentido de melhorá-lo.

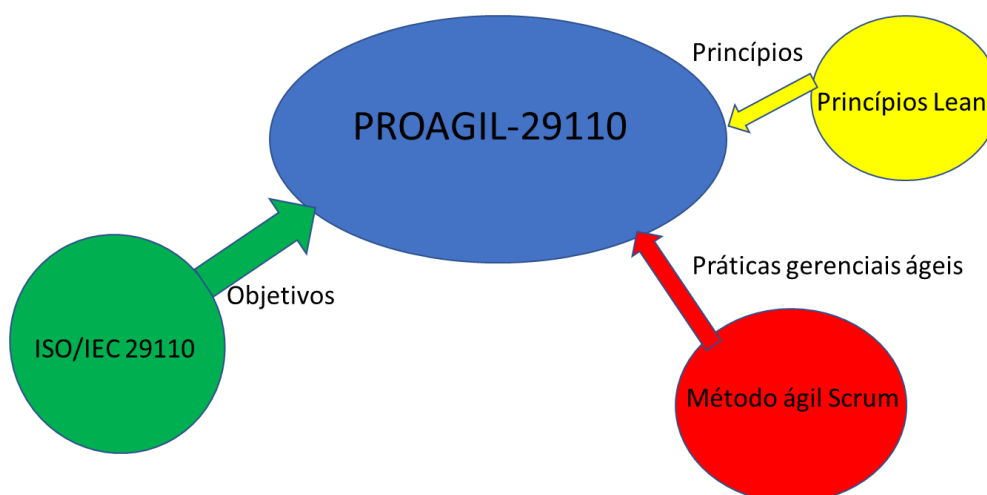
A partir de entrevistas com os colaboradores que utilizaram o PROÁGIL- 29110 e da observação direta foram elencados benefícios e dificuldades em relação ao uso do processo. Adicionalmente, foi realizada uma análise documental que contemplou artefatos produzidos ao longo das *Sprints* dos projetos.

5. MODELO E PROCESSO PROÁGIL-29110 PROPOSTO

Nesse capítulo é descrito o processo PROÁGIL-29110 a partir do modelo proposto.

O modelo proposto, apresentado na Figura 15, é resultado das questões de pesquisa elaboradas e respondidas na revisão da literatura efetuada. O modelo proposto teve como propósito desenvolver e implantar em pequenas empresas de software brasileiras um processo ágil aderente à norma ISO/IEC 29110, contemplando o método ágil Scrum e princípios Lean adaptados a software.

Figura 15: Modelo Proposto



O processo PROÁGIL-29110 será composto por:

- a) Um conjunto de objetivos preconizados pela norma ISO/IEC 29110 que são organizados em atividades ou tarefas necessárias para um processo que seja aplicável a pequenas empresas de software (VSE). Todos os objetivos da norma estão contemplados na solução devido à possibilidade de as organizações poderem obter uma certificação internacional com a finalidade de ter um diferencial de mercado. Esse conjunto é representado no modelo como ISO/IEC 29110.

- b) A esses objetivos da norma é acrescida um conjunto de práticas ágeis, obtido do método ágil Scrum, voltado para gerenciamento do processo PROÁGIL-29110. Esse conjunto de práticas ágeis é representado no modelo como o método ágil Scrum. A técnica “Burndown” é usada no método Scrum.
- c) Com relação ao processo PROÁGIL-29110 ainda são incorporados os princípios da filosofia Lean visando a integração das pessoas envolvidas nas equipes de desenvolvimento. A técnica “Kanban” é usada na implementação do Lean.

O processo PROÁGIL-29110 está organizado da seguinte forma:

- a) Está dividido em duas etapas: a primeira contemplando Gerência de Projetos e a segunda contemplando Implementação de software, que formam os dois processos do perfil básico da norma ISO/IEC 29110;
- b) Cada etapa contém uma série de subprocessos (4 para o processo de Gerência de Projetos e 6 para o processo de Implementação de software), organizadas dessa forma com o intuito de representar as atividades de cada processo descritas na norma ISO/IEC 29110;
- c) Cada subprocesso possui requisitos padrão que são: objetivo, descrição, responsável, envolvidos, templates associados, ferramentas, tarefas, entradas e saídas.

O processo PROÁGIL-29110 está descrito no ANEXO A.

5.1 VALIDACAO DO PROÁGIL-29110

Antes da aplicação da pesquisa-ação em campo, o processo PROÁGIL- 29110 foi enviado para dois especialistas para validação. A primeira especialista trabalha há 27 anos em melhoria de processos, é avaliadora do modelo MPS.Br para software e serviços e participou do comitê brasileiro que auxiliou na definição da norma ISO/IEC 29110. Os critérios de validação foram: conformidade (capacidade de aderência à norma ISO/IEC 29110), aceitabilidade (capacidade do processo ser aceito nas

organizações) e reprodutibilidade (capacidade de obtenção de resultados semelhantes se usado em diversas organizações).

Em relação à conformidade, o processo foi considerado aderente, porém foram feitas 12 modificações, todas sugeridas pela especialista e mostradas no Quadro 11.

Quadro 11: Alterações no Processo PROÁGIL-29110

Item	Antes da verificação	Após a verificação	Motivo do aceite das modificações sugeridas
1	Saída de artefato “Backup do projeto” não contemplada no Processo	Saída do artefato “Backup do projeto”, contemplado no item 1.1 do Processo	O artefato faz parte das saídas de artefatos exigidos pela norma
2	Tarefa: “Analisar eventuais solicitações de mudanças avaliando impacto, utilizando o Template Solicitação de Mudança – SM”, contemplada no item 1.2 do Processo	Tarefa: Analisar eventuais solicitações de mudanças avaliando impacto, utilizando o Template Solicitação de Mudança – SM, atualizar o Plano de Projeto se necessário e realizar as devidas correções, contemplada no item 1.2 do Processo	Uma solicitação de mudança leva a uma eventual atualização do Plano de Projeto, bem como a realização de todas as correções necessárias para que requisitos e correspondentes produtos de trabalho sejam consistentes.
3	Saída de artefato “Registro de correções” não contemplada no Processo	Saída do artefato: “Registro de correções” incorporada ao item 1.2 do Processo	Incorporação para atender os requisitos da norma.
4	Tarefa não contemplada no item 2.2 do Processo	Tarefa: “Elaborar documentação do usuário” no item 2.2 do Processo	Tarefa incorporada para atender os objetivos da norma
5	Saídas de artefato não contemplado no item 2.3 do Processo	Saídas dos artefatos “Resultados de Verificação” “Resultados de Validação” Incorporadas no item 2.3 do Processo	Os artefatos fazem parte das saídas de artefatos exigidos pela norma

6	Tarefa: “Elaborar o Plano de Testes”, contemplada no item 2.2 do Processo	Tarefa: “Elaborar o Plano de Testes”, contemplada no item 2.3 do Processo	Alteração da localização do item com o objetivo de aderência à norma
7	Saída do artefato: “Plano de Testes elaborado” no item 2.2 do Processo	Saída do artefato: “Plano de Testes elaborado” no item 2.3 do Processo	Alteração para compatibilizar com o exposto no item 6
8	Saídas de artefato não contemplado no item 2.3 do Processo	Saída do artefato “Documentos para implementação dos requisitos elaborados (design)” no item 2.3 do Processo	O artefato faz parte das saídas de artefatos exigidos pela norma
9	Entrada de artefato não contemplado no item 2.4 do Processo	Entrada do artefato “Documentos para implementação dos requisitos elaborados (design)” no item 2.4 do Processo	O artefato passa a compor o conjunto de artefatos necessários para a realização das tarefas do item 2.4 do Processo
10	Tarefa não contemplada no item 2.5 do Processo	Tarefa: “Integrar os componentes de software”, incorporada ao item 2.5 do Processo	Tarefa incorporada para atender os objetivos da norma
11	Tarefa: “Executar os testes, utilizando o Plano de Testes, identificando inconsistências a partir da análise de documentação do projeto e registrar os resultados, contemplada no item 2.5 do Processo	Tarefa: “Executar os testes, utilizando o Plano de Testes, identificando inconsistências a partir da análise de documentação do projeto, registrar os resultados e estabelecer o software”, contemplada no item 2.5 do Processo	Mudança de redação, contemplando “estabelecimento do software” para atendimento aos objetivos da norma
12	Saídas de artefato não contemplado no item 2.5 do Processo	Saída do artefato “Software”, incorporado ao item 2.5 do Processo	O artefato faz parte das saídas de artefatos exigidos pela norma

Realizadas as modificações, de acordo com a especialista, o PROÁGIL-29110 atende todos os objetivos dos processos de Gerenciamento de Projetos e Implementação de Software. Além disso, possui todas as atividades contempladas

nesses processos e os artefatos gerados pelo PROÁGIL-29110 atendem os requeridos pela norma.

A especialista relatou que as pequenas organizações necessitam de um processo padronizado, razão pela qual acredita na aceitabilidade do processo por elas. Da mesma forma, a reprodutibilidade deve acontecer, porém ressaltou que as organizações devem adequar o processo para o uso de ferramentas que possam auxiliar na implementação e sua utilização.

Em sequência o processo foi apresentado a segunda especialista que é professora universitária, possui experiência de 6 anos em gerenciamento de projetos com a utilização de métodos ágeis, mais especificamente usando-se o método Scrum. Possui as seguintes certificações: Scrum Professional, Scrum Master, Product Owner e Lean IT Foundation.

A especialista fez as seguintes ressalvas:

- a) “Na questão de levantamento de requisitos existe a cerimônia ou evento, que não é obrigatória, chamada Refinement Session ou Product Backlog Refinement ou Backlog Grooming. Essa sessão faz com que o Product Owner (PO) que tem total domínio do Backlog Product converse com o cliente e, também com o time alguns dias antes do planejamento do Sprint para priorizar e refinar a escrita das histórias do Product Backlog. Essa prática otimiza o entendimento de cada história durante o planejamento e leva a uma priorização muito mais correta onde se pode agregar valor ao cliente ou usuário”.

O fato da cerimônia ser opcional, a ressalva foi incorporada ao processo PROÁGIL-29110 como uma observação e sua realização fica por conta da necessidade do projeto ou exigências contratuais.

- b) “A nomenclatura utilizada no processo PROÁGIL-29110 denominada “Scrum Poker” deveria ser trocada por “Planning Poker”, que é mais usada nas organizações”.

A sugestão foi aceita e a nova denominação foi incorporada ao processo.

- c) “Existe uma cerimônia, chamada Sprint Review, em que é realizada a entrega do incremento ao solicitante (PO) e este dá o seu “de acordo”

ou não, baseando-se no que foi estipulado no início da Sprint (Planning)”.

Essa ressalva foi mencionada na atividade de “Fechamento do Projeto”.

A cerimônia ou evento já estava presente no processo na atividade “Entrega do Produto”.

Em relação aos princípios da filosofia Lean, a observação foi de que as atividades do processo PROÁGIL-29110 estão aderentes aos mesmos, conforme exposto no Quadro 12.

No Quadro 12 mostra-se o relacionamento entre as atividades do processo PROÁGIL-29110 (primeira coluna), as atividades da ISO/IEC 29110 (segunda coluna), os artefatos, técnicas, eventos ou definições extraídas dos autores Liubchenko (2016), Sabbagh (2013), Schwaber e Sutherland (2016) que são aplicadas no método Scrum (terceira coluna) e o princípio Lean aplicado ou técnica utilizada, segundo os autores Massari (2017), e Poppendieck, (2003), (quarta coluna).

A escolha dos autores Liubchenko (2016), Sabbagh (2013), Schwaber e Sutherland (2016) deve-se ao fato dos mesmos, em seus trabalhos, terem associado o método Scrum com artefatos, técnicas e eventos.

Da mesma forma a escolha pelos autores Massari (2017) e Poppendieck (2003) se dá por conta que os mesmos nos seus trabalhos, definirem os Princípios da filosofia Lean (no caso de Poppendieck (2003)) ou exemplificar a aplicação dos princípios (no caso de Massari (2017)).

Quadro 12: Relacionamento entre ISO/IEC 29110 x Scrum X Lean

PROÁGIL-29110	Atividades da ISO/IEC 29110	Artefatos, técnicas, eventos ou práticas (Pn) aplicadas no método Scrum	Princípio Lean aplicado ou técnica utilizada
<p>1.1 Planejamento do Projeto</p> <ul style="list-style-type: none"> A partir da Declaração de Trabalho revisada, definir o Product Backlog, realizar a definição de pronto e preencher o Template do Plano do Projeto; <p>Obs: o evento (opcional) chamado Refinement Session ou Product Backlog Refinement ou Backlog Grooming poderá ser realizada para o refinamento das histórias do Product Backlog</p>	PM.1	<ul style="list-style-type: none"> ✓ Product Backlog (Artefato) ✓ Definição de Pronto (evento) ✓ Refinement Session (opcional) 	<ul style="list-style-type: none"> ✓ Princípio II – Amplificar o conhecimento: uso de <i>guidelines</i> ou <i>templates</i> contribuem para o amadurecimento da equipe e do processo. (POPPENDIECK, 2003) ✓ Princípio III: Adiar decisões: totalmente aderente ao conceito do Product Backlog emergente. Requisitos prioritários do Backlog devem estar bem detalhados, requisitos menos prioritários devem estar menos detalhados (MASSARI, 2017 e POPPENDIECK, 2003)
<ul style="list-style-type: none"> Realizar a estimativa de esforço, custo e tempo para cada item do backlog utilizando uma técnica associada (APF, Planning Poker, ou método próprio); 	PM.1	<ul style="list-style-type: none"> ✓ Planning Poker (técnica) ou APF (técnica) 	<ul style="list-style-type: none"> ✓ Princípio II – Amplificar o conhecimento: uso de <i>guidelines</i> ou <i>templates</i> contribuem para o amadurecimento da equipe e do processo. (POPPENDIECK, 2003) ✓ Princípio V: Fortalecer a equipe: a atividade de estimativa de esforço está associada ao trabalho em equipe (MASSARI, 2017)
<ul style="list-style-type: none"> Planejar a Sprint: duração, equipe, identificar riscos (na Planilha de Riscos) tarefas e respectivas duração e dependências em um cronograma; 	PM.1	<ul style="list-style-type: none"> ✓ Sprint Planning - evento (SABBAGH, 2013) ✓ Pequenas Equipes (P1): recomendação para o tamanho da equipe até sete pessoas (LIUBCHENKO, 2016) 	<ul style="list-style-type: none"> ✓ Princípio II – Amplificar o conhecimento: O uso do desenvolvimento iterativo representado pelo Sprint (POPPENDIECK, 2003)

		<ul style="list-style-type: none"> ✓ Pacotes de Baixo Acoplamento (P3), e recomenda o particionamento de tarefas de trabalho e de equipe em partições ou pacotes limpos e de baixo acoplamento (LIUBCHENKO, 2016) ✓ Controles Iterativos (P6): calibrar os requisitos, a funcionalidade, o risco e permitir ao PO planejar as iterações (LIUBCHENKO, 2016) ✓ O Sprint é um ciclo de desenvolvimento que se repete em formato ao longo de todo o projeto e possui sempre duração de uma a quatro semanas. Essa duração ajuda a estabelecer um ritmo regular para o trabalho do time de desenvolvimento (SABBAGH, 2013) e (SCHWABWE, 2016) 	<ul style="list-style-type: none"> ✓ Princípio IV: Entregas rápidas: totalmente aderente como conceito de Sprints de duração entre uma e quatro semanas. (MASSARI, 2017)
<ul style="list-style-type: none"> • Criar repositório do projeto e estabelecer uma estratégia de controle de versão de artefatos e baseline do projeto; 	PM.1	<ul style="list-style-type: none"> ✓ Sprint Planning - evento (SABBAGH, 2013) 	<ul style="list-style-type: none"> ✓ Princípio V: Fortalecer a equipe: equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção. (MASSARI, 2017)
<ul style="list-style-type: none"> • Estabelecer uma estratégia de Garantia da Qualidade do projeto. 	PM.1	<ul style="list-style-type: none"> ✓ Sprint Planning - evento (SABBAGH, 2013) 	<ul style="list-style-type: none"> ✓ Princípio VI: Construir a qualidade: A integridade percebida significa que a totalidade do produto alcança um equilíbrio entre funções, usabilidade, confiabilidade, economia e isso encanta o cliente. A integridade conceitual significa que os conceitos centrais do sistema desenvolvidos são facilitados e coesos. (POPPENDIECK, 2003)

<p>1.2 Execução do Plano de Projeto</p> <ul style="list-style-type: none"> Realizar a reunião diária (Daily Meeting) – tempo máximo de 15 minutos e anotar dificuldades e obstáculos 	PM.2	✓ Daily Meeting - evento (SABBAGH, 2013)	<ul style="list-style-type: none"> ✓ Técnica utilizada: Kanban ✓ Princípio II – Amplificar o conhecimento: uso de meios de compartilhamento de informações. (POPPENDIECK, 2003) ✓ Princípio IV: Entregas rápidas: Iterações curtas trazem mais experiência para a equipe e aumentam a sua segurança para tomar decisões. (POPPENDIECK, 2003) ✓ Princípio V: Fortalecer a equipe: equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção. (MASSARI, 2017)
<ul style="list-style-type: none"> Utilizar o gráfico de Burndown para estabelecer o registro de status do projeto 	PM.2	✓ Daily Meeting - evento (SABBAGH, 2013)	<ul style="list-style-type: none"> ✓ Princípio II – Amplificar o conhecimento: uso de meios de compartilhamento de informações. (POPPENDIECK, 2003) ✓ Princípio VII: Otimizar o todo: o sucesso de um Sprint não é gerado pelas ações individuais de cada membro da Equipe Scrum e sim pelo trabalho colaborativo em equipe. (MASSARI, 2017)
<ul style="list-style-type: none"> Realizar as revisões e aceites entre a equipe de trabalho e/ou cliente, quando pertinente 	PM.2	✓ Reunião de Sprint Retrospective – evento (SABBAGH, 2013)	<ul style="list-style-type: none"> ✓ Princípio II – Amplificar o conhecimento: A reunião de Sprint Retrospective permite a aplicação do <i>hansei</i>¹⁰, ou seja, identificar possíveis erros cometidos no Sprint ou melhorias que possam ser usadas no futuro. (SABBAGH, 2013) ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de

¹⁰ Hansei é uma palavra japonesa que significa “reflexão profunda”

			Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
<ul style="list-style-type: none"> Estabelecer o Backup do repositório do projeto 	PM.2	✓ Sprint Planning - evento (SABBAGH, 2013)	✓ Princípio V: Fortalecer a equipe: equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção. (MASSARI, 2017)
<p>1.3 Avaliação e Controle do Projeto</p> <ul style="list-style-type: none"> Avaliar a realização do Plano de Projeto e progresso frente aos objetivos; 	PM.3	✓ Daily Meeting - evento (SABBAGH, 2013)	<ul style="list-style-type: none"> ✓ Kanban ✓ Princípio I – Eliminar desperdício: Aplicação de duas categorias de desperdício abordada por Liker, 2003. A primeira chamada <i>muri</i>, definido com sobrecarga nas pessoas ou equipamentos além dos limites naturais. A segunda chamada <i>mura</i>, definido como irregularidades, flutuações ou desbalanços no ritmo da produção. (POPPENDIECK, 2003). ✓ Princípio II – Amplificar o conhecimento: uso de <i>guidelines</i> ou <i>templates</i> contribuem para o amadurecimento da equipe e do processo. (POPPENDIECK, 2003).
<ul style="list-style-type: none"> Revisar os riscos do projeto, identificando novos riscos, se necessário; 	PM.3	✓ Daily Meeting - evento (SABBAGH, 2013)	<ul style="list-style-type: none"> ✓ Princípio II – Amplificar o conhecimento: uso de <i>guidelines</i> ou <i>templates</i> contribuem para o amadurecimento da equipe e do processo. (POPPENDIECK, 2003) ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
<ul style="list-style-type: none"> Levantar possíveis desvios significativos em relação a custo, cronograma e desempenho técnico ou problemas relativos ao 	PM.3	✓ Daily Meeting - evento (SABBAGH, 2013)	✓ Princípio II – Amplificar o conhecimento: uso de <i>guidelines</i> ou <i>templates</i> contribuem para o

projeto e registrar as ações corretivas;			amadurecimento da equipe e do processo. (POPPENDIECK, 2003) ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
<ul style="list-style-type: none"> Analisar eventuais solicitações de mudanças avaliando impacto, utilizando o Template Solicitação de Mudança – SM, atualizar o Plano de Projeto se necessário e realizar as devidas correções; 	PM.3	✓ Daily Meeting - evento (SABBAGH, 2013)	<ul style="list-style-type: none"> ✓ Princípio II – Amplificar o conhecimento: uso de <i>guidelines</i> ou <i>templates</i> contribuem para o amadurecimento da equipe e do processo. (POPPENDIECK, 2003) ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
<ul style="list-style-type: none"> Garantir que os artefatos a serem alterados em função das solicitações de mudanças foram atualizados; 	PM.3	✓ Daily Meeting - evento (SABBAGH, 2013)	<ul style="list-style-type: none"> ✓ Princípio II – Amplificar o conhecimento: uso de <i>guidelines</i> ou <i>templates</i> contribuem para o amadurecimento da equipe e do processo. (POPPENDIECK, 2003) ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
<ul style="list-style-type: none"> Atualizar a Matriz de Rastreabilidade. 	PM.4	✓ Daily Meeting - evento (SABBAGH, 2013)	<ul style="list-style-type: none"> ✓ Princípio II – Amplificar o conhecimento: uso de <i>guidelines</i> ou <i>templates</i> contribuem para o amadurecimento da equipe e do processo. (POPPENDIECK, 2003) ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de

			Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção.
1.4. Fechamento de Projeto <ul style="list-style-type: none"> Realizar uma reunião de <i>Sprint Retrospective</i> com os Desenvolvedores; 		✓ Reunião de Sprint Retrospective – evento (SABBAGH, 2013)	✓ Princípio II – Amplificar o conhecimento: A reunião de Sprint Retrospective permite a aplicação do <i>hansei</i> ¹¹ , ou seja, identificar possíveis erros cometidos no Sprint ou melhorias que possam ser usadas no futuro. (SABBAGH, 2013) ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
<ul style="list-style-type: none"> Armazenar as lições aprendidas; 	PM.4	✓ Reunião de Sprint Retrospective – evento (SABBAGH, 2013)	✓ Princípio II – Amplificar o conhecimento: Lições podem ser extraídas das experiências vividas pelas equipes e incorporada ao processo, fazendo com que as dificuldades passadas sejam fonte de conhecimento e contribuam para o amadurecimento da equipe e do processo. (POPPENDIECK, 2003)
<ul style="list-style-type: none"> Entregar o produto de software e documentos aplicáveis de acordo com as instruções de entrega e registrar o aceite. 	PM.4	✓ Capacidade de declarar o projeto feito a qualquer momento (P7) que representa uma filosofia de gerenciamento que enfatiza a usabilidade e a correção do sistema, em vez do crescimento estrito de recursos. Essa capacidade afeta os tipos de recursos ou correções que são incorporados na próxima versão e também altera a mentalidade com a qual o gerente gerencia o projeto. (LIUBCHENKO, 2016)	✓ Princípio II – Amplificar o conhecimento: uso de <i>guidelines</i> ou <i>templates</i> contribuem para o amadurecimento da equipe e do processo. (POPPENDIECK, 2003)

