FUNDAÇÃO MINEIRA DE EDUCAÇÃO E CULTURA - FUMEC

ALAN REIS SCATOLINO

INFLUÊNCIA DA APLICAÇÃO DE MÉTODOS ÁGEIS E DA GESTÃO DO CONHECIMENTO NA QUALIDADE DE *SOFTWARE*: uma análise multivariada com profissionais de tecnologia da informação

Belo Horizonte

2019

ALAN REIS SCATOLINO

INFLUÊNCIA DA APLICAÇÃO DE MÉTODOS ÁGEIS E DA GESTÃO DO CONHECIMENTO NA QUALIDADE DE *SOFTWARE*: uma análise multivariada com profissionais de tecnologia da informação

Dissertação apresentada ao Curso de Mestrado Profissional em Sistemas de Informação e Gestão do Conhecimento da Universidade FUMEC, como parte dos requisitos para a obtenção do título de Mestre em Sistemas de Informação e Gestão do Conhecimento.

Área de Concentração: Gestão de Sistemas de Informação e Gestão do Conhecimento.

Linha de Pesquisa: Gestão da Informação e do Conhecimento.

Orientador: Prof. Dr. Ronaldo Camilo

Belo Horizonte

2019

Dados Internacionais de Catalogação na Publicação (CIP)

S277i Scatolino, Alan Reis, 1968 -

Influência da aplicação de métodos ágeis e da gestão do conhecimento na qualidade de software: uma análise multivariada com profissionais de tecnologia da informação / Alan Reis Scatolino. – Belo Horizonte, 2019.

91 f : il. ; 29,7 cm

Orientador: Ronaldo Darwich Camilo

Dissertação (Mestrado em Sistemas de Informação e Gestão do Conhecimento), Universidade FUMEC, Faculdade de Ciências Empresariais, Belo Horizonte, 2019.

1. Gestão do Conhecimento - Brasil. 2. Software - Controle de qualidade - Brasil. 3. Tecnologia da informação - Brasil. I. Título. II. Camilo, Ronaldo Darwich. III. Universidade FUMEC, Faculdade de Ciências Empresariais.

CDU: 65.01:001

Ficha catalográfica elaborada pela Biblioteca Universitária-FUMEC

UNIVERSIDADE
FUMEC

Dissertação intitulada "**A influência da aplicação de métodos ágeis e da gestão do conhecimento na qualidade de software: uma análise multivariada com profissionais de TI**" de autoria de Alan Reis Scatolino, aprovada pela banca examinadora constituída pelos seguintes professores:

Prof. Dr. Ronaldo Darwich Camilo – Universidade FUMEC
(Orientador)

Prof. Dr. Armando Sérgio de Aguiar Filho – Universidade FUMEC
(Examinador Interno)

Prof. Dr. Rodrigo Baroni de Carvalho – PUC MINAS
(Examinador Externo)

Eric de Paula Ferreira, Me. – Axxiom Soluções Tecnológicas
(Consultor *Ad Hoc*)

Prof. Dr. Fernando Silva Parreiras
Coordenador do Programa de Pós-Graduação em Sistemas de Informação e Gestão do Conhecimento da Universidade FUMEC

Belo Horizonte, 15 de fevereiro de 2019.

À Solange,
pela paciência e incentivo.

Aos meus filhos, Lucas e Tiago,
pelo apoio.

Aos meus pais, Guido e Nilsa,
pelo apoio e incentivo incondicional.

Agradecimentos

A Deus, Que me deu forças para concluir este projeto.

Ao meu orientador, Prof. Dr. Ronaldo Camilo, por seu apoio, disponibilidade e suporte incansável ao longo do projeto.

Ao corpo docente do Programa de Pós-Graduação em Sistemas de Informação e Gestão do Conhecimento (PPGSIGC) da FUMEC, na pessoa do Prof. Dr. Fernando Silva Parreiras.

Aos funcionários da Secretaria, que foram sempre prestativos e atenciosos.

A todos que, direta ou indiretamente, contribuíram para a realização deste trabalho.

“Não importa se a estação do ano muda...
Se o século vira, se o milénio é outro.
Se a idade aumenta...
Conserva a vontade de viver,
Não se chega a parte alguma sem ela”.

Fernando Pessoa.

Resumo

INFLUÊNCIA DA APLICAÇÃO DE MÉTODOS ÁGEIS E DA GESTÃO DO CONHECIMENTO NA QUALIDADE DE *SOFTWARE*: UMA ANÁLISE MULTIVARIADA COM PROFISSIONAIS DE TECNOLOGIA DA INFORMAÇÃO

Atualmente, as empresas de tecnologia da informação têm buscado incessantemente a produtividade e a melhoria do posicionamento competitivo, mas elas não podem abrir mão da qualidade do *software* gerado ou da preservação dos conhecimentos obtidos nesse processo. As metodologias ágeis encaixam-se bem nessas necessidades, pois se propõem a permitir entregas de *software*, em prazos menores e com mais maleabilidade. Nesse ambiente, espera-se que conceitos aplicados de gestão do conhecimento tenham importante papel, porque os maiores ativos das empresas de desenvolvimento de *software* atuais são seus funcionários e o conhecimento que trazem com eles. Nesse contexto, para garantir entregas de qualidade, as empresas de desenvolvimento de *software* têm percebido a importância de processos de desenvolvimento de *software* que utilizem as melhores práticas do mercado. O presente trabalho apresentou, a partir de análise multivariada operacionalizada por meio de questionário com profissionais de tecnologia da informação (TI), a influência e a contribuição dos métodos ágeis e da gestão do conhecimento na qualidade de *software* desenvolvido pelas empresas. A pesquisa mostrou que tanto os métodos ágeis quanto a gestão do conhecimento influenciam a qualidade de *software*, sendo que a influência desta última foi maior.

Palavras-chave: Gestão do Conhecimento. Métodos Ágeis. Qualidade de *Software*.

Abstract

INFLUENCE OF THE APPLICATION OF AGE METHODS AND THE MANAGEMENT OF KNOWLEDGE IN SOFTWARE QUALITY: A MULTIVARIATE ANALYSIS WITH INFORMATION TECHNOLOGY PROFESSIONALS

Nowadays, information technology companies have been constantly looking for productivity and improving their competitive positioning, but they cannot give up the quality of the software generated or the preservation of the knowledge obtained in this process. Agile methodologies fit well with these needs, as they propose to allow software deliveries, in shorter and more flexible terms. In this environment, applied knowledge management concepts are expected to play an important role, because the biggest assets of today's software development companies are their employees and the knowledge they bring with them. In this context, to ensure quality deliveries, software development companies have realized the importance of software development processes that use the best practices on the market. The present work presented, based on a multivariate analysis using a questionnaire with information technology (IT) professionals, the influence and contribution of agile methods and knowledge management in software quality developed by companies. The research showed that both agile methods and knowledge management influence the quality of software, and the influence of the latter was greater.

Keywords: Knowledge Management. Agile Methods. Software Quality.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

AC	Alfa de Cronbach
AVE	Variância média extraída
CC	Confiabilidade composta
CMM	*Capability Maturity Model*
CMMI	*Capability Maturity Model Integration*
DP	Desvio-padrão
EIA/IS	*Eletronic Industries Alliance Ínterim Standard*
ETM	Equipe Técnica do Modelo
FDD	*Feature Driven Development*
FFC	Fórum de Credenciamento e Controle
FUMEC	Fundação Mineira de Educação e Cultura
IA	Instituições avaliadoras
IC	Intervalo de confiança
IEEE	Instituto de Engenheiros Eletricistas e Eletrônicos
II	Instituições implementadoras
IPD-CMM	*Integrated Product Development Capability Maturity Model*
ISO	*International Standardization for Organization*
MA-MPS	Método de Avaliação Melhoria de Processo do *Software*
MANOVA	Análise multivariada da variância
MPS-BR	Melhoria de Processo do *Software* Brasileiro
MR-MPS-SV	Modelo de Referência Melhoria de Processo do *Software* para Serviços
MR-MP-SW	Modelo de Referência Melhoria de Processo do *Software* para *Software2*
PLS	*Partial Least Square*
PPGSIGC	Programa de Pós-Graduação em Sistemas de Informação e Gestão do Conhecimento
SEI	Sistema Eletrônico de Informações
SEM	Modelagem de equações estruturais
SOFTEX	Associação para Promoção da Excelência do *Software* Brasileiro
SW-CMM	*Capability Maturity Model for Software*
TI	Tecnologia da informação
UEP	Unidade de Execução do Programa
XP	*Extreme Programming*

Sumário[1]

[1] Este trabalho foi revisado de acordo com as novas regras ortográficas aprovadas pelo Acordo Ortográfico assinado entre os países que integram a Comunidade de Países de Língua Portuguesa (CPLP), em vigor no Brasil desde 2009. E foi formatado de acordo com as Instruções para Formatação de Trabalhos Acadêmicos – Norma APA, 2017

1 Introdução

O mundo do desenvolvimento de *software* vive em constante mudança. Novas tecnologias, novas metodologias, novos ambientes de desenvolvimento fazem com que os desafios nessa área sejam significativos. Segundo o CHAOS *Report* (Standish Group, 2015), o percentual de falha de muitos projetos de desenvolvimento de *software* é considerável e os *softwares* gerados, muitas vezes, não satisfazem aos usuários. Com o intuito de minimizar essas falhas, muitas metodologias e *frameworks* de referência vêm sendo desenvolvidos nas empresas, entidades normativas e associações ao longo dos anos.

A gestão do conhecimento, por sua vez, vem sendo avaliada em diversas áreas das organizações como uma forma eficaz de desenvolver soluções de maior valor, contribuindo, também, para o desenvolvimento de *software*. Para acompanhar o ritmo acelerado de mudanças ocorridas em âmbito digital, os métodos ágeis - metodologias mais "leves" que as tradicionais, baseadas em equipes multifuncionais e auto-organizadas (Gomes, Willi & Rehem, 2014) - propõem processos mais efetivos de desenvolvimento de *software*, para oferecer rapidez, flexibilidade e entregas de qualidade. Já os modelos de maturidade dos processos são propostos para garantir níveis avançados de qualidade, considerando as melhores práticas de desenvolvimento de *software.* Assim, será objeto de estudos desta dissertação compreender a inter-relação entre métodos ágeis, gestão do conhecimento e a qualidade do *software* gerado, para descobrir se essas metodologias estão sendo efetivas e têm realmente contribuído para a qualidade do *software*.

Os desafios e as pressões das mudanças econômicas, tecnológicas e sociais obrigam as empresas a buscarem produtividade, qualidade e conhecimento, para poderem reagir às demandas do mercado e suas oportunidades e ameaças, em tempo hábil. Além de tecnologia e mercado para seus produtos e serviços, para serem competitivas, as empresas precisam de colaboradores adequadamente capacitados, pois, segundo Korobinski (2001), é pelos seus talentos que é gerada a criatividade e amplia-se o aprendizado organizacional. Assim, a maior vantagem competitiva de uma empresa está em seu capital humano, com base nos talentos, valores, competências e habilidades dos seus colaboradores (Matos & Lopes, 2008; Terra, 2000). Cavalcante (2000) afirma que, para que as empresas atinjam posição de destaque, a gestão do conhecimento oferece suporte muito importante. Para Ponchirolli e Fialho (2005), as empresas podem se distinguir pelo seu conhecimento e pela maneira como são capazes de usar esse conhecimento, o que, em uma economia global, se transforma em um dos seus maiores diferenciais competitivos.

De acordo com Fadel e Silveira (2010), as metodologias ágeis despontaram para que as empresas pudessem fazer entregas dos *softwares* solicitados pelos clientes, em prazo apropriado. Já para Nerur, Mahapatra & Mangalaraj (2005), as metodologias ágeis atendem a projetos de desenvolvimento de *software* com mudanças tanto em requisitos, quanto nas habilidades dos envolvidos e, também, nas tecnologias empregadas. Assim, é de se esperar que a utilização dos métodos ágeis, pelas empresas, seja cada vez maior. O 12º relatório anual *State of Agile* de 2018 confirma isso, trazendo a informação de que, atualmente, 97% das empresas utilizam métodos ágeis. Esse relatório foi composto de dados oriundos de 55% de empresas norte-americanas, 27% da Europa, 7% da Ásia, 7% da América do Sul, 3% da Austrália ou Nova Zelândia e 1% da África. Entre as empresas, 39% tinham menos de 1.000 funcionários, 21% de 1.001 a 5.000, 16% de 5.001 a 20.000 e 28% acima de 20.001 funcionários (Versionone, 2018).

Chrissis, Konrad & Shrum (2003) acreditam que o desenvolvimento de *software* é crítico para as empresas, porque vários processos corporativos têm suporte de *softwares* de todo tipo. Os investimentos em tecnologia da informação (TI), notadamente em *software*, representam, atualmente, considerável parcela do orçamento de várias empresas. Dessa forma, a qualidade do *software* gerado é primordial para evitar prejuízos de todo tipo, entre eles financeiros e de imagem (Lowry & Wilson, 2016; Oliveira, Petrini & Pereira, 2015).

Em contrapartida, a concorrência cada vez maior das empresas de TI vem pressionando a indústria de *software* a desenvolver produtos cada vez mais rapidamente (Oliveira *et al*., 2015). Os vários obstáculos encontrados no desenvolvimento de *software*, que abrangem recursos humanos e questões técnicas de negócio, burocráticas e políticas, aliados à alta complexidade dos produtos gerados, torna a gestão da qualidade do *software* muito difícil. “Idealmente, os sistemas de *software* devem não só fazer corretamente o que o cliente precisa, mas também fazê-lo de forma segura, eficiente e escalável e serem flexíveis, de fácil manutenção e evolução” (Bernardo e Kon, 2008).

Em decorrência disso, a qualidade do *software* gerado muitas vezes diminui, gerando aumento de custos de produção na correção das falhas que foram causadas por planejamento inadequado e prazos de desenvolvimento insuficientes. O estudo de Hughes, Dwivedi, Rana & Simintiras (2016) revela várias causas de falhas dos projetos de desenvolvimento de *software*:

a) Relação deteriorada entre a organização e o prestador de serviço;
b) falta ou pouco suporte do patrocinador (*sponsor*) ao projeto;

c) má-definição do processo de negócio, com benefícios do projeto maldefinidos e sua má-gestão financeira;
d) má-*performance* da equipe do projeto;
e) não aprendizado com os erros do passado;
f) projetos muito grandes e complexos;
g) mau gerenciamento dos requisitos e do escopo;
h) problemas de comunicação entre os envolvidos (*stakeholders*) no projeto.

Considerando uma análise por grupamentos, podem-se verificar várias causas relacionadas a problemas de comunicação ou ao processo de desenvolvimento de *software*. O 12º Relatório Anual *State of Agile* (Versionone, 2018), por sua vez, apresenta as principais dificuldades para a adoção dos métodos ágeis:

a) Cultura organizacional em desacordo com valores ágeis;
b) resistência à mudança da organização como um todo;
c) gerenciamento e patrocínio inadequados;
d) falta de habilidades/ experiência com métodos ágeis;
e) treinamento e educação insuficientes.

Sendo assim, "manter produtividade, eficiência e qualidade em equilíbrio tornou-se vital para o sucesso das empresas do ramo" (Oliveira *et al*., 2015).

Nesse sentido, as empresas de TI têm procurado ampliar a maturidade de seus processos de desenvolvimento de *software* com base nos modelos de qualidade, deixando-os mais previsíveis e eliminando, ou pelo menos controlando, os principais motivos da geração de produtos com baixa produtividade e pouca qualidade (Ahern Dennis, Aaron, & Richard, 2008). Essas empresas buscam, então, alterar seus processos com base em práticas validadas, atingindo a maturidade (Oliveira *et al*., 2015).

Existem lacunas teóricas no relacionamento entre métodos ágeis e qualidade de *software* e gestão do conhecimento e qualidade de *software*, que podem ser mais bem exploradas, trazendo ganhos tanto acadêmicos quanto práticos. Em termos acadêmicos, a pesquisa se propõe a investigar se existem relacionamentos entre os constructos citados. Por outro lado, a resposta a essa pergunta pode trazer ganhos práticos a partir do momento em que permite a análise das ações relacionadas a cada um dos constructos que devem ser executadas para garantir a geração de *software* de qualidade.

1.1 Pergunta de pesquisa

Apresentado o cenário, tem-se a seguinte questão de pesquisa: **qual a influência dos métodos ágeis, alinhados à gestão do conhecimento, na qualidade de *software* na perspectiva de profissionais de TI?**

1.2 Objetivo geral

Analisar a influência da utilização de métodos ágeis e da gestão do conhecimento na qualidade de *software*.

1.3 Objetivos específicos

a) Apresentar os conhecimentos relevantes e relacionamentos hipotéticos entre métodos ágeis, gestão conhecimento e qualidade de *software*.
b) Desenvolver uma modelagem de mensuração da qualidade de *software*, na forma de questionário, com a aplicação da análise multivariada.

1.4 Justificativa

A pesquisa de Campanelli, Camilo & Parreiras (2018), feita com profissionais de empresas desenvolvimento de *software* de Belo Horizonte, Brasil, mostrou que essas empresas, estão empregando métodos ágeis, em substituição aos métodos tradicionais. Entretanto, elas vêm fazendo adaptações para adequar os métodos ágeis ao seu ambiente, gerando métodos customizados ou híbridos, que se encaixam melhor às suas necessidades de negócio e muitas vezes não implicam fazer alterações significativas em seus processos de desenvolvimento de *software*.