¹¹ Hansei é uma palavra japonesa que significa “reflexão profunda”

2.1 Início da Implementação do Software	SI.1	✓ Daily Meeting – evento (SABBAGH, 2013)	✓ Princípio II – Amplificar o conhecimento: uso de <i>guidelines</i> ou <i>templates</i> contribuem para o amadurecimento da equipe e do processo. (POPPENDIECK, 2003) ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
<ul style="list-style-type: none"> Realizar a reunião de Kick-off com todos os desenvolvedores com o objetivo de revisar o plano de projeto atividades, e obter o compromisso da equipe; 			
<ul style="list-style-type: none"> Criar o repositório do projeto. 	SI.1	✓ Daily Meeting – evento (SABBAGH, 2013)	✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
2.2 Analise de Requisitos do Software	SI.2	✓ Daily Meeting – evento (SABBAGH, 2013)	✓ Princípio II – Amplificar o conhecimento: uso de <i>guidelines</i> ou <i>templates</i> contribuem para o amadurecimento da equipe e do processo. (POPPENDIECK, 2003)
<ul style="list-style-type: none"> Verificar e validar os requisitos e elaborar a planilha de Especificação de Requisitos; 			
<ul style="list-style-type: none"> Detalhar os requisitos. Descrever as informações de cada requisito com a identificação de atributos (tela, campos e outros) de forma resumida; 	SI.2	✓ Documentação Constante (P5), pela razão de fornecer uma “foto” atualizada do sistema e seu progresso. O PO pode usá-las para rastreamento ou para trazer novas pessoas para aumentar velocidade do projeto. A prática determina que a documentação seja produzida e mantida atualizada não apenas para o código, mas também para os requisitos e o planejamento da Liberação. (LIUBCHENKO, 2016) ✓ Daily Meeting – evento (SABBAGH, 2013)	✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
<ul style="list-style-type: none"> Elaborar a documentação do usuário. 	SI.2	✓ Daily Meeting – evento (SABBAGH, 2013)	✓ Princípio II – Amplificar o conhecimento: uso de <i>guidelines</i> ou <i>templates</i> contribuem para o amadurecimento da equipe e do processo (POPPENDIECK, 2003)

2.3 Projeto de Arquitetura e Detalhamento do Software	SI.3	✓ Daily Meeting – evento (SABBAGH, 2013)	✓ Princípio I: Eliminar os desperdícios: como as Sprints são iterações curtas, a Equipe Scrum deve trabalhar de forma mais otimizada possível evitando falta de testes (MASSARI, 2017) ✓ Princípio II – Amplificar o conhecimento: uso de <i>guidelines</i> ou <i>templates</i> contribuem para o amadurecimento da equipe e do processo (POPPENDIECK, 2003). ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
• Elaborar o Plano de Testes;			
• Elaborar os documentos para implementação dos requisitos;	SI.3	✓ Documentação Constante (P5), pela razão de fornecer uma “foto” atualizada do sistema e seu progresso. O PO pode usá-las para rastreamento ou para trazer novas pessoas para aumentar velocidade do projeto. A prática determina que a documentação seja produzida e mantida atualizada não apenas para o código, mas também para os requisitos e o planejamento da Liberação (LIUBCHENKO, 2016). ✓ Daily Meeting – evento (SABBAGH, 2013)	✓ Princípio II – Amplificar o conhecimento: uso de <i>guidelines</i> ou <i>templates</i> contribuem para o amadurecimento da equipe e do processo. (POPPENDIECK, 2003) ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
• Controlar a versão dos artefatos produzidos;	SI.3	✓ Daily Meeting – evento (SABBAGH, 2013)	✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
• Realizar a verificação e as devidas correções dos artefatos.	SI.3	✓ Daily Meeting – evento (SABBAGH, 2013)	✓ Princípio I: Eliminar os desperdícios: como as Sprints são iterações curtas, a Equipe Scrum deve trabalhar de forma mais otimizada possível evitando defeitos. (MASSARI, 2017)

			<ul style="list-style-type: none"> ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
<p>2.4 Construção do Software</p> <ul style="list-style-type: none"> • Revisão da documentação disponível esclarecendo duvidas com a equipe de desenvolvimento ou Scrum Master no Daily Meeting e estabelecer o Kanban; 	SI.4	<ul style="list-style-type: none"> ✓ Daily Meeting – evento (SABBAGH, 2013) 	<ul style="list-style-type: none"> ✓ Técnica utilizada: Kanban ✓ Princípio IV: Entregas rápidas: Iterações curtas trazem mais experiência para a equipe e aumentam a sua segurança para tomar decisões. O Kanban é uma prática associada. (POPPENDIECK, 2003) ✓ Princípio V: Fortalecer a equipe: Para um desenvolvimento de software “Lean”, a estratégia é fornecer ao cliente versões que vão se refinando em intervalos regulares. A sinalização é realizada no uso de gráficos visíveis, reuniões diárias, integração constante e testes abrangentes. (POPPENDIECK, 2003) ✓ Princípio VII: Otimizar o todo: Lean ainda recomenda a escolha de métricas de alto nível que sejam representativas para identificar a evolução. No caso o Kanban auxilia a valorizar o trabalho em equipe. (POPPENDIECK, 2003)
<ul style="list-style-type: none"> • Implementar o código fonte (componentes do software) 	SI.4	<ul style="list-style-type: none"> ✓ Compilações Frequentes (P2), e recomenda fornecer a construção de executáveis que podem ser frequentemente inspecionados, ajustados, testados, documentados e compilados. Os Builds permitem que o PM veja o estado do sistema a qualquer momento para acompanhar o progresso. Essa prática permite que os Desenvolvedores comuniquem o status do sistema com outros Desenvolvedores e PM 	<ul style="list-style-type: none"> ✓ Princípio II – Amplificar o conhecimento: pelo uso de Padrão de Codificação ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção. ✓ Princípio VI: Construir a Qualidade: O software com integridade tem uma arquitetura coerente, pontuação

		<p>a qualquer momento, além de ajudar o PM a planejar e monitorar Liberações e Desenvolvimento (LIUBCHENKO, 2016).</p> <p>✓ Daily Meeting – evento (SABBAGH, 2013)</p>	<p>alta no que diz respeito a usabilidade e adequação, e é sustentável, adaptável e extensível. Isso pode ser conseguido através de liderança sensata, experiência relevante, comunicação eficaz, e disciplina saudável. A promoção para a qualidade pode ser feita por meio de práticas, tais como Integração Continua. (POPPENDIECK, 2003)</p>
<ul style="list-style-type: none"> Realizar os testes unitários, caso erros sejam identificados, fazer a devida correção 	SI.4	<p>✓ Testes Constantes (P4), pela razão de fornecer ao PO uma visão geral do sistema e permite que os desenvolvedores conheçam o estado atual do sistema a qualquer momento (LIUBCHENKO, 2016).</p>	<p>✓ Princípio I: Eliminar os desperdícios: como as Sprints são iterações curtas, a Equipe Scrum deve trabalhar de forma mais otimizada possível evitando falta de testes (MASSARI, 2017)</p> <p>✓ Princípio II – Amplificar o conhecimento: O uso do desenvolvimento iterativo representado pelo Sprint (POPPENDIECK, 2003).</p> <p>✓ Princípio V: Fortalecer a equipe: Para um desenvolvimento de software “Lean”, a estratégia é fornecer ao cliente versões que vão se refinando em intervalos regulares. A sinalização é realizada no uso de gráficos visíveis, reuniões diárias, integração constante e testes abrangentes. (POPPENDIECK, 2003)</p>
<ul style="list-style-type: none"> Atualizar a matriz de rastreabilidade com o código fonte produzido 	SI.4	<p>✓ Daily Meeting – evento (SABBAGH, 2013)</p>	<p>✓ Princípio II – Amplificar o conhecimento: uso de <i>guidelines</i> ou <i>templates</i> contribuem para o amadurecimento da equipe e do processo. (POPPENDIECK, 2003)</p> <p>✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção</p>
<ul style="list-style-type: none"> Armazenar o código fonte 	SI.4	<p>✓ Daily Meeting – evento (SABBAGH, 2013)</p>	<p>✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de</p>

			Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
2.5 Integração e Testes do Software <ul style="list-style-type: none"> Integrar os componentes do software; 	SI.5	<ul style="list-style-type: none"> ✓ Testes Constantes (P4) no sentido de permitir que os desenvolvedores conheçam o estado atual do sistema a qualquer momento (LIUBCHENKO, 2016). ✓ Daily Meeting – evento (SABBAGH, 2013) 	<ul style="list-style-type: none"> ✓ Princípio I: Eliminar os desperdícios: como as Sprints são iterações curtas, a Equipe Scrum deve trabalhar de forma mais otimizada possível evitando falta de testes (MASSARI, 2017) ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
<ul style="list-style-type: none"> ✓ Executar os testes no produto identificando inconsistências a partir da análise de documentação do projeto, registrar os resultados e estabelecer o software; 	SI.5	<ul style="list-style-type: none"> ✓ Testes Constantes (P4) no sentido de permitir que os desenvolvedores conheçam o estado atual do sistema a qualquer momento (LIUBCHENKO, 2016). ✓ Daily Meeting – evento (SABBAGH, 2013) 	<ul style="list-style-type: none"> ✓ Princípio I: Eliminar os desperdícios: como as Sprints são iterações curtas, a Equipe Scrum deve trabalhar de forma mais otimizada possível evitando falta de testes (MASSARI, 2017) ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção ✓ Princípio VI: Construir a Qualidade: Estabelecimento da técnica Lean chamada “jidoka”¹². O “jidoka” estabelece que os próprios trabalhadores devem realizar o controle de qualidade, paralisando imediatamente a produção caso algum defeito seja encontrado para se buscar a causa raiz, resolvê-la e evitar que o problema volte a acontecer. No caso de software problemas recorrentes devem ser investigados (SABBAGH, 2013). ✓

¹² Jidoka, significa “automação com um toque humano” em japonês

<ul style="list-style-type: none"> Atualizar a matriz de rastreabilidade, se necessário; 	SI.5	<ul style="list-style-type: none"> ✓ Daily Meeting – evento (SABBAGH, 2013) 	<ul style="list-style-type: none"> ✓ Princípio II – Amplificar o conhecimento: uso de <i>guidelines</i> ou <i>templates</i> contribuem para o amadurecimento da equipe e do processo. (POPPENDIECK, 2003) ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção.
<ul style="list-style-type: none"> Elaborar documentação da operação; 	SI.5	<ul style="list-style-type: none"> ✓ Documentação Constante (P5), pela razão de fornecer uma “foto” atualizada do sistema e seu progresso. O PO pode usá-las para rastreamento ou para trazer novas pessoas para aumentar velocidade do projeto. A prática determina que a documentação seja produzida e mantida atualizada não apenas para o código, mas também para os requisitos e o planejamento da Liberação (LIUBCHENKO, 2016). ✓ Daily Meeting – evento (SABBAGH, 2013) 	<ul style="list-style-type: none"> ✓ Princípio II – Amplificar o conhecimento: uso de <i>guidelines</i> ou <i>templates</i> contribuem para o amadurecimento da equipe e do processo. (POPPENDIECK, 2003) ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
<ul style="list-style-type: none"> Verificar a consistência da baseline. 	SI.5	<ul style="list-style-type: none"> ✓ Daily Meeting – evento (SABBAGH, 2013) 	<ul style="list-style-type: none"> ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção

<p>2.6 Entrega do Produto</p> <ul style="list-style-type: none"> Elaborar o documento da Instrução de entrega; 	SI.6	<ul style="list-style-type: none"> ✓ Documentação Constante (P5), pela razão de fornecer uma “foto” atualizada do sistema e seu progresso. O PO pode usá-las para rastreamento ou para trazer novas pessoas para aumentar velocidade do projeto. A prática determina que a documentação seja produzida e mantida atualizada não apenas para o código, mas também para os requisitos e o planejamento da Liberação (LIUBCHENKO, 2016). ✓ Daily Meeting – evento (SABBAGH, 2013) 	<ul style="list-style-type: none"> ✓ Princípio II – Amplificar o conhecimento: uso de <i>guidelines</i> ou <i>templates</i> contribuem para o amadurecimento da equipe e do processo. (POPPENDIECK, 2003) ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
<ul style="list-style-type: none"> Verificar a documentação de manutenção, se necessário; 	SI.6	<ul style="list-style-type: none"> ✓ Daily Meeting – evento (SABBAGH, 2013) 	<ul style="list-style-type: none"> ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
<ul style="list-style-type: none"> Atualizar a configuração do software; 	SI.6	<ul style="list-style-type: none"> ✓ Daily Meeting – evento (SABBAGH, 2013) 	<ul style="list-style-type: none"> ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção
<ul style="list-style-type: none"> Realizar a reunião de Sprint Review. 	SI.6	<ul style="list-style-type: none"> ✓ Sprint Review – evento (SABBAGH, 2013) 	<ul style="list-style-type: none"> ✓ Princípio V: Fortalecer a equipe: reunião equipe é uma das principais características de uma Equipe de Desenvolvimento para que seja possível atingir uma alta maturidade em sua auto-organização e sua auto direção ✓ Princípio VI: Construir qualidade: efetuar testes automatizados, mostrar protótipos ao Product Owner e demais partes interessadas e eliminar defeitos e bugs identificados durante o Sprint.

6. ANÁLISE DE DADOS E DISCUSSÃO DOS RESULTADOS

Esse capítulo expõe os aspectos associados com a análise dos dados capturados na pesquisa e promove a discussão dos resultados encontrados frente à literatura existente.

A aplicação do método pesquisa-ação foi realizada e cada uma das fases é descrita em detalhes. Na fase “Planejar”, conforme exposto no método de Pesquisa-Ação, em relação ao ambiente da pesquisa, foram definidas duas empresas, aqui chamadas de A e B.

a) Pesquisa-ação realizada na empresa A:

A empresa A está sediada nas imediações da Avenida Paulista em São Paulo e executa projetos de especificação, construção, implantação e sustentação de sistemas. É formada atualmente por 18 colaboradores, sendo 1 Gerente de Desenvolvimento, 2 Coordenadores de Projetos, 6 Analistas de Sistemas e 9 Programadores. Não possui a cultura Scrum, porém todo o corpo técnico conhece esse método ágil.

O ponto focal nessa empresa foi o Gerente de Desenvolvimento. Foi decidido em reunião dia 04/01/2019 que o piloto seria um projeto cujo objetivo era a construção de um “front-end” de um CRM. A data do treinamento no processo PROÁGIL-29110 foi agendada. Nesse mesmo dia uma entrevista coletiva com os 6 recursos do projeto foi realizada. Participaram da entrevista 1 Coordenador de Projeto, 1 Analista de Sistemas e 4 Programadores. O objetivo desta entrevista foi de complementar a análise de dados disponibilizados pela empresa com a opinião e o depoimento dos indivíduos envolvidos sobre as práticas de desenvolvimento. A principal preocupação levantada pelos entrevistados foi que o desenvolvimento realizado por eles era feito sem o apoio de um processo definido, sem troca de experiências no que tange às lições aprendidas e o desperdício de tempo em algumas atividades. Da mesma forma, gostariam de usar o conhecimento adquirido em métodos ágeis, no dia a dia de trabalho. Também ressaltaram o receio de que uma grande documentação de projeto poderia ser um obstáculo para a agilidade do time.

Na fase “Agir”, o treinamento no PROÁGIL-29110 foi aplicado. Todas as dúvidas foram tiradas e o time treinado se mostrou entusiasmado com a possibilidade do uso do processo. A equipe afirmou que estava precisando de um processo definido, visto que não seguia um padrão para os projetos. A organização decidiu usar todos os templates do processo PROÁGIL-29110, menos do Plano de Testes, já existente e disseminado nos projetos. O projeto piloto teve seu início em 14/01/2019. Foi realizada uma reunião de Sprint Planning com a participação das 6 pessoas envolvidas no projeto. Nessa reunião foram definidos o Product Owner e o Scrum Master.

Em seguida o processo foi implementado. O sprint foi iniciado com o seu planejamento, de acordo com o PROÁGIL-29110, em que os requisitos a serem desenvolvidos no sprint atual são selecionados com a participação do cliente ou com seu representante (PO). Cada requisito foi trabalhado por todo o time do projeto, desde a sua análise até sua homologação interna, na ordem de maior valor para o cliente. Somente após a equipe concluir o primeiro requisito do sprint, o segundo requisito começou a ser trabalhado. O Sprint 1 possuiu 3 requisitos implementados. O Sprint 2 possuiu 4 requisitos implementados. Durante todos os 2 sprints, foram utilizadas as práticas do método Scrum e definidas no PROÁGIL-29110: Planejamento do Sprint (Sprint Planning), Gráfico Burndown, reuniões diárias (Daily meeting) para monitoração do projeto, com a participação do Scrum Master e a equipe do projeto, reunião de revisão do Sprint (Sprint Review) reunião de retrospectiva do Sprint (Sprint Retrospective). Além disso, foram realizadas reuniões semanais do Scrum Master do projeto com a gerência superior nas quais o andamento do projeto da semana foi apresentado, juntamente com os impedimentos ainda não solucionados, e foram definidas ações para tratá-los.

Na fase “Avaliar”, foi realizada a coleta dos dados. O processo de coleta de dados, análise e observação foi realizado nas duas empresas, utilizando-se as fontes de dados, descritas abaixo:

- a) Dados obtidos pela observação do pesquisador durante a execução dos sprints;
- b) Notas das reuniões;
- c) Entrevistas;

d) Notas de reunião obtidas nos seminários.

Ao finalizar o primeiro Sprint e ao término do segundo Sprint foram realizados seminários com os participantes do projeto, a fim de validar o modelo proposto e levantamento das melhorias.

Cada um dos Sprints se encerrou com as reuniões Sprint Review, nas quais as funcionalidades desenvolvidas foram apresentadas para o cliente, homologadas e implantadas e Sprint Retrospective, na qual foram coletadas lições aprendidas.

Para cada organização foram realizados dois Sprints. Ao término do segundo Sprint, as observações se mostraram suficientes para a análise de dados.

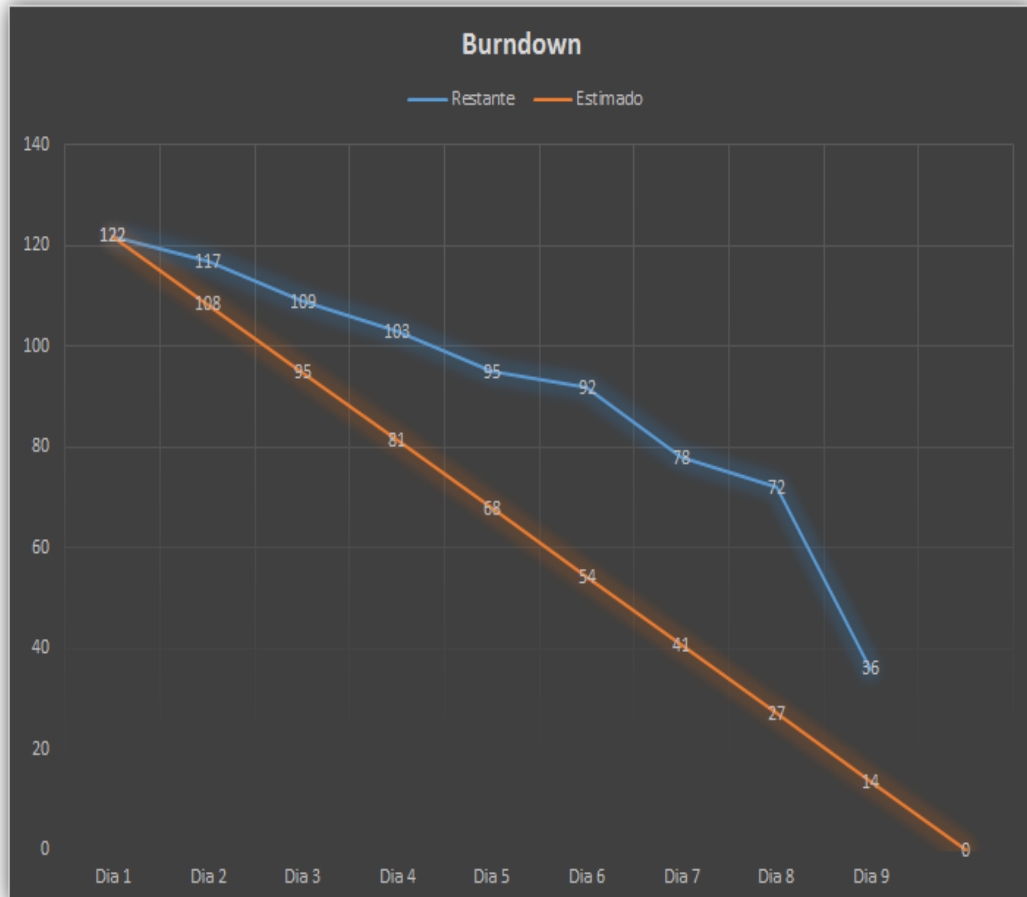
Em relação a fase “Refletir”, o pesquisador, a partir das coletas de dados realizadas, apresenta uma síntese dos resultados:

- O “time Scrum” trabalhou de forma colaborativa e tiveram grande capacidade de desempenhar várias funções simultâneas. Porém, eles não trabalharam lado a lado, e alguns participantes do time trabalhavam na organização em horários diferentes. Em muitas oportunidades fizeram horas extras e tinham dificuldade em se auto gerenciar.
- As estimativas foram feitas usando-se o método proposto no PROÁGIL-29110 (Planning Poker), e foi bem aceito pelo time.
- Em relação à reunião de Planejamento do Sprint, o time definiu qual seria o Backlog do primeiro e segundo Sprints, as priorizações e também definiu as datas de término. Todos do time contribuíram e concordaram com o planejamento elaborado, havendo, portanto, um consenso nas tarefas a serem realizadas.
- As reuniões diárias existiram, mas não foram realizadas em horários fixos no primeiro Sprint. Além disso, elas não foram realizadas em alguns dias, o que deve ser evitado. Percebeu-se que os membros do time conseguiram conviver com a ausência da figura do gerente de projetos, o que também diminuiu bastante a necessidade de realização de horas extras. Nessas reuniões muitos problemas foram colocados por integrantes do time e resolvidos por meio de discussões no grupo. No

segundo Sprint, as reuniões diárias passaram a ser mais produtivas quando o time começou a se auto gerenciar corretamente. Com isso eles passaram a buscar as tarefas e a cobrar entre si sua realização. Além disso, as reuniões diárias passam a ser realizadas sempre no mesmo horário e local.

- O Scrum master trabalhou para que o processo PROÁGIL-29110 fosse executado e para que não houvesse impedimentos na realização do trabalho do time.
- No Sprint Review do primeiro Sprint, o time apontou alguns erros cometidos. O maior deles foi ter criado um Backlog do Sprint muito grande, que acabou resultando em um atraso no Sprint, conforme gráfico burndown, apresentado na Figura 16.
- A realização das tarefas sempre esteve aquém do estimado. Um fato importante a ser relatado foi que a partir do gráfico da Figura 16, no oitavo dia do Sprint em que o realizado estava muito distante do estimado, foi tomada uma ação corretiva pelo time com a incorporação de mais um integrante, o que fez com que no nono dia do Sprint esse desvio sofresse uma considerável redução.
- O gráfico da Figura 16 foi atualizado pelo Scrum Master cada vez que uma funcionalidade era implementada e apresentado ao time nas reuniões diárias, dando visibilidade a todos. Ele foi elaborado em uma planilha eletrônica, de modo a ficar o mais simples possível para o entendimento de todos.

Figura 16: Gráfico burndown do primeiro Sprint



- Na realização da Sprint Retrospective, pontos positivos e negativos foram apontados pelo time:
 - a) Atualização do GIT (Ferramenta de Gestão de Configuração). Este ponto foi definido como negativo. A atualização dos artefatos foi realizada na última semana, o que acabou gerando muitos conflitos em relação ao versionamento dos mesmos. A solução encontrada, foi realizar a atualização no GIT diariamente, diminuindo assim os conflitos e melhorando o processo.

- b) Realização de testes unitários. Este ponto foi definido como positivo, onde a equipe precisou manter e fazer mais testes unitários, entregando assim um produto com mais qualidade.
- c) Detalhamento de atividades e tarefas. Este ponto foi definido como positivo, contribuindo para uma definição de atividades claras e com prazo definido.

Em relação aos seminários, algumas considerações advindas dos participantes foram importantes:

- A equipe como um todo se mostrou mais animada por conta da visibilidade que o projeto proporcionou.
- O método aumentou a motivação do time Scrum. O time também considerou que as reuniões diárias (Daily Scrum) aumentaram consideravelmente o controle do projeto e diminuíram seus riscos. Além disso, foi constatado que um bom planejamento do Sprint é fundamental para seu sucesso.
- Nas Sprint 1 e Sprint 2 foram realizadas duas Sprint Review, uma no final da primeira semana e outra no final da segunda semana, o que permitiu um maior suporte para solucionar problemas e antecipar, possíveis imprevistos.
- Algumas frases ditas nas entrevistas com o time a respeito do PROÁGIL-29110:
 - a) “Melhorou a comunicação e a colaboração de todos os envolvidos”, dita por 1 analista de sistemas e 1 programador;
 - b) “Aumentou nossa motivação”, observação vinda de 3 programadores;
 - c) “Diminuiu os riscos e a possibilidade de insucesso”, observou o coordenador de projetos.

Em relação ao seminário final realizado com todo o time, as análises em relação a algumas cerimônias¹³/artefatos Scrum e princípios Lean são apresentadas no Quadro 13.

Quadro 13: Análise dos itens da cerimônia / artefatos Scrum e princípios Lean

Item	Vantagens/ Facilidades	Desvantagens / Dificuldades
Backlog do produto	Sua adoção melhorou o planejamento do time	O time sentiu dificuldade para elaborar sua primeira versão que geralmente é muito extenso.
<i>Backlog</i> do Sprint	Sua boa elaboração é crucial para o sucesso ou fracasso do Sprint	Seu superdimensionamento causou um atraso no primeiro Sprint.
Reunião diária	Sua utilização aumentou muito o controle do projeto. Os riscos foram minimizados	Houve algumas falhas de periodicidade devido a desencontros de horários dos membros da equipe.
Sprint	A divisão do projeto em Sprints aumentou a motivação do time	Devido à inexperiência do time, alguma dificuldade na execução do processo foi sentida
Planejamento do Sprint	Sua realização melhorou o planejamento em relação ao projeto como um todo	A primeira reunião foi pouco produtiva devido à inexperiência da equipe
Revisão do Sprint	Importante reunião para o recolhimento das lições aprendidas	Cuidado para deixar de ser executada quando o time está ansioso para o planejamento do próximo Sprint
Amplificar o conhecimento	A realização das reuniões diárias aumentou significativamente o entendimento das tarefas a serem realizadas no dia	No início alguns membros do time estavam tímidos em expressarem suas opiniões. O Scrum master incentivou a participação de todos. O aspecto motivacional se mostra nesse aspecto fundamental no processo.

b) Pesquisa-ação realizada na empresa B

A empresa B sediada no bairro de Santo Amaro em São Paulo, desenvolve aplicativos para celulares e está em expansão contemplando projetos maiores. Possui

¹³ Cerimônias no Scrum são eventos que acontecem dentro de um ciclo de desenvolvimento. Existem três tipos cerimônias: a reunião de Planejamento do Sprint, as reuniões diárias e reunião de revisão do Sprint.

uma cultura Scrum e é formada por 8 colaboradores, sendo 1 Gerente de Projeto e 7 Analistas / Programadores, ou seja, sem assumir os papéis preconizados pelo método Scrum.

A empresa utiliza a Ferramenta da Microsoft Azure DevOps que possui um conjunto de serviços hospedados na nuvem. O DevOps do Azure funciona com qualquer linguagem, visando qualquer plataforma, com serviços extensíveis. Os serviços incluem:

- a) Conexão com o GitHub (Gestão de configuração). O código pode ser extraído de sistemas populares de controle de origem. Hosted MacOS, Linux e Windows Build são alguns dos agentes oferecidos.
- b) Azure Boards: para acompanhar o trabalho com quadros Kanban, pendências, painéis de equipe e relatórios personalizados. O trabalho pode ser rastreado entre as equipes.
- c) Planos de teste do Azure: apresentam um kit de ferramentas de teste manual e exploratório. Os testes podem ser gerenciados em várias plataformas e configurações.

Na fase “Agir”, foi definido o projeto piloto que teve como objetivo a produção de um aplicativo a ser usado para Gestão de Condomínios. A primeira reunião ocorreu em 10/01/2019 e foi realizada uma adequação do processo PROÁGIL-29110 à ferramenta Azure DevOps e outros artefatos organizacionais, como mostra a Figura 17.

Figura 17: Criação do PROÁGIL-29110 no Azure DevOps

The screenshot shows the Azure DevOps web interface. The top navigation bar includes the Azure DevOps logo, the user 'smitbr', and the breadcrumb 'Organization Settings / Process'. A search bar and a filter button 'Filter by process name' are also present. The left sidebar is titled 'Organization Settings' and contains sections for 'General', 'Security', 'Boards', and 'Pipelines'. The 'Boards' section is expanded, and 'Process' is selected. The main content area is titled 'All processes' and shows a table of process templates.

Name	Description	Team proje...
Basic	This template is flexible for any process and great for teams getting started with Azure DevOps.	0
Agile (default)	This template is flexible and will work great for most teams using Agile planning methods, including those ...	27
PROÁGIL-29110	Template PROÁGIL-29110 - Nilson Salvetti	1
Scrum	This template is for teams who follow the Scrum framework.	1
CMMI	This template is for more formal projects requiring a framework for process improvement and an auditable...	0

Outros pontos estudados para a devida customização e posteriormente implementados, foram:

- a) Declaração de trabalho foi substituída pela Proposta;
- b) Para a estimativa do trabalho foi utilizado um método próprio baseado em uma planilha de estimativas;
- c) Os requisitos foram registrados na ferramenta Azure DevOps, aba Work item, na forma de Features;
- d) O Sprint Planning foi realizado na ferramenta Azure DevOps não sendo necessário a utilização do template disponível;
- e) Controle de versão foi feito utilizando-se a ferramenta Git dentro do Azure DevOps;
- f) O gráfico Burndown foi implementado dentro da ferramenta Azure DevOps;
- g) Para evidencia de monitoramento do projeto ficou definido que seria realizada uma reunião de 20 minutos do último dia da primeira semana do Sprint. Essa sugestão de melhoria foi incorporada ao processo PROÁGIL-29110;
- h) As lições aprendidas foram colocadas na página Wiki da ferramenta Azure DevOps, não sendo necessário a utilização do template disponibilizado;
- i) O Azure DevOps permitiu a inserção de cada atividade do projeto, sua duração e os responsáveis, não sendo necessário a elaboração de um cronograma em uma outra ferramenta;
- j) Os artefatos contemplados a partir dos requisitos (Features) foram as User Stories, Plano de Testes e o código, compondo dessa forma a matriz de rastreabilidade. Essa matriz é produzida dentro da ferramenta Azure DevOps por meio das identificações dos “Work Items”;
- k) O Kanban foi realizado na própria ferramenta Azure DevOps, dando visibilidade a todos do projeto;
- l) As inconsistências de projeto advindas de testes foram tratadas como “Bug” na ferramenta Azure DevOps.


O Plano do Projeto foi elaborado, usando-se o template do PROÁGIL-29110, porém incorporado na *Wiki* da ferramenta Azure DevOps, mostrado na  Figura 18. O pesquisador auxiliou no preenchimento do artefato, elucidando dúvidas existentes no time.

Figura 18: Plano do Projeto na ferramenta Azure DevOps

The screenshot displays the Azure DevOps interface for a project named 'Condominio'. The left sidebar contains navigation links: Overview, Summary, Dashboards, Wiki (selected), Boards, Repos, Pipelines, Test Plans, and Artifacts. The main content area shows the 'Sistema Condomínio' wiki page, dated 2/9/2019. The page includes a table of infrastructure resources and a section on quality management.

Sistema Condomínio

2/9/2019 Revisions Edit page + New page More

Recursos de Infra-estrutura

Identificar os recursos de infra-estrutura necessários para o projeto (hardware especial, software específico, recursos de rede, links especiais, memória, entre outros).

#	Recurso de Infra-estrutura
1.	Servidor web Linux (Apache + PHP 7.1)
2.	Servidor de Banco de dados (MySql ou PgSQL)
3.	Servidor
4.	Serviço de versionamento de código (GIT - Azure DevOps)
5.	Serviço de acompanhamento do projeto (Azure DevOps)
6.	Notebooks.
7.	Ferramenta Azure DevOps
8.	Framework Laravel

Gerenciamento da Qualidade do Projeto

Esse item aborda a estratégia utilizada para a Garantia da Qualidade do Projeto. Fazem parte da estratégia a descrição dos artefatos a serem verificados e validados e as revisões que são feitas envolvendo equipe de trabalho e cliente

Itens a serem revisados

- O código será verificado e validado através de um plano de teste elaborado pela equipe.
- Reuniões com o time

Serão realizadas as reuniões de Sprint Planning, Daily Meeting, Sprint Retrospective.

- Reuniões com o cliente
- Reuniões presenciais e online.
- Gerenciamento de Configuração_

- Sistema de biblioteca

O sistema de biblioteca de controle de versão usado neste projeto é o:

- GIT Azure DevOps

Link: <https://smitbr.visualstudio.com/>

Pages

- Sistema Condomínio
 - Controle de acesso
 - Plano de Teste
 - Reuniões
 - Matriz de Risco
 - Requisitos
 - Lições aprendidas

As informações iniciais utilizadas no template da Matriz de Risco foram customizadas e posteriormente cadastradas como um *Work Item* no Azure DevOps. Esta personalização permite a utilização das informações do projeto de maneira uniforme e online, evitando-se o preenchimento em múltiplos formatos de arquivos, facilitando assim a sua gestão e acompanhamento. O exemplo é apresentado na Figura 19.

Figura 19: Work Item da Matriz de Risco no Azure DevOps

The screenshot displays an Azure DevOps Work Item interface. At the top, the work item is identified by a red icon, the ID 'RISK 1050', and the title '1050 Configuração de servidor'. Below the title, the creator 'Daniel' is listed, along with '0 comments' and an 'Add tag' button. Action buttons include 'Save & Close', 'Follow', and refresh/undo icons. The work item's metadata shows the 'State' as 'New', 'Area' as 'Condominio', and 'Reason' as 'Moved to state New'. The 'Iteration' is 'Condominio\Sprint 01', and it was 'Updated just now'. A 'Details' button and a link icon are also present. The main content area is divided into three sections: 'Description' with the text 'O servidor de homologação está em um servidor compartilhado (Locaweb)', 'Mitigation Plan' with 'Pode haver instabilidade e indisponibilidade do sistema a homologar.', and 'Discussion' with a placeholder for a comment. On the right, a 'Custom' section lists 'Probability' (1), 'Severity' (2 - High), 'Priority' (4), and 'Impact' (2 - Medium). Below this, the 'Development' section shows a '+ Add link' button and a message 'Development hasn't started on this item.' with a 'Create a new branch' link. The 'Related Work' section includes another '+ Add link' button and a 'Parent' link to '957 Sistema de acesso', which was updated on 1/29/2019.

RISK 1050

1050 Configuração de servidor

Daniel 0 comments Add tag Save & Close Follow

State: ● New Area: Condominio Updated just now

Reason: Moved to state New Iteration: Condominio\Sprint 01

Details (1)

Description

O servidor de homologação está em um servidor compartilhado (Locaweb)

Mitigation Plan

Pode haver instabilidade e indisponibilidade do sistema a homologar.

Discussion

Add a comment. Use # to link a work item, ! to link a pull request, or @ to mention a person.

Custom

Probability
1

Severity
2 - High

Priority
4

Impact
2 - Medium

Development

+ Add link

Development hasn't started on this item.
[Create a new branch](#)

Related Work

+ Add link

Parent

957 Sistema de acesso
Updated 1/29/2019, ● New

Para o projeto em questão a estimativa de prazo foi realizada apenas por um dos participantes que possui experiência em desenvolvimento de aplicativos, e houve um compromisso do mesmo em treinar a equipe no método do “Planning Poker”, para que esse método seja usado nos próximos projetos. Para o primeiro Sprint foi realizado o Sprint Planning que resultou na implementação de 3 Features, apresentado na Figura 20.

Figura 20: Requisitos do sistema na forma de "Features"

Order	Work Item Type	Title
+	Epic	👑 Sistema Galloro Condomínio
	Feature	> 🏆 Planos de acesso ao usuário
	Feature	> 🏆 Sistema de controle do Administrador
	Feature	> 🏆 Dashboard do Usuário
	Feature	> 🏆 Acesso a histórico e logs de auditorias
	Feature	> 🏆 Geração e aprovação de Relatórios de auditoria.
	Feature	> 🏆 Solicitação e controle de auditorias
	Feature	> 🏆 Sistema de controle de nível do usuário
	Feature	> 🏆 Login do sistema

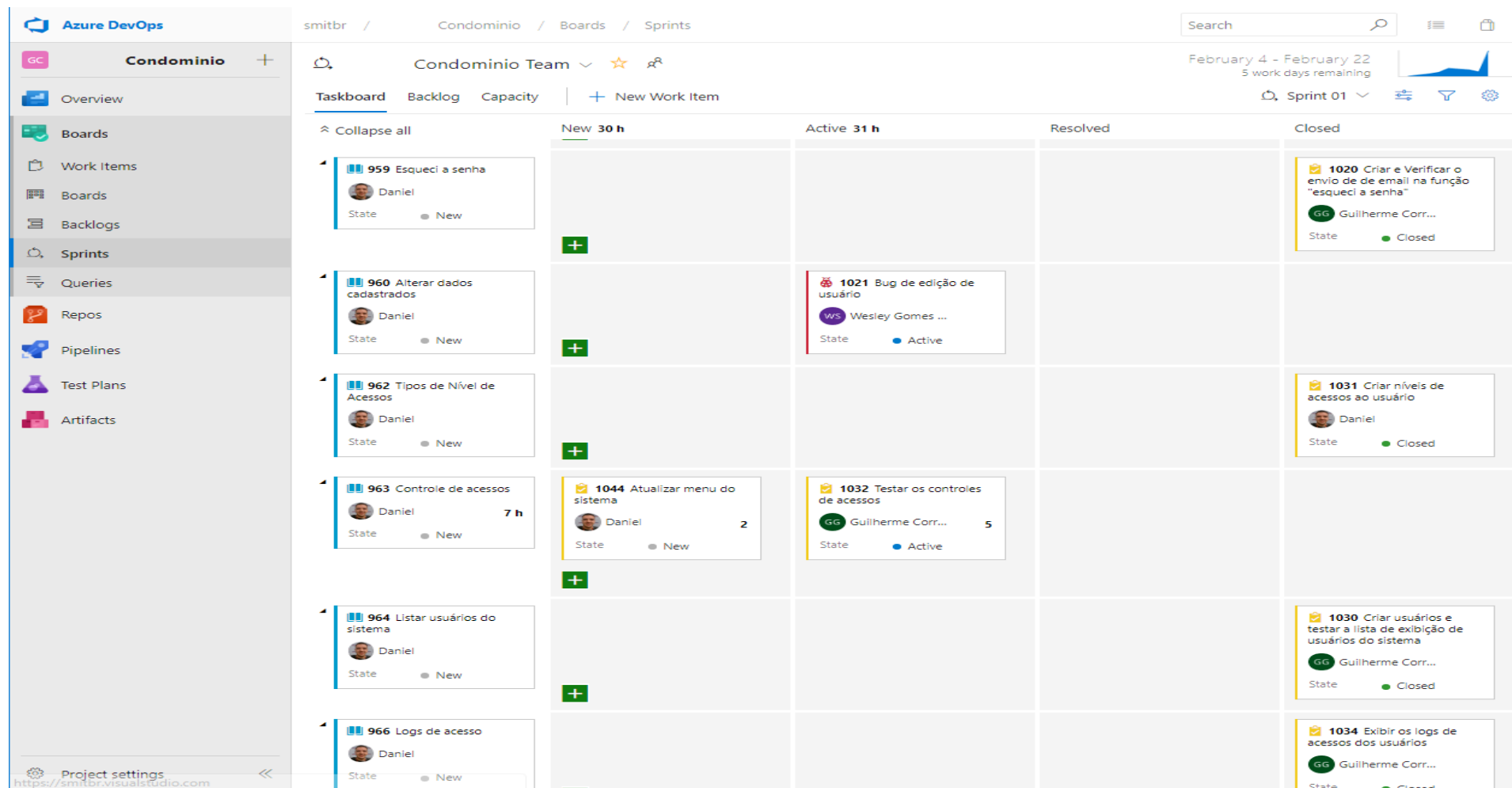
Os requisitos, aqui chamados de “Features”, foram detalhado em “User Stories” para o Sprint 1, e apresentados na Figura 21.

Figura 21: "Feature" x User Story x Sprint 1

Order	Work Item Type	Title	Iteration Path
	Feature	Sistema de controle do Administrador	Galloro Condominio\Sprint 01
	User Story	Criar, Aprovar e Editar administradora de condomínio	Galloro Condominio\Sprint 01
	User Story	Criar Template para administradora de condomínio	Galloro Condominio\Sprint 01
	Feature	Solicitação e controle de auditorias	Galloro Condominio\Sprint 01
	User Story	Solicitar nova auditoria	Galloro Condominio\Sprint 01
	User Story	Listar e Editar auditorias solicitadas	Galloro Condominio\Sprint 01
	User Story	Enviar arquivos da auditoria	Galloro Condominio\Sprint 01
	User Story	Verificar lista de arquivos enviados	Galloro Condominio\Sprint 01
	User Story	Visualizar código identificador	Galloro Condominio\Sprint 01
	User Story	Verificar status da auditoria	Galloro Condominio\Sprint 01
	User Story	Visualizar Relatório final da Auditoria	Galloro Condominio\Sprint 01
	User Story	Solicitar cadastro de um novo condomínio no sistema	Galloro Condominio\Sprint 01
	Feature	Sistema de controle de nível do usuário	Galloro Condominio\Sprint 01
	User Story	Tipos de Nível de Acessos	Galloro Condominio\Sprint 01
	User Story	Controle de acessos	Galloro Condominio\Sprint 01
	User Story	Listar usuários do sistema	Galloro Condominio\Sprint 01
	User Story	Logs de acesso	Galloro Condominio\Sprint 01
	Feature	Login do sistema	Galloro Condominio\Sprint 01
	User Story	Sistema WEB Responsivo	Galloro Condominio\Sprint 01
	User Story	Sistema de acesso	Galloro Condominio\Sprint 01
	User Story	Cadastrar novo usuário	Galloro Condominio\Sprint 01
	Task	Criar tela de cadastro de usuário	Galloro Condominio\Sprint 01
	User Story	Esqueci a senha	Galloro Condominio\Sprint 01
	User Story	Alterar dados cadastrados	Galloro Condominio\Sprint 01
	User Story	Inatividade do usuário	Galloro Condominio\Sprint 01

As atividades do processo padrão de desenvolvimento com a sequência e a dependência entre elas, os papéis responsáveis pela execução de cada atividade e as atividades consideradas marcos do projeto foram colocadas e disponibilizadas para o time através do método Kanban, apresentado na Figura 22.

Figura 22: Kanban e distribuição de tarefas



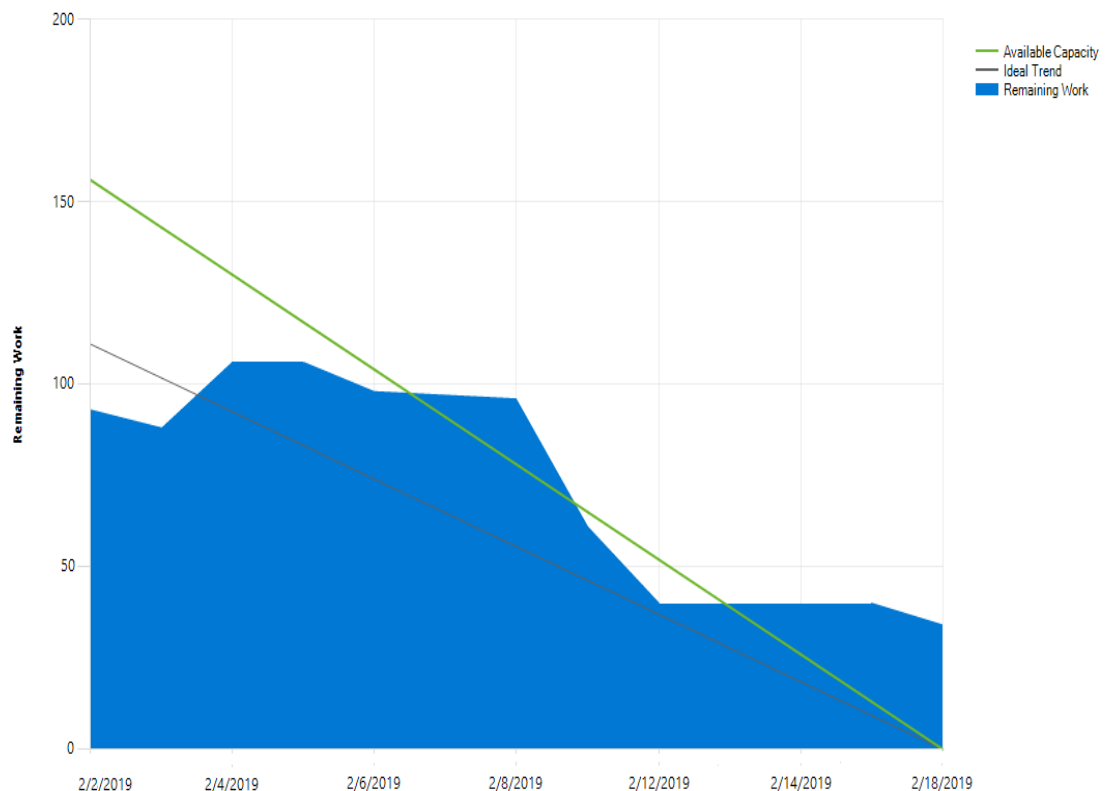
Após a definição do cronograma do projeto, um plugin do Azure DevOps é utilizado para gerar uma tarefa correspondente para cada atividade do cronograma, já importando as datas de início e fim e o esforço estimado para cada atividade, que passa a ser gerenciado no Azure DevOps.