A maior dificuldade da gestão do conhecimento é a transferência do conhecimento tácito para o conhecimento explícito, assim como a transferência do conhecimento explícito de pessoas para os grupos, dentro da organização. A atividade de desenvolvimento de *software* demanda, dos envolvidos no processo, grande conhecimento relativo às tecnologias envolvidas, ao processo de desenvolvimento e ao gerenciamento de projetos e requer destes tanto o conhecimento tácito quanto o explícito. Esse conhecimento não é estático, pelo contrário, ele se modifica com a tecnologia, a cultura organizacional e o processo de

desenvolvimento de *software* da empresa (Aurum, Ross, Wohlin & Meliha, 2003). Assim, o sucesso do desenvolvimento de *software* depende das atividades de planejamento e do conhecimento do conteúdo relativos ao *software* desenvolvido (Levy & Hazzan, 2009).

No tocante à qualidade de *software*, Bernardo e Kon (2008) destacam que a ideologia de vários métodos ágeis, entre eles o *lean*, o *scrum* e o *Extreme Programming* (XP), sugere que evitar erros é mais fácil e menos oneroso do que detectá-los e, depois, corrigi-los. Para isso, todos os envolvidos no projeto devem trabalhar para garantir a qualidade do *software* em todo o projeto.

Silva (2007) informa que os resultados de pesquisas indicam aumento da utilização de modelos de melhoria dos processos de desenvolvimento de *software* nas empresas de TI, no Brasil, demonstrando preocupação com os indicadores de qualidade dessas empresas, para cumprir requisitos cada vez mais exigentes dos seus usuários e clientes.

Pelo que foi exposto, uma pesquisa que apresente, de um lado, os pressupostos teóricos da gestão conhecimento, dos métodos ágeis e, principalmente, o relacionamento dessas disciplinas com a qualidade de *software* irá contribuir para complementar os estudos acadêmicos nessa área. Por outro lado, uma pesquisa de campo que apresente a influência dos métodos ágeis e da gestão do conhecimento na qualidade do *software* poderá auxiliar as empresas a avaliar as práticas e métodos utilizados no seu processo de desenvolvimento de *software*, em especial destes dois primeiros constructos, para que elas possam gerar *softwares* de melhor qualidade, mesmo no contexto atual de projetos de prazos e custos apertados.

1.4.1 Relação com o programa do mestrado

Esta dissertação se relaciona à linha de pesquisa Gestão da Informação e Conhecimento, mais precisamente com a trilha conhecimento, estratégia e modelos de negócios, uma vez que busca avaliar influências dos métodos ágeis e da gestão do conhecimento na qualidade de *software*, tendo natureza interdisciplinar, avaliando desempenho organizacional e processos de gestão do conhecimento.

1.5 Estrutura da dissertação

Esta dissertação está estruturada em seis capítulos e um apêndice.

No primeiro capítulo tem-se a introdução.

O segundo capítulo apresenta o Referencial Teórico, exibindo o embasamento teórico dos constructos da pesquisa (métodos ágeis para desenvolvimento de *software*, gestão do conhecimento e qualidade de *software*), bem como os trabalhos anteriores sobre o relacionamento entre eles.

A metodologia é descrita no terceiro capítulo, onde é explicada a natureza da pesquisa, listados os passos seguidos para a sua realização e apresentado o seu modelo conceitual.

No quarto capítulo faz-se a análise dos dados do questionário desenvolvido para a pesquisa desta dissertação.

O quinto capítulo traz a conclusão do trabalho.

O sexto capítulo mostra as referências utilizadas no texto.

Finalmente, é acrescentado um apêndice: o questionário utilizado na pesquisa.

2 Referencial Teórico

O referencial teórico representa uma base conceitual organizada, com teorias, abordagens e estudos que possibilitem o entendimento do fenômeno a ser estudado (Rodrigues, 2007). A seguir são apresentados os constructos da pesquisa e o relacionamento entre eles:

a) métodos ágeis;
b) gestão do conhecimento;
c) relacionamento entre métodos ágeis e gestão do conhecimento;
d) qualidade de *software*;
e) relacionamento entre métodos ágeis e qualidade de *software*;
f) relacionamento entre gestão do conhecimento e qualidade de *software;*
g) modelo conceitual desta dissertação.

2.1 Métodos ágeis

A Engenharia de *Software* engloba o desenvolvimento de sistemas com base no tratamento sistemático e disciplinado, com o objetivo de desenvolvimento, operação e manutenção de *software* (Kasse, 2008). Inicialmente, ela era baseada nos processos de manufatura, notadamente nos da indústria automobilística. A partir da metade dos anos 90, surgiram novos métodos, mais "leves" (*lightweight*), em detrimento aos métodos tradicionais, mais burocráticos, lentos e "pesados" (*heavyweight*). Os métodos "leves" tinham como principais características a utilização de equipes multifuncionais (cujos integrantes apresentavam diferentes conhecimentos e habilidades) e auto-organizadas, para possibilitar entregas rápidas e de qualidade (Gomes *et al*., 2014).

A partir de 2001, essas metodologias passaram a ser chamadas de "ágeis", quando um grupo de 17 especialistas se reuniu para debater formas de desenvolvimento de *software* de forma mais rápida e baseada nas pessoas envolvidas no processo. Eles criaram, então, o Manifesto Ágil, que é a base do desenvolvimento ágil de *software* ou dos métodos ágeis. Ele é composto de alguns valores e 12 princípios (Gomes *et al*., 2014).

Os valores são: a) indivíduos e interação; b) *software* em funcionamento; c) colaboração com o cliente; d) resposta a mudanças.

Beck *et al.* (2001), sugerem estes 12 princípios:

1. Nossa maior prioridade é satisfazer o cliente através da entrega antecipada e contínua de software valioso.
2. Bem-vindo mudanças nos requisitos, mesmo no final do desenvolvimento. Os processos ágeis aproveitam as mudanças para a vantagem competitiva do cliente.
3. Entregue software de trabalho com frequência, de algumas semanas a alguns meses, com uma preferência pela menor escala de tempo.
4. Empresários e desenvolvedores devem trabalhar juntos diariamente durante todo o projeto.
5. Construa projetos em torno de indivíduos motivados. Dê a eles o ambiente e o suporte de que precisam e confie neles para realizar o trabalho.
6. O método mais eficiente e eficaz de transmitir informações para e dentro de uma equipe de desenvolvimento é a conversa face a face.
7. O software de trabalho é a principal medida de progresso.
8. Processos ágeis promovem o desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente.
9. A atenção contínua à excelência técnica e ao bom design aumenta a agilidade.
10. Simplicidade - a arte de maximizar a quantidade de trabalho não realizado - é essencial.
11. As melhores arquiteturas, requisitos e projetos emergem de equipes auto-organizadas.
12. Em intervalos regulares, a equipe reflete sobre como se tornar mais eficiente, depois ajusta e ajusta seu comportamento de acordo.

Posteriormente, parte desses especialistas concebeu a *Star Alliance* (2014) para impulsionar o desenvolvimento ágil. A Figura 1 apresenta os métodos ágeis mais utilizados atualmente, segundo o 12º *State of Agile* (Versionone, 2018).

No Brasil, segundo Goldman, Melo, Kon, Corbucci e Santos (2014), os métodos ágeis foram vistos, inicialmente, com muita desconfiança. Mas essa situação mudou muito nos últimos anos e a utilização dos métodos ágeis está crescendo, fazendo com que eles sejam aplicados por cada vez mais profissionais e equipes.

A seguir são apresentados os princípios básicos dos principais métodos ágeis.

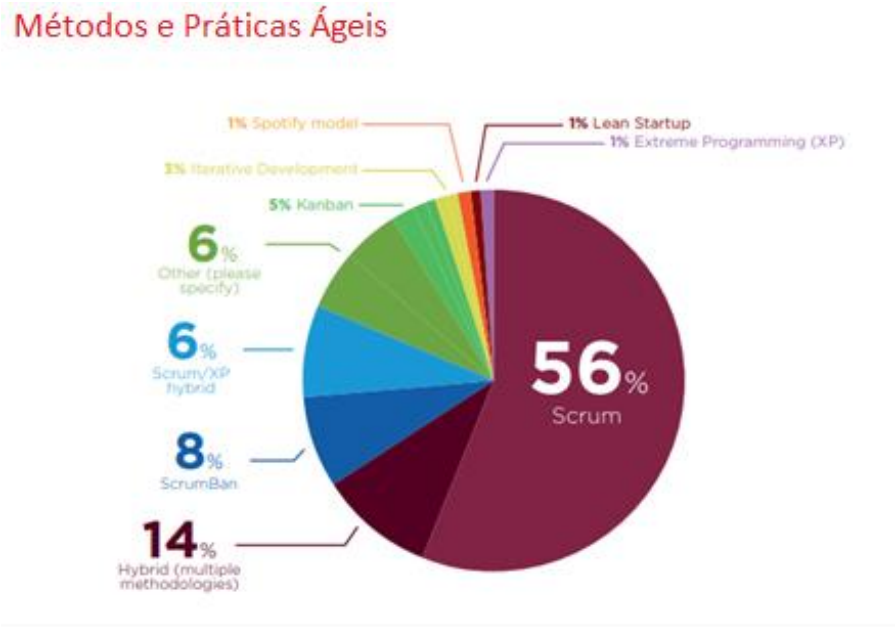

Figura 1
Métodos ágeis mais utilizados.
Fonte: Versionone. (2018). *Anais do 12 Annual State of Agile Report*. Retrieved from: https://explore.versionone.com/state-of-agile/versionone-12th-annual-state-of-agile-report.

2.1.1 Scrum

O desenvolvimento de sistemas é uma atividade complicada e imprevisível, pois envolve diversas variáveis (técnicas e de ambiente) que podem ser alteradas ao longo do projeto e, por isso, necessita de um processo flexível e que possa absorver as mudanças (Franco, 2007).

Cruz (2013) e Prikladnicki e Magno (2014) reconhecem o *scrum* como um *framework* de gerenciamento ágil de projetos de *software*, mas pode ser utilizado para o desenvolvimento de qualquer tipo de produto. Takeuchi & Nonaka (1986) relacionaram a formação *scrum* do esporte *rugby* à equipes pequenas e multidisciplinares que, muitas vezes, geravam melhores resultados. Cruz (2013), por sua vez, afirma que o *scrum* é uma jogada do *rugby*, em que todos os jogadores participam da disputa de reposição de bola e se um deles errar todos erram. Araújo e Galina (2005) prelecionam que ele foi normatizado, em 1995, por Ken Schwaber, que dedicou a estabelecê-lo como método de desenvolvimento de *software* no mundo todo.

A metodologia do *scrum* se propõe a fazer entregas contínuas de *software* (Fadel & Silveira, 2010). Cruz (2013) mostra que o *scrum* é composto de equipes pequenas (times) que têm papéis e responsabilidades e executam eventos de duração fixa, com o apoio de artefatos específicos.

a) Papéis e responsabilidades

O time *scrum* é constituído por três papéis: *scrummaster*, *product owner* (dono do produto) e o time (Cruz, 2013).

b) *Scrummaster*

A responsabilidade do *scrummaster* é garantir que o time *scrum* siga o fluxo dessa metodologia e atuar na remoção dos impedimentos que atrapalhem os objetivos do time. É ele que assegura que as regras e práticas do *scrum* estão sendo utilizadas, ajudando o time a compreender o autogerenciamento, permitindo que ele seja mais produtivo e desenvolva produtos de qualidade (Schwaber & Sutherland, 2017).

c) *Product owner* (dono do produto)

O *product owner* é responsável por entender o negócio e manter o *backlog* do produto, priorizando os itens a serem entregues em cada uma das interações e assegurando valor ao trabalho do time. É ele que garante o entendimento do produto a ser entregue pelo time (Cruz, 2013).

d) Equipe de desenvolvimento

Segundo Prikladnicki e Magno (2014), a equipe de desenvolvimento é a equipe de desenvolvedores responsável pelo desenvolvimento de incrementos do produto a serem entregues ao final de cada ciclo. Os membros da equipe de desenvolvimento devem ser multidisciplinares, ou seja, ter todo o conhecimento necessário para o desenvolvimento do produto (Cruz, 2013). Também é a equipe de desenvolvimento que faz as estimativas do tamanho dos itens a serem desenvolvidos e, por conseguinte, é responsável pela meta de entregas de um ciclo (Prikladnicki & Magno, 2014).

e) Artefatos

Prikladnicki e Magno (2014) admitem que os artefatos do *scrum* proporcionam um panorama da execução do projeto e de cada ciclo. São eles: *backlog* do produto, *backlog* da *sprint* e incremento do produto.

f) *Backlog* do produto

O *backlog* do produto é uma lista ordenada com os itens a serem incrementados no produto, cujos itens priorizados pelo *product owner* ficam no início e serão implementados antes (Schwaber & Sutherland, 2017). Para esses itens, têm-se conhecimento e melhor detalhamento. Cruz (2013, p. 35), por sua vez, salienta que essa lista contém "características, funções, tecnologias, melhorias e correções que constituem a versão futura do produto a ser entregue".

Ao longo do projeto, Prikladnicki e Magno (2014) realçam que o *backlog* do produto pode receber itens que serão priorizados em detrimento a outros, que perderão importância ou até mesmo serão retirados da lista.

g) *Backlog* da *sprint*

O *backlog* da *sprint*, de acordo com Schwaber & Sutherland (2017), contém o conjunto de itens selecionados para compor uma *sprint* (ciclo) e o planejamento para implementá-los e gerar o incremento do produto. O *backlog* da *sprint* é criado na reunião de planejamento de *sprint*, trazendo visibilidade ao trabalho necessário para atingir a meta da *sprint* para a equipe de desenvolvimento.

h) Incremento do produto

De acordo com Schwaber & Sutherland (2017), o incremento do produto é o resultado gerado na *sprint*. Ele deve ser um produto ou parte de um produto que, a critério do *product owner*, poderá ser colocado em produção. Para isso, o incremento do produto deverá estar "pronto", que é um conceito ligado à definição da entrega de algo com um nível de qualidade que atenda aos requisitos dos clientes, conforme acordado com a equipe de desenvolvimento.

i) Cerimônias do *scrum*

No entendimento de Prikladnicki e Magno (2014) e Cruz (2013), o *scrum* possui alguns eventos (cerimônias) com um trabalho definido para ser executado em intervalos fixos (*time-boxed*). São eles: *sprint*, reunião de planejamento da *sprint*, reunião diária, reunião de revisão da *sprint*, reunião de retrospectiva da *sprint*.

j) *Sprint*

Prikladnicki e Magno (2014) e Cruz (2013) afirmam que a *sprint* é um ciclo completo de desenvolvimento que tem um objetivo definido, uma meta estipulada, em que se gera um incremento do produto, que poderá ser colocado em produção. Ela tem duração de duas a quatro semanas.

Esses autores acrescentam que a *sprint* é composta de outras cerimônias do *scrum*: reunião de planejamento da *sprint*, reunião diária, reunião de revisão da *sprint*, reunião de retrospectiva da *sprint*.

k) Reunião de planejamento da *sprint*

A reunião de planejamento da *sprint* é uma cerimônia que, conforme Schwaber & Sutherland (2017), ocorre no início da *sprint*, quando esta é planejada pela equipe *scrum*. Ela tem a duração de oito horas, para uma *sprint* de um mês, tendo duração proporcionalmente menor para *sprints* de menor duração.

Ainda na linha de pensamento de Prikladnicki e Magno (2014), a reunião de planejamento da *sprint* é dividida em duas partes com duração fixa de exatamente a metade do tempo da reunião. Na primeira parte, o objetivo é responder a seguinte pergunta: o que será gerado no incremento da *sprint?* Assim, a equipe *scrum* verifica, com o *product owner*, quais são funcionalidades priorizadas no *backlog* do produto que serão desenvolvidas na *sprint*. Depois disso, a equipe de desenvolvimento avalia a quantidade de itens possíveis de serem desenvolvidos na *sprint* e define, desse modo, a meta da *sprint*. Na segunda parte da reunião, a equipe de desenvolvimento planeja como implementar os itens priorizados, decompondo-os "em unidades de um dia ou menos".

l) Reunião diária

Segundo Prikladnicki e Magno (2014) e Cruz (2013), na reunião diária a equipe de desenvolvimento se encontra diariamente, por 15 minutos, para responder três perguntas:

a) O que eu fiz desde a última reunião diária?
b) O que eu pretendo fazer até a próxima reunião diária?
c) Existe algum impedimento para eu terminar alguma atividade?

A equipe de desenvolvimento tentará, então, resolver os problemas detectados pelos seus membros e repassará, ao *scrummaster*, os impedimentos que não se considerar capaz de solucionar (Prikladnicki & Magno, 2014).

m) Revisão da *sprint*

Cruz (2013) refere que a revisão da *sprint* é uma cerimônia de quatro horas, cujo objetivo é a inspeção, por parte do *product owner* ou do cliente, do incremento gerado na *sprint*. Nessa reunião será avaliada a entrega em relação à meta da *sprint* (Cruz, 2013; Prikladnicki & Magno, 2014).

n) Retrospectiva da *sprint*

A reunião de retrospectiva da *sprint* ocorre logo após a revisão do *sprint* e tem como objetivo verificar o que obteve êxito e o que precisa ser melhorado para as próximas *sprints*. Ela conta com a participação de todos os membros da equipe *scrum* (Schwaber & Sutherland, 2017). A retrospectiva da *sprint* é uma cerimônia de três horas, que deve ter como resultado a geração de ações exequíveis para serem implementadas nas próximas *sprints* (Cruz, 2013).

2.1.2 XP - Extreme Programming - programação extrema

A XP foi criado para ajudar pequenas equipes a desenvolver *software* quando os requisitos são vagos e mudam frequentemente (Beck, 2000). É considerada uma metodologia leve e focada em economia de custos, testes de unidade antes e durante as atividades de

código, integração frequente de sistemas completos, programação em pares, *design* simples e publicações frequentes de *software* (Beck, 2000).