Para a monitoração do projeto foi utilizado apenas o Azure DevOps. A equipe do projeto reporta as datas de início e fim reais através das mudanças realizadas do estado das tarefas no Azure DevOps e o coordenador do projeto e a equipe do projeto recebem os e-mails de mudança de estado das tarefas, evidenciando o andamento do projeto. Além disso, a ferramenta possui funcionalidades específicas para projetos com práticas de metodologias ágeis, como o Dashboard e o Burndown Chart do projeto. A Figura 23 mostra o Burndown do Sprint 1.

Figura 23: Burndown do Sprint 1

Burndown for: Sprint 01

[How do I read this chart?](#)



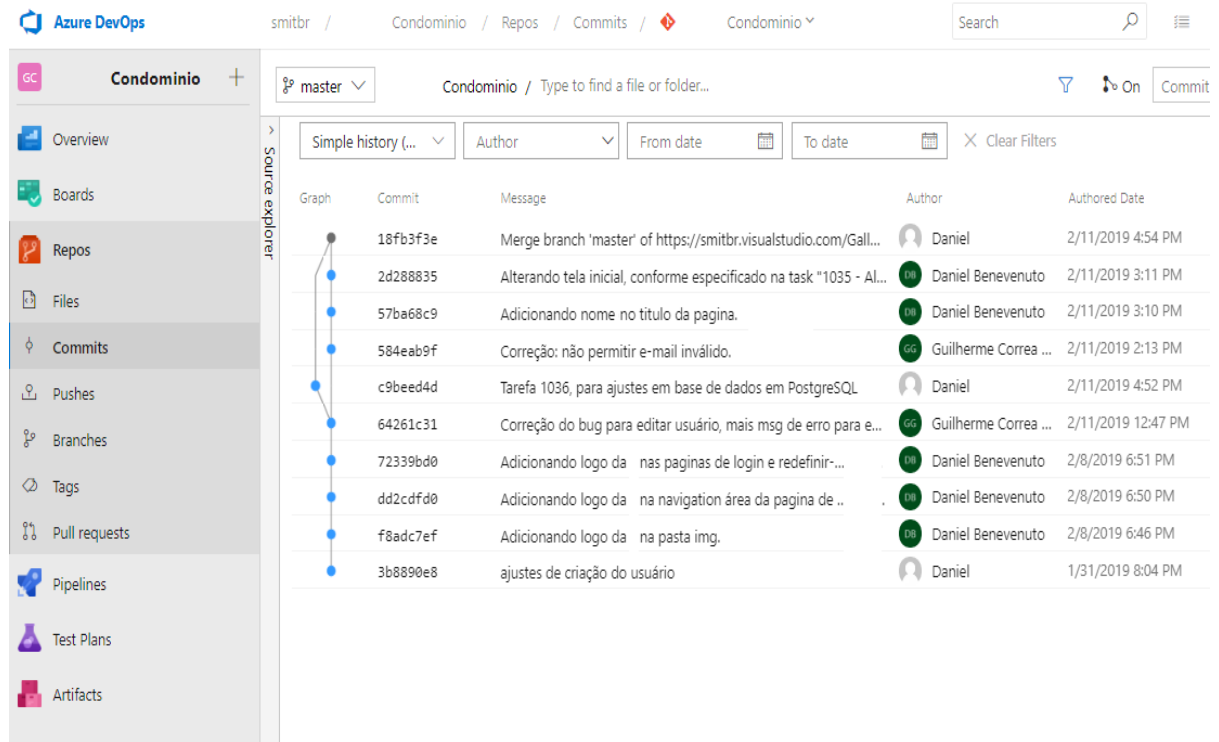
A fase “Avaliar” foi realizada de forma idêntica à empresa A em relação a coleta de dados.

Em relação a fase “Refletir”, o pesquisador, a partir das coletas de dados realizadas, apresenta uma síntese dos resultados:

- Todas as informações de andamento do projeto se encontram apenas no Azure DevOps, o que facilita o controle do projeto por parte dos coordenadores e permite que a geração das atas de monitoração seja feita rapidamente. Nas atas constam as informações do andamento do projeto retiradas de relatórios do Azure DevOps, os impedimentos identificados na reunião e as ações definidas para tratá-los. As ações são cadastradas e acompanhadas no Azure DevOps.
- O principal desafio com relação a Gerência de Configuração foi decidir em quais momentos seriam geradas as baselines. Como os sprints possuem normalmente a duração de duas semanas, optou-se por gerar apenas uma baseline ao final de cada sprint, o que não compromete a qualidade final do produto, já que as auditorias ocorrem antes de a documentação ser entregue ao cliente.
- Além do problema do curto período dos sprints, dentro de um mesmo sprint, cada caso “User Story” foi tratado por vez, indo desde a sua análise até sua homologação interna, o que faz com que os documentos sejam finalizados apenas no final do sprint. Concluiu-se que, neste caso específico, baselines intermediárias travariam muito o processo, prejudicando sua agilidade e impedindo que sua duração permanecesse em duas semanas.
- A ferramenta Git foi utilizada para o versionamento e controle de configuração dos documentos e código fonte. Os mecanismos de branch e tags são utilizados para gerar baselines, efetuar correções e controlar as alterações nos itens de configuração, juntamente com o Azure DevOps, onde os pedidos de alteração são registrados e analisados. O Azure DevOps possui integração com o GIT, mostrado na Figura 24 possibilitando a ativação de uma restrição que exige que cada operação

de commit no repositório do projeto esteja associada a alguma tarefa definida no Azure DevOps.

Figura 24: Repositório GIT



- O Azure DevOps é uma ferramenta colaborativa, fato esse que contribui para que os princípios Lean que tratam de amplificação do conhecimento e fortalecimento da equipe sejam plenamente disseminados.

No seminário final envolvendo o pesquisador e o time pode-se concluir através dos depoimentos do time:

- A empresa se apresentou motivada quanto a reprodução do modelo PROAGIL-29110 em projetos futuros, uma vez que a mesma observou que os novos processos adicionados a sua rotina de trabalho não geraram lentidão ou burocratização em seu processo de desenvolvimento.

- Um grande facilitador a esta iniciativa foi a percepção de que estes novos processos poderiam ser facilmente absorvidos e organizados na ferramenta já conhecida e utilizada em projetos anteriores. Neste sentido, o Azure DevOps mostrou-se configurável e flexível na adoção do PROÁGIL-29110, tornando-se assim um aliado na implementação do novo processo.

Em relação a empresa B, as análises em relação a algumas cerimônias/artefatos Scrum e princípios Lean são apresentadas no Quadro 13.

Quadro 14: Análise dos itens da cerimônia / artefatos Scrum e princípios Lean

Item	Vantagens/ Facilidades	Desvantagens / Dificuldades
Backlog do produto	Fundamental para o planejamento do time	Não houve dificuldade em função do projeto não apresentar uma lista extensa de requisitos.
<i>Backlog</i> do Sprint	Fundamental para o planejamento do time	Não houve dificuldade em função do projeto não apresentar uma lista extensa de requisitos.
Reunião diária	Sua utilização aumentou muito a discussão dos problemas enfrentado no dia a dia do projeto.	No primeiro Sprint, em algumas reuniões foram abordados assuntos fora do escopo do projeto.
Sprint	Permite visibilidade do que vai ser construído.	Assim como na empresa A, existiu dificuldade na execução do processo pela falta de experiência do time.
Planejamento do Sprint	Deu ao time a oportunidade de troca de experiência no que se refere ao desenvolvimento de requisitos.	O tempo estabelecido para reunião deve ser estabelecido e mantido.
Revisão do Sprint	Momento ideal para coleta de oportunidades de melhorias.	Deve ser sempre executada.
Amplificar o conhecimento	Assim como na empresa A, a realização das reuniões diárias e a revisão do Sprint, aumentou significativamente o entendimento das tarefas a serem realizadas no dia	Deve ser sempre incentivada a participação de todos do time. O aspecto motivacional se mostra nesse aspecto fundamental no processo.

Dessa forma a pesquisa-ação atingiu seus objetivos que era avaliar o uso do processo PROÁGIL-29110 em empresas brasileiras produtoras de software que contribuíram com o aprimoramento do processo.

...

7. CONCLUSÕES

Essa tese teve como principal motivação desenvolver e implementar um processo ágil para pequenas empresas baseado na norma ISO/IEC 29110 e aderente ao método Scrum e os princípios da filosofia Lean, visando principalmente a construção de software de alta qualidade e aderente às práticas internacionais, porém aplicáveis às VSE's brasileiras.

Para isso o pesquisador realizou uma revisão bibliométrica e sistemática utilizando-se o método científico Systematic Literature Review (SLR) para buscar em bases científicas o estado da arte envolvendo os pilares dessa tese, ou seja, a ISO/IEC 29110, Scrum e Lean.

O processo PROÁGIL-29110 foi desenvolvido e enviado a duas especialistas, cujas observações foram incorporadas às tarefas elaboradas com o objetivo de validá-lo sob a ótica dos constructos que formam a tese. O processo PROÁGIL foi estruturado nos pilares ISO/IEC 29110, Scrum e Lean.

Duas empresas foram escolhidas para que através do método de pesquisa-ação o processo PROÁGIL-29110 pudesse ser implementado em projetos e o pesquisador realizasse a análise de dados coletados durante a execução das práticas do processo para que se fizesse adaptações no processo e uma análise dos resultados encontrados.

Em relação à implementação do processo, um benefício relatado pelos envolvidos nos projetos de ambas empresas foi que houve uma grande melhoria na comunicação e aumento na colaboração entre a equipe, em função da semântica da comunicação ser padronizada pelo PROÁGIL-29110. A pesquisa detectou a necessidade da construção de uma cultura de autogerenciamento, de forma a convencer o time de desenvolvimento que eles são os próprios gerentes.

A implementação na empresa B mostrou a importância de se encontrar ferramentas computacionais que contribuam para implementação do processo. A automação torna-se importante no sentido de otimizar tarefas e dar visibilidade a todos do time a respeito do andamento do projeto.

Os pilares dessa tese (ISO/IEC 29110, método ágil Scrum e princípios da filosofia Lean) se mostraram aplicáveis e alguns exemplos podem ser vistos no Quadro 15 que apresenta uma comparação do cenário e dos resultados obtidos nas empresas que participaram da pesquisa-ação, em relação a itens associados a desenvolvimento de projetos.

Quadro 15: Quadro comparativo antes e após a implantação do PROÁGIL-29110 nas duas empresas

Item	Antes do PROÁGIL-29110	Depois da implementação do PROÁGIL-29110
Gerência do projeto	Os projetos tinham coordenadores que possuíam como responsabilidades a monitoração do projeto, o contato com o cliente. O progresso do projeto é pouco visível.	O time se autogerencia. Todos os membros do time têm comprometimento com seu sucesso. O progresso do projeto pode ser acompanhado pela técnica Kanban.
Contato com o cliente	Somente o gerente do projeto tinha contato com o cliente.	Criou-se a figura do dono do produto, o qual tem grande interação com o cliente e conhece suas regras de negócio.
Velocidade	A velocidade do time não era medida. O único controle eram os prazos.	A velocidade do projeto é medida e um gráfico de controle visual de toda a equipe é atualizado diariamente (Burndown).
Planejamento do projeto	O projeto era inteiramente planejado em seu início em detalhes.	O projeto é rapidamente planejado no início e são realizados replanejamentos em cada <i>Sprint</i> . A realização dessa prática em outros projetos futuros resultará em fortalecimento da equipe.
Funcionalidades	As funcionalidades eram implementadas sem uma ordem específica.	Os itens de maior prioridade são implementados antes, de modo que o retorno do investimento ocorra mais rapidamente e funcionalidades opcionais são implementadas se o tempo permitir.
Lições Aprendidas / amplificação do conhecimento	Não havia método formal para documentar as lições aprendidas.	O time realiza reuniões de revisão para documentar as lições aprendidas e discussões para promover a disseminação do conhecimento.
Riscos	A gestão de riscos não era uma tarefa formal e ficava sempre a cargo do gerente do projeto.	Existe o <i>Backlog</i> de Impedimentos, que é uma boa ferramenta para todo o time conhecer os riscos e cobrar a mitigação de riscos conhecidos.
Entregas	São definidas no início do projeto. Geralmente é feita apenas uma entrega, já com o produto final.	As entregas são diversas (rápidas) durante todo o projeto, sempre com um produto com melhorias incrementais em relação ao anterior.

Como exemplo dos ganhos obtidos nessas empresas cita-se a velocidade do time que pode ser mensurada no dia a dia do projeto com a utilização da técnica de Burndown e no progresso do projeto visualizado pela técnica Kanban.

7.1 CONTRIBUIÇÕES

Dentre as principais contribuições desse trabalho pode-se destacar:

- Utilizar o método de revisão bibliográfica SLR que permite esquematizar todo o processo de pesquisa em bases científicas;
- Propor um processo que permita à pequenas empresas se tornarem competitivas no mercado globalizado;
- Mostrar de forma prática e acadêmica o impacto da implementação de um processo ágil em duas pequenas empresas desenvolvedoras de software;
- Apresentar os resultados da aplicação do processo desenvolvido em duas pequenas empresas desenvolvedoras de software que podem ser replicados em outras organizações;
- Demonstrar que o uso combinado dos pilares do processo proposto traz benefícios para as empresas, bem como a eficácia da aplicação do PROÁGIL-29110.

Essas contribuições forneceram dados para:

- ✓ Artigo “Análise crítica e comparativa entre a Norma ISO/IEC 29110 e o modelo MPS.BR nível G”, apresentado e premiado como um dos 3 melhores artigos no Setii - Seminário em Tecnologia da Informação Inteligente, São Paulo, 2017;
- ✓ Livro a ser escrito pelo pesquisador e o orientador como contribuição para disseminação do processo para as pequenas empresas;
- ✓ Registro da marca PROÁGIL-29110 junto ao INPI;

- ✓ Desenvolvimento de artigos contemplando os resultados obtidos e submissão a periódicos indexados.

7.2 LIMITAÇÕES DA PESQUISA

Essa pesquisa foi realizada em duas organizações. A reprodutibilidade, característica levantada no item correspondente a validação do processo, se concretizou, porém, uma aplicação em maior escala nas organizações brasileiras se torna necessário para consolidar o processo desenvolvido.

7.3 TRABALHOS FUTUROS

Propõe-se a investigação que mostre o ganho de qualidade no produto de software, da satisfação dos clientes e da diminuição do custo no projeto. Essa mensuração, bem como outras métricas podem ser desenvolvidas para que uma quantificação possa ser associada à implementação do processo PROÁGIL-29110.

Um outro ponto a ser explorado é a integração do DevOps ao processo PROÁGIL-29110 e sua implementação em empresas.

Acredita-se que fatores motivacionais e relações interpessoais entre o time de projetos exercem influência na evolução do mesmo. Uma pesquisa abrangente envolvendo aspectos emocionais relacionados com processos contemplando métodos ágeis poderia ser aplicada.

Outros benefícios que podem ser identificados e para tanto outros trabalhos poderiam contemplar:

- Redução de custos em projetos de software;
- Menos retrabalho e por consequência um aumento de eficiência e competitividade;
- Exemplos de empresas que implementaram o PROÁGIL-29110 e obtiveram a certificação ISO/IEC 29110;
- Facilidades e barreiras na implementação do PROÁGIL-29110.

GLOSSÁRIO

DESENVOLVIMENTO: Todas as atividades a serem realizadas para criação de um produto de software.

HISTÓRIA DE USUÁRIO: Requisito capturado através de conversas com o *Product Owner*, onde é descrito as necessidades de forma clara utilizando uma linguagem simples.

MÉTODO: Abordagem técnica, passo a passo, para realizar uma ou mais tarefas indicadas na metodologia.

MÉTODO CIENTÍFICO: Conjunto das normas básicas que devem ser seguidas para a produção de conhecimentos que têm o rigor da ciência, ou seja, é um método usado para a pesquisa e comprovação de um determinado conteúdo.

METODOLOGIA: Maneira ou forma de se utilizar um conjunto coerente e coordenado de métodos para atingir um objetivo, de modo que se evite, tanto quanto possível, a subjetividade na execução do trabalho.

MODELO: Estratégia de desenvolvimento que engloba processos, métodos, ferramentas e fases de desenvolvimento.

MODELO DE QUALIDADE: Conjunto de práticas ligadas aos processos de gestão e ao desenvolvimento de projetos.

PROCESSO: Conjunto de passos parcialmente ordenados que visam atingir um objetivo. Na engenharia de software, a meta é criar um produto de software ou aprimorar um já existente.

PROCESSO DE SOFTWARE: Conjunto de ferramentas, métodos e práticas que as pessoas alocadas em um projeto usam para desenvolver um produto de software.

PRODUTO DE SOFTWARE: Conjunto completo de programas de computador, procedimentos e documentação correlata, assim como dados designados para entrega a um usuário.

TÉCNICA: Modo apropriado de se investigar sistematicamente um universo de interesse ou domínio do problema.

REFERÊNCIAS

ABESSOFTWARE, 2018. **Mercado Brasileiro de Software: panorama e tendências.** Disponível em: <http://central.abessoftware.com.br/Content/UploadedFiles/Arquivos/Dados%202011/ABES-Publicacao-Mercado-2017.pdf> . Acesso em: 12 abr.2018.

ABNT, ISO/IEC TR 29110-5-1-2 – **Engenharia de Software – Perfis de ciclo de vida para micro organizações (VSEs) - Parte 5-1-2: Guia de Engenharia e Gestão** – válida a partir de 2012.

ABRAHAMSSON, P.; SALO, O; RONKAINEN, J; **Agile Software Development Methods.** VTT Technical research centre of Finland, 2002.

AI-BAIK, O, MILLER, J. **Empirical Software Engineering** vol. 20, DOI: 10.1007/s10664-014-9340-x, 1861–1897, 2015.

AI-HAWARI, F.; ALUFEISHAT, A.; ALSHAWABKEH, M.; BARHAM, H. **The software engineering of a three-tier web-based student information system** Journal Computer Applications in Engineering Education, March 2017, Vol.25(2), pp.242-263, 2017.

AHERN, D., CLOUSE, A. TURNER, R. **CMMI Distilled.** Addison-Wesley, 2008.

ALEEM, S.; CAPRETZ, L.; AHMED, F. **Critical Success Factors to Improve the Game Development Process from a Developer's Perspective** Journal of Computer Science and Technology, Sep 2016, Vol.31(5), pp.925-950, 2016.

ALMEIDA, C. D. A. **Continuidade da Execução dos Processos de Software em Empresas Avaliadas no MPS.BR.** Dissertação de mestrado. Universidade de Fortaleza, 2011.

ALVEAR, S.E. **Gestión documental por procesos con AURA PORTAL, integrado al proceso de Desarrollo de Software en la Universidad Técnica del Norte** Maestria em Engenharia de software, Universidad técnica del norte, Equador, 2018.

ANDERSON, D. **Kanban-Successful Evolutionary Change for Your Technology Business** WA: David J. Anderson & Associates Inc, Seattle, 2010.

ANDERSON, D; CONCAS, G; LUNESU, M.I; MARCHESI, M. **Studying Lean–Kanban approach using software process simulation** International Conference on Agile Software Development, Berlin Heidelberg, Springer, pp. 12-26, 2011.

BALASUBRAMANIAN, L.; MNKANDLA, E. **An evaluation to determine the extent and level of Agile** Software Development Methodology adoption and implementation in the Botswana Software Development Industry International Conference on Advances in Computing and Communication Engineering (ICACCE), Durbanm, South Africa, 2016.

BASKERVILLE, R. **Investigating Information Systems with Action** Research Communications of the Association for Information Systems: Vol. 2, Article 19, 1999.

BERG, V.; BIRKELAND, J.; NGUYEN-DUC, A.; PAPPAS, I. O. **Software startup engineering: A systematic mapping study** The Journal of Systems & Software, October 2018, Vol.144, pp.255-274, 2018.

BIRÓ, M.; MESSNARZ, R.; COMO-PALACIOS, R. **Process improvement approaches fertilised by advances in SPI** Journal of software: evolution and process. Published online 4 July 2015 in Wiley Online Library (wileyonlinelibrary.com). <<http://doi.org/10.1002/smr.1725>> , 2015.

BLAS, M.J.; GONNET, S.; LEONE, H. **An ontology to document a quality scheme specification of a software product.** Wiley Online Library <https://onlinelibrary.wiley.com/doi/abs/10.1111/exsy.12213>> Acesso em: 07 fev.2018.

BOEHM, B. W., & TURNER, R. **Balancing agility and discipline: a guide for the perplexed.** Addison-Wesley, 2003

BRERETON, P *et al.*, **Lessons from applying the systematic literature review process within the software engineering domain.** Journal of Systems and Software, 571–583, 2006.

BRYMAN, A. **Research methods and organization studies** London: Unwin Hyman, London, 1989

BUCHALCEVOVA, A., **Application of Methodology Evaluation System on Current IS Development Methodologies** International Journal of Information Technologies and Systems Approach (IJITSA), 2018.

BUCHALCEVOVA, A. **Software Process Improvement in small companies** Department of Information Technologies, Prague University of Economics, W. Churchill sqr. 4, 13067 Prague 3, 2015.

BUSTAMANTE A.F., RINCÓN, R.C. **WYDIWYN – What You Define, Is What You Need: Defining Agile/Traditional Mixed Methodologies** Springer International Publishing AGJ. Mejia et al. (eds.), Trends and Applications in Software Engineering, 2018.

CALDERÓN, A.; RUIZ, M.; O'CONNOR, R.V. **A multivocal literature review on serious games for software process standards education** Journal Computer Standards & Interfaces, Elsevier, March 2018, Vol.57, pp.36-48, 2018.