Bassi (2014) reporta que a proposta da XP é que os desenvolvedores possam focar a programação, sem se importar com artefatos intermediários, enquanto o cliente não precisa definir todo o *software* no início do projeto, podendo adicionar funcionalidades ao longo do projeto. Nesse contexto, a XP permite a variabilidade dos requisitos e pressupõe que o desenvolvimento de *software* seja flexível e colaborativo.

Tudo começou quando Kent Beck assumiu a gerência de um projeto de desenvolvimento de um sistema crítico e problemático na Chrysler e foi bem-sucedido utilizando boas práticas de Engenharia de *Software*, que se tornaram a base da XP (Bassi, 2014). Assim, a XP pode ser definida como um conjunto de boas práticas de Engenharia de *Software* que tem por objetivo a entrega de *softwares* de qualidade, entre elas: revisão de código, integração rápida, testes automatizados, *feedback* do cliente e *design* simples.

A XP tem cinco valores que resumem os princípios da metodologia:

a) **Comunicação** - a criação de um *software* de qualidade necessita de muita integração e transparência entre todos os envolvidos, em uma relação franca, em que são expostos desde estimativas realistas até problemas de difícil solução (Bassi, 2014).
b) **Simplicidade** - a ideia é manter a simplicidade da modelagem à codificação, fazendo apenas o essencial, sem deixar a qualidade cair (Bassi, 2014).
c) **Coragem** - o desenvolvimento de *software* é um trabalho complicado que exige, além do comprometimento das pessoas envolvidas, coragem para inovar e admitir que os envolvidos não têm o conhecimento de tudo (Bassi, 2014).
d) ***Feedback*** - o *feedback* de um cliente em relação a um projeto deve acontecer o quanto antes, para garantir que o projeto está no rumo correto ou para serem feitas as correções necessárias (Bassi, 2014).
e) **Respeito** - a partir do pressuposto de que as pessoas constituem o componente fundamental dos projetos de *software*, o respeito é primordial para garantir comunicação e *feedback* de qualidade (Bassi, 2014).

Bassi (2014) adverte que uma equipe XP deve possuir todas as competências técnicas e negócio para desenvolver o *software* e, desse modo, deve conter "programadores, analistas de negócio, analistas de testes, *designers* de interação, arquitetos, gerentes de projeto, gerentes de produto, redatores técnicos, executivos e usuários". Na XP, desenvolvedores e clientes têm

o objetivo comum de gerar um *software* de qualidade e que atenda às necessidades de negócio.

Como na XP as pessoas são a essência do desenvolvimento de *software*, Bassi (2014) presume uma documentação enxuta, visto que a inconstância dos requisitos leva à dificuldade na sua manutenção em relação ao código gerado.

2.1.3 Lean

Crescêncio (2014) conceitua o *lean* como uma metodologia fortemente eficaz, que se propõe a entregar produtos de valor, com o mínimo esforço.

a) Jidoka

Uma das bases do *lean*, no entendimento de Crescêncio (2014), é o que pode ser chamado de automação inteligente, ou *jidoka*, que introduz o toque humano na automação. Em se tratando de *software*, a automação pode encurtar o prazo das tarefas e garantir a uniformização dos resultados. Processos como a compilação, testes, empacotamento e instalação, se automatizados, garantem mais qualidade e diminuição de custo.

b) Testes manuais e automatizados

A automatização dos testes garante mais rapidez nos testes e mais qualidade nas entregas, visto que o sistema como um todo pode ser testado, a cada entrega, evitando principalmente erros de integração (Crescêncio, 2014).

c) *Just in time*

O conceito do *just in time*, conforme Crescêncio (2014), surgiu das gôndolas dos supermercados americanos, onde os produtos eram repostos conforme iam sendo vendidos. Essa ideia foi implantada nas linhas de produção do carro da Toyota, numa técnica que foi chamada de *just in time*.

d) *Poka-Yoke*

Outro conceito importante do *lean* é o *poka-yoke*, que consiste em gerar formas de evitar o erro. Transportando esse conceito para o *software*, a utilização de interfaces de código bem-definidas ou a programação em pares, em que o código é gerado por dois desenvolvedores, são exemplos de maneiras de evitar erros (Crescêncio, 2014).

e) *Kanban*

Para Vale (2014), o *kanban*, que pode ser traduzido como "etiqueta de instrução", é uma ferramenta efetiva no controle da produção, ao passo que permite a integração entre diferentes áreas de um projeto. Ele pode ser representado por meio de um sistema de cartões, informando qual a especificidade e o material necessário, permitindo que uma área termine um item e repasse-o a outra área o mais rápido o possível.

Em projetos de *software*, Vale (2014) explicita que o *kanban* é utilizado para ilustrar o andamento dos itens selecionados para uma iteração, geralmente com as situações no projeto: não iniciado, andamento e finalizado. Os itens, representados por cartões, são colocados em quadros situados na área onde o projeto está sendo executado.

f) O sistema Toyota de produção

Segundo Crescêncio (2014), para aumentar a produtividade, a Toyota utiliza os conceitos citados anteriormente, como o *jidoka*, o *just in time* e o *kanban*, evitando o desperdício de tempo, dinheiro e material.

g) O sistema Toyota de desenvolvimento do produto

O sistema Toyota permite, também, o desenvolvimento mais rápido de novos produtos, contando desde a concepção até o início da produção (Crescêncio, 2014).

Sob o ponto de vista do desenvolvimento de *software*, os princípios e práticas do *lean* compõem a base do desenvolvimento ágil de *software*:

a) Entregar uma sequência ininterrupta de valor ao cliente;
b) criar uma organização que adquire continuamente conhecimento;

c) criar um espaço de melhoria contínua;
d) acabar totalmente com o desperdício.

2.1.4 FDD - Feature driven development - desenvolvimento dirigido por funcionalidades

A história do desenvolvimento dirigido por funcionalidades (FDD) iniciou-se em 1997, em Cingapura, quando um importante banco internacional montou uma equipe de 50 pessoas para fazer uma reengenharia em boa parte de seus processos, após tentativa frustrada de quase dois anos (Retamal, 2014). A equipe, liderada por Jeff De Lucca, contou com notável projetista de sistemas orientados a objetos. Ela conseguiu entregar 2.000 funcionalidades em 15 meses.

Retamal (2014) realça que o FDD, em 2001, foi considerada uma das metodologias do movimento ágil. Ele se propõe a unir boas práticas de Engenharia de *Software* e da gestão de projetos. Uma *feature* é uma funcionalidade pequena que pode ser desenvolvida em uma interação de duração de aproximadamente duas semanas. A descrição dessa funcionalidade, por sua vez, deve ser detalhada de forma que um desenvolvedor consiga fornecer uma estimativa correta para sua implementação e o testador ter segurança para desenvolver seus casos de teste.

O FDD é uma metodologia que contém cinco processos.

O processo número 1, “desenvolver um modelo abrangente”, ocorre quando o conhecimento tácito é compartilhado para gerar conhecimento explícito e elicitam-se os requisitos. Depois disso, são detalhados e cada domínio de negócio envolvido é modelado (Retamal, 2014).

O processo número 2, “construir a lista de funcionalidades”, tem com ideia principal a decomposição dos requisitos em funcionalidades, com o objetivo de "identificar de que forma o sistema deve apoiar os processos sendo automatizados" (Retamal, 2014, p. 79).

O processo número 3, “planejar por funcionalidade”, é executado pelo gerente de projeto, pelo gerente de desenvolvimento e pelos programadores líderes, que planejam o desenvolvimento das funcionalidades segundo suas dependências e a carga de trabalho envolvida (Retamal, 2014).

O processo número 4, “detalhar por funcionalidade”, desenvolve o projeto (*design*) das funcionalidades a serem implementadas a partir do pacote de funcionalidades planejado no processo anterior (Retamal, 2014).

No processo número 5, “construir por funcionalidade”, as funcionalidades detalhadas no processo anterior são, agora, implementadas, passam por testes unitários e são inspecionadas por um programador líder. A partir daí, é gerada uma compilação (*build*) com as funcionalidades (Retamal, 2014).

O FDD baseia-se em oito práticas essenciais com o objetivo de assegurar a qualidade do processo de desenvolvimento e do produto gerado:

a) **Modelagem de objetos do domínio.** A geração de um modelo de domínio é fundamental para a determinação das funcionalidades: fornece uma estrutura abrangente na qual adiciona funcionalidade; ajuda a manter a integridade conceitual do sistema; reduz a necessidade e a quantidade de refatoração (*refactoring*); é uma forma de armazenamento e comunicação concisa, relativamente acessível e reutilizável, para todos os envolvidos no processo (Retamal, 2014, p. 89).
b) **Desenvolvimento por funcionalidade.** As funcionalidades serão utilizadas pelo cliente e são valorizadas por ele. Assim, o projeto deve ser conduzido para gerar entregáveis (Retamal, 2014).
c) **Posse individual de classe/código.** Essa prática determina o responsável por uma classe ou parte do código-fonte, garantindo que sempre existirá um especialista que consiga explicar o funcionamento dessa porção de código, facilitando o entendimento e a manutenção da funcionalidade (Retamal, 2014).
d) **Equipes de funcionalidades.** São equipes estabelecidas dinamicamente para o desenvolvimento das funcionalidades, enfatizando o trabalho em equipe (Retamal, 2014).
e) **Inspeções.** A proposta do FDD é entregar funcionalidades que agreguem valor ao cliente. Assim, para garantir a qualidade do produto, são feitas inspeções do desenho das funcionalidades (processo número 4) e do código fonte gerado (processo número 5), conforme apregoam várias fontes de literatura técnica (Retamal, 2014).
f) **Geração de *builds* frequentes.** A geração de *builds* periódicos garante a disponibilização das funcionalidades desenvolvidas e ajuda a antever problemas de integração (Retamal, 2014).
g) **Gestão de configuração.** A gestão de configuração trata das evoluções e modificações em artefatos do produto, com o objetivo de descomplicar a implementação do *software* e garantir sua integridade. "A cada iteração a equipe de desenvolvimento realiza as alterações no código em um ambiente separado da versão

atual (ramificação, ou *branch*/*fork*), evitando que problemas gerados localmente se espalhem por todo o sistema" (Retamal, 2014, p. 92).

h) **Relatório e visibilidade de resultados.** A ideia é proporcionar uma visão precisa da posição do projeto aos gestores e clientes (Retamal, 2014).

Após a apresentação de conceitos do constructo dos métodos ágeis, o tópico 2.2 discorre sobre o próximo constructo: a gestão do conhecimento.

2.2 Gestão do conhecimento

Davenport & Prusak (1998), inicialmente, apresentam definições de dado, informação e conhecimento, argumentando que as empresas têm dificuldade em diferenciar esses três conceitos. Para esses autores, no contexto orgarnizacional, dados são documentos organizados de transações, muitas vezes armazenados em sistemas de informação. E os dados nada dizem sobre o evento em si, não tendo valor intrínseco. Já a informação é formada por dados que agregam valor ao destinatário de uma mensagem. O conhecimento é diferente de dado e de informação e a maioria das pessoas tem a percepção de que ele é mais abrangente, mais relevante e mais rico que os outros dois. Assim como a informação descende dos dados, o conhecimento descende da informação. Os autores argumentam que essa conversão acontece com palavras que se iniciam com “C”:

a) Comparação: como informações de uma situação se relacionam a outras situações?
b) Consequências: em que uma informação vai influenciar em uma tomada de decisão?
c) Conexões: como esse conhecimento se compara a outros?
d) Conversa: qual a opinião de outras pessoas sobre uma informação?

O conhecimento, como conceituam Davenport & Prusak (1998), é a combinação de vários componentes, difícil de ser definido em palavras ou ser entendido integralmente, existindo no íntimo das pessoas. É a reunião de experiências, princípios, informações dentro de um âmbito e intuições de especialistas, que permitem a análise e a absorção de novas experiências e informações. Nas empresas, o conhecimento está presente tanto em documentos como também em rotinas, processos e normas organizacionais.

Davenport & Prusak (1998) reportam que o conhecimento agrega mais valor que informações ou dados e, ainda que se encontre na cabeça das pessoas, ele tem relação mais

estreita com a ação. O conhecimento carece de ser medido em relação à tomada de decisões que ele gera ou às ações que ele acarreta, pois ele pode trazer grande sucesso no desenvolvimento e produção de serviços.

O conhecimento é alcançado no decorrer do tempo, por meio de experiências obtidas em cursos, livros e instrutores, além do aprendizado informal. Os especialistas são pessoas com extenso conhecimento sobre um assunto, obtido com a experiência. A partir da experiência podem-se compreender novas situações, com base nos eventos ocorridos no passado. O conhecimento que é concebido com a experiência permite identificar padrões conhecidos e relações das situações atuais e passadas. Davenport & Prusak (1998) concluem que é esse conhecimento dos especialistas que é valorizado pelas empresas.

Os valores e crenças das pessoas influenciam o que a pessoa vê, aprende e deduz e, desse modo, pessoas diferentes podem ter percepções diferentes sobre um mesmo acontecimento. Davenport & Prusak (1998) opinam, ainda, que os valores e crenças são primordiais para o conhecimento das pessoas e concluem que "o poder do conhecimento de organizar, selecionar, aprender e julgar vem de valores e crenças tanto quanto, e provavelmente mais do que, de informação e lógica" (Davenport & Prusak, 1998, p. 12). Da mesma forma, os valores e crenças das pessoas têm grande repercussão no conhecimento organizacional, pois influenciam as suas ações nas empresas e, por consequência, influenciam as crenças corporativas.

Nonaka e Takeuchi (1997), por sua vez, afirmam que o conhecimento é o ponto principal, não somente para o progresso econômico de uma empresa, mas, antes de tudo, para o seu sucesso corporativo.

Segundo Davenport & Prusak (1998), as empresas contratam seus funcionários muito mais por sua experiência e conhecimento do que pela sua inteligência e educação. É mais provável que gerentes que tomam decisões complexas valorizem mais pessoas experientes do que informações em bancos de dados, porque eles vão se aconselhar com elas, em questões específicas. Os autores concluem que as empresas entenderam a necessidade de gerenciar o conhecimento organizacional com o mesmo cuidado que gerenciam seus ativos tangíveis e a necessidade de empregar esse conhecimento para conquistar o maior valor possível.

As empresas precisam, inicialmente, saber que têm um conhecimento, para poder usá-lo. Para empresas de grande porte e, principalmente, nas globais, com escritórios e fábricas espalhados pelo mundo, usar o conhecimento de que dispõem é, muitas vezes, difícil e não é raro que elas "reinventem a roda", solucionando os mesmos problemas mais de uma vez, porque o conhecimento não foi compartilhado entre seus funcionários (Davenport & Prusak,

1998). O conhecimento que, para empresas de pequeno e médio porte, de no máximo 300 funcionários, pode estar na sala ao lado, nas empresas de grande porte pode não ser, na prática, um ativo com alguma valia, se elas não o documentam de alguma forma. Davenport & Prusak (1998) concluem que, no ambiente competitivo atual, as empresas precisam ter um sistema para armazenar e localizar o conhecimento.

Choo (2006) demonstra, na Figura 2, que o *sense making*, a construção do conhecimento e a tomada de decisão estão interligados.

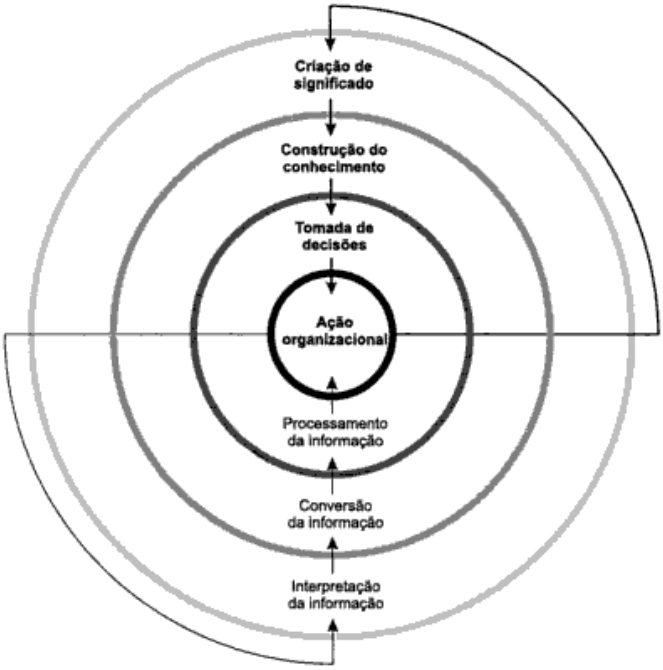

Figura 2
Sense making, construção do conhecimento e tomada de decisão.
Fonte: Choo, C. W. (2006). *The knowing organization: How organizations use information to construct meaning, create knowledge, and make decisions* (2. ed., Education + Training, v. 43). https://doi.org/10.1108/EUM0000000005482

Para Choo (2006), conforme apresentado na Figura 2, a informação parte do ambiente externo (do lado de fora do círculo) e, aos poucos, é compreendida pela empresa, para permitir as ações organizacionais. Primeiramente, as informações acerca do ambiente organizacional são identificadas, permitindo a criação do conhecimento e a tomada de decisão. Nesse aspecto, o conhecimento que se encontra na mente das pessoas precisa ser transformado em conhecimento comum, para ser aplicado. O entendimento e o conhecimento suportam a ação. Esta, por sua vez, muda o ambiente e faz o fluxo ser executado novamente, criando novas experiências e gerando um ciclo contínuo.