CHRISSIS, M. B.; KONRAD, M.; SHURM, S. **CMMI: Guidelines for Process Integration and Product Improvement**. Boston: Addison-Wesley, 2004

CHECKLAND, P.; HOLWELL, S. **Information, systems, and information systems: making sense of the field**. Chichester: Wiley, 1988.

CLARKE, P.; YILMAZ, M. **Software Process Improvement and Capability Determination** Conference 2016 Computer Standards & Interfaces, November 2016, Vol.54, pp.117-118, 2016.

CMMI Institute, <https://sas.cmmiinstitute.com/pars/pars.aspx>. Acesso em: 09 fev. 2018.

COCKBURN, A. HIGHSMIYH, J. **Agile Software Development: The business of innovation**. Journal Computer vol 34 pg 120 –122, IEEE Computer, Los Alamitos, CA, 2001.

COCCO. L. *et al.* **Simulating Kanban and Scrum vs. Waterfall with System Dynamics**. In: Lecture Notes in Business Information Processing vol. 77, pp. 117–131, 2011.

COLOMO-PALACIOS, R.; BIRÓ, M.; MESSNARZ, R., **Special issue on software and service improvement in the scope of SMEs** Software Quality Journal, Sep 2016, Vol. 24(3), pp.485-487, 2016.

CONCAS, M.; I. LUNESU, M. MARCHESI; ZHANG, H. **Simulation of software maintenance process, with and without a work-in-process limit** Journal Software. Evolution Process, pp. 1225-1248, 2013.

COSTA, I. **Como implantar métodos ágeis em desenvolvimento de software** ITAC – Instituto de Tecnologia Aragon Costa, 2013.

COUGHLAN, P.; COUGHLAN, D. **Action research for operations management. International Journal of Operations & Production Management**, v.22, n. 2, p. 220-240, 2002.

COUTO, A.B. **CMMI: Integração dos Modelos de Capacitação e Maturidade de Sistemas**. Ed. Ciência Moderna, São Paulo: 2007.

CROSBY, P.B. **Quality is free: The Art of Making Quality Certain**. New York: McGraw-Hill, 1979.

CRUZ, F **Scrum e Agile em projetos: guia completo** 2.ed. Rio de Janeiro: Brasport, 2018.

CRUZ, E; TRUJILLO, M. **Disciplinando Equipos Pequeños con Prácticas Ágiles** Grupo de Investigación KUALI-KAANS, Universidad Nacional Autónoma de México. Published by DIF U100ci@ <<https://nautilus.uaz.edu.mx/difu100cia>> Ciudad Universitaria, Distrito Federal, México, 2014.

CURTIS, W. et al, **Process Modeling** Communications of the ACM, set, 1992.

DAHLEM, D. **Agile Practices in practice: a mapping study** In 18th International Conference on Evaluation and Assessment in Software Engineering. New York: ACM, 2014.

DAVINSON, R.; MARIS, M.; KOCK, N. **Principles of canonical action research**. Inf. Syst. J.14. 65-. 10.1111/j.1365-2575.2004.00162. x., 2004.

DHARMAPAL, S.; SIKAMANI, K. **Applying Lean on Agile Scrum Development Methodology** In: Compusoft, An international journal of advanced computer technology 3 (Mar. 2014), pp. 633–639, 2014.

DEMING, W. E. **Quality, Productivity, and Competitive Position** Cambridge: MIT Press, 1982.

DIAZ, A.; DE JESUS, C.; MELENDEZ, K.; DAVILA, A. **The ISO/IEC 29110 Implementation on two Very Small Software Development Companies in Lima** Lessons Learned IEEE Latin America Transactions, May 2016, Vol.14 (5), pp.2504-2510, 2016.

DINGSOYR, T. **A decade of agile methodologies: Towards explaining agile software development** Journal of Systems and Software, 85 (6), 1213–1221., 2012.

DOWSON, M. et al. **Concepts for Process Definition and Support** Proc. of 6th International Software Process Workshop, Washington DC, 1991.

DUARTE, C.H.C. **Productivity paradoxes revisited** Springer Science Business Media, New York, 2017.

DYBA, T.; DINGSOYR, T. **Empirical studies of agile software development: A systematic review** Information and Software Technology, 833–859, 2008.

EBSE Technical Report, **Guidelines for performing Systematic Literature Reviews in Software Engineering** Version 2.3, 2007.

EITO-BRUN, R. **Incorporating Innovation Management Practices to ISO/IEC 29110** 24th European Conference Proceedings, 24th European Conference EURO SPI p15-25, 2017, Ostrava, Czech Republic, 2017.

EITO-BRUN, R.; AMESCUA, A. **Dealing with software process requirements complexity: an information access proposal based on semantic technologies** Requirements Engineering Journal, Nov 2017, Vol. 22 (4), pp.527-542, 2017.

EITO-BRUN, R.; SICILIA, M. **An innovation activity model for Very Small Entities in the software sector: an empirical study** R&D Management, November 2017, Vol.47 (5), pp. E13-E25, 2017.

FEIGENBAUM, A. **Total Quality Control** 3 ed. São Paulo: McGraw-Hill, 1991.

FELDERER, M *et al.*, **Software Paradigms, Assessment Types and Non-Functional Requirements in Model-Based Integration Testing: A Systematic Literature Review**, EASE '14, May 13 - 14 2014, London, England, BC, United Kingdom, ACM 978-1-45 <http://dx.doi.org/10.1145/2601248.2601257>, 2014.

FELDERER, M.; RAMLER, R. **Risk orientation in software testing processes of small and medium enterprises: an exploratory and comparative study** Software Quality Journal, Sep 2016, Vol. 24 (3), pp.519-548, 2016.

FREITAS, J.; CALBINO, D.; SANTOS, A.; PEREIRA, R.D. **Em Defesa do uso da Pesquisa-Ação na Pesquisa em Administração no Brasil** Administração: Ensino e Pesquisa - Rio de Janeiro - v. 11 - n. 3 - p. 425-445 - Jul/Ago/Set, 2010.

FUERTES A.Y.; SEPÚLVEDA C.J. **Scrum, Kanban and Canvas in the commercial, industrial and educational sector - A literature review** Revista Antioqueña de las Ciencias Computacionales, Vol. 6 Issue 1, p46-50. 5p, jun, 2016.

GALVAN, S.; MORA, M.; O'CONNOR, R.; ACOSTA, F.; ALVARZ, F. **A Compliance Analysis of Agile Methodologies with the ISO/IEC 29110** Project Management Process, Conference on Enterprise Information Systems / International Conference on Project, Vol.64, pp.188-195, 2015.

GHANE, K. **Quantitative planning and risk management of Agile Software Development** IEEE Technology & Engineering Management Conference (TEMSCON), Santa Clara, USA, 2017.

GARCÍA-MIRELES, G. **A Identifying Relevant Product Quality Characteristics in the Context of Very Small Organizations** Journal Computer Science and Information Systems, Vol. 13, No. 3, 875–900, 2017.

GARZÁS, J.; PINO, F.J.; PIATTINI, M.; FERNÁNDEZ, C. **A maturity model for the Spanish software industry based on ISO standards** Journal Computer Standards & Interfaces, Elsevier, November 2013, Vol. 35 (6), pp.616-628, 2013.

GIARDINO, C., PATERNOSTER, N., UNTERKALMSTEINER, M., GORSCHKE, T., ABRAHAMSSON, P., **Software development in startup companies: the greenfield startup model** IEEE Trans. Softw. Eng. Vol. 42, 585–604. <http://dx.doi.org/10.1109/TSE.2015.2509970>, 2016.

GÓMEZ, C.L; GARCIA, A.A.; DEL DEDO, R; **Métodos Ágiles, Scrum, Kanban, Lean (Manuales Imprescindibles)** Anaya Multimedia, 2017.

GONZÁLEZ, J.D. **AgileFM: Modelo de desarrollo ágil formal basado en la ISO/IEC 29110 para las micro, pequeñas y medianas empresas** Tesis de Maestría. Escuela de Ingeniería Medellín, Colombia, 2016.

GONZALEZ-PEREZ, C.; HENDERSON-SELLERS, B.; MCBRIDE, T.; LOW, G.C. **An Ontology for ISO software engineering standards: Proof of concept and application** Journal Computer Standards & Interfaces, Elsevier, November 2016, Vol.48, pp.112-123, 2016.

GORDON, M.L.; O'CONNOR, R. **Understanding the gap between software process practices and actual practice in very small companies** Software Quality Journal, Springer, 2016.

GRÄßLER, I.; HENTZE, J. **Application Potentials of Systems Engineering for Small and Middle-sized Enterprises** 11th CIRP Conference on Intelligent Computation in Manufacturing Engineering, 19-21 July 2017, Gulf of Naples, Italy, 2018, Vol.67, pp.510-515, 2018.

GRAZIOTIN, D.; JEDLITSCHKA, A. **Recent developments in product-focused software process improvement** ACM SIGSOFT Software Engineering Notes, 11 November 2013, Vol. 38 (6), pp.29-34, 2013.

GRAZIOTIN, D.; WANG, X.; ABRAHAMSSON, P. **How do you feel, developer? An explanatory theory of the impact of effects on programming performance** PeerJ Computer Science, August 19, 2015, Vol.1, p.e18, 2015.

GUMMESSON, E. **Qualitative Methods in Management Research** 2nd ed. Sage, London, 2000.

HAMMER, M.; CHAMPY, J. **Reengenharia: revolucionando a empresa em função dos clientes, da concorrência e das grandes mudanças da gerência**. Rio de Janeiro: Campus, 1994.

HAMMER, M. **A realidade da reengenharia**. Hsm Management: Informação e Conhecimento para Gestão Empresarial, São Paulo, v. 1, n. 2, p.20-23, maio/jun., 1997.

HOSS, M. **Soft Systems Methodology como forma de operacionalizar o processo de estruturação da transformação Lean sob a perspectiva da escola de pensamento evolucionária: uma pesquisa-ação** Tese de Doutorado apresentada no Programa de Pós-Graduação em Engenharia de Produção (PPGEP) da Universidade Federal do Rio Grande do Sul, UFRGS, 2011.

HOUDE, R.; LAPORTE, C.Y.; BLONDELLE, G. **ISO/IEC 29110 Deployment Packages and Case Study for Systems Engineering: The “Not-So-Secret” Ingredients That Power the Standard** INCOSE International Symposium, July 2016, Vol. 26 (1), pp.1276-1292, 2016.

HUMPHREY, S. A. e KELLNER, M. I., **Software Process Modeling: Principles of Entity Process Models** Proc. of 11th International Conference on Software Engineering, Los Alamitos, CA, 1989.

HURTADO, J. A. **A meta-process for defining adaptable software process** Tesis para optar al grado de Doctor em Ciencias Computacionais, Universidad del Chile, 2012.

IBRAHIM, L.; WALLMÜLLER, E.; DASCHNER, W. **Using Enterprise SPICE in Very Small Entities Software** Journal Quality Professional, Mar 2017, Vol. 19 (2), pp.43-49, 2017.

IEEE Std 1061-1998 - **IEEE Standard for a Software Quality Metrics Methodology**, 1998.

IRAMA, K. **Engenharia de Software: Qualidade e Produtividade com Tecnologia** São Paulo: Ed. Campus, 2012.

ISHIKAWA, K. **What Is Total Quality Control? The Japanese Way** Hardcover, 1985.

HAYAT, M.; M. QURESHI, R. J. **Measuring the Effect of CMMI Quality Standard on Agile Scrum Model** Cornell University Library, 2016.

JIMMY R.M.R. **Estado del Arte: Métricas de calidad para el desarrollo de aplicaciones web** Revista 3C Tecnología, Edición 26, 2017, Vol.6 (4), pp.1-12, 2017.

JOHNSON, R.B; ONWUEGBUIZIE, A.J. **Mixed Methods Research: A Research Paradigm** Educational Researcher, Vol. 33, No. 7 (Oct., 2004), pp. 14-26, 2004.

JURAN, J.M. **Quality Control Handbook** New York: McGraw-Hill, 1988.

KALINOWSKI, M; SANTOS, G.; REINEHR, S; MONTONI, M; ROCHA, A.R.; WEBER, K.C.; TRAVASSOS, G.H. **MPS.BR: Promovendo a Adoção de Boas Práticas de Engenharia de Software pela Indústria Brasileira** CibSE 2010: XIII Conferencia Iberoamericana en Software Engineering, Cuenca, Equador, 2010.

KARAMBIRI, A. **A review of agile methodology in software development** India International Research Journal of Engineering and Technology (IRJET) Volume: 03 Issue: 03 March-2016, 2016.

KASURINEN, J.; RISTO LAINE, R.; SMOLANDER, K; HEIDRICH, J **How Applicable Is ISO/IEC 29110 in Game Software Development?** Profes 2013, LNCS 7983, pp. 5–19, 2013. © Springer-Verlag Berlin Heidelberg, 2013.

KEMMIS, S.; McTAGGART, R. **Participatory Action Research: Communicative Action and the Public Sphere.** The Sage handbook of qualitative research (pp. 559-603). Thousand Oaks, CA: Sage Publications Ltd, 2005.

KEOGH, L. **Scrum and Kanban both the Same, only Different** Disponível em <http://lizkeogh.com/2011/11/20/scrum-and-kanban-both-the-same-only-different/>, 2011. Acesso em: 08 jan. 2018.

KETO, H.; JAAKKOLA, H. **Identifying up-to-the-minute topics in software process improvement research** Tampere University of Technology, Proceedings of the SQAMIA 2017: 6th Workshop of Software Quality, Analysis, Monitoring, Improvement, 2017.

KITCHENHAM, B; PLEEGER, S.L. **Software Quality: The Elusive Target** IEEE Computer Society Press Los Alamitos, CA, USA, 1996.

KITCHENHAM, B., CHARTES, S. **Guidelines for performing Systematic Literature Reviews in Software Engineering.** Staffordshire: Elsevier, 2007.

KIVAL, C.W *et al* MPS.BR - **Melhoria de Processo do Software Brasileiro: resultados alcançados e lições aprendidas (2004 - 2008).** Disponível em <http://www.softex.br/mps.html>. Acesso em: 05 jan. 2018.

KLIEM, R. **Managing Lean Projects** 1sted. Florida, USA: CRCPress, ISBN: 978-1-4822-5183-8, 2016.

KNIBERG, H; SKARIN, M. **Kanban and Scrum—making the most of both** 1st ed. Washington, USA: C4Media Inc. ISBN: 978-0-557-13832-6, 2010. Acesso em: 21 mai 2018.

KOCK, N.F. JR.; MCQUEEN, R.J.; SCOTT, J.L. **Can Action Research be Made More Rigorous in a Positivist Sense? The Contribution of an Iterative Approach** Journal of Systems and Information Technology, v.1, n.1, p. 1- 24, 1997.

KOSCIANSKI, A.; SOARES, M.S. **Qualidade de Software: Aprenda as metodologias e técnicas mais modernas para o desenvolvimento de software.** 2ª Ed. Ed, Novatec, 2006.

KUHRMANN, M.; DIEBOLD, P.; MUNCH, J. **Software process improvement: a systematic mapping study on the state of the art**. PeerJ Computer Science, May 23, 2016, Vol.2, p.e62, 2016.

KUHRMANN, M. **How Does Software Process Improvement Address Global Software Engineering?** IEEE 11th International Conference on Global Software Engineering (ICGSE), CA, USA, 2016.

KUMAR, T.H; SHANKAR, C. **Lean as Agile methodology – A Study** International Journal of Advanced Networking and Applications vol. 7 ISSN: 0975-0290, pp. 2949–2952, 2016.

LAKATOS, E. M.; MARCONI, M. A. **Fundamentos da Metodologia Científica**. 4. ed. São Paulo: Atlas, 2006.

LARMAN, C. **Agile and Iterative Development: A Manager's Guide** 1st Edition, Agile Software Development Series, 2004.

LAPORTE, C.Y.; O'CONNOR, R.V. **Systems and Software Engineering Standards for Very Small Entities: Accomplishments and Overview** IEEE Computer, Aug. 2016 Vol.49 (8), pp.84-87, 2016.

LAPORTE, C.Y.; O'CONNOR, R.V. **Systems and Software Engineering Standards for Very Small Entities: Implementation and Initial Results** 9th International Conference on the Quality of Information and Communications Technology, Sept. 2014, pp.38-47, 2014.

LAPORTE, C.; O'CONNOR, R. **Software Process Improvement in Industry in a Graduate Software Engineering Curriculum** Journal Software Quality Professional Journal, Jun 2016, Vol.18(3), pp.4-17, 2016.

LAPORTE, C.; MUNOZ, M.; MEJIA M.J.; OCONNOR, R.V. **Applying Software Engineering Standards in Very Small Entities: From Startups to Grownups** IEEE Software, January/February 2018 Vol. 35 (1), pp.99-103, 2018.

LAPORTE, C.; HÉBERT, C.; MINEAU, C. **Development of a Social Network Website Using the New ISO/IEC 29110 Standard Developed Specifically for Very Small Entities** Software Quality Professional, Vol.16 (4), pp.4-25, 2014.

LAPORTE, C.; BERRHOUMA, N.; DOUCET, M. **Measuring the Cost of Software Quality of a Large Software Project at Bombardier Transportation: A**

Case Study Software Quality Professional Journal, Jun 2012, Vol.14 (3), pp.14-31, 2012.

LAPORTE, C.Y., O'CONNOR, R.V., **Implementing process improvement in very small enterprises with ISO/IEC 29110: a multiple case study analysis.** Proceedings of the 10th International Conference on the Quality of Information and Communications Technology (QUATIC). IEEE, pp. 125–130, 2016

LARRUCEA, X.; O'CONNOR, R.V.; COLOMO-PALACIOS, R.; LAPORTE, C.Y., **Software Process Improvement in Very Small Organizations** IEEE Software, Mar.-Apr.2016 Vol.33 (2), pp.85-89, 2016

LARRUCEA, X.; SANTAMARÍA, I.; COLOMO-PALACIOS, R., **Assessing ISO/IEC29110 by means of ITMark: results from an experience factory** Journal of Software: Evolution and Process, November 2016, Vol. 28 (11), pp.969-980, 2016.

LEI, H. *et al.* **A statistical analysis of the effects of Scrum and Kanban on software development projects** Robotics and Computer- Integrated Manufacturing vol. 43, pp. 59–67, 2017.

LÓPEZ-LIRA HINOJO, F. **Agile, CMMI®, RUP®, ISO/IEC 12207: is there a method in this madness?** ACM SIGSOFT Software Engineering Notes, 29 March 2014, Vol. 39 (2), pp.1-5, 2014.

MACIEL, A. C. F.; VALLS, C.; SAVOINE, M. M. **Análise da Qualidade de Software utilizando as normas 12207, 15504, ISO 9000-3 e os modelos CMM/CMMI e MPS.** BR Revista Científica do ITPAC, Araguaína, vol. 4, n. 4, Pub. 5, out, 2011.

MANIFESTO, Agile. **Principles behind the Agile Manifesto** Disponível em <http://agilemanifesto.org/principles.html>. Acesso em: 10 jun 2018.

MARTINS, G. A.; THEÓPHILO, C. R. **Metodologia da investigação científica para ciências sociais aplicadas** São Paulo: Atlas, 2009.

MAS, A.; MESQUIDA, A. **Software process improvement and capability determination** Journal Computer Standards & Interfaces, Elsevier, 2017 <https://doi.org/10.1016/j.csi.2018.05.002> vol 60, 2017.Acesso em 28 ago. 2018.

MASSARI, V.L. **Agile Scrum Master no gerenciamento avançado de projetos** São Paulo: Brasport, 2017.

McKAY, J.; MARSHALL, P. **The Dual Imperatives of Action Research**. Information Technology & People, v. 14, n. 1, p. 46-59, 2001. <https://www.emeraldinsight.com/doi/abs/10.1108/09593840110384771> Acesso em: 04 set. 2018.

MEJIA, J.; MUÑOZ, M.; ROCHA, A.; QUIÑONEZ, Y.; MANZANO, J.C **Trends and Applications in Software Engineering** Proceedings of the 6th International Conference on SPI, CIMPS, 2017.

MESQUIDA, A.; MAS, A. **A project management improvement program according to ISO/IEC 29110 and PMBOK** Journal of Software: Evolution and Process, September Vol. 26(9), pp.846-854, 2014.

MITTNER, J.; BUCHALCEVOVA, A. **Towards the IT Support of Processes in Small Software** 2014 Central European Conference on Information and Intelligent Systems, pp.200-206, 2014.

MONTONI, M. A. **Uma Investigação sobre os Fatores Críticos de Sucesso em Iniciativas de Melhoria de Processos de Software** Tese de doutorado. Universidade Federal do Rio de Janeiro/ Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia - UFRJ/COPPE, 2010.

MORALES-TRUJILLO, M.; OKTABA, H.; PIATTINI, M. **The making of an OMG standard** Journal Computer Standards & Interfaces, Elsevier, November 2015, Vol.42, pp.84-94, 2015.

MORGADO, G.P.; GESSER, I.; SILVEIRA, D.S.; MANSO, F.S.P.; LIMA, P.M. V.; SCHMITZ, E.A. **Práticas do CMMI® como regras de negócio** Produção, v. 17, n. 2, p. 383-394, Maio/Ago. 2007.

MTIGWE, B. **The entrepreneurial firm internationalization process in the Southern African context: A comparative approach** International Journal of Entrepreneurial Behavior & Research, vol. 11, 358–377, 2005.

MUNOZ, O., **Especialización de MoProSoft basada en el método ágil Scrum** Editorial académica Española, 2011.

MUÑOZ, M.; GASCA, G.; VALTIERRA, C. **Caracterizando las Necesidades de las Pymes para Implementar Mejoras de Procesos Software: Una Comparativa entre la Teoría y la Realidad** Revista Ibérica de Sistemas e Tecnologías de Informação, Mar 2014, Issue E1, pp.1-15, 2014.

NONOYAMA, T.; WEN, L.; ROUT, T.; TUFFLEY, D.; O'CONNOR, R. **The Impact of Cultural Issues on the Software Process of Very Small Entities** Journal Software Quality Professional, Mar 2018, Vol. 20 (2), pp.59-68, 2018.

O'CONNOR, R.; COLEMAN, G. **An Investigation of Barriers to the Adoption of Software Process Best Practice Models.** Australian Conference on Information Systems, p.780-788, 2007.

O'CONNOR, R.; LAPORTE, C. **An Innovative Approach to the Development of an International Software Process Lifecycle Standard for Very Small Entities** International Journal of Information Technologies and Systems Approach, Jan 1, 2014, Vol.7(1), pp.1-22, 2014.

O'HAGAN, A.; COLEMAN, G. **Software Development Processes for Games: A Systematic Literature Review** EuroSPI 2014, CCIS 425, pp. 182–193, 2014. © Springer-Verlag Berlin Heidelberg, 2014.

OMRAN, A. **Agile CMMI from SMEs perspective** 3rd International Conference (pp. 1–8). ICTTA, 2008.