No *sense making*, as pessoas decidem quais são as informações mais importantes e apresentam coletivamente seus pontos de vista, permitindo um processo de construção do conhecimento, analisando as opções disponíveis para atingir os objetivos e, assim, realizar a tomada de decisão (Choo, 2006). Para Choo (2006, p. 4), “todos os três modos de uso da informação - interpretação, conversão e processamento - são processos sociais dinâmicos que continuamente constituem e reconstituem significado, conhecimento e ação.” Finalmente, o autor conclui que as empresas que verdadeiramente conseguem integrar o *sense making*, a construção do conhecimento e a tomada de decisão são chamadas de organização do conhecimento.

O *sense making*, como sugere Choo (2006), inicia quando ocorrem mudanças no ambiente, trazendo mensagens e sugestões que permitem várias interpretações. Essas mudanças, chamadas de mudanças ecológicas, fazem com que os membros das empresas procurem compreender o seu significado. As interpretações geradas podem ser equivocadas e é necessário fazer uma retrospectiva e analisar acontecimentos passados, para tornar os ambientes mais estáveis e previsíveis.

A partir daí, Choo (2006) alerta que a empresa tentar isolar essas mudanças. O primeiro passo nesse sentido é chamado de promulgação, cujos membros selecionam e isolam as informações importantes e, depois, as categorizam e associam aos seus atores. Os dados gerados, entretanto, ainda podem ser equivocados.

No próximo passo, como propõe Choo (2006), chamado de seleção, as pessoas analisam os dados promulgados em relação a situações anteriores que poderiam esclarecer, da melhor forma, a situação atual. Como resultado, são geradas relações de causa e efeito que deixam o ambiente mais compreensível.

Por fim, Choo (2006) discorre sobre o processo de retenção: os produtos que permitiram explicações bem-sucedidas da mudança, obtidos nos passos anteriores do *sense making*, são retidos, para que possam ser utilizados no futuro.

Takeuchi e Nonaka (2008) registram que o conhecimento tem dois componentes supostamente antagônicos: o conhecimento tácito e o conhecimento explícito.

O conhecimento explícito pode ser manifestado com palavras, números e áudios e explicado por meio de fórmulas científicas, vídeos, manuais e especificações de produtos. Ele pode ser compartilhado com as pessoas de forma rápida, formal e sistemática. Ele "inclui coisas das quais estamos cientes e podemos discutir com os outros. É facilmente capturado, expresso e codificado" (Bradley, Paul & Seeman, 2006).

Em contrapartida, o conhecimento tácito está muito ligado a experiências e às atividades de uma pessoa, sendo difícil de transmitir ou explicar. Ele incorpora intuições, palpites e subjetividade (Takeuchi & Nonaka, 2008). Normalmente, o conhecimento tácito é inconsciente e usa a intuição ou a habilidade natural das pessoas. É de difícil articulação ou codificação (Bradley *et al.*, 2006).

Existem duas dimensões para o conhecimento tácito. Uma é a dimensão técnica, que abrange as habilidades técnicas e informais de uma pessoa e, muitas vezes, é chamada de *know-how*. É o conhecimento representado pelas especialidades das pessoas. A outra dimensão é a cognitiva. Ela é composta de "crenças, percepções, ideais, valores, emoções e modelos mentais tão inseridos em nós que os consideramos naturais" e não é transmitida de forma fácil (Takeuchi & Nonaka, 2008, p. 19).

Takeuchi e Nonaka (2008) defendem que o conhecimento organizacional é gerado a partir da transformação do conhecimento tácito em explícito, e vice-versa, por meio de um ciclo chamado espiral do conhecimento, espiral SECI ou, simplesmente, SECI (Figura 3). O ciclo apresenta quatro tipos de transformação, a saber:

1. **Socialização (de tácito para tácito):**

Na socialização, o conhecimento tácito é criado e compartilhado entre os indivíduos, a partir da experiência direta.

2. **Externalização (de tácito para explícito):**

Na externalização, o conhecimento é debatido por meio do diálogo e da reflexão.

3. **Combinação (de explícito para explícito):**

Na combinação, o conhecimento explícito é sistematizado e empregado.

4. **Internalização (de explícito para tácito):**

Na internalização, o indivíduo assimila o novo conhecimento tácito pela prática.

Takeuchi e Nonaka (2008, p. 23) concluem sobre a espiral do conhecimento:

> A espiral também é amplificada à medida que passa para os níveis ontológicos, do indivíduo para o grupo e, então, para a organização. Cada modo do processo SECI envolve uma combinação diferente das entidades de criação do conhecimento, como mostrado abaixo:
> 1. Socialização: indivíduo para indivíduo.
> 2. Externalização: indivíduo para grupo.
> 3. Combinação: grupo para organização.
> 4. Internalização: organização para indivíduo.

Figura 3
Espiral do conhecimento.
Fonte: Takeuchi, H., & Nonaka, I. (2008). *Gestão do conhecimento: paradoxo e conhecimento*. Retrieved from: http://srvd.grupoa.com.br/uploads/imagensExtra/legado/T/takeuchi_Hirotaka/Gestao_Do_Conhecimento/Liberado/Cap_01.pdf.

Por fim, Bradley *et al*. (2006, p. 89) afirmam que "o conhecimento tácito complexo pode ser eliciado e codificado com precisão". Dessa forma, Scorsolini-Comin, Inocente e Miura (2011) demonstram que a aprendizagem organizacional pode ser conceituada como a forma como as empresas interagem com o seu ambiente interno e seus indivíduos, aprendendo, extraindo e absorvendo informações, com o intuito de se desenvolver e agregar vantagens competitivas.

Choo (2006) apregoa que a construção do conhecimento pode iniciar pelos *insights* ou intuições dos membros das empresas, para a solução de problemas. Entretanto, enquanto esse conhecimento, que é tácito, não for compartilhado com outros membros, as empresas não serão capazes de explorá-lo plenamente. Para que uma ideia, entendida como viável, seja convertida em produto, a empresa precisa adicionar recursos operacionais e de produção. Então, Choo (2006, p. 10) conclui: “finalmente, a experiência de criar e aplicar o novo conhecimento é assimilada e internalizada como novas práticas e modelos mentais que fornecem a base para mais ciclos de criação de conhecimento”.

Idealmente, uma tomada de decisão racional demandaria uma pesquisa de todas as opções disponíveis e suas consequências, para a análise dos resultados (Choo, 2006). No mundo real, isso não é possível, o que torna limitada a racionalidade das tomadas de decisão. Choo (2006) explica que essa racionalidade limitada leva os membros das empresas a

escolher ações que sejam satisfatórias, simplificando a busca das ações possíveis, ao contrário de buscar uma solução ótima.

Nesse aspecto, Choo (2006) explica que existem dois tipos de premissas para a tomada de decisão. Primeiramente, existem as premissas de valor, a partir das quais o tomador de decisão avalia o que é valioso ou bom em uma opção, para definir qual é o resultado esperado de uma ação. Existem, também, as premissas factuais, em que o tomador de decisão observa do ambiente o que é relevante para a tomada de decisão.

Choo (2006) apregoa que a pesquisa das opções de ação é causada pela ocorrência de um problema, que é avaliado segundo soluções anteriores para problemas semelhantes e dirigido pela experiência e pelos objetivos do tomador de decisão. As premissas e a busca simplificada por ações satisfatórias constituem a base dos programas organizacionais de tomada de decisão.

Os três modos do uso da informação sugeridos por Choo (2006), embora tratem de questões diferentes do comportamento das empresas, estão inter-relacionados. O *sense making* produz interpretações compartilhadas para servir de base para ações organizacionais. A criação do conhecimento possibilita a inovação organizacional, materializada em novos produtos e novas competências. Por fim, os tomadores de decisão escolhem uma ação para responder a um acontecimento, entre várias opções possíveis, baseado em premissas e regras.

Para Korobinski (2001), uma empresa pode atingir resultados convincentes por meio de muitas estratégias e técnicas gerenciais, mas é necessária a análise se elas convergem para a aprendizagem organizacional, oferecendo condições para a criatividade e satisfação pessoal e viabilizando novas oportunidades de negócio.

Patriotta (2003) complementa que o desempenho organizacional está condicionado à institucionalização do conhecimento. E Scorsolini-Comin *et al*. (2011) afirmam que a aprendizagem organizacional é uma ferramenta para absorção do aprendizado dos indivíduos, que está ligada a uma política de desenvolvimento organizacional e está em conformidade com a sua estratégia. Os resultados dessa aprendizagem podem ser sentidos, por exemplo, quando é quebrada a dependência entre um indivíduo responsável pela mudança de um processo organizacional e a adoção do novo processo, na organização.

Nesse sentido, Gonzalez e Campos (2015) acrescentam que a globalização traz novas oportunidades e desafios, em um mercado fortemente competitivo. As empresas dependem de informação e conhecimento para implantar processos de inovação, para o quais seus profissionais podem trazer novas ideias, permitindo sua evolução e permanência no mercado. A inovação é, desse modo, um desses desafios, mas ela só ocorrerá se processos como a

gestão do conhecimento, a aprendizagem organizacional e a gestão da inovação estejam implantadas nessas empresas.

Morum (2005) mostra que a gestão do conhecimento trata da transformação do conhecimento das pessoas em conhecimento de grupo. E o conhecimento dos grupos, por sua vez, em conhecimento organizacional. Já segundo Pandey (2016, p. 1), "gestão do conhecimento é uma vasta área que compreende criação, aquisição, colaboração, compartilhamento, uso, reutilização e capitalização de conhecimento em uma organização".

Assim, nas últimas três décadas, as empresas se propuseram a adotar a gestão do conhecimento para obterem vantagens competitivas, evitando a "reinvenção da roda" e permitindo a economia de tempo e recursos (Pandey, 2016).

Após a apresentação de conceitos do constructo de gestão do conhecimento, o tópico 2.3 apresenta estudos do relacionamento entre gestão do conhecimento e TI/métodos ágeis.

2.3 Relacionamento entre gestão do conhecimento e TI / métodos ágeis

Para embasar o relacionamento entre gestão do conhecimento e métodos ágeis, foram selecionados artigos sobre os temas *knowledge management*, *agile* e *scrum*, pesquisados nos seus títulos e resumos, nas bases do Instituto de Engenheiros Eletricistas e Eletrônicos (IEEE), *Science Direct* e *Google* acadêmico. Como critérios de exclusão foram eliminados da pesquisa livros, trabalhos de conclusão de curso, dissertações, teses e artigos que continham apenas um dos termos citados.

Após a seleção citada anteriormente, foram encontrados 273 artigos, sendo 140 da base do *Google* acadêmico e 133 da *Science Direct* e, a princípio, não foi encontrado algum artigo do IEEE que atendesse ao protocolo da pesquisa. Após a leitura e análise mais apurada, 135 artigos do *Google* acadêmico e os 133 da *Science Direct* foram descartados, por não estarem realmente relacionados à pesquisa, embora contivessem os termos pesquisados. Sendo assim, foram selecionados cinco artigos para a pesquisa. Estranhamente, desses artigos, dois pertencem à base IEEE, embora só tenham sido retornados no mecanismo de pesquisa do *Google* acadêmico. Somados a eles, foram adicionados mais quatro arquivos obtidos da referência bibliográfica de alguns artigos e de pesquisas menos estruturadas.

A Tabela 1 exibe os artigos selecionados para a pesquisa.

Rossetti e Morales (2007) destacam a evolução tecnológica do mundo, que abrange pessoas, empresas e engloba praticamente todas as atividades e disponibiliza significativa quantidade de informação, notadamente pela internet. Esses autores reconhecem que a

associação entre a gestão do conhecimento e a TI é demasiadamente complicada, pois abrange, de um lado, pessoas, conhecimentos tácitos, explícitos e empresariais e, de outro, sistemas de informação.

O estudo dos autores chegou a algumas conclusões sobre o assunto que, em termos mais abrangentes, pode resumir o relacionamento entre os dois constructos:

a) As empresas que valorizam seus profissionais e os recompensam por partilhar seus conhecimentos criam um ambiente mais favorável à gestão do conhecimento;
b) a gestão do conhecimento é um fator crítico de sucesso para a tomada de decisão nas empresas. A integração entre a gestão do conhecimento e a TI pode melhorar essa tomada de decisão e ser um diferencial da empresa no mercado;
c) a literatura que trata do relacionamento entre gestão do conhecimento e TI é escassa;
d) algumas empresas creem que a simples implantação de ferramentas informatizadas de gestão do conhecimento pode caracterizá-las como empresas que implementam a gestão do conhecimento, o que é um grande erro;
e) assim, a TI exerce uma função de infraestrutura para a gestão do conhecimento. É uma ferramenta de apoio à tomada de decisão e análise de mercado, favorecendo a interação entre pessoas e grupos nas empresas;
f) a TI permite, então, a rápida disseminação do conhecimento na empresa, sendo um elemento estratégico para a sobrevivência das empresas.

Tabela 1
Trabalhos sobre o relac. entre gestão do conhecimento e TI/ métodos ágeis

Artigo	Base
Rossetti, A. G., & Morales, A. B. T. (2007). O papel da tecnologia da informação na gestão do conhecimento. *Ciência da Informação*, *36*(1), 124–135. https://doi.org/10.1590/S0100-19652007000100009.	Outras pesquisas
Levy, M., & Hazzan, O. (2009). Knowledge management in practice: The case of agile software development. *Anais do Icse Workshop on Cooperative and Human Aspects of Software Engineering*, 60–65. https://doi.org/10.1109/CHASE.2009.5071412.	Outras pesquisas
Dingsøyr, T., Nerur, S., Balijepally, V., & Moe, N. B. (2012). A decade of agile methodologies: Towards explaining agile software development. *Journal of Systems and Software*, *85*(6), 1213–1221. https://doi.org/10.1016/j.jss.2012.02.033.	Outras pesquisas
Dorairaj, S., Noble, J., & Malik, P. (2012, August). Knowledge management in distributed agile software development. *Anais da Agile Conference (AGILE)*, 2012 (pp. 64-73). IEEE.	IEE *Google* Acadêmico
Gandomani, T. J., Zulzalil, H., Ghani, A. A. A., Sultan, A. B. M., & Nafchi, M. Z. (2013). Obstacles in moving to agile software development methods; At a Glance. *Journal of Computer Science*, *9*(5), 620–625. https://doi.org/10.3844/jcssp.2013.620.625.	Outras pesquisas
Dingsøyr, T., & Smite, D. (2014). Managing knowledge in global software development projects. *IT Professional*, *16*(1), 22-29.	*Google* Acadêmico
Razzak, M. A., & Ahmed, R. (2014, September). Knowledge sharing in distributed agile projects: Techniques, strategies and challenges. *Anais do Computer Science and Information Systems (FedCSIS)*, 2014 *Federated Conference on* (pp. 1431-1440). IEEE.	IEEE *Google* Acadêmico
Singh, A., Singh, K., & Sharma, N. (2014). Agile knowledge management: a survey of Indian perceptions. *Innovations in Systems and Software Engineering*, *10*(4), 297-315.	*Google* Acadêmico
Yanzer Cabral, A. R., Ribeiro, M. B., & Noll, R. P. (2014). Knowledge management in agile software projects: A systematic review. *Journal of Information & Knowledge Management*, *13*(01), 1450010.	*Google* Acadêmico

Fonte: elaborada pelo autor.

Na linha de pensamento de Dingsoyr & Smite (2014), o desempenho de uma equipe depende do conhecimento que é compartilhado entre seus membros, evitando erros e permitindo que membros da equipe assumam atividades de outros membros, em situações de alto volume de trabalho, garantindo o bom andamento do projeto.

No desenvolvimento de sistemas, Levy & Hazzan (2009) afirmam que a documentação gerada pelos métodos tradicionais é mais pesada, dando mais suporte à gestão do conhecimento, em vez dos métodos ágeis, cuja documentação é reduzida. Segundo Dingsøyr, Nerur, Balijepally & Moe (2012), isto é, inclusive, entendido por muitos como ausência de documentação. Singh *et al.* (2014) referem que a predisposição técnica das equipes de desenvolvimento de *software* tende a sobrepor o valor de "indivíduo e interação" do manifesto ágil. Singh *et al.* (2014), Gandomani *et al.* (2013) e Yanzer Cabral *et al.* (2014), salientam que isso sustenta a crença de que a maior parcela do conhecimento empregada no desenvolvimento de *software*, utilizando-se métodos ágeis, é tácito e está em poder da equipe de desenvolvimento. Já Dorairaj *et al.* (2012) enfatiza que o conhecimento necessário para o

desenvolvimento de *software* utilizando métodos ágeis depende muito do projeto e frequentemente é complicado aplicá-lo em outros projetos.

Por outro lado, no estudo de Singh *et al.* (2014), feito em empresas indianas, os defensores dos métodos ágeis argumentam que esses métodos contêm várias práticas da gestão do conhecimento entre suas atividades. Assim, de acordo com Dorairaj *et al.* (2012), os métodos ágeis promoveriam o compartilhamento do conhecimento. Entretanto, não existem estudos concretos que garantam a adoção dessas práticas de gestão do conhecimento nesse contexto. Razzak & Ahmed (2014) entendem que o compartilhamento do conhecimento em projeto baseado em métodos ágeis depende da iniciativa dos membros da equipe, pois algumas técnicas desses próprios métodos favorecem esse compartilhamento: programação em pares, colaboração de clientes, quadros do *scrum* e do *kanban*. Outras técnicas, como *workshops*/seminários, apresentações técnicas e técnicas de discussão também são utilizadas.