PALOMINO, M.; DÁVILA, A; MELENDEZ, K.; PESSOA, M **Analysis of Agile Practices Adoption on CMMI Organizations through a Systematic Literature Review** Computacion e Informática ReCIBE, noviembre, 2017.

PANDE, P. **Estratégia Seis Sigma: Como a GE, a Motorola e outras grandes empresas estão aguçando seu desempenho.** 1. ed. Rio de Janeiro: Qualitymark, 2002.

PANE, E. S; SARNO, R. **Capability Maturity Model Integration (CMMI) for Optimizing Object-Oriented Analysis and Design (OOAD)** Procedia Computer Science pag 40 – 48, 2015.

PAREDES, L.; BENAVIDES, L.; SÁNCHEZ, N. **Guía de recomendación para la selección de un modelo de calidad para la mejora de procesos de software (SPI)** Revista Ciencias Estrategicas, Jan-Jun 2017, Vol.25 (37), pp.25-51, 2017.

PAPATHEOCHAROUS, E.; ANDREOU, A. **Empirical evidence and state of practice of software agile teams** Journal of Software: Evolution and Process, <https://doi.org/10.1002/smr.1664> , 2014. Acesso em: 18 ago. 2018.

PARNELL-KLABO, E. **Introducing Lean Principles with Agile Practices at a Fortune 500 Company** Proceedings of AGILE 2006 Conference, 2006.

PAUCAR, L.; LAPORTE, C.; ARTEAGA, J.; BRUGGMANN, M. **Implementation and Certification of ISO/IEC 29110 in an IT Startup in Peru** Software Quality Professional, Mar 2015, Vol.17 (2), pp.16-29, 2015.

PAULK, M.C. et al. **Capability Maturity Model for Software - Version 1.1, Relatório Técnico**. Software Engineering Institute, Carnegie Mellon University, Pittsburgh: SEI, 1993.

PETERSEN, K.; MUJTABA, R. F. **Systematic mapping studies in software engineering** In 12th International Conference on Evaluation and Assessment in Software Engineering (pp. 1–10), 2008.

PINO, F.; GARCÍA, F.; PIATTINI, M.; OKTAB, H. **A research framework for building SPI proposals in small organizations: the COMPETISOFT experience** Software Quality Journal, Sep 2016, Vol. 24(3), pp.489-518, 2016.

POLITOWSKI, C.; FONTOURA, L.M; PETRILLO, F. **Learning from the past: A process recommendation system for video game projects using postmortems experiences** Journal Information and Software Technology, Elsevier, August 2018, Vol.100, pp.103-118, 2018.

POPPENDIECK, M.; POPPENDIECK, T. **Lean Software Development** Addison-Wesley, 2003.

POPPENDIECK, M. **Lean Software Development** 29th International Conference on Software Engineering (ICSE'07 Companion), IEEE, 2007.

PRESSMAN, R.S. **Software Engineering – A Practitioner’s Approach**. 5. Ed. McGraw-Hill, 2010.

PUONTI, P.V **Tailored Project Management Framework from SCRUM and Lean Practices: Case Study of Two Colombian Companies** Master’s Thesis, Alborg University, 2017.

PYTEL, P.; HOSSIAN, A.; BRITOS, P. **Feasibility and effort estimation models for medium and small size information mining projects** Information Systems, Elsevier, January 2015, Vol.47, pp.1-14, 2015.

RAZZAK, M. A. **An Empirical Study on Lean and Agile Methods in Global Software Development** IEEE Computer Society, 2016.

REYES-DELGADO, P.Y **The strengths and weaknesses of software architecture design in the RUP, MSF, MBASE and RUP-SOA methodologies: A conceptual review** Computer Standards & Interfaces Volume 47, Elsevier, August 2016, Pages 24-41 <<http://doi.org/10.1016/j.csi.2016.02.005>> , 2016. Acesso em: 25 jul. 2018.

ROCHA, A. R. C.; MALDONADO, J. C.; WEBER, K. C. **Qualidade de Software: Teoria e Prática** São Paulo: Prentice Hall, 2001.

ROJAS-MONTES, M.; PINO-CORREA, F.; MARTÍNEZ, J. **Proceso de pruebas para pequeñas organizaciones desarrolladoras de software** Revista Facultad de Ingenieria, 2015, Vol. 24 (39), pp.55-70, 2015.

RUMESER, D.; EMSLEY, M. **A systematic review of project management serious games: Identifying gaps, trends, and directions for future research** The Journal of Modern Project Management Vol 6, No 1, 2018.

SABBAGH, R **Scrum- Gestão Ágil para projetos de sucesso** São Paulo: Casa do Código, 2013.

SALVIANO, C. F.: **Uma proposta orientada a perfis de capacidade de processo para evolução da melhoria de processo de software**. PhD thesis, FEEC Unicamp, 2006.

SANCHEZ-GORDON, M.; DE AMESCUA, A.; O'CONNOR, R.; **A standard-based framework to integrate software work in small settings** Journal Computer Standards & Interfaces, Nov 2017, Vol.54, p.162 Larrucea, X., 2017.

SÁNCHEZ-GORDÓN, M.; O'CONNOR, R., **Understanding the gap between software process practices and actual practice in very small companies** Software Quality Journal, 2016, Vol.24(3), pp.549-570, 2016.

SANTOS, R. N. M.; KOBASHI, N. Y. **Bibliometria, cientometria, infometria: conceitos e aplicações** Pesquisa Brasileira em Ciência da Informação, v. 2, n. 1, p. 155–172, 2009.

SANTOS, M. K.; ROCHA, A.; TRAVASSOS, G.H.; WEBBER, K.C.; ANTONIONI, J.A. **MPS.BR Program and MPS Model: Main Results, Benefits and Beneficiaries of Software Process Improvement in Brazil**," 2012 Eighth International Conference on the Quality of Information and Communications Technology (QUATIC), Lisbon, TBD, Portugal, pp 137-142 2012.

SCACCHI, W. e Mi, P. **Process Integration in CASE Environments** IEEE Software, março, 1992.

SCHWABER, K; BEEDLE, M **Agile Software Development with Scrum** Prentice Hall, 2001.

SCHWABER, K.; SUTHERLAND, J. **Guia do SCRUM. Um guia definitivo para o SCRUM: As regras do jogo**, 2013 Disponível em: <https://www.scrumguides.org/scrum-guide.html> Acesso em: 08 set.2018.

SEI **CMMI® for Development, Version 1.3**”, CMMI Product Team Software Engineering Institute, Pittsburgh, November 2010.

SEI, 2017. **Maturity Profile**. Disponível em: <http://partners.cmmiinstitute.com/wp-content/uploads/2017/09/Maturity-Profile-30-June-2017.pdf> Acesso em: 08 abr.2018

SHEA, B. J., GRIMSHAW, J. M., WELLS, G. A., BOERS, M.; ANDERSSON, N.; HAMEL, C. **Development of AMSTAR: A measurement tool to assess the methodological quality of systematic reviews** BMC Medical Research Methodology, 2007.

SJØBERG, D.K., JOHNSEN, A., SOLBERG, J. **Quantifying the Effect of Using Kanban versus Scrum: A Case Study** IEEE Software, pp. 47–54, 2012.

SOFTEX, MPS.BR. **Melhoria de Processo de Software Brasileiro, Guia Geral: 2013** Disponível em: <http://www.softex.br/mpsbr/> Acesso em: 12 dez. 2017.

SOFTEX, 2018. <<https://www.softex.br/mpsbr/>>, 2018 Acesso em: 17 jan. 2018.

SORIA, F. G. **Implantação do CMMI: metodologia baseada na abordagem por processos** Dissertação (mestrado) – Pontifícia Universidade Católica do Paraná, Curitiba, 2006.

SOUZA, R., MALTA, K., ALMEIDA, E.S.D., **Software engineering in startups: A single embedded case study** Proceedings of the 1st IEEE/ACM International Workshop on Software Engineering for Startups, SoftStart. Institute of Electrical and Electronics Engineers Inc., pp. 17–23. <http://dx.doi.org/10.1109/SoftStart.2017.2>, 2017. Acesso em: 15 mai 2018.

SUETIN, S *et al.* **Results of Agile Project Management Implementation in Software Engineering Companies** 6th Seminar on Industrial Control Systems: Analysis, Modeling and Computation In: ITM Web Conferences, pp. 0–4, 2016.

TAKEUCHI, M.; KOHTAKE, N.; SHIRASAKA, S.; KOISHI, Y.; SHIOYA, T.; SHIOYA, K. **Report on an assessment experience based on ISO/IEC 29110** Journal of Software: Evolution and Process, March 2014, Vol.26 (3), pp.306-312, 2014.

TAKEUCHI, H.; NONAKA, I. **The new Product Development Game** Harvard Business Review, 1986.

TANIGUCHI, K; CORREA, F.E. **Metodologias ágeis e a motivação de pessoas em projeto de desenvolvimento de software** Revista de Ciências Exatas e Tecnologia Vol. IV, Nº 4, 2009.

TELES, V.M. **Extreme Programming: aprenda como encantar seus usuários desenvolvendo software com agilidade e alta qualidade** 3. ed. São Paulo: Novatec, 2006.

THIOLLENT, M. **Metodologia da Pesquisa-ação** 16ª ed. São Paulo: Cortez (Coleção temas básicos de pesquisa-ação), 2008.

THIOLLENT, M. **Metodologia da Pesquisa-ação** 18ª ed. São Paulo: Cortez, 2018.

THIRY, M.; GRESSE von W.; ZOUCAS, A.; PICLER, K. **Uma Abordagem para a Modelagem Colaborativa de Processos de Software em Micro e Pequenas Empresas** V Simpósio Brasileiro de Qualidade de Software – SBQS’2006, 2006.

VASQUEZ, L.L.B. **Implementacion de sistema de pre-venta para propuestas de proyectos de software em avantica Technologies** Facultad de ingenieria y arquitectura, Lima, Peru, 2014.

WANG, X; CONBOY, K; Cawley, O. **“Leagile” software development: An experience report analysis of the application of lean approaches in agile software development** J. Syst. Software, 2012.

WIKLUND, k *et al.*, **Impediments for software e test automation: A systematic literature review**, wileyonlinelibrary.com/journal/stvr 1 of 20 <https://doi.org/10.1002/stvr.1639>, 2017 Acesso em: 12 ago. 2018.

WINKLER, D.; O’CONNOR, R.V; MESSNARZ, R. **Systems, Software and Services Process Improvement** 19th European Conference Springer, Austria, 2012.

XABIER, L.; SANTAMARIA, I., **Survival studies based on ISO/IEC29110: industrial experiences** Computer Standards & Interfaces, Elsevier, <<http://doi.org/10.1016/j.csi.2018.04.006>>, 2018 Acesso em: 21 set. 2018.

YIN, R. K. **Case Study Research – Design and Methods** 3.ed. Applied Social Research Methods Series, v. 5, ISBN 0-7619-2553-8, EUA: Sage Publications, 2003.

ZAROOUR, M.; ABRAN, A.; DESHARNAIS, J. **An investigation into the best practices for the successful design and implementation of lightweight software process assessment methods: A systematic literature review** The Journal of Systems & Software, March 2015, Vol.101, pp.180-192, 2015.

APÊNDICE A – ESTUDOS IDENTIFICADOS

Identificador	Referência
ID 1	Takeuchi, M.; Kohtake, N.; Shirasaka, S.; Koishi, Y.; Shioya, T.; Shioya, K. Report on an assessment experience based on ISO/IEC 29110 Journal of Software: Evolution and Process, March 2014, Vol.26(3), pp.306-312, 2014.
ID 2	Galvan, S.; Mora, M.; O'connor, R.; Acosta, F.; Alvarez, F. A Compliance Analysis of Agile Methodologies with the ISO/IEC 29110 Project Management Process, Conference on ENTERprise Information Systems / International Conference on Project, Vol.64, pp.188-195, 2015.
ID 3	Mesquida, A.; Mas, A. A project management improvement program according to ISO/IEC 29110 and PMBOK Journal of Software: Evolution and Process, September Vol. 26(9), pp.846-854, 2014.
ID 4	Paucar, L.; Laporte, C.; Arteaga, J.; Bruggmann, M. Implementation and Certification of ISO/IEC 29110 in an IT Startup in Peru Software Quality Professional, Mar 2015, Vol.17 (2), pp.16-29, 2015.
ID 5	Laporte, C.; Hébert, C.; Mineau, C. Development of a Social Network Website Using the New ISO/IEC 29110 Standard Developed Specifically for Very Small Entities Software Quality Professional, Vol.16(4), pp.4-25, 2014
ID 6	Laporte, C.; Munoz, M.; Mejia M.J.; Oconnor, R.V. Applying Software Engineering Standards in Very Small Entities: From Startups to Grownups IEEE Software, January/February 2018 Vol.35(1), pp.99-103, 2018
ID 7	Laporte, C.Y.; O'Connor, R.V. Systems and Software Engineering Standards for Very Small Entities: Accomplishments and Overview Computer, Aug. 2016 Vol. 49(8), pp.84-87, 2016.
ID 8	Larrucea, X.; O'Connor, R.V.; Colomo-Palacios, R.; Laporte, C.Y., Software Process Improvement in Very Small Organizations IEEE Software, Mar.-Apr.2016 Vol.33(2), pp.85-89, 2016
ID 9	Al-Hawari, F.; Alufeishat, A.; Alshawabkeh, M.; Barham, H., The software engineering of a three-tier web-based student information system Computer Applications in Engineering Education, March 2017, Vol.25(2), pp.242-263, 2017
ID 10	Larrucea, X.; Santamaría, I.; Colomo-Palacios, R., Assessing ISO/IEC29110 by means of ITMark: results from an experience factory Journal of Software: Evolution and Process, November 2016, Vol.28(11), pp.969-980, 2016

ID 11	O'Connor, R.; Laporte, C. An Innovative Approach to the Development of an International Software Process Lifecycle Standard for Very Small Entities International Journal of Information Technologies and Systems Approach, Jan 1, 2014, Vol.7(1), pp.1-22, 2014
ID 12	García-Mireles, G. A Identifying Relevant Product Quality Characteristics in the Context of Very Small Organizations Computer Science and Information Systems, Vol. 13, No. 3, 875–900, 2017.
ID 13	Nonoyama, T.; Wen, L.; Rout, T.; Tuffley, D.; O'Connor, R. The Impact of Cultural Issues on the Software Process of Very Small Entities Software Quality Professional, Mar 2018, Vol.20(2), pp.59-68, 2018
ID 14	Sanchez-Gordon, M.; de Amescua, A.; O'connor, R.; A standard-based framework to integrate software work in small settings Computer Standards & Interfaces, Nov 2017, Vol.54, p.162 Larrucea, X., 2017
ID 15	Xabier, L.; Santamaria, I., Survival studies based on ISO/IEC29110: industrial experiences Computer Standards & Interfaces, Elsevier, https://doi.org/10.1016/j.csi.2018.04.006 , 2018.
ID 16	Sánchez-Gordón, M.; O'Connor, R., Understanding the gap between software process practices and actual practice in very small companies Software Quality Journal, 2016, Vol.24(3), pp.549-570, 2016.
ID 17	Eito-Brun, R.; Sicilia, M. An innovation activity model for Very Small Entities in the software sector: an empirical study R&D Management, November 2017, Vol. 47(5), pp.E13-E25, 2017.
ID 18	Laporte, C.Y. ; O'Connor, R.V. Systems and Software Engineering Standards for Very Small Entities: Implementation and Initial Results 9th International Conference on the Quality of Information and Communications Technology, Sept. 2014, pp.38-47 , 2014.
ID 19	Mas, A.; Mesquida, A. Software process improvement and capability determination Computer Standards & Interfaces, 2017 https://doi.org/10.1016/j.csi.2018.05.002 , 2017
ID 20	Clarke, P.; Yilmaz, M. Software Process Improvement and Capability Determination Conference 2016 Computer Standards & Interfaces, November 2016, Vol.54, pp.117-118, 2016.
ID 21	Calderón, A.; Ruiz, M.; O'Connor, R.V. A multivocal literature review on serious games for software process standards education Computer Standards & Interfaces, March 2018, Vol.57, pp.36-48, 2018
ID 22	Laporte, C.; O'Connor, R. Software Process Improvement in Industry in a Graduate Software Engineering Curriculum Software Quality Professional, Jun 2016, Vol.18(3), pp.4-17, 2016.

ID 23	Colomo-Palacios, R.; Biró, M.; Messnarz, R., Special issue on software and service improvement in the scope of SMEs Software Quality Journal, Sep 2016, Vol.24(3), pp.485-487, 2016
ID 24	Gonzalez-Perez, C.; Henderson-Sellers, B.; McBride, T.; Low, G.C. An Ontology for ISO software engineering standards: Proof of concept and application Computer Standards & Interfaces, November 2016, Vol.48, pp.112-123, 2016.
ID 25	Eito-Brun, R.; Amescua, A. Dealing with software process requirements complexity: an information access proposal based on semantic technologies Requirements Engineering Journal, Nov 2017, Vol. 22(4), pp.527-542, 2017.
ID 26	Gräßler, I.; Hentze, J. Application Potentials of Systems Engineering for Small and Middle-sized Enterprises 11th CIRP Conference on Intelligent Computation in Manufacturing Engineering, 19-21 July 2017, Gulf of Naples, Italy, 2018, Vol.67, pp.510-515, 2018.
ID 27	Paredes, L.; Benavides, L.; Sánchez, N. Guía de recomendación para la selección de un modelo de calidad para la mejora de procesos de software (SPI) Revista Ciencias Estrategicas, Jan-Jun 2017, Vol. 25(37), pp.25-51, 2017.
ID 28	Laporte, C.; Berrhouma, N.; Doucet, M. Measuring the Cost of Software Quality of a Large Software Project at Bombardier Transportation: A Case Study Software Quality Professional, Jun 2012, Vol.14(3), pp.14-31, 2012.
ID 29	Jimmy R.M.R. Estado del Arte: Métricas de calidad para el desarrollo de aplicaciones web 3C Tecnología, Glosas de innovación aplicadas a la pyme” Edición 26, 2017, Vol. 6(4), pp.1-12, 2017.
ID 30	Ibrahim, L.; Wallmüller, E.; Daschner, W. Using Enterprise SPICE in Very Small Entities Software Quality Professional, Mar 2017, Vol. 19(2), pp.43-49, 2017.
ID 31	Rojas-Montes, M.; Pino-Correa, F.; Martínez, J. Proceso de pruebas para pequeñas organizaciones desarrolladoras de software/Testing process for small software development organizations/Processos de provas para pequenas organizações desenvolvedoras de software Revista Facultad de Ingenieria, 2015, Vol.24(39), pp.55-70, 2015.
ID 32	Houde, R.; Laporte, C.Y.; Blondelle, G. ISO/IEC 29110 Deployment Packages and Case Study for Systems Engineering: The “Not-So-Secret” Ingredients That Power the Standard INCOSE International Symposium, July 2016, Vol. 26(1), pp.1276-1292, 2016.
ID 33	Pino, F.; García, F.; Piattini, M.; Oktaba, H. A research framework for building SPI proposals in small organizations: the COMPETISOFT experience Software Quality Journal, Sep 2016, Vol. 24(3), pp.489-518, 2016.
ID 34	Muñoz, M.; Gasca, G.; Valtierra, C. Caracterizando las Necesidades de las Pymes para Implementar Mejoras de Procesos Software: Una Comparativa entre la Teoría y la

	Realidad/Characterizing SME's needs for implementing a software process improvement: A comparative between the reality and the theory Revista Ibérica de Sistemas e Tecnologias de Informação, Mar 2014, Issue E1, pp.1-15 , 2014.
ID 35	López-Lira Hinojo, Francisco Agile, CMMI®, RUP®, ISO/IEC 12207: is there a method in this madness? ACM SIGSOFT Software Engineering Notes, 29 March 2014, Vol.39(2), pp.1-5, 2014.
ID 36	Morales-Trujillo, M.; Oktaba, H.; Piattini, M. The making of an OMG standard Computer Standards & Interfaces, November 2015, Vol.42, pp.84-94, 2015.
ID 37	Kuhrmann, M.; Diebold, P.; Munch, J. Software process improvement: a systematic mapping study on the state of the art. PeerJ Computer Science, May 23, 2016, Vol.2, p.e62, 2016.
ID 38	Felderer, M.; Ramler, R. Risk orientation in software testing processes of small and medium enterprises: an exploratory and comparative study Software Quality Journal, Sep 2016, Vol. 24(3), pp.519-548, 2016.
ID 39	Berg, V.; Birkeland, J.; Nguyen-Duc, A.; Pappas, I. O. Software startup engineering: A systematic mapping study The Journal of Systems & Software, October 2018, Vol.144, pp.255-274, 2018.
ID 40	Politowski, C.; Fontoura, L.M; Petrillo, F. Learning from the past: A process recommendation system for video game projects using postmortems experiences Information and Software Technology, August 2018, Vol.100, pp.103-118, 2018.
ID 41	Aleem, S.; Capretz, L.; Ahmed, F. Critical Success Factors to Improve the Game Development Process from a Developer's Perspective Journal of Computer Science and Technology, Sep 2016, Vol.31(5), pp.925-950, 2016.
ID 42	Graziotin, D.; Jedlitschka, A. Recent developments in product-focused software process improvement: PROFES 2013 conference report ACM SIGSOFT Software Engineering Notes, 11 November 2013, Vol. 38(6), pp.29-34, 2013.
ID 43	Graziotin, D.; Wang, X.; Abrahamsson, P. How do you feel, developer? An explanatory theory of the impact of effects on programming performance PeerJ Computer Science, August 19, 2015, Vol.1, p.e18, 2015.
ID 44	Diaz, A.; De Jesus, C.; Melendez, K.; Davila, A. The ISO/IEC 29110 Implementation on two Very Small Software Development Companies in Lima. Lessons Learned IEEE Latin America Transactions, May 2016, Vol.14(5), pp.2504-2510, 2016
ID 45	Mittner, J.; Buchalcevova, A. Towards the IT Support of Processes in Small Software 2014 Central European Conference on Information and Intelligent Systems, pp.200-206, 2014
ID 46	Zarour, M.; Abran, A.; Desharnais, J. An investigation into the best practices for the successful design and implementation of lightweight software process assessment

	methods: A systematic literature review The Journal of Systems & Software, March 2015, Vol.101, pp.180-192, 2015.
ID 47	Pytel, P.; Hossian, A.; Britos, P. Feasibility and effort estimation models for medium and small size information mining projects Information Systems, January 2015, Vol.47, pp.1-14, 2015
ID 48	Garzás, J.; Pino, F.J.; Piattini, M.; Fernández, C. A maturity model for the Spanish software industry based on ISO standards Computer Standards & Interfaces, November 2013, Vol. 35(6), pp.616-628, 2013.
ID 49	Kuhrmann, M. How Does Software Process Improvement Address Global Software Engineering? IEEE 11th International Conference on Global Software Engineering (ICGSE), CA, USA, 2016.
ID 50	Ghane, K. Quantitative planning and risk management of Agile Software Development IEEE Technology & Engineering Management Conference (TEMSCON), Santa Clara, USA, 2017.
ID 51	Reyes-Delgado, P.Y The strengths and weaknesses of software architecture design in the RUP, MSF, MBASE and RUP-SOA methodologies: A conceptual review Computer Standards & Interfaces Volume 47, Elsevier, August 2016, Pages 24-41 https://doi.org/10.1016/j.csi.2016.02.005 , 2016.
ID 52	Cruz, E; Trujillo, M. Disciplinando Equipos Pequeños con Prácticas Ágiles Grupo de Investigación KUALI-KAANS, Universidad Nacional Autónoma de México. Published by DIF U100ci@ http://nautilus.uaz.edu.mx/difu100cia Ciudad Universitaria, Distrito Federal, México, 2014.
ID 53	Buchalcevová, A. Software Process Improvement in small companies Department of Information Technologies, Prague University of Economics, W. Churchill sqr. 4, 13067 Prague 3, 2015.
ID 54	Karambiri, A. A review of agile methodology in software development India International Research Journal of Engineering and Technology (IRJET) Volume: 03 Issue: 03 March-2016, 2016
ID 55	Bustamante A.F., Rincón, R.C. WYDIWYN – What You Define, Is What You Need: Defining Agile/Traditional Mixed Methodologies Springer International Publishing AG J. Mejia et al. (eds.), Trends and Applications in Software Engineering, 2018
ID 56	Winkler, D.; O'Connor, R.V; Messnarz,R. Systems, Software and Services Process Improvement: 19th European Conference Springer, Austria, 2012
ID 57	González, J.D. AgileFM: Modelo de desarrollo ágil formal basado en la ISO/IEC 29110 para las micro, pequeñas y medianas empresas Tesis de Maestría. Escuela de Ingeniería Medellín, Colombia, 2016.