Gandomani *et al.* (2013) sublinham, então, a importância de uma estratégia adequada da gestão do conhecimento e sua disseminação na empresa. Razzak & Ahmed (2014) complementam que, desse modo, faz-se necessária a análise criteriosa do conhecimento que pode ser reutilizado. Dingsoyr & Smite (2014) concluem que é necessária uma avaliação proativa para definir o que deve ser compartilhado. Para projetos globais, deve ser avaliado o que será compartilhado entre os membros das equipes locais e o que vai ser compartilhado globalmente. Localmente, podem ser adotadas estratégias menos dispendiosas, enquanto que globalmente a utilização de *sites*, para compartilhar o conhecimento, pode ser uma boa opção. Mas a colaboração entre as equipes deve ser incentivada constantemente. Vale ressaltar que os casos de sucesso no compartilhamento de conhecimento devem fazer parte do aprendizado organizacional.

Após a apresentação de conceitos do constructo gestão do conhecimento, o tópico 2.4 apresenta o próximo constructo: qualidade de *software*.

2.4 Qualidade de *software*

Em uma definição clássica de Juran & Godfrey (1998), qualidade significa ausência de deficiências. Já Garvin (1992) propõe oito dimensões para entender a qualidade: desempenho, características/especificações, confiabilidade, conformidade, durabilidade, atendimento ao cliente, estética/imagem e qualidade percebida. Valls (2004) informa que a norma *International Standardization for Organization* (ISO) 9000, por sua vez, apresenta a definição de que qualidade é o grau no qual um conjunto de características inerentes satisfaz requisitos.

Bueno e Campelo (2006) acreditam que é mais fácil explicar a qualidade de *software* por meio de um conjunto de características ou requisitos dos produtos ou dos clientes que os demandam. Pode-se classificar a qualidade de *software* por meio de fatores de qualidade internos e externos:

a) Fatores externos são aqueles que podem ser percebidos pelos usuários do produto, como, por exemplo, facilidade de uso ou velocidade;
b) fatores internos são aqueles que podem ser observados apenas pelos profissionais da computação (exemplos: modularidade, manutenibilidade, etc.).

Ainda segundo Bueno e Campelo (2006), são os fatores internos que garantem a qualidade externa do *software*. E argumentam que, em qualquer processo de Engenharia, a medição é um elemento primordial. Sendo assim, é importante que utilizemos medidas para mensurar a qualidade dos produtos de engenharia ou os *softwares* que são desenvolvidos. Entretanto, Bueno e Campelo (2006) argumentam que, diferentemente de outras engenharias, a Engenharia de *Software* não é constituída por medidas absolutas ou leis quantitativas básicas.

Nas últimas décadas, a busca da garantia de qualidade vem sendo estruturada por processos que se baseiam na aplicação de boas práticas de mercado ou de modelos de qualidade utilizados com êxito em outras áreas do conhecimento (Oliveira *et al*., 2015). Kasse (2008, p. 17) preconiza que a Engenharia de *Software* tem como premissa que “a qualidade de um sistema ou produto é altamente influenciada pela qualidade do processo usado para desenvolvê-lo ou mantê-lo”. Os próximos tópicos fornecem informações sobre dois modelos de qualidade: CMMI e MPS-BR.

2.4.1 CMMI

Em 1984, foi criado o *Carnegie Melon Software Engineering Institute* (Instituto de Engenharia de *Software* Carnegie Melon), com o objetivo de desenvolver e transmitir práticas de Engenharia de *Software*, segurança de computadores e melhoria de processos, atendendo empresas de tecnologia de informação (Ataídes, 2008).

"O SEI trabalha junto a organizações de defesa e governo, indústria e universidades para melhorar continuamente seus sistemas e modelos de referência" (Ataídes, 2008, p. 18.) O alvo do SEI é apoiar as empresas a aprimorarem seus processos de Engenharia de *Software*.

Ataídes (2008) acrescenta que, em 1991, o SEI criou o *Capability Maturity Model* (CMM) - modelo de maturidade de capacidade), que pode ser entendido como um guia que auxilia na melhoria de processos das empresas, sejam quais forem as suas áreas-fim. O CMM era baseado nos princípios de gerenciamento da qualidade total de Deming, Juran e Crosby e recomendava o foco na qualidade como forma de as empresas alcançarem um nível de maturidade superior (Kasse, 2008).

O projeto *Capability Maturity Model Integration* (CMMI) teve como objetivo combinar três modelos: *Capability Maturity Model for Software* (SW-CMM), *Eletronic Industries Alliance Ínterim Standard* (EIA/IS) 731 e *Integrated Product Development Capability Maturity Model* (IPD-CMM) em um *framework* simplificado de melhoria de processos (Ataídes, 2008).

Já Chrissis *et al*. (2003) entendem o CMMI como o conjunto das melhores práticas para o desenvolvimento e a manutenção de produtos e serviços e abrange todo o ciclo de vida do produto, desde a sua concepção até a entrega e posterior manutenção. Para eles, o CMMI é "composto de corpos de conhecimento essenciais no desenvolvimento de produtos, que eram tratados em separado no passado como a Engenharia de *Software*, a engenharia de sistemas e aquisições". E desse modo provém um *framework* que compreende o desenvolvimento e a manutenção de produtos e serviços. Ahern *et al.* (2008), por sua vez, afirmam que a suíte de produtos do CMMI pode ajudar uma empresa a aperfeiçoar seus processos, por conter imensa quantidade de orientações

A legitimização dos processos de TI de uma empresa por meio de políticas, padrões e sua estrutura organizacional permite que ela ganhe maturidade (Ataídes, 2008). Segundo Ataídes (2008), O CMMI propõe três formas de se analisar os processos de TI:

a) Pela capacidade: essa análise retrata os resultados que podem ser obtidos com a utilização dos processos;
b) pelo desempenho: reflete os resultados atuais atingidos com o uso dos processos;
c) pela maturidade: exibe o quanto um processo está definido, gerenciado, mensurado, controlado e traz resultados.

Ataídes (2008) detalha que o modelo do CMMI apresenta cinco níveis de maturidade, que podem ser alcançados em estágios pelas empresas. Cada nível contém um grupo de objetivos predefinidos em áreas-chave estabelecidas pelo modelo, para que a empresa possa

atingir o próximo nível. A Figura 4 ilustra os cinco níveis de maturidade do modelo CMMI. A Tabela 2 apresenta os níveis de maturidade do CMMI e suas respectivas áreas-chave.

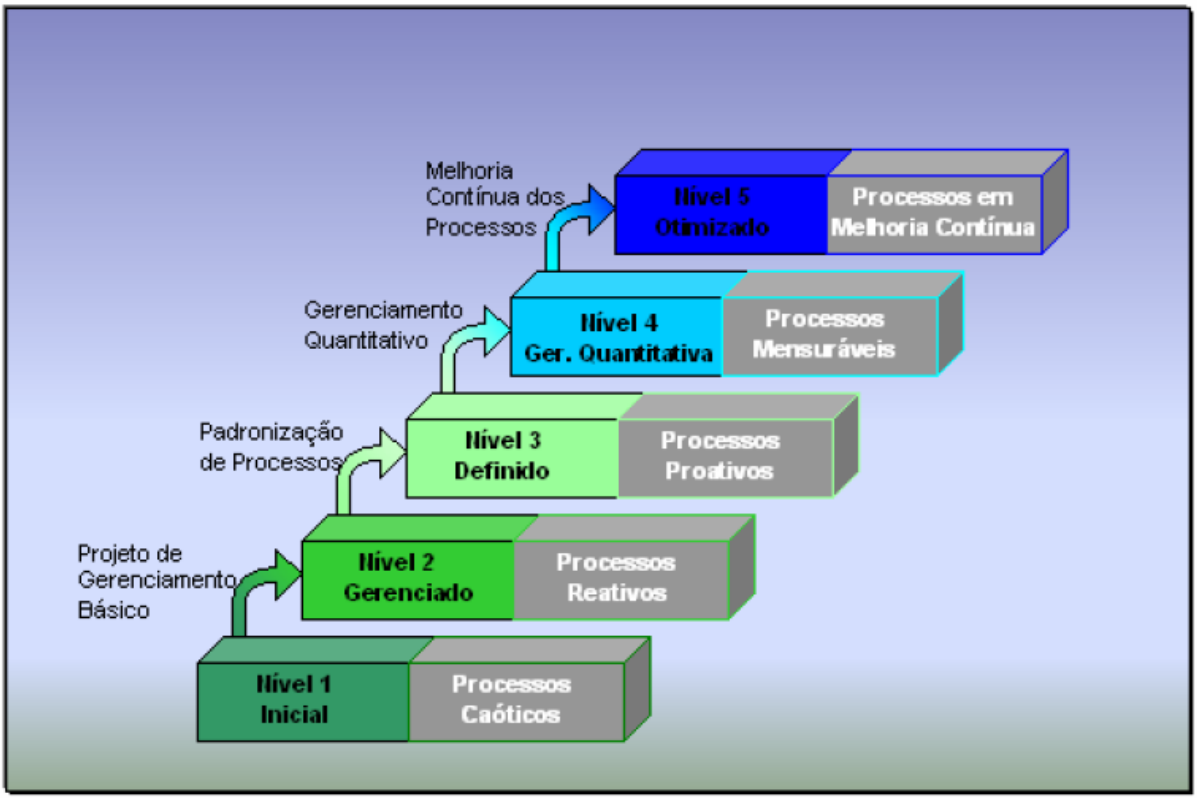

Figura 4
Níveis de maturidade do modelo CMMI.
Fonte: Ataídes, A. D. C. (2008). Um método para acompanhamento e controle da implantação do CMMI. Dissertação (Mestrado em Engenharia Elétrica) - Faculdade de Tecnologia da Universidade de Brasília.

Assim, Kasse (2008, p. 27) admite que "a proposta do CMMI é sustentar os processos de desenvolvimento de *software* e a transformação de culturas organizacionais para trazer benefícios mensuráveis para os negócios das empresas". Ainda segundo Kasse (2008, p. 44), "o CMMI provê um *framework* para estruturar e priorizar atividades de engenharia, pessoas e negócios. Ele pode servir de apoio na construção de projetos de sucesso, provendo descrições detalhadas para permitir a melhoria dos processos técnicos, gerenciais e de serviços".

Tabela 2
Níveis de maturidade

5 - Otimizado	O foco do processo é na melhoria contínua quantitativa do processo.	Inovação organizacional e implantação Análise causal e resolução
4 - Quantitativamente gerenciado	O processo é mensurado e controlado.	Desempenho de processo organizacional Gerenciamento quantitativo de projeto
3 - Definido	O processo é caracterizado pela organização e é proativo.	Desenvolvimento de requisitos Solução técnica Integração de produto Verificação Validação Foco de processo organizacional Definição de processo organizacional Treinamento organizacional Gerenciamento integrado de projeto Gerenciamento de riscos Análise de decisão e resolução
2 - Gerenciado	O processo é caracterizado por projetos e frequentemente reativo.	Gerenciamento de requisitos Planejamento de projeto Acompanhamento e controle de projeto Gerenciamento de acordo com fornecedor Medição e análise Garantia da qualidade de processo e produto Gerência de configuração
1 - Inicial	O processo é imprevisível, não controlado e reativo.	

Fonte: Oliveira, A., Petrini, M., & Pereira, D. L. (2015). Avaliação da adoção do CMMI considerando o custo de qualidade de software. *Revista de Gestão e Projetos - GeP*, *06*(01), 45–62. https://doi.org/10.5585/gep.v6i1.281.

O próximo tópico apresenta informações sobre o MPS-BR.

2.4.2 MPS-BR

O guia de Melhoria de Processo do *Software* Brasileiro (MPS-BR) (Associação para Promoção da Excelência do *Software* Brasileiro - Softex, 2016) é um programa (MPS.BR) criado pela Softex em 2003 e que se baseia nos conceitos de capacidade de processo e maturidade, cujo objetivo é a melhoria da qualidade e produtividade de *software* e serviços.

No MPS-BR (Softex, 2016), o programa MPS.BR possui uma Unidade de Execução do Programa (UEP) e duas estruturas de apoio para a execução de suas atividades, o Fórum de Credenciamento e Controle (FCC) e a Equipe Técnica do Modelo (ETM). Ainda segundo o guia do MPS-BR (Softex, 2016 p. 4-5), estas são as atribuições das duas estruturas:

a) Cabe ao FCC: (i) emitir parecer que subsidie decisão da Softex sobre o credenciamento de instituições implementadoras (II) e instituições avaliadoras (IA);

(ii) monitorar os resultados das IIs e IA)s, emitindo parecer propondo à Softex o seu descredenciamento no caso de comprometimento da credibilidade do modelo MPS.

b) Cabe à ETM (i) apoiar a SOFTEX nas questões estratégicas relacionadas ao programa MPS.BR e aos modelos MPS, com o envolvimento dos sênior advisor, (ii) tomar decisões sobre os aspectos técnicos relacionados aos Modelos MPS no que se refere à sua criação e aprimoramento contínuo; (iii) propor ações visando a capacitação de profissionais das empresas ,dos implementadores dos modelos e guias MPS e dos avaliadores MPS; (iv) apoiar a SOFTEX nas tarefas relacionadas à divulgação, disseminação e internacionalização dos Modelos MPS; (v) apoiar a SOFTEX na organização dos workshops do MPS (WAMPS).

Finalmente, é com base nessas estruturas que instituições governamentais e privadas podem colaborar e agregar valor ao programa.

Nesse cenário, o MPS apresenta quatro componentes:

a) Modelo de Referência MPS para *Software* (MR-MPS-SW);
b) Modelo de Referência MPS para Serviços (MR-MPS-SV);
c) Método de Avaliação (MA-MPS);
d) Modelo de Negócio para Melhoria de Processo de *Software* e Serviços.

Ele é detalhado a partir de guias:

a) Guia geral MPS de *Software*: descreve o MR-MPS-SW, com seus componentes e as definições comuns que permitem seu entendimento e utilização prática;
b) guia geral MPS de Serviços: descreve o MR-MPS-SV, com seus componentes e as definições comuns que permitem o seu entendimento e utilização prática;
c) guia de aquisição: especifica um processo de aquisição de *software* e serviços;
d) guia de avaliação: especifica o processo e o método de avaliação MA-MPS;
e) guia de implementação: compreende um conjunto de documentos que contém orientações para implementar nas organizações os níveis de maturidade detalhados no MR-MPS-SW.

Consta no guia do MPS-BR (Softex, 2016) que o modelo de referência MPS para *Software* estabelece níveis de maturidade, que são estágios de evolução dos processos nas

empresas, possibilitando a previsão da *performance* de execução dos processos. Os níveis de maturidade são uma associação entre processos e suas capacidades respectivas. Cada processo contém um propósito que detalha seu objetivo geral e "os resultados a serem obtidos com a efetiva implementação do processo". A capacidade do processo é composta de atributos do processo, que devem ser atendidos com base nos resultados esperados dos atributos do processo.

O MR-MPS-SW apresenta sete níveis de maturidade, como consta no guia do MPS-BR (Softex, 2016):

a) A (em otimização);
b) B (gerenciado quantitativamente);
c) C (definido);
d) D (largamente definido);
e) E (parcialmente definido);
f) F (gerenciado);
g) G (parcialmente gerenciado).

Esses níveis são acumulativos. Assim, se uma empresa está no nível F, ela detém os atributos de processo dos níveis G e F, e assim sucessivamente. Além disso, quando a empresa avança para um nível de maturidade superior, os processos deverão ser realizados com o nível de exigência das capacidades desse nível.

A seguir, a descrição dos oito níveis:

a) Nível G – parcialmente gerenciado

O nível de maturidade G do guia MPS-BR (Softex, 2016) contém os processos de Gerência de Projetos e de Gerência de Requisitos.

No processo de Gerência de Projetos, o objetivo é desenvolver os planos de atividades, recursos e responsabilidades, além de executar correções dos desvios em relação ao planejamento do projeto (Softex, 2016).

Na Gerência de Requisitos, a proposta é controlar e perceber inconsistências nos requisitos do produto (Softex, 2016).

b) Nível F – gerenciado

O nível maturidade F contém os processos do nível G, somados aos processos de aquisição, garantia da qualidade, gerência de configuração, gerência de portfólio de projetos e medição (Guia do MPS-BR) (Softex, 2016).

O objetivo do processo de aquisição é controlar a aquisição de produtos que atendam às necessidades da empresa. O processo de Gerência de Configuração tem como intuito controlar a integridade dos produtos e disponibilizá-los aos envolvidos. A finalidade do processo de Garantia da Qualidade é certificar que os produtos entregues se encontrem em conformidade com o planejamento e com os padrões definidos. O processo de Gerência de Portfólio de Projetos tem o fito de promover e sustentar os projetos realmente necessários, que cumpram os objetivos estratégicos da empresa. E o objetivo do processo de Gerência de Medição é levantar, analisar e apresentar os dados referentes aos produtos desenvolvidos na empresa, para sustentar os seus objetivos (Softex, 2016).

c) Nível E – parcialmente definido

Segundo o guia do MPS-BR (Softex, 2016), o nível maturidade E contém os processos dos níveis F e G, somados aos processos de avaliação e melhoria do processo organizacional, definição do processo organizacional, gerência de recursos humanos e gerência de reutilização. Além disso, o processo de Gerência de Projetos tem um novo objetivo: a gerência será baseada no processo estabelecido e em planos integrados.

O objetivo do processo de avaliação e melhoria do processo organizacional é avaliar os processos da empresa em relação aos seus objetivos de negócio e ajudá-la a executar e implantar melhorias contínuas. O processo de definição do processo organizacional almeja determinar e manter ativos de processo organizacionais apropriados ao negócio da empresa. O objetivo do processo de Gerência de Recursos Humanos é garantir os recursos humanos e suas respectivas competências, necessários aos projetos e ao negócio da empresa. E o processo de gerência de reutilização busca gerenciar o ciclo de vida dos ativos reutilizáveis (Softex, 2016).

d) Nível D – largamente definido

O nível maturidade D contém os processos dos níveis anteriores (G, F e E), somados aos processos de desenvolvimento de requisitos, integração do produto, projeto e construção do produto, validação e verificação, registrados no guia do MPS-BR (Softex, 2016).

Os objetivos dos processos, de acordo com o Softex (2016), são:

a) Desenvolvimento de requisitos: definir os requisitos do cliente, do produto e dos componentes do produto;
b) processo integração do produto: estruturar os componentes do produto de forma a deixá-lo coerente com o seu projeto e seus requisitos;
c) processo projeto e construção do produto: projetar, desenvolver e implementar soluções para atender aos requisitos;
d) processo de validação: comprovar que o produto entregue vai ter a serventia desejada, quando instalado;
e) processo verificação: atestar que o produto do projeto satisfaz os requisitos especificados.

e) Nível C – definido

No guia do MPS-BR (Softex, 2016), o nível maturidade C contém os processos dos níveis anteriores (G ao D), somados aos processos de desenvolvimento para reutilização, Gerência de Decisões e Gerência de Riscos.

O objetivo do processo desenvolvimento para reutilização é identificar ativos com possibilidades de reutilização e, se possível, criar um programa de reutilização, na empresa. O processo Gerência de Decisões mira utilizar um processo formal, de critérios predefinidos, para a tomada de decisões. E o intuito do processo Gerência de Riscos é identificar, analisar, tratar, monitorar e reduzir continuamente os riscos em nível organizacional e de projeto (Softex, 2016).

f) Nível B – gerenciado quantitativamente

De acordo com o guia do MPS-BR (Softex, 2016), o nível maturidade B contém os mesmos processos dos níveis anteriores (G ao C), mas são acrescidos resultados a estes

processos, para permitir o seu gerenciamento de forma quantitativa, satisfazendo requisitos de análise de desempenho.

g) Nível A – em otimização

O nível maturidade A contém os mesmos processos dos níveis anteriores (G ao C), como mostra o MPS-BR. Nesse nível, entretanto, a empresa deve ser capaz de identificar necessidades de mudança "a partir da análise de defeitos, problemas, causas comuns de variação do desempenho e da investigação de enfoques inovadores para a definição e implementação do processo" (Softex, 2016, p. 20). Além disso, a empresa deve conseguir avaliar se as mudanças foram efetivas para atender aos objetivos de melhoria do processo.

Após a apresentação de conceitos do constructo dos modelos de qualidade de *software*, o tópico 2.5 discorre um pouco sobre o relacionamento dos métodos ágeis e qualidade de *software*.

2.5 Relacionamento entre métodos ágeis e qualidade de *software*

Para embasar o relacionamento entre métodos ágeis e qualidade de *software*, foram selecionados artigos sobre os temas “*agile*”, “*scrum*” “*software development quality*”, “*software quality*”, “*software development*" e "*quality*", pesquisados nos seus títulos e resumos, nas bases IEEE, *Science Direct* e *Google* acadêmico. Como critérios de exclusão foram eliminados da pesquisa livros, trabalhos de conclusão de curso, dissertações e teses e artigos que continham apenas um dos termos citados.

Após a seleção citada anteriormente, foram encontrados 1.015 artigos, sendo 242 da base do *Google* acadêmico e 772 da *Science Direct* e um artigo do IEEE, que atenderam ao protocolo da pesquisa. Depois de leitura e análise mais apurada, 235 artigos do *Google* acadêmico, todos os artigos da *Science Direct* e o artigo da IEEE foram descartados, por não estarem realmente relacionados à pesquisa, embora contivessem os termos pesquisados. Sendo assim, foram selecionados quatro artigos do *Google* acadêmico para a pesquisa. Somados a eles, foram adicionados mais cinco arquivos obtidos da referência bibliográfica de alguns artigos e de pesquisas menos estruturadas.

Tabela 3
Trabalhos sobre o relac. entre métodos ágeis e modelos de qualidade de *software*

Artigo	Base
Jia, R., & Reich, B. (2008). IT service climate: The validation of an antecedent construct for it service quality. *Anais do International Conference on Information Systems* (*ICIS*).	Outras pesquisas
Prabhakar, G. P. (2008). What is project success: A literature review. *International Journal of Business and Management, 3*, 3–10.	Outras pesquisas
Acuña, S. T., Gómez, M., & Juristo, N. (2009). How do personality, team processes and task characteristics relate to job satisfaction and software quality? *Information and Software Technology, 51*(3), 627–639. https://doi.org/10.1016/j.infsof.2008.08.006.	Outras pesquisas
Ullah, M. I., & Zaidi, W. A. (2009). *Quality assurance activities in agile*. Retrieved from: http://www.cin.ufpe.br/~scls/Claudia/QA_Activities_in_Agile.pdf.	*Google* acadêmico
Imreh, R., & Raisinghani, M. (2011). Impact of agile software development on quality within information technology organizations. *Journal of Emerging Trends in Computing and Information Sciences, 2*(10), 460–475.	*Google* acadêmico
Softex (2016). Ministério da Previdência Social - MPS.BR - *Melhoria de processo do software brasileiro guia geral*. Brasília: Softex, 2016.	Outras pesquisas
Koka, A. (2015). Software quality assurance in scrum projects: A case study of development processes among scrum teams in South Africa. Retrieved from: http://digitalknowledge.cput.ac.za/jspui/handle/11189/3194.	*Google* acadêmico
Sagheer, M., Zafar, T., & Sirshar, M. (2015). A framework for software quality assurance using agile methodology. *International Journal of Scientific & Technology Research, 4*(2), 44–50.	*Google* acadêmico
Oliveira, A., Petrini, M., & Pereira, D. L. (2015). Avaliação da adoção do CMMI considerando o custo de qualidade de software. *Revista de Gestão e Projetos - GeP, 06*(01), 45–62. https://doi.org/10.5585/gep.v6i1.281.	*Google* Acadêmico
Lowry, P. B., & Wilson, D. W. (2016). Creating agile organizations through IT: The influence of Internal IT service perceptions on IT service quality and ITagility. *Journal of Strategic Information Systems* (*JSIS*).	Outras pesquisas

Fonte: elaborada pelo autor.

Embora existam mais trabalhos que no relacionamento entre métodos ágeis e gestão do conhecimento, a literatura que relaciona a métodos ágeis e a qualidade de *software* também é escassa.

A seleção de artigos que embasou o relacionamento entre métodos ágeis e qualidade de *software* trouxe conclusões importantes dos seus respectivos autores sobre essa relação. A qualidade de *software* tem, como alicerce, a maturidade dos processos do seu desenvolvimento. Assim, o s*oftware* desenvolvido por equipes de tecnologia da informação que têm como principal objetivo a satisfação das necessidades de seus clientes e da empresa e que recebem a correta valorização por suas entregas propiciam o desenvolvimento de *softwares* de melhor qualidade. Ainda na visão desses autores, a utilização dos métodos ágeis deve garantir a constituição de equipes participativas e comunicativas e com práticas de testes adequadas, para também permitir a geração de *software* de qualidade.

Prabhakar (2008) reconhece que a qualidade do *software* gerado está relacionada a pontos como especificação e implementação dos objetivos funcionais e o desempenho técnico

da equipe. Nesse contexto, Lowry & Wilson (2016) afirmam que departamentos de TI, com objetivos diferentes desses (como, por exemplo, a implementação de novas tecnologias), tendem a produzir serviço de pior qualidade. Jia & Reich (2008) admitem que quando a equipe de TI entende que sua função é satisfazer as necessidades da empresa, certamente vai entregar serviços de qualidade e mais alinhados aos seus objetivos.

Lowry & Wilson (2016) argumentam que a correta valorização das entregas das equipes de TI é um dos fatores de sua motivação e geração de um clima de trabalho favorável. O autor argumenta que um clima favorável permite a formação de um ambiente que permite flexibilidade e eficácia do departamento de TI que, dessa forma, garantirá entregas de qualidade, alinhadas às necessidades das empresas, cada vez mais inseridas em mercados que carecem de agilidade.

Em estudo feito com o método ágil XP, Acuña *et al.* (2009) mostram que as equipes que o utilizam devem ser fortemente participativas e comunicativas, para obterem um *software* de qualidade. Os autores concluem que a extroversão dos membros da equipe está substancialmente relacionada à melhor qualidade de *softwares* desenvolvidos, utilizando-se uma metodologia ágil. Imreh & Raisinghani (2011) argumentam que uma implantação de sucesso dos métodos ágeis depende de uma equipe fortemente colaborativa, experiente e localizada em uma mesma área geográfica ou escritório.

Ullah & Zaidi (2009) sugerem que o teste é a principal atividade para a qualidade do produto de *software*. Adicionado a ele, práticas como a programação em pares e o desenvolvimento orientado a testes podem ser eficazes para se minimizar erros. Os achados de Koka (2015) corroboram essa afirmação e indicam que as três atividades mais significativas para a garantia da qualidade de *software*, a partir da adoção do *scrum*, são: as revisões de código, o teste unitário e o teste de aceitação do usuário.

Os métodos ágeis fornecem um *software* de qualidade, desenvolvido em menos tempo, aumentando a satisfação do cliente. Isso ocorre por causa da maior cooperação dos membros da equipe de desenvolvimento e da comunicação contínua com o cliente. Ao longo do desenvolvimento novos recursos vão sendo adicionados ao *software*, conforme as necessidades do cliente, reduzindo tempo e custo e auxiliando na qualidade do *software* gerado (Sagheer *et al*., 2015).

Assim, Koka (2015) acredita que é importante que as equipes *scrum* e, em especial, os líderes de projeto garantam a execução das atividades dessa metodologia, independentemente do tempo disponível para o desenvolvimento do *software* ou do tamanho do projeto. Koka (2015) recomenda que as atividades do *scrum* sejam sempre aplicadas, para garantir que os

problemas sejam descobertos no início do desenvolvimento, quando é menos custosa a descoberta de defeitos.

A cultura empresarial não deve ser ignorada pelas empresas de TI, para garantir um ambiente ágil, eficiente e que gera *software* de qualidade (Imreh & Raisinghani, 2011). Para o manual da Softex (2016), as mudanças ocorridas nos ambientes de negócio empresariais, por sua vez, têm provocado modificações em seus processos e ambientes produtivos.

Para Imreh & Raisinghani (2011), os métodos ágeis têm forte impacto na qualidade de *software* e dependem de implementação de mudanças organizacionais e da melhoria de processos para obter sucesso. Assim, qualidade de *software* gerado com equipes ágeis está relacionada a uma apropriada implantação dos métodos ágeis nas empresas.

Obviamente, a qualidade é um fator crítico de sucesso para vários setores e isso não é diferente para a indústria de *software.* Para terem um *software* competitivo, as empresas precisam investir na eficiência dos seus processos (Softex, 2016). Oliveira *et al.* referem que essa melhoria de processos está sendo baseada nos modelos de qualidade, com o objetivo de agregar maturidade em processos de produção. "Ao utilizar um modelo, as empresas podem modificar ou criar processos baseadas em práticas que comprovadamente melhoraram a capacidade dos processos e, assim, atingir maturidade" (Oliveira *et al.*, 2015).

Após a apresentação do relacionamento entre métodos ágeis e modelos de qualidade de *software*, o tópico 2.6 aborda sobre o relacionamento de gestão de conhecimento e os modelos de qualidade de *software*.

2.6 Relacionamento entre gestão do conhecimento e qualidade de *software*

Para embasar o relacionamento entre Gestão do conhecimento e qualidade de *software,* foram selecionados artigos sobre os temas "*knowledge management*", "*software development quality*", visto que o *string* de busca "*software quality*" retornou um número muito elevado de documentos, a maioria deles relacionada à pesquisa de *softwares* para atender à gestão do conhecimento, o que não é foco da pesquisa. Os temas foram pesquisados nos seus títulos e resumos, nas bases IEEE, *Science Direct* e *Google* acadêmico. Foram eliminados da pesquisa livros, trabalhos de conclusão de curso, dissertações, teses e artigos que continham apenas um dos termos citados.

Após a seleção citada anteriormente, foram encontrados 181 artigos, sendo 175 da base do *Google* acadêmico e seis da *Science Direct* e não foi encontrado artigo algum do IEEE que atendesse ao protocolo da pesquisa. Depois da leitura e análise mais apurada dos

artigos, 173 artigos do *Google* acadêmico e os seis da *Science Direct* foram descartados, por não estarem realmente relacionados à pesquisa, embora contivessem os termos pesquisados. Sendo assim, foram selecionados dois artigos para a pesquisa. Somados a eles, foram adicionados mais cinco arquivos obtidos da referência bibliográfica de alguns artigos e de pesquisas menos estruturadas.

A Tabela 4 mostra os artigos selecionados para a pesquisa.

Tabela 4
Trabalhos sobre o relacionamento entre métodos ágeis e modelos de qualidade

Artigo	Base
Arent, J., & Norbjerg, J. (2000). Software process improvement as organizational knowledge creation: a multiple case analysis. System Sciences, 2000. *Proceedings of the 33d Hawaii International Conference., 00*(c), 1–11. Retrieved from: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=926775.	Outras pesquisas
Aurum, A., Ross, J., Wohlin, C., & Meliha, H. (2003). *Managing software engineering knowledge*. https://doi.org/10.1007/978-3-662-05129-0.	Outras pesquisas
Salo, O. (2005). Systematical validation of learning in agile software development environment. *Biennial Conference on Professional Knowledge Management/ Wissensmanagement*. Berlin, Heidelberg: Springer, 106–110.	Outras pesquisas
Al-shehab, A. J., Hughes, R. T., & Winstanley, G. (2005). Facilitating organisational learning through causal mapping techniques in IS/ IT Project Risk Management (October 2004, p. 145–154). Springer Verlag, Kaiserslautern, Germany (p. 145–154).	Outras pesquisas
Lee, M.C., & Chang, T. (2006). Applying TQM, CMM and ISO 9001 in knowledge management for software development process improvement. *International Journal of Services and Standards, 2*(1), 101–115. https://doi.org/10.1504/IJSS.2006.008161.	*Google* acadêmico
Bjørnson, F. O., & Dingsøyr, T. (2008). Knowledge management in software engineering: A systematic review of studied concepts, findings and research methods used. *Information and Software Technology, 50*(11), 1055–1068. https://doi.org/10.1016/j.infsof.2008.03.006.	Outras pesquisas
Shang, S. S. C., & Lin, S.-F. (2009). Understanding the effectiveness of capability maturity model integration by examining the knowledge management of software development processes. *Total Quality Management & Business Excellence, 20*(5), 509–521. https://doi.org/10.1080/14783360902863671.	*Google* acadêmico

Fonte: elaborada pelo autor.

Conforme apresentado na Tabela 4, é escassa a literatura que relaciona a gestão do conhecimento e a qualidade de *software*. Entretanto, existem, também, trabalhos relacionando a gestão do conhecimento à Engenharia de *Software*.

A seleção de artigos que deu suporte ao relacionamento entre gestão do conhecimento e qualidade de *software* trouxe conclusões muito pertinentes dos seus respectivos autores sobre essa relação. Entre elas, a de que a qualidade do *software* gerado está relacionada à qualidade dos entregáveis que são produzidos ao longo do processo de desenvolvimento de *software*. A gestão do conhecimento pode estar presente em todo esse processo, visto que

define e sustenta as suas atividades, além de aprimorar sua a comunicação. Mas tudo isso só trará ganhos palpáveis às empresas se elas conseguirem transformar esse conhecimento tácito em conhecimento organizacional, visto que o maior ativo das empresas de TI, hoje, está no conhecimento de seus funcionários e esse conhecimento, muitas vezes, passa a ficar embutido em produtos e processos.

Shang & Lin (2009, p. 512) apregoam que:

> É amplamente aceito que o conhecimento organizacional está embutido em produtos ou processos, à medida que as empresas criam, armazenam e analisam dados e conhecimento sobre oportunidades de mercado e desenvolvimento tecnológico e as usam para desenvolver bens e procedimentos apropriados.

Para Bjørnson & Dingsøyr (2008), a Engenharia de *Software* é uma atividade que depende muito do conhecimento. Assim, o maior ativo das empresas de TI é o conhecimento de seus funcionários e não suas máquinas ou fábricas de *software*. Lee & Chang (2006) advertem que os principais problemas do desenvolvimento de *software* estão relacionados à qualidade do produto gerado e à produtividade das equipes de desenvolvimento.

O CMMI tem sido citado como a melhor prática para aprimorar a qualidade do desenvolvimento de *software* (Shang & Lin, 2009) Embora tanto o CMMI quanto a gestão do conhecimento apresentem o aumento da eficácia organizacional como um dos seus objetivos, não existem muitas pesquisas que avaliam o valor do conhecimento no CMMI. Mesmo assim, Shang & Lin (2009) asseguram que existe uma forte ligação entre a gestão do conhecimento e o CMMI, dessa forma, é possível avaliar a eficácia do primeiro em relação ao segundo. A qualidade do *software* gerado está relacionada aos entregáveis apresentados durante o processo desenvolvimento de *software*, entre eles precisão funcional, eficiência, manutenção e integridade da documentação.