ID 58	Eito-Brun, R. Incorporating Innovation Management Practices to ISO/IEC 29110 24th European Conference Proceedings, 24th European Conference EURO SPI p15-25, 2017, Ostrava, Czech Republic, 2017.
ID 59	O'Hagan, A.; Coleman, G. Software Development Processes for Games: A Systematic Literature Review EuroSPI 2014, CCIS 425, pp. 182–193, 2014. © Springer-Verlag Berlin Heidelberg, 2014.
ID 60	Buchalceva, A., Application of Methodology Evaluation System on Current IS Development Methodologies International Journal of Information Technologies and Systems Approach (IJITSA), 2018.
ID 61	Puonti, P.V. Tailored Project Management Framework from SCRUM and Lean Practices: Case Study of Two Colombian Companies Master of Science in Engineering, Faculty of Engineering and Science Department of Mechanical and Manufacturing Engineering, Aalborg University, Dinamarca, 2017.
ID 62	Fuertes A.Y.; Sepúlveda C.J. Scrum, Kanban and Canvas in the commercial, industrial and educational sector - A literature review Revista Antioqueña de las Ciencias Computacionales, Vol. 6 Issue 1, p46-50. 5p, jun, 2016.
ID 63	Mejia, J.; Muñoz, M.; Rocha, A.; Quiñonez, Y.; Manzano, J.C Trends and Applications in Software Engineering Proceedings of the 6th International Conference on SPI, CIMPS, 2017.
ID 64	Alvear, S.E. Gestión documental por procesos con AURA PORTAL, integrado al proceso de Desarrollo de Software en la Universidad Técnica del Norte Maestría en Ingeniería de software, Universidad técnica del norte, Ecuador, 2018.
ID 65	Papatheocharous, E.; Andreou, A. Empirical evidence and state of practice of software agile teams Journal of Software: Evolution and Process, https://doi.org/10.1002/smr.1664 , 2014.
ID 66	Balasubramanian, L.; Mnkandla, E. An evaluation to determine the extent and level of Agile Software Development Methodology adoption and implementation in the Botswana Software Development Industry International Conference on Advances in Computing and Communication Engineering (ICACCE), Durban, South Africa, 2016.
ID 67	Vasquez, L.L.B. Implementacion de sistema de pre-venta para propuestas de proyectos de software em avantica Technologies Facultad de ingeniería y arquitectura, Lima, Peru, 2014.
ID 68	Rumeser, D.; Emsley, M. A systematic review of project management serious games: Identifying gaps, trends, and directions for future research The Journal of Modern Project Management Vol 6, No 1, 2018.
ID 69	Hurtado, J. A. A meta-process for defining adaptable software process Tesis para optar al grado de Doctor em Ciencias Computacionais, Universidad del Chile, 2012.

ID 70	Keto, H.; Jaakkola, H. Identifying up-to-the-minute topics in software process improvement research Tampere University of Technology, Proceedings of the SQAMIA 2017: 6th Workshop of Software Quality, Analysis, Monitoring, Improvement, 2017.
ID 71	Biró, M.; Messnarz, R.; Como-Palacios, R. Process improvement approaches fertilised by advances in SPI JOURNAL OF SOFTWARE: EVOLUTION AND PROCESS J. Softw. Evol. and Proc. Published online 4 July 2015 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/smr.1725, 2015.
ID 72	Kasurinen, J.; Risto Laine, R.; Smolander, K; Heidrich, J How Applicable Is ISO/IEC 29110 in Game Software Development? PROFES 2013, LNCS 7983, pp. 5–19, 2013. © Springer-Verlag Berlin Heidelberg, 2013.

APÊNDICE B – ESTUDOS X CANAL DE PUBLICAÇÃO

Canal de Publicação	Total
Computer Standards & Interfaces	10
IEEE	7
Software Quality Professional Journal	6
Journal of Software: Evolution and Process	5
Software Quality Journal	4
part of the book © Springer	3
The Journal of Systems & Software	3
Conference on Enterprise Information Systems / International Conference on Project	1
Computer Applications in Engineering Education	1
International Journal of Information Technologies and Systems Approach	1
R&D Management	1
9th International Conference on the Quality of Information and Communications Technology	1
Requirements Engineering Journal	1
11th CIRP Conference on Intelligent Computation in Manufacturing Engineering	1
Revista Ciencias Estrategicas	1
3C Tecnología, Glosas de innovación aplicadas a la pyme	1
Revista Facultad de Ingenieria	1
PeerJ Computer Science	1
Revista Ibérica de Sistemas e Tecnologias de Informação,	1
ACM SIGSOFT Software Engineering Notes	1
Information and Software Technology	1
Journal of Computer Science and Technology	1
ACM SIGSOFT Software Engineering Notes	1

2014 Central European Conference on Information and Intelligent Systems	1
Grupo de Investigación KUALI-KAANS - online library	1
Department of Information Technologies, Prague University of Economics - online library	1
India International Research Journal of Engineering and Technology (IRJET)	1
Trends and Applications in Software Engineering,Advances in Intelligent Systems and Computing	1
Universidad Eafit Tesis de Maestría. Escuela de Ingeniería Medellín	1
24th European Conference EURO SPI	1
EuroSPI 2014	1
International Journal of Information Technologies and Systems Approach (IJITSA)	1
Master of Science in Engineering, Faculty of Engineering and Science - Dinamarca	1
Revista Antioqueña de las Ciencias Computacionales	1
Proceedings of the 6th International Conference on SPI	1
Maestria em Ingenieiria de Sosftware, Universidad Tecnica del Norte - Ecuador	1
International Conference on Advances in Computing and Communication Engineering (ICACCE)	1
Tesis de maestria Facultad de Ingenieria Y Arquitectura Escuela Profesional de Computacion Y Sistemas - Peru	1
The Journal of Modern Project Management	1
Tesis para doctor em Ciencias en Computacion - Facultad de Ciencias Físicas y Matematicas - Universidad de Chile	1
6th Workshop of Software Quality, Analysis, Monitoring, Improvement, and Applications (SQMIA)	1

APÊNDICE C – FORMATAÇÃO DE TEMPLATES DO PROÁGIL-29110

Esse documento estabelece as regras para a concepção dos templates para o PROÁGIL-29110.

- a) Todos os templates devem possuir um cabeçalho padrão, com no mínimo as informações: Nome do projeto e Cliente.
- b) Como orientação de preenchimento dos campos, colocar quando possível, alguns exemplos e com cores destacadas, preferencialmente “Azul”.

APÊNDICE D – TEMPLATE DE SPRINT BACKLOG

Nome do Projeto	<Preencher com o nome do Projeto>
Cliente	<Preencher com o nome do Cliente>
Data de Criação	<colocar a data da criação da Sprint>
Identificação da Sprint	<identificar a Sprint>

- Insira os itens na Sprint Backlog por ordem de prioridade.

ID	Item	Data de inserção	Tempo de construção (em minutos)		Data de Validação	Retrabalho (sim ou não)
			Planejado	Realizado		
1	Cadastrar empresas	18/11/2018	240	<u>250</u>	18/11/2018	Não
2	Agendar atendimentos	18/11/2018	360	350	18/11/2018	Não

APÊNDICE E – TEMPLATE DE PRODUCT BACKOLOG

Nome do Projeto	<Preencher com o nome do Projeto>
Nome do Product Owner	<colocar o nome>
Nome do Scrum Master	<colocar o nome>
Cliente	<Preencher com o nome do Cliente>
Data de Criação	<colocar a data>

ID	Descrição	Data da inserção	Data de seleção para a Sprint
1	Cadastrar empresas	11/11/2018	13/11/2018
2	Agendar atendimentos	11/11/2018	13/11/2018
3	Realizar atendimentos	11/11/2018	
4	Relatar atendimentos realizados	11/11/2018	
5	Relatar atendimentos a realizar	11/11/2018	

APÊNDICE E – TEMPLATE DE SOLICITAÇÃO DE MUDANÇA (SM)

Nome do Projeto	<Preencher com o nome do Projeto>
Nome do Product Owner	<colocar o nome>
Nome do Scrum Master	<colocar o nome>
Cliente	<Preencher com o nome do Cliente>
Data de Criação	<colocar a data>

Sprint:

Número da Solicitação de Mudança: SM _ _ _

Data da Solicitação:

Descrição da Mudança:

Nome do solicitante:

Nome do responsável pela análise da solicitação:

Data da aprovação/recusa da solicitação:

Situação: Aprovado/recusado:

Análise de Impacto:

- **Prazo:**
- **Custo:**
- **Esforço:**

Produtos de Trabalho / Artefatos a serem alterados:

APÊNDICE F – TEMPLATE DE CRONOGRAMA

Nome do Projeto	<Preencher com o nome do Projeto>
Cliente	<Preencher com o nome do Cliente>
Data de Criação	<colocar a data da criação da Sprint>
Identificação da Sprint	<identificar a Sprint>

Semanas

Pacote de trabalho	Semanas			
	1	2	3	4
<Cadastrar empresas>				
<Agendar atendimentos>				
<Realizar atendimentos>				
<Relatar atendimentos realizados>				
<Relatar atendimentos a realizar>				

APÊNDICE G – TEMPLATE DE VERIFICAÇÃO

Nome do Projeto	<Preencher com o nome do Projeto>
Nome do Product Owner	<colocar o nome>
Cliente	<Preencher com o nome do Cliente>
Data de Criação	<colocar a data>

- **Autor do documento:** Nome do autor do documento
- **Participantes:** Nome e contato dos participantes da verificação
- **Lugar:** Local da verificação
- **Duração:** Tempo decorrente da verificação
- **Check-List:** Lista de itens avaliados na verificação
- **Itens Aprovados:** Lista de itens aprovados na verificação
- **Itens Reprovados:** Lista de itens reprovados na verificação
- **Itens Pendentes:** Lista de itens pendentes na verificação
- **Defeitos:** Defeitos identificados durante a verificação

APÊNDICE H – TEMPLATE DE REGISTRO DE REUNIÃO

Nome do Projeto	<Preencher com o nome do Projeto>
Nome dos Participantes	<Preencher com o nome dos participantes>
Cliente	<Preencher com o nome do Cliente>
Data da Reunião	<Preencher com a data da reunião>

ID	Item abordado	Observações
<1>	<Prazo>	<Projeto atrasado 3 dias em relação ao planejado>

Eventuais dúvidas que surgirem:

APÊNDICE I – TEMPLATE DE MATRIZ DE RISCOS

Client e: <Nome do Cliente> 1.0
 Projeto: <Nome do Projeto> Data de Alteração
 Produto: <Nome do Product Owner> 23/11/2018
 Scrum Master: <Nome do Scrum Master>

MATRIZ DE RISCOS

Número	Descrição do Risco	Data de Identificação	Consequência	Probabilidade	Impacto	Severidade	Prioridade	Contingência
RK_001	<O projeto possui um "dead-line" crítico por parte do Cliente, impedindo o adiamento da implantação.>							
RK_002	<O sistema do projeto envolve mais de uma plataforma (mainframe e WEB, mainframe e aplicação client etc).>							
RK_003								

APÊNDICE I – TEMPLATE DE MATRIZ DE RISCOS (ABA 2)

Cliente: <Nome do Cliente>

1.0

Projeto: <Nome do Projeto>

Data de Alteração 23/11/2018

Product Owner <Nome do Product Owner>

Scrum Master <Nome do Scrum Master>

Plano de Ação ao Risco (Somente se o risco ocorrer)

Número	Ação de Resposta	Status da Ação	Responsável	Data da Ocorrência	Data Limite	Observação

APÊNDICE J – TEMPLATE DE LIÇÕES APRENDIDAS

PLANILHA DE LIÇÕES APRENDIDAS						
Nº	Categoria	Descrição da lição aprendida	Tipo	Consequências	Ação Tomada	Data
1			Positivo			
2			Negativo			
3						
4						
5						
6						
7						
8						
9						

APÊNDICE K – TEMPLATE DO PLANO DE PROJETO

	Plano de Projeto	[LOGO_CLIENTE]
--	-------------------------	----------------

Nome do Projeto	<Preencher com o nome do Projeto>
Nome do Product Owner	<colocar o nome>
Cliente	<Preencher com o nome do Cliente>
Data de Criação	<colocar a data>

1. Objetivos do Projeto

<Descrever os objetivos do projeto>

2. Ciclo de vida

O ciclo de vida do projeto é baseado no método Scrum e realizado através de Sprints.

3. Tarefas

<Informar as tarefas e as respectivas durações>

4. Composição da Equipe

<Identificar a composição da equipe relacionando os perfis, papéis e os nomes de cada recurso definido para o projeto.>

Nome	Papel
<Nome do colaborador>	Product Owner
<Nome do colaborador>	Scrum Master
<Nome do colaborador>	Desenvolvedor
<Nome do colaborador>	Desenvolvedor
<Nome do colaborador>	Desenvolvedor
<Nome do colaborador>	Desenvolvedor

5. Recursos de Infraestrutura

Identificar os recursos de infraestrutura necessários para o projeto (hardware especial, software específico, recursos de rede, links especiais, memória, entre outros).

#	Recurso de Infraestrutura
1.	<descrever o recurso de infra>
2.	<descrever o recurso de infra>
3.	<descrever o recurso de infra>

6. Gerenciamento da Qualidade do Projeto

Esse item aborda a estratégia utilizada para a Garantia da Qualidade do Projeto. Fazem parte da estratégia a descrição dos artefatos a serem verificados e validados e as revisões que são feitas envolvendo equipe de trabalho e cliente

6.1 Itens a serem revisados

<Descrever os itens a serem verificados / validados >

6.2 Reuniões com o time

Serão realizadas as reuniões de Daily Meeting, Sprint Review e Sprint Retrospective.

6.3 Reuniões com o cliente

<Informar as reuniões que terão a participação do cliente> .

7. Gerenciamento de Configuração

7.1 Sistema de biblioteca

O sistema de biblioteca de controle de versão usado neste projeto é o:

<Especificar: (descrever a ferramenta (para gestão de configuração de códigos fontes) que será utilizada para o projeto)>

7.2 Baselines do Projeto:

<Descrever como serão geradas as baselines do projeto>

APÊNDICE L – TEMPLATE DO PLANO DE TESTES

	Plano de Testes	[LOGO_CLIENTE]
--	------------------------	----------------

Nome do Projeto	<Preencher com o nome do Projeto>
Nome do Product Owner	<colocar o nome>
Cliente	<Preencher com o nome do Cliente>
Data de Criação	<colocar a data>

1. Introdução

1.1 Escopo

<Descrição do escopo do documento, a qual projeto está associado>

1.2 Referências

Título	Versão	Data	Onde pode ser obtido

<Incluir a identificação de todos os documentos referenciados no presente documento. Para cada documento indicar o título, versão (caso tenha), data e local o documento pode ser encontrado para consulta.>

1.3 Estratégia de Teste

<Identificar nesta seção todos os tipos de teste a serem feitos no projeto. Para cada tipo de teste é necessário identificar quando serão feitos, quais os recursos necessários para sua realização, quem ficará responsável, quais os requisitos a serem submetidos a esse tipo de teste e os casos de teste.

Existem momentos diferentes e níveis diferentes de testes a serem feitos. Em cada um desses momentos, podem ser executados tipos diferentes de testes como testes funcionais, testes de desempenho, testes de stress, etc. O objetivo desse plano é identificar, de acordo com a criticidade e tamanho do projeto, o quanto se deve investir em testes, ou seja, quais testes

serão feitos em cada momento.

São tipos de testes que podem ser feitos:

testes funcionais

testes de desempenho

testes de stress

testes de carga

testes de tolerância a falhas

testes de controle de acesso e segurança

testes de instalação e configuração

etc>

1.4 Teste <tipo de teste>

<Identificar o tipo do teste a ser feito.>

Recursos necessários

<Identificar os recursos necessários para o teste, sejam humanos, de software ou de hardware. Se forem testes que envolvam um ambiente controlado pela equipe de Infra-estrutura (ambiente de Homologação ou Produção), é importante repassar as informações do plano para essa equipe>

Requisitos a serem testados

<Identificar quais requisitos do sistema serão submetidos a esse tipo de teste. Por exemplo, pode ser que exista um requisito de desempenho relacionado a um ou dois casos de uso somente e sendo assim não teria sentido submeter todo o sistema ao teste>

1.5 Resultados dos Testes

<Para cada tipo de teste a ser feito, em cada momento especificado, registrar os resultados dos testes com as análises feitas e ações tomadas>

APÊNDICE M – TEMPLATE DO TERMO DE ACEITE

Termo de Aceite da Entrega	[LOGO_CLIENTE]
Nome do Projeto:	

1. Objetivos deste documento

Este documento formaliza o aceite da entrega considerando-a em conformidade com os requisitos e os critérios de aceitação definidos.

2. Entrega

<Descrever a entrega com seus requisitos e critérios de aceitação ou referencie o documento que será anexado.>

3. Questões em Aberto

<Usar caso haja alguma questão pendente em relação à entrega. Retirar a seção caso não houver nenhum ponto em aberto.>

Questão em aberto	Responsável	Previsão

Aceite da Entrega		
Os participantes abaixo atestam o cumprimento dos requisitos e dos critérios de aceitação da entrega.		
Participante	Assinatura	Data
Patrocinador do Projeto		
Cliente		

APÊNDICE N – PROCESSO PROÁGIL-29110

Esse documento tem o objetivo de fornecer a base para a melhoria contínua da qualidade do desenvolvimento de software e o controle de projetos com base na norma ISO/IEC 29110, contemplando o método ágil Scrum e princípios da filosofia Lean adaptados a software.

As decisões devem representar o consenso entre os envolvidos. Os resultados serão alcançados a partir da consciência para a qualidade e do envolvimento de cada equipe de projeto. A qualidade e o êxito dos serviços são responsabilidade de todos.

O envolvimento dos colaboradores é estimulado pela participação ativa. Todos os colaboradores se aperfeiçoam continuamente utilizando-se a criatividade e a aprendizagem.

O processo de melhoria contínua baseia-se na utilização de métodos, bem como na efetiva troca de conhecimentos, e é baseado nas melhores práticas.

Novos colaboradores contratados deverão ser treinados no processo PROÁGIL-29110, e estarão cientes das Políticas Organizacionais e atividades relacionadas com a sua função.

Políticas Organizacionais

As políticas aqui descritas possuem o apoio irrestrito da alta direção. Cada uma endereça uma área de processo, e deverá nortear todos os projetos desenvolvidos na organização.

Gerência de Projeto (PM)

Política: A Gerência de Projeto se propõe a estabelecer e manter os planos que definem as atividades para o projeto, possíveis solicitações de mudanças de requisitos, além de acompanhar o progresso do projeto para que ações corretivas possam ser tomadas quando a execução do projeto desviar do planejado. Da mesma forma garante a qualidade de software e estabelece um sistema de configuração.

Implementação de Software (SI)

Política: Os projetos devem analisar os requisitos de software, garantindo sua rastreabilidade, realizar a arquitetura, construção, integração, testes e liberação do produto. Da mesma forma deve garantir a configuração do software desenvolvido.

Descrição do Processo – PROÁGIL-29110

O processo PROÁGIL-29110 define as fases que conectam o início do projeto ao seu final. A transição de uma fase para a outra dentro do ciclo de vida do projeto envolve e normalmente é definida por transferência técnica ou entrega. O ciclo de vida do projeto define:

- Quais as atividades devem ser realizadas em cada fase;
- Quando as entregas devem ser geradas em cada fase;
- Como cada entrega será revisada, verificada e validada;
- Quem está envolvido em cada fase e cada tarefa;
- Como controlar os artefatos e quais deles aprovar em cada fase;
- Quais são as técnicas, eventos e ferramentas utilizadas em cada tarefa do processo.

Processo – PROÁGIL-29110

O processo PROÁGIL-29110 é dividido em duas fases:

1. GP – Gerência de Projetos
2. SI – Implementação de Software

1 - GP - Gerência de Projetos

1.1 Planejamento do Projeto – PM.1

Objetivo: Elaborar o Plano do Projeto	Descrição: Esse sub-processo define as atividades que mostram como é realizada a elaboração do Product Backlog, Sprint Backlog e Plano de Projeto, a partir da Declaração de Trabalho, bem como a estimativa e o dimensionamento de tarefas e recursos.
Responsável:	Time Scrum
Envolvidos:	Cliente
Templates associados	Product Backlog, Sprint Backlog, Plano de Projeto, Matriz de Riscos
Ferramentas associadas	Word, Excel
Tarefas:	<ul style="list-style-type: none">• A partir da Declaração de Trabalho revisada, o Product Owner define o <i>Product Backlog</i>, e o Time realiza a definição de pronto e preenche os Templates de Product Backlog, Sprint Backlog e Plano do Projeto. <p>Obs: o evento (opcional) chamado Refinement session ou Product Backlog Refinement ou Backlog Grooming poderá ser realizada para o refinamento das histórias do Product Backlog.</p> <ul style="list-style-type: none">• Realizar a estimativa de esforço, custo e prazo para cada item do backlog utilizando uma técnica associada (APF, Planning Poker, ou método próprio).• Planejar o Sprint (Sprint Planning): duração, equipe, identificar riscos (na Matriz de Riscos) tarefas e respectivas duração e dependências em um cronograma.• Criar o repositório do projeto e estabelecer uma estratégia de controle de versão de artefatos e baseline do projeto.• Estabelecer uma estratégia de Garantia da Qualidade do projeto.