Nesse sentido, Bjørnson & Dingsøyr (2008) lembram que a indústria de desenvolvimento de *software* vem percebendo a utilidade da aplicação dos princípios da gestão do conhecimento na Engenharia de *Software*. Lee & Chang (2006) comentam que a gestão do conhecimento pode ser utilizada em várias atividades relacionadas ao desenvolvimento de *software*, entre elas, definição de processos de *software*, alocação de recursos humanos, estimativa, análise de requisitos, planejamento de qualidade, entre outros. Por meio da gestão do conhecimento, o conhecimento gerado com o desenvolvimento de *software* pode ser capturado, armazenado, disseminado e reutilizado, com o intuito de melhorar a qualidade e a produtividade. Assim, a gestão do conhecimento pode dar suporte à

melhoria contínua do processo de desenvolvimento de *software* e, assim, de seus produtos. Bjørnson & Dingsøyr (2008), por sua vez, complementam que disciplinas da gestão do conhecimento, como a ciência cognitiva, ergonomia e gestão, podem contribuir muito para a administração desse conhecimento.

Arent & Nørbjerg (2000) propõem modelos teóricos que unem a gestão do conhecimento, o capital intelectual e o gerenciamento de projetos. E salientam que a gestão do conhecimento tem grande importância na gestão de projetos de *softwares* complexos, principalmente nas negociações e na solução de problemas inerentes a essa atividade. Aurum *et al*. (2003) asseveram que a gestão conhecimento está presente na melhoria de *software* e processos, na medida em que dá suporte ao processo de desenvolvimento de *software*, com base na definição de atividades e na orientação para sua execução, bem como mediante o aperfeiçoamento da comunicação do projeto. Esses autores defendem que as atividades devem ser acompanhadas no decorrer de todo o ciclo de desenvolvimento do *software* para assegurar a entrega de um produto de qualidade. "O objetivo é identificar e eliminar defeitos o mais cedo possível ao longo do ciclo de vida, reduzindo assim os custos de teste e manutenção" (Aurum *et al*., 2003, p. 255).

Na mesma linha, Birk (2000, citado por Bjørnson & Dingsøyr, 2008) entende que, mais que abranger apenas o desenvolvimento de *software*, a Engenharia de *Software* deve disponibilizar ferramentas para dar suporte às atividades cooperativas, gerenciamento de conflitos, comunicação e gerenciamento do projeto. E Salo (2005) comenta que o conhecimento e o raciocínio adquiridos em melhorias de processo só serão efetivos para as empresas se tornarem conhecimento explícito, contribuindo para o aprendizado organizacional. Finalmente, Bjørnson & Dingsøyr (2008) explicitam que a Engenharia de *Software* é muito ampla e apresenta, entre suas disciplinas mais relevantes, a gestão do conhecimento e a melhoria de processos de *software*.

Por outro lado, segundo Al-Shehab *et al.* (2005), apesar de a Engenharia de *Software* abranger várias disciplinas e diversos atores, os modelos de Engenharia de *Software* não dão a devida importância à cooperação e à gestão do conhecimento. Finalmente, o estudo de Shang & Lin (2009) mostrou que, mesmo com a implantação do CMMI, a criação do conhecimento não conseguiu incorporar o conhecimento dos clientes, ficando limitado ao conhecimento interno.

Após a apresentação do relacionamento entre Métodos ágeis e modelos de qualidade de *software*, o tópico 2.7 apresenta o modelo conceitual proposto para a pesquisa.

2.7 Modelo conceitual proposto

Alguns autores afirmam que a maior parcela do conhecimento empregada no desenvolvimento de *software*, utilizando-se métodos ágeis, é tácito e está em poder da equipe de desenvolvimento (Gandomani *et al*., 2013; Singh *et al*., 2014; Yanzer Cabral *et al*., 2014). A esse respeito, Gandomani *et al*. (2013) ressaltam a importância de uma estratégia adequada da gestão do conhecimento e sua disseminação na empresa. Desse modo, o modelo conceitual proposto incorpora a influência dos métodos ágeis na gestão do conhecimento.

Acuña *et al*. (2009) afirmam que a participação e a comunicação das equipes que trabalham com métodos ágeis é primordial para o desenvolvimento de um *software* de qualidade. Para atestar a influência dos métodos ágeis na qualidade de *software*, o modelo conceitual proposto incorpora essa relação.

Como conceituam Aurum *et al*. (2003) e Lee & Chang (2006), a gestão do conhecimento dá suporte ao processo de desenvolvimento de *software*. Lee & Chang (2006) acrescentam que o conhecimento gerado com desenvolvimento de *software* pode ser incorporado ao conhecimento organizacional, para melhorar qualidade e produtividade. Assim, o modelo conceitual proposto incorpora, também, a influência da gestão do conhecimento na qualidade de *software*.

A Figura 5 apresenta o modelo conceitual proposto para a pesquisa.

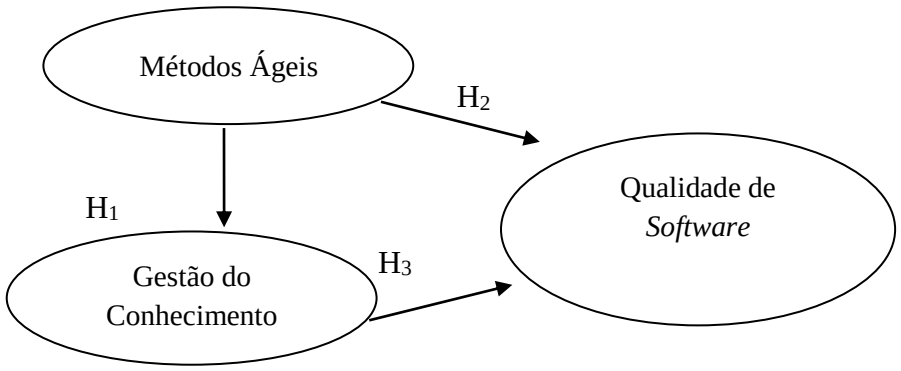

Figura 5
Modelo conceitual proposto.
Fonte: elaborada pelo autor.

Desse modelo conceitual foram geradas as seguintes hipóteses de pesquisa:

a) H_1: os métodos ágeis influenciam positivamente a gestão do conhecimento;
b) H_2: os métodos ágeis influenciam positivamente a qualidade de *software*;

c) H_3: a gestão do conhecimento influencia positivamente a qualidade de *software*.

O próximo capítulo descreve a metodologia de estudo prevista para a dissertação.

3 Metodologia

A pesquisa realizada para atender a este estudo foi exploratória, de natureza aplicada e abordagem quantitativa, operacionalizada a partir de um *survey* (levantamento).

Em relação à natureza, Gil (2002) entende que uma pesquisa de natureza aplicada é desenvolvida para trazer conhecimento, com o objetivo de se executar algo de modo mais eficaz. E aduz que as pesquisas exploratórias têm o objetivo de expor ideias, com o intuito de gerar hipóteses que, por sua vez, podem ser testadas em investigações futuras. Se, por um lado, essas pesquisas podem ser baseadas em um planejamento mais flexível, para se obter observações empíricas ou determinar as relações entre os fenômenos estudados elas devem ser orientadas por procedimentos com um nível razoável de sistematização.

Hair Jr., Black, Babin, Anderson & Tatham (2009) mostram que a análise multivariada compreende várias técnicas de análise de dados e abrange amplo domínio de pesquisa. Ainda indicam que a modelagem de equações estruturais (SEM), em especial, pode ser entendida como um conjunto de modelos estatísticos que tem o objetivo de explicar relações entre múltiplas variáveis. "Fazendo isso, ela examina a estrutura de inter-relações expressas em uma série de equações, semelhante a uma série de equações de regressão múltipla. Tais equações descrevem todas as relações entre constructos (as variáveis dependentes e independentes) envolvidos na análise" (Hair Jr. *et al*., 2009, p. 543).

Para estruturar este estudo, o primeiro passo foi a investigação de questionários relacionados ao tema da dissertação. Como não foram encontrados questionários que atendessem à pesquisa, foi criado um novo questionário. O questionário proposto foi baseado na escala de verificação Likert que, segundo Silva Júnior e Costa (2014), é criada a partir de um conjunto de questões relativas a um constructo, para as quais os respondentes manifestam seu grau de concordância. A partir daí se deduz a medida do constructo.

O questionário foi codificado por meio da ferramenta *Google forms* e, então, validado por especialistas que sugeriram as adaptações e correções necessárias. Segundo Gil (2002), no pré-teste o questionário criado é validado por especialistas experientes que conhecem as peculiaridades da pesquisa e avaliam o seu grau de dificuldade de escolha das opções de resposta. A partir daí são feitas as adaptações necessárias. Entre os especialistas estavam profissionais com vasta experiência em projetos de desenvolvimento de sistemas: um mestre e doutorando em Sistemas de Informação e Gestão do Conhecimento, um mestre em Sistemas de Informação e Gestão do Conhecimento, uma gerente de projetos de TI muito experiente e um estatístico.

O questionário foi, então, enviado em meio digital para os respondentes. O perfil dos respondentes escolhidos para a pesquisa foi o de pessoas envolvidas no desenvolvimento de *software* em empresas de TI, como desenvolvedores, analistas de requisitos/negócio, analistas de testes, gerentes de projeto e gestores.

Hair Jr. *et al*. (2009, p. 40) sublinha que, de posse dos dados, o pesquisador estima o modelo e lhe faz o ajuste geral. "No processo de estimação, o pesquisador pode escolher entre opções para atender a características específicas dos dados (p. ex., uso de covariáveis estatísticas em análise multivariada da variância - MANOVA) ou maximizar o ajuste dos dados (p. ex., rotação de fatores ou funções discriminantes)". Depois da estimação, o pesquisador analisará se será necessário ajuste geral dos dados, para garantir que eles atinjam níveis aceitáveis em relação a critérios estatísticos (p. ex., o nível de significância).

Após o ajuste do modelo feito no passo anterior, fez-e a interpretação das variáveis estatísticas, para "identificar evidência empírica de relações multivariadas nos dados da amostra que possam ser generalizadas para a população total" (Hair Jr. *et al*., 2009, p. 40).

Finalmente, Hair Jr. *et al*. (2009, p. 40) consideram que o pesquisador deve fazer análises diagnósticas nos resultados obtidos, para garantir a sua generalidade. "Essas análises diagnósticas acrescentam pouco à interpretação dos resultados, mas podem ser vistas como uma “garantia” de que os resultados são os melhores descritivos dos dados e ainda generalizáveis à população".

Assim, após o recebimento dos questionários, os dados foram tabulados, estimados, interpretados e validados, conforme descrito anteriormente. Para isso, foi utilizado o *software* R, versão 3.5.1. Seguidos todos esses passos, os dados foram analisados.

3.1 Modelo conceitual e siglas dos indicadores do modelo

Conforme descrito anteriormente, seguem-se o modelo conceitual do projeto e a tabela com a relação das questões, com seus constructos, siglas e o referencial teórico que lhes deu suporte.

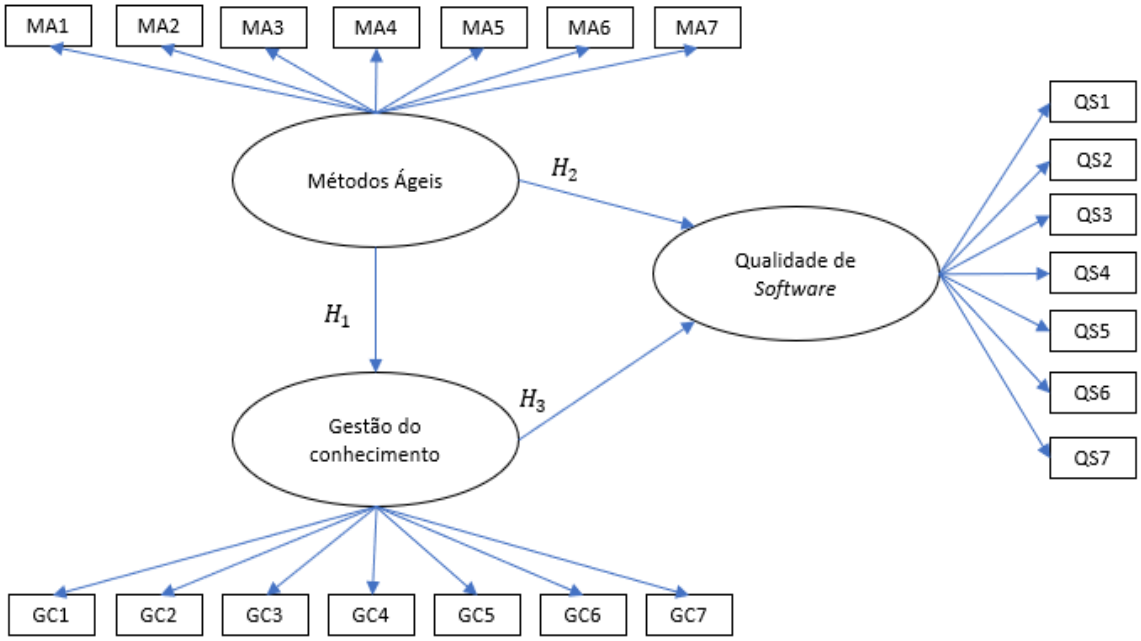

Figura 6
Modelo conceitual e as siglas do questionário.
Fonte: elaborada pelo autor.

Tabela 5
Relação das siglas por item

Const.	Sigla	Questão	Referencial Teórico
Métodos ágeis	MA1	A interação entre colaboradores no meu ambiente de trabalho favorece a obtenção de melhores resultados no desenvolvimento de *software*.	Gomes, Willi e Rehem (2014)
	MA2	A equipe de desenvolvimento com a qual trabalho se compromete com o cumprimento dos prazos previamente estabelecidos, para as entregas do *software* acordado.	Cruz (2013)
	MA3	O cliente participa ativamente dos processos de desenvolvimento de *software* nos quais estou envolvido.	Gomes *et al.* (2014)
	MA4	O processo de desenvolvimento de *software* do meu ambiente de trabalho dá suporte a solicitações de mudanças de escopo de *software*, por parte do cliente.	Beck (2000)
	MA5	O processo de desenvolvimento de *software* do meu ambiente de trabalho garante entregas parciais (versões) do *software* acordado.	Cruz (2013) Beck (2000)
	MA6	O *software* entregue pela equipe de desenvolvimento com a qual trabalho atende plenamente aos requisitos do cliente.	Fadel e Silveira (2010) e Bernardo e Kon (2008)
	MA7	Os líderes de desenvolvimento de *software* de meu ambiente de trabalho atuam mais como facilitadores.	Gome *et al.* (2014)
Gestão do conhecimento	GC1	O meu ambiente de trabalho valoriza as entregas dos profissionais envolvidos no desenvolvimento de *software*.	Lowry & Wilson (2016)
	GC2	O desenvolvimento de *software* do meu ambiente de trabalho conta mais com o conhecimento dos envolvidos do que com a documentação de metodologias, processos e práticas documentadas aplicáveis.	Gandomani et all (2013) Bjørnson & Dingsøyr (2008)
	GC3	Em meu ambiente de trabalho há incentivo para a troca de informações entre os envolvidos no desenvolvimento dos *softwares*.	Gandomani *et al.* (2013)
	GC4	O processo de desenvolvimento de *software* do meu ambiente de trabalho se preocupa em disseminar as lições aprendidas entre os diferentes projetos de *software*.	Choo (2006)
	GC5	O processo de desenvolvimento de *software* do meu ambiente de trabalho adota técnicas para incentivar a criatividade de seus profissionais.	Korobinski (2001)
	GC6	O meu ambiente de trabalho adota estratégias para institucionalizar as novas práticas que deram certo, para execução de atividades.	Patriotta (2003) Scorsolini-Comin et. all, (2011)
	GC7	Na minha organização há clara percepção de que a gestão do conhecimento pode propiciar a obtenção de vantagens competitivas.	Pandey (2016) Cavalcante (2000)
Qualidade de *software*	QS1	O processo de desenvolvimento de *software* do meu ambiente de trabalho contempla o acompanhamento das atividades, dos recursos e das responsabilidades das pessoas.	Softex, 2016
	QS2	Os requisitos dos *softwares* desenvolvidos no meu ambiente de trabalho são controlados.	Softex, 2016
	QS3	O meu ambiente de trabalho apresenta um processo de controle de qualidade do *software* desenvolvido.	Softex, 2016
	QS4	O processo de desenvolvimento de *software* do meu ambiente de trabalho estabelece que o desempenho das atividades seja medido.	Softex, 2016
	QS5	O processo de desenvolvimento de *software* do meu ambiente de trabalho apresenta uma atividade de gerência de configuração, garantindo que os *softwares* desenvolvidos serão instalados adequadamente no ambiente dos clientes.	Softex, 2016
	QS6	No meu ambiente de trabalho existe um processo de gerência de riscos de projetos.	Softex, 2016
	QS7	No meu ambiente de trabalho são identificadas necessidades de mudança nos processos, a partir da análise de defeitos e problemas de desempenho dos projetos de desenvolvimento de *software*.	Softex, 2016

Fonte: elaborada pelo autor.

3.2 Estratégia da pesquisa x tratamento dos dados de campo

O perfil dos entrevistados foi descrito por meio do cálculo das frequências absolutas e relativas. Os itens de cada constructo foram relacionados utilizando-se o intervalo percentílico de *bootstrap* de 95% de confiança que, segundo Efron & Tibshirani (1993), é muito utilizado na realização de inferências, em que não se conhece a distribuição de probabilidade da variável de interesse. As questões foram respondidas com base na escala Likert, onde 1 representou "discordo totalmente" e, de outro lado, 5 representou "concordo totalmente".

De acordo com Hair Jr, Hult, Ringle & Sarstedt (2016), o tamanho mínimo de entrevistados para a pesquisa é apresentado na Tabela 6 em relação ao maior número de setas que chegam a um constructo. No caso da pesquisa em questão, para um nível de significância de 5% são necessárias no mínimo 110 respostas, para o nível de 0,10 da medida de qualidade de ajuste do modelo estrutural - R^2 (0,10), considerando que o maior número de setas chega ao constructo qualidade de *software* (duas setas).