Entradas: <ul style="list-style-type: none"> • Declaração de Trabalho • Template do Product Backlog • Template do Sprint Backlog • Template do Plano de Projeto • Template da Matriz de Riscos • Cronograma 	Saídas: <ul style="list-style-type: none"> • Product Backlog preenchido • Sprint Backlog preenchido • Plano de Projeto preenchido • Matriz de Riscos preenchida • Cronograma elaborado • Repositório do projeto
--	--

1.2 Execução do Plano de Projeto – PM.2

Objetivo: Realizar a execução do Plano de Projeto	Descrição: Esse sub-processo define as atividades que mostram como é realizada a atividade de acompanhamento periódico do projeto e como são realizadas as avaliações de impacto em mudanças de requisitos
Responsável:	Scrum Master, Time de Desenvolvimento
Envolvidos:	Product Owner
Templates associados	Solicitação de Mudanças, Registro de reunião
Ferramentas associadas	Word, Excel, gráfico de Burndown
Tarefas:	<ul style="list-style-type: none"> • Realizar a reunião diária (Daily Meeting) – tempo máximo de 15 minutos e anotar dificuldades e obstáculos; • Utilizar o gráfico de Burndown para estabelecer o registro de status do projeto; • Realizar as revisões e aceitações entre a Equipe de Trabalho e/ou Cliente quando pertinente. • Estabelecer uma rotina de Backup do repositório do projeto
Entradas: <ul style="list-style-type: none"> • Template de Registro de reunião • Gráfico Burndown • Plano de Projeto 	Saídas: <ul style="list-style-type: none"> • Registro de Reunião preenchido • Gráfico Burndown atualizado • Plano de Projeto atualizado • Backup do repositório

1.3 Avaliação e Controle do Projeto – PM.3

Objetivo: realizar a avaliação e controle do projeto	Descrição: Esse sub-processo define as atividades que mostram como são realizadas a avaliação e controle do projeto e o tratamento de ações corretivas
Responsável:	Scrum Master, Time de Desenvolvimento
Envolvidos:	Product Owner,
Templates associados	Registro de reunião
Ferramentas associadas	Word e Excel

Tarefas:	<p>Ao final de cada semana realizar uma reunião de acompanhamento para:</p> <ul style="list-style-type: none"> • Avaliar a realização do Plano de Projeto e progresso frente aos objetivos; • Revisar os riscos do projeto, identificando novos riscos, se necessário; • Levantar possíveis desvios significativos em relação a custo, cronograma e desempenho técnico ou problemas relativos ao projeto e registrar as ações corretivas; • Analisar eventuais solicitações de mudanças avaliando impacto, utilizando o Template Solicitação de Mudança – SM, atualizar o Plano de Projeto se necessário e realizar as devidas correções; • Garantir que os artefatos a serem alterados em função das solicitações de mudanças foram atualizados. • Atualizar a Matriz de Rastreabilidade 	
Entradas:	<ul style="list-style-type: none"> • Plano de Projeto • Cronograma do Projeto • Template de Registro de reunião • Template da Planilha de Controle de Pendências • Template de Solicitação de Mudanças • Matriz de Rastreabilidade 	Saída:
		<ul style="list-style-type: none"> • Plano de Projeto atualizado • Planilha de Controle de Pendências atualizada • Registro de reunião preenchido • Registro de correções • Template de Solicitação de mudanças preenchido • Matriz de Rastreabilidade atualizada

1.4 Fechamento de Projeto – PM.4

Objetivo: Realizar o fechamento do projeto	Descrição: Esse sub-processo define as atividades que permitem fazer o fechamento do projeto	
Responsável:	Scrum Master, Time de Desenvolvimento	
Envolvidos:	Product Owner	
Templates associados	Controle de Pendências, Planilha de Lições Aprendidas, Termo de aceite	
Ferramentas associadas	Word, Excel	
Tarefas:	<ul style="list-style-type: none"> • Realizar uma reunião de Sprint Retrospective; • Armazenar as lições aprendidas; • Entregar o produto de software e documentos aplicáveis de acordo com as Instruções de entrega e registrar o aceite 	
Entradas:	<ul style="list-style-type: none"> • Planilha de Lições Aprendidas • Termo de aceite 	Saída:
		<ul style="list-style-type: none"> • Planilha de Lições Aprendidas atualizada • Registro de aceite

2- SI - Implementação do Software

2.1 Início da Implementação do Software – SI.1

Objetivo: Iniciar a implementação do software	Descrição: Esse sub-processo define as atividades que mostram a inicialização da implementação do software a ser construído
Responsável:	Scrum Master, Time de Desenvolvimento
Envolvidos:	Product Owner
Templates associados	Registro de reunião
Ferramentas associadas	Word e Excel
Tarefas:	<ul style="list-style-type: none">• Realizar a reunião de Kick-off com todos os desenvolvedores com o objetivo de revisar o plano de projeto, atividades, marcos do projeto e obter o compromisso da equipe;• Criar repositório do projeto;
Entrada: <ul style="list-style-type: none">• Plano de Projeto preenchido• Cronograma• Matriz de Riscos• Estrutura Padrão de Diretórios• Template da Registro de reunião	Saída: <ul style="list-style-type: none">• Registro de reunião preenchido• Ambiente do projeto definido na ferramenta

2.2 Análise de Requisitos do Software – SI.2

Objetivo: Analisar os requisitos do software	Descrição: Esse sub-processo define as atividades que mostra como são analisados os requisitos do software
Responsável:	Time de Desenvolvimento
Envolvidos:	Product Owner, Scrum Master
Templates associados	Plano de Testes
Ferramentas associadas	Word, Excel
Tarefas:	<ul style="list-style-type: none">• Verificar e validar os requisitos e elaborar a planilha de Especificação de Requisitos;• Detalhar os requisitos. Descrever as informações de cada requisito com a identificação de atributos (tela, campos e outros) de forma resumida;• Elaborar documentação do usuário

Entrada: <ul style="list-style-type: none"> Planilha de “Especificação de Requisitos (E.R)” Template de Plano de Testes 	Saída: <ul style="list-style-type: none"> Planilha de “Especificação de Requisitos (E.R)” elaborada Resultados de Verificação Resultados de Validação Documentação do usuário
---	---

2.3 Projeto de Arquitetura e Detalhamento do Software – SI.3

Objetivo: Definir a arquitetura e detalhar o software	Descrição: Esse sub-processo define as atividades que mostram como é realizada a atividade da arquitetura e detalhamento do software
Responsável:	Time de Desenvolvimento
Envolvidos:	Scrum Master
Templates associados	Resultado de Verificação
Ferramentas associadas	Word e Excel
Tarefas:	<ul style="list-style-type: none"> Elaborar o Plano de Testes Elaborar os Documentos para implementação dos requisitos; Controlar a versão dos artefatos produzidos; Realizar a verificação e as devidas correções dos artefatos.
Entradas: <ul style="list-style-type: none"> Planilha de Especificação de Requisitos Template de Resultado de Verificação 	Saídas: <ul style="list-style-type: none"> Plano de Testes elaborado Documentos para implementação dos requisitos elaborados (design) Matriz de Rastreabilidade atualizada Resultado de Verificação preenchida

2.4 Construção do Software – SI.4

Objetivo: Construir o software	Descrição: Esse sub-processo define as atividades que mostram como é realizada a construção do software
Responsável:	Time de Desenvolvimento
Envolvidos:	Product Owner, Scrum Master
Templates associados	Planilha de Especificação de Requisitos, Matriz de Rastreabilidade
Ferramentas associadas	Kanban

Tarefas:	<ul style="list-style-type: none"> • Revisão da documentação disponível esclarecendo dúvidas com a equipe de desenvolvimento ou Scrum Master no Daily Meeting e estabelecer o Kanban; • Implementar o código fonte (componentes do software); • Realizar os testes unitários e, caso erros sejam identificados, fazer a devida correção; • Atualizar a Matriz de Rastreabilidade com o código fonte produzido; • Armazenar o código fonte.
Entradas: <ul style="list-style-type: none"> • Planilha de Especificação de Requisitos • Documentos para implementação dos requisitos elaborados (design) • Matriz de Rastreabilidade 	Saída: <ul style="list-style-type: none"> • Kanban disponibilizado • Código fonte (componentes do software) • Resultados de teste • Matriz de Rastreabilidade atualizada

2.5 Integração e Testes do Software – SI.5

Objetivo: realizar a integração e testes do software	Descrição: Esse sub-processo define as atividades que mostram como é realizada a integração e testes do software
Responsável:	Time de Desenvolvimento
Envolvidos:	Product Owner, Scrum Master
Templates associados	Plano de Testes
Ferramentas associadas	Word
Tarefas:	<ul style="list-style-type: none"> • Integrar os componentes de software; • Executar os testes, utilizando o Plano de Testes, identificando inconsistências a partir da análise de documentação do projeto, registrar os resultados e estabelecer o software; • Atualizar a Matriz de Rastreabilidade, se necessário; • Elaborar a documentação da operação; • Verificar a consistência da baseline.
Entradas: <ul style="list-style-type: none"> • Planilha de Especificação de Requisitos • Plano de Testes • Documentação do usuário • Matriz de Rastreabilidade • Baseline 	Saída: <ul style="list-style-type: none"> • Relatório de Teste • Software • Matriz de Rastreabilidade atualizada • Documento do usuário • Guia de operação • Baseline verificada

2.6 Entrega do Produto – SI.6

Objetivo: Entregar o Produto	Descrição: Esse sub-processo define as atividades que mostram como é realizada a entrega do produto desenvolvido	
Responsável:	Time Scrum	
Envolvidos:	Cliente, convidados pelo Product Owner	
Templates associados	Termo de Aceitação	
Ferramentas associadas	Word	
Tarefas:	<ul style="list-style-type: none"> • Elaborar o documento de Instrução de entrega; • Verificar Documentação de Manutenção, caso necessário; • Atualizar a configuração do software; • Realizar a reunião de Sprint Review. 	
Entradas: <ul style="list-style-type: none"> • Documentação de Manutenção • Instruções de entrega • Ata de Reunião de <i>Sprint Review</i> 	Saída:	<ul style="list-style-type: none"> • Documento de Instrução de entrega • Documento de Manutenção • Ata de Reunião de Sprint Review

Para assumir os papéis e responsabilidades no processo define-se o TIME Scrum que será composto pelo Product Owner, Scrum Master e o Time de Desenvolvimento. Fica a critério da organização que vier a implementar o PROÁGIL-29110 acrescentar outros papéis existentes, sendo que nesse caso a descrição deve ser adicionada no Quadro 16.

Quadro 16: Papel x Responsabilidade

Papel	Responsabilidade	Conhecimentos desejáveis
Product Owner	<ul style="list-style-type: none"> • Interagir com os “Stakeholders” do projeto; • Acompanhar o andamento dos projetos e tomada de ações corretivas; • Avaliar os recursos do projeto; 	<ul style="list-style-type: none"> • ISO/IEC 29110 • Certificação Scrum Master • Gestão de Projeto (PMBOK);

	<ul style="list-style-type: none"> • Elaborar Plano de Projeto; • Estimar prazos e custos do projeto; • Avaliar qual será o impacto causado no projeto devido às solicitações de mudança. • Gerenciar o backlog do produto • Expressar claramente os itens do backlog do produto; • Ordenar os itens do Backlog do Produto para alcançar melhor as metas e missões; • Garantir o valor do trabalho realizado pelo Time de Desenvolvimento; • Garantir que o Backlog do Produto seja visível, transparente, claro para todos, e mostrar o que o Time Scrum vai trabalhar a seguir; • Garantir que o Time de Desenvolvimento entenda os itens do Backlog do Produto no nível necessário 	<ul style="list-style-type: none"> • Gestão de Pessoas; • Conhecimento em Métricas e Estimativas de Esforço de Projeto; • Conhecimento em ferramenta de Gestão de Configuração • Atitudes (Liderança, responsabilidade, colaborador, comprometimento)
Scrum Master	<ul style="list-style-type: none"> • Facilitar o trabalho do projeto; • Introduzir e fomentar as práticas ágeis; • Garantir a qualidade do processo e artefatos; • Analisar a velocidade do projeto; • Treinar o Time de Desenvolvimento em autogerenciamento e interdisciplinaridade; • Ensinar a Time Scrum a criar itens de Backlog do Produto de forma clara e concisa; • Remover impedimentos para o progresso do Time de Desenvolvimento; • Facilitar os eventos Scrum conforme exigidos ou necessários; • Planejar implementações 	<ul style="list-style-type: none"> • ISO/IEC 29110 • Certificação Scrum Master • Gestão de Projeto (PMBOK) • Levantamento e Análise de Requisitos; • Conhecimento em Modelagem UML • Conhecer alguma Linguagem de Programação • Conhecer Banco de Dados (SQL, DB2, Oracle, SyBase). • Conhecimento em ferramenta de Gestão de Configuração • Atitudes (Responsabilidade, colaborador, comprometimento)
Time de Desenvolvimento	<ul style="list-style-type: none"> • Analisar os requisitos do projeto; • Gerar solução do projeto; • Gerar modelagem de dados com descritivo e acessos à base de dados; 	<ul style="list-style-type: none"> • ISO/IEC 29110 • Scrum • Levantamento e Análise de Requisitos

	<ul style="list-style-type: none"> • Gerar especificações, protótipos, fluxos e outros artefatos necessários para desenvolvimento dos requisitos; • Implementar os códigos; • Realizar testes unitários dos programas • Realizar testes integrados do projeto; 	<ul style="list-style-type: none"> • Conhecimento em Modelagem UML • Conhecer alguma Linguagem de Programação • Conhecer Banco de Dados (SQL, DB2, Oracle, SyBase). • Conhecimento em ferramenta de Gestão de Configuração • Atitudes (Responsabilidade, colaborador, comprometimento)
--	--	---

Obs: Outros papéis que a empresa vier a adotar outros deverão ser acrescentados no Quadro 16, incluindo-se as responsabilidades e conhecimentos desejáveis.

No Quadro 17 e Quadro 18 encontra-se o “Guia rápido de implementação”, que descreve cada uma das fases do processo PROÁGIL-29110, contemplando a correspondência entre artefato, técnicas, ferramentas e responsabilidades para cada atividade/tarefa.

O Guia deve ser sempre atualizado de acordo com as adequações realizadas pela organização levando-se em consideração mudanças em relação aos artefatos, técnicas, ferramentas e responsabilidades.

Guia rápido de implementação

Quadro 17: Fase 1 – Gerência de Projetos

ATIVIDADES / TAREFAS	ARTEFATOS	TÉCNICAS / EVENTO	FERRAMENTAS	RESPONSABILIDADES		
				PO	SM	D
1.1 Planejamento do Projeto ✓ A partir da Declaração de Trabalho revisada, definir o Product Backlog, realizar a definição de pronto e preencher o Template do Plano do Projeto. Obs: o evento (opcional) chamado Refinement Session ou Product Backlog Refinement ou Backlog Grooming poderá ser realizada para o refinamento das histórias do Product Backlog.	✓ Declaração de trabalho ✓ Product Backlog ✓ Plano de Projeto	✓ Definição de pronto ✓ Refinement Session (Opcional)	✓ Word, Excel	X		
✓ Realizar a estimativa de esforço, custo e tempo para cada item do backlog utilizando uma técnica associada (APF, Planning Poker, ou método próprio).	✓ Product Backlog	✓ APF ✓ Planning Poker ✓ Método próprio	✓ Word, Excel	X	X	X
✓ Planejar o Sprint (Sprint Planning): duração, equipe, identificar riscos (na Planilha de Riscos) tarefas e respectivas duração e dependências em um cronograma.	✓ Backlog do Sprint ✓ Planilha de Riscos ✓ Cronograma	✓ Sprint Planning	✓ Word, Excel	X	X	X

✓ Criar o repositório do projeto e estabelecer uma estratégia de controle de versão de artefatos e baseline do projeto.	✓ Baseline do projeto ✓ Configuração de Software ✓ Repositório do projeto	✓ Sprint Planning	✓ Ferramenta de Controle de Configuração	X	X	X
✓ Estabelecer uma estratégia de Garantia da Qualidade do projeto.	✓ Plano de projeto	✓ Sprint Planning		X	X	X
1.2 Execução do Plano de Projeto						
✓ Realizar a reunião diária (Daily Meeting – tempo máximo de 15 minutos) e anotar dificuldades e obstáculos;	✓ Ata de reunião	✓ Daily Meeting	✓ Word		X	X
✓ Utilizar o gráfico de Burndown para estabelecer o registro de status do projeto;	✓ Registro de Status	✓ Daily Meeting	✓ Gráfico Burndown		X	X
✓ Realizar as revisões e aceitações entre a Equipe de Trabalho e/ou Cliente quando pertinente	✓ Ata de reunião	✓ Daily Meeting / Sprint Retrospective	✓ Word		X	X

✓ Estabelecer o Backup do repositório do projeto.	✓ Backup do repositório do projeto	✓ Daily Meeting	✓ Ferramenta de Controle de Configuração		X	X
1.3 Avaliação e controle						
✓ Avaliar a realização do Plano de Projeto e progresso frente aos objetivos;	✓ Ata de reunião	✓ Daily Meeting	✓ Gráfico Burndown		X	X
✓ Revisar os riscos do projeto, identificando novos riscos, se necessário;	✓ Planilha de riscos	✓ Daily Meeting	✓ Excel		X	X
✓ Levantar possíveis desvios significativos em relação a custo, cronograma e desempenho técnico ou problemas relativos ao projeto e registrar as ações corretivas;	✓ Registro de correção ✓ Planilha de Controle de pendencias	✓ Daily Meeting	✓ Word, Excel		X	X
✓ Analisar eventuais solicitações de mudanças avaliando impacto, utilizando o Template Solicitação de Mudança – SM, atualizar o Plano de Projeto se necessário e realizar as devidas correções;	✓ Solicitação de mudança	✓ Daily Meeting	✓ Word		X	
✓ Garantir que os artefatos a serem alterados em função das solicitações de mudanças foram atualizados.	✓ Ata de reunião	✓ Daily Meeting	✓ Word, Excel		X	X

1.4 Fechamento do Projeto						
✓ Realizar uma reunião de Sprint Retrospective com os Desenvolvedores;	✓ Ata de reunião	✓ Sprint Retrospective	✓ Word, Excel		X	X
✓ Armazenar as lições aprendidas	✓ Planilha de lições aprendidas	✓ Sprint Retrospective	✓ Word, Excel		X	X
✓ Entregar o produto de software e documentos aplicáveis de acordo com as Instruções de entrega	✓ Instruções de entrega	✓ Sprint Retrospective	✓ Word, Excel	X	X	X
	✓ Registro de aceite					

Quadro 18: Fase 2 – Implementação do Software

ATIVIDADES / TAREFAS	ARTEFATOS	TÉCNICAS / EVENTO	FERRAMENTAS	RESPONSABILIDADES		
				PO	SM	D
2.1 Início da implementação do software ✓ Realizar a reunião de Kick-off com todos os desenvolvedores com o objetivo de revisar o plano de projeto, atividades, marcos do projeto e obter o compromisso da equipe;	✓ Ata de reunião de kick-off		✓ Word, Excel		X	X
✓ Criar repositório do projeto;	✓ Repositório do projeto	✓ Daily Meeting	✓ Ferramenta Controle Configuração			X
2.2 Análise de Requisitos do Software ✓ Verificar e validar os requisitos e elaborar a planilha de Especificação de Requisitos;	✓ Resultados da validação ✓ Especificação de Requisitos	✓ Daily Meeting	✓ Word, Excel			X
✓ Detalhar os requisitos. Descrever as informações de cada requisito com a identificação de atributos (tela, campos e outros) de forma resumida;	✓ Design de software	✓ Daily Meeting	✓ Word, Excel			X
✓ Elaborar a documentação do usuário.	✓ Documentação do usuário	✓ Daily Meeting	✓ Word, Excel			X

2.3 Projeto de Arquitetura e Detalhamento do Software	✓ Plano de Testes	✓ Daily Meeting	✓ Word			X
✓Elaborar o Plano de Testes						
✓ Elaborar os Documentos para implementação dos Requisitos;	✓ Componentes de software	✓ Daily Meeting	✓ Word			X
✓ Controlar a versão dos artefatos produzidos;	✓ Configuração do software	✓ Daily Meeting	✓ Ferramenta de Controle de Configuração			X
✓ Realizar a verificação e correção dos artefatos.	✓ Resultados de verificação	✓ Daily Meeting	✓ Word			X
2.4 Construção do software						
✓ Revisão da documentação disponível esclarecendo duvidas com a equipe de desenvolvimento ou Scrum Master no Daily Meeting e estabelecer o Kanban;	✓ Ata de reunião	✓ Daily Meeting	✓ Kanban		X	X
✓ Implementar o código fonte (componentes do software);	✓ Código fonte (componentes de software)	✓ Daily Meeting	✓ Ferramenta de Controle de Configuração			X
✓ Realizar os testes unitários e, caso erros sejam identificados, fazer a devida correção;	✓ Relatório de Teste	✓ Daily Meeting	✓ Word			X

✓ Atualizar a Matriz de Rastreabilidade com o código fonte produzido;	✓ Matriz de rastreabilidade	✓ Daily Meeting	✓ Excel			X
✓ Armazenar o código fonte.	✓ Código fonte (componentes de software)	✓ Daily Meeting	✓ Ferramenta Controle Configuração	de		X
2.5 Integração e testes do software						
✓ Integrar os componentes do software;	✓ Software	✓ Daily Meeting	✓ Word			X
✓ Executar os testes, utilizando o Plano de Testes, identificando inconsistências a partir da análise de documentação do projeto e registrar os resultados	✓ Relatório de Teste	✓ Daily Meeting	✓ Word			X
✓ Atualizar a Matriz de Rastreabilidade, se necessário;	✓ Matriz de rastreabilidade	✓ Daily Meeting	✓ Excel			X
✓ Elaborar documentação da operação;	✓ Guia de Operação do produto	✓ Daily Meeting	✓ Word			X
✓ Verificar a consistência da Baseline.	✓ Configuração do software	✓ Daily Meeting	✓ Ferramenta Controle Configuração	de		X

2.6 Entrega do Produto						
✓ Elaborar o documento de Instrução de entrega;	✓ Instrução de entrega	✓ Daily Meeting	✓ Word			X
✓ Verificar documentação de Manutenção, caso necessário	✓ Documentação de Manutenção	✓ Daily Meeting	✓ Word			X
✓ Atualizar a configuração do software	✓ Configuração do software – Backlog da versão de entrega	✓ Daily Meeting	✓ Controle de Versão			X
✓ Realizar a reunião de Sprint Review	✓ Ata de Reunião	✓ Daily Meeting	✓ Word	X	X	X