Tabela 6
Recomendação de tamanho de amostra para PLS-SEM

Número máximo de setas que apontam para o constructo	**Nível de Significância**											
	1%				**5%**				**10%**			
	R^2 Mínimo				**R^2 Mínimo**				**R^2 Mínimo**			
	0,10	0,25	0,50	0,75	0,10	0,25	0,50	0,75	0,10	0,25	0,50	0,75
2	158	75	47	38	110	52	33	26	88	41	26	21
3	176	84	53	42	124	59	38	30	100	48	30	25
4	191	91	58	46	137	65	42	33	111	53	34	27
5	205	98	62	50	147	70	45	36	120	58	37	30
6	217	103	66	53	157	75	48	39	128	62	40	32
7	228	109	69	56	166	80	51	41	136	66	42	35
8	238	114	73	59	174	84	54	44	143	69	45	37
9	247	119	76	62	181	88	57	46	150	73	47	39
10	256	123	79	64	189	91	59	48	156	76	49	41

Fonte: Hair Jr, J. F., Hult, G. T. M., Ringle, C., & Sarstedt, M. (2016). *A primer on partial least squares structural equation modeling (PLS-SEM)*. New York: Sage Publications.

A pesquisa foi realizada com 144 indivíduos, mas 34 não participaram porque não se enquadravam ao perfil necessário: participar, de alguma forma, do processo de desenvolvimento de *software*. Sendo assim, 110 indivíduos responderam efetivamente a pesquisa. O questionário, apresentado no Apêndice B, contém cinco variáveis de caracterização dos entrevistados e 21 questões sobre o objeto do estudo, divididas igualmente em sete questões para cada constructo (métodos ágeis, gestão do conhecimento e qualidade de *software*). Todos os indivíduos responderam 100% das perguntas.

Para verificar a linearidade, foi realizado o teste Bartlett (Mingoti, 2005), que indica que valores de p inferiores a 0,05 sugerem sinais importantes de linearidade nos constructos.

As relações entre os constructos foram avaliadas pela técnica *Partial Least Square* (PLS) que, segundo Tenenhaus, Vinzi, Chatelin e Lauro (2005), é uma abordagem mais flexível que a tradicional baseada na matriz de covariância.

Hair Jr. *et al*. (2009) classificam o processo de modelagem de equações estruturais em modelo de mensuração e modelo estrutural. O primeiro tem o objetivo de aferir se todos os itens de um constructo medem o seu conceito; e o outro determina as relações de causa ou associação entre as variáveis. Os autores recomendam que os itens com cargas fatoriais inferiores a 0,50 devem ser eliminados, pois diminuem a qualidade dos indicadores que representam um constructo.

Para garantir a validade do modelo de mensuração, ou seja, da capacidade do conjunto de indicadores de cada constructo representar com precisão seu respectivo conceito, foram avaliadas a dimensionalidade, a confiabilidade e a validade convergente. Na avaliação da validade convergente utilizou-se o critério da variância média extraída (AVE) proposto por Fornell & Larcker (1981), que representa o percentual médio de variância compartilhada entre o constructo e seus itens. Esse critério garante a validade convergente para valores da AVE acima de 40% no caso de pesquisas exploratórias (Nunnally & Bernstein, 1994). Para verificar a confiabilidade foram utilizados os indicadores Alfa de Cronbach (AC) e confiabilidade composta (CC) (Chin, 1998). Conforme Tenenhaus *et al.* (2005), os indicadores AC e CC devem apresentar valores acima de 0,60 para uma indicação de confiabilidade do constructo no caso de pesquisas exploratórias. Para a validade discriminante foi utilizado o critério de Fornell & Larcker (1981), que garante a validade discriminante quando a AVE de um constructo for maior que a variância compartilhada desse constructo com os demais. A validação discriminante garante que o constructo medido é empiricamente único (Hair Jr. *et al*., 2009). Para verificar a dimensionalidade dos constructos foi utilizado o critério de Kaiser (1958), que retorna a quantidade de dimensões do constructo.

Na avaliação da qualidade do ajuste do modelo foi utilizado o R^2, que representa o quanto os constructos independentes explicam os dependentes, sendo que valores inferiores a 25% representam capacidade explicativa fraca, valores entre 25% e 50% indicam capacidade explicativa moderada e valores acima de 50% evidenciam considerável capacidade explicativa (Hair Jr. *et al*., 2009).

Após a descrição da metodologia de pesquisa, no próximo capítulo faz-se a análise dos dados coletados.

4 Análise e Discussão dos Dados

4.1 Análise de dados faltantes, *outliers*, linearidade e normalidade

Não foram encontrados valores fora do intervalo da escala de sua respectiva variável, não evidenciando o tipo de *outlier* relacionado ao erro na tabulação dos dados. Também não foram encontradas observações consideradas *outliers* univariados nem *outliers* multivariados.

De acordo com o teste de Bartlett, houve linearidade significativa nos constructos, uma vez que todos os valores-p foram inferiores a 0,05. Além disso, foram observadas 183 das 210 relações significativas no nível de significância de 5%, o que representa 87,1% das correlações possíveis, pela matriz de correlação de Pearson, na qual se indica a linearidade entre os itens.

Por definição, o conjunto de dados não apresentou distribuição normal univariada nem mesmo multivariada, pois estão limitados em uma escala discreta e finita. De qualquer forma, foram utilizados os testes Anderson-Darling, Shapiro-Wilk e Mardia para verificar a normalidade univariada e multivariada. De acordo com esses testes, os itens não foram normalmente distribuídos (valor-p < 0,001) de forma univariada, tampouco multivariada.

4.2 Descrição das variáveis de caracterização da amostra

A Tabela 7 mostra as frequências absolutas e relativas para as variáveis de caracterização dos entrevistados.

Tabela 7
Frequências absolutas e relativas para a caracterização dos entrevistados

Variáveis		N=110	
		N	**%**
Sexo	Feminino	19	17,3
	Masculino	91	82,7
Grau de instrução	Ensino Fundamental	3	2,7
	Ensino Médio	0	0,0
	Ensino Superior	38	34,5
	Pós-graduação (*Lato senso*, Mestrado/Doutorado)	69	62,7
Faixa etária	Até 29 anos	17	15,5
	Entre 30 e 39 anos	48	43,6
	Entre 40 e 49 anos	31	28,2
	50 ou mais anos	14	12,7
Função	Analista Desenvolvedor	41	37,3
	Analista de Requisitos	15	13,6
	Analista de Testes	8	7,3
	Gerente / Líder de Projeto	16	14,5
	Gerente	10	9,1
	Diretor ou Dono	7	6,4
	Outros	13	11,8
Quantidade de funcionários da empresa	Até 49 funcionários	20	18,2
	De 50 a 149 funcionários	28	25,5
	De 150 a 299 funcionários	12	10,9
	300 ou mais funcionários	50	45,5

Fonte: dados da pesquisa.

Desta tabela podem-se destacar:

a) 82,7% dos entrevistados eram do sexo masculino;
b) 97,3% possuíam ao menos curso superior, sendo que 62,7 curaram a pós-graduação. Isso significa alto nível de qualificação dos entrevistados;
c) 43,6% tinham entre 30 e 39 anos e apenas 12,7% tinham 50 anos ou mais;
d) 37,3% exerciam a função de analista desenvolvedor, 14,5% eram gerente ou líder de projeto;
e) 45,5% dos indivíduos trabalhavam/prestavam serviço em empresas com 300 funcionários ou mais e 25,5% trabalhavam/prestavam serviço em empresas com 50 a 149 funcionários.

4.3 Análise descritiva dos constructos

A Tabela 8 demonstra a média, o desvio-padrão (DP) e o intervalo de confiança de cada item. Intervalos estritamente menores que três indicam que os indivíduos tendem a discordar, enquanto que intervalos estritamente maiores que três que os indivíduos tendem a concordar.

Tabela 8
Média, desvio-padrão e intervalo de confiança dos itens dos constructos

Constructos	Itens	Média	DP	IC - 95%[1]
Métodos Ágeis	MA1	4,5	0,7	[4,37; 4,62]
	MA2	4,0	0,8	[3,88; 4,16]
	MA3	3,6	1,0	[3,40; 3,79]
	MA4	4,0	0,8	[3,86; 4,17]
	MA5	4,0	1,0	[3,83; 4,22]
	MA6	4,2	0,8	[4,08; 4,37]
	MA7	3,8	1,1	[3,65; 4,06]
Gestão do Conhecimento	GC1	3,8	1,0	[3,63; 3,98]
	GC2	3,9	0,9	[3,77; 4,12]
	GC3	3,9	1,0	[3,68; 4,04]
	GC4	3,3	1,3	[3,02; 3,48]
	GC5	3,2	1,2	[2,99; 3,41]
	GC6	3,3	1,2	[3,07; 3,54]
	GC7	3,3	1,3	[3,11; 3,56]
Qualidade de *Software*	QS1	3,9	0,9	[3,67; 4,03]
	QS2	3,7	1,0	[3,55; 3,90]
	QS3	3,5	1,0	[3,27; 3,66]
	QS4	3,2	1,1	[3,00; 3,40]
	QS5	3,5	1,1	[3,25; 3,67]
	QS6	2,8	1,2	[2,56; 2,99]
	QS7	3,4	1,2	[3,19; 3,64]

[1] Intervalo de confiança (IC) *bootstrap*.
Fonte: dados da pesquisa

Tabela 9
Relação das siglas por item – métodos ágeis

Cons-tructos	Siglas	Itens
Métodos ágeis	MA1	A interação entre colaboradores no meu ambiente de trabalho favorece a obtenção melhores resultados no desenvolvimento de *software*.
	MA2	A equipe de desenvolvimento com a qual trabalho se compromete com o cumprimento dos prazos previamente estabelecidos, para as entregas do *software* acordado.
	MA3	O cliente participa ativamente dos processos de desenvolvimento de *software* nos quais estou envolvido.
	MA4	O processo de desenvolvimento de *software* do meu ambiente de trabalho dá suporte a solicitações de mudanças de escopo de *software*, por parte do cliente.
	MA5	O processo de desenvolvimento de *software* do meu ambiente de trabalho garante entregas parciais (versões) do *software* acordado.
	MA6	O *software* entregue pela equipe de desenvolvimento com a qual trabalho atende plenamente aos requisitos do cliente.
	MA7	Os líderes de desenvolvimento de *software* de meu ambiente de trabalho atuam mais como facilitadores.

Fonte: elaborada pelo autor.

Para o constructo métodos ágeis, a questão MA1 apresentou a maior média (4,5 IC [4,37; 4,62]) entre todos os itens do constructo, indicando que os métodos ágeis realmente favorecem a interação no ambiente de trabalho. Por outro lado, a questão MA3 apresentou a menor média (3,6 IC [3,40; 3,79]) de todo o constructo, indicando que o cliente não teria participação tão ativa conforme a prevista pela teoria dos métodos ágeis. Por fim, em todos os itens desse constructo os indivíduos tenderam a concordar, visto que todos os intervalos de confiança foram estritamente maiores que três.

Tabela 10
Relação das siglas por item – gestão conhecimento

Cons-tructos	Siglas	Itens
Gestão do conhecimento	GC1	O meu ambiente de trabalho valoriza as entregas dos profissionais envolvidos no desenvolvimento de *software*.
	GC2	O desenvolvimento de *software* do meu ambiente de trabalho conta mais com o conhecimento dos envolvidos do que com a documentação de metodologias, processos e práticas documentadas aplicáveis.
	GC3	Em meu ambiente de trabalho há incentivo para a troca de informações entre os envolvidos no desenvolvimento dos *softwares*.
	GC4	O processo de desenvolvimento de *software* do meu ambiente de trabalho preocupa-se em disseminar as lições aprendidas entre os diferentes projetos de *software*.
	GC5	O processo de desenvolvimento de *software* do meu ambiente de trabalho adota técnicas para incentivar a criatividade de seus profissionais.
	GC6	O meu ambiente de trabalho adota estratégias para institucionalizar as novas práticas que deram certo, para execução de atividades.
	GC7	Na minha organização, há clara percepção de que a gestão do conhecimento pode propiciar a obtenção de vantagens competitivas.

Fonte: elaborada pelo autor.

Em todos os itens os indivíduos tenderam a concordar no constructo "gestão do conhecimento", exceto para o item GC5. As questões GC2 (3,9 IC [3,77; 4,12]) e GC3 (3,9 IC [3,68; 4,04]) apresentaram as maiores médias entre todos os itens do constructo, indicando que o conhecimento realmente está em poder dos envolvidos no processo de desenvolvimento de *software*, embora haja grande interação entre eles. Por outro lado, a questão GC5 apresentou a menor média (3,2 IC [2,99; 3,41]) de todo o constructo, indicando que os ambientes não incentivam, pelo menos como deveriam, a criatividade e a inovação.

A média dos itens GC4, GC5, GC6 e GC7 foram significativamente menores que a média dos itens GC1, GC2 e GC3, uma vez que não houve sobreposição de seus intervalos de confiança. Isso leva a crer que, embora o conhecimento e a interação entre os envolvidos no desenvolvimento de *software* sejam valorizados, que a retenção e a gestão do conhecimento ainda não recebem o devido valor nas empresas.

Tabela 11
Relação das siglas por item – qualidade de *software*

Cons-tructos	Siglas	Itens
Qualidade de *software*	QS1	O processo de desenvolvimento de *software* do meu ambiente de trabalho contempla o acompanhamento das atividades, dos recursos e das responsabilidades das pessoas.
	QS2	Os requisitos dos *softwares* desenvolvidos no meu ambiente de trabalho são controlados.
	QS3	O meu ambiente de trabalho apresenta um processo de controle de qualidade do *software* desenvolvido.
	QS4	O processo de desenvolvimento de *software* do meu ambiente de trabalho estabelece que o desempenho das atividades seja medido.
	QS5	O processo de desenvolvimento de *software* do meu ambiente de trabalho apresenta uma atividade de gerência de configuração, garantindo que os *softwares* desenvolvidos sejam instalados adequadamente no ambiente dos clientes.
	QS6	No meu ambiente de trabalho existe um processo de gerência de riscos de projetos.
	QS7	No meu ambiente de trabalho são identificadas necessidades de mudanças nos processos, a partir da análise de defeitos e problemas de desempenho dos projetos de desenvolvimento de *software*.

Fonte: elaborada pelo autor.

Em relação ao constructo "qualidade do *software*", a questão QS1 (3,9 IC [3,67; 4,03]) apresentou a maior média entre todos os itens do constructo, enquanto que a questão QS6 apresentou a menor média (2,8 IC [2,56; 2,99]) de todo o constructo.

Vale ressaltar que em todos os itens os entrevistados tenderam a concordar, exceto quanto ao item QS6 em que os indivíduos tenderam a discordar, visto que o seu intervalo de confiança foi estritamente menor que três. Sua média foi significativamente menor que a média dos demais itens, sendo que seu intervalo de confiança não se sobrepôs a nenhum intervalo de confiança dos itens desse constructo, o que indica que uma gerência efetiva dos riscos dos projetos ainda não está sendo exercida nas empresas de TI. A média do item QS1 foi significativamente maior que a média dos itens QS3, QS4, QS6 e QS7, uma vez que não houve sobreposição do seu intervalo de confiança. Isso indica que, embora as atividades de desenvolvimento de *software* sejam acompanhadas, seu desempenho não está sendo medido como deveria e não está sendo feito controle de qualidade como esperado.

4.4 Modelagem de equações estruturais (PLS)

4.4.1 Resultado do modelo de mensuração

A Tabela 12 apresenta os pesos, as cargas fatoriais e comunalidades no modelo inicial e final de mensuração.

Tabela 12
Pesos, cargas fatoriais e comunalidades no modelo inicial e final de mensuração

Constructos	Modelo inicial					Modelo final			
	Itens	**Peso**	**I.C. - 95%[1]**	**C.F.[2]**	**Com.[3]**	**Peso**	**I.C. - 95%[1]**	**C.F.[2]**	**Com.[3]**
Métodos Ágeis	MA1	0,20	[0,13; 0,26]	0,61	0,37	0,20	[0,13; 0,25]	0,61	0,37
	MA2	0,17	[0,07; 0,23]	0,63	0,40	0,17	[0,07; 0,23]	0,63	0,40
	MA3	0,23	[0,18; 0,30]	0,71	0,50	0,23	[0,17; 0,30]	0,71	0,50
	MA4	0,24	[0,18; 0,31]	0,66	0,43	0,24	[0,18; 0,31]	0,66	0,43
	MA5	0,17	[0,09; 0,24]	0,55	0,30	0,18	[0,10; 0,25]	0,55	0,31
	MA6	0,27	[0,20; 0,36]	0,72	0,52	0,27	[0,19; 0,36]	0,72	0,52
	MA7	0,26	[0,18; 0,35]	0,64	0,41	0,26	[0,19; 0,35]	0,64	0,41
Gestão do Conhecimento	GC1	0,19	[0,17; 0,22]	0,73	0,52	0,20	[0,17; 0,22]	0,73	0,53
	GC2	0,06	[0,00; 0,12]	0,16	0,02	-	-	-	-
	GC3	0,19	[0,16; 0,21]	0,76	0,58	0,19	[0,16; 0,22]	0,76	0,58
	GC4	0,21	[0,19; 0,24]	0,88	0,78	0,22	[0,19; 0,24]	0,88	0,78
	GC5	0,21	[0,18; 0,24]	0,89	0,79	0,21	[0,19; 0,24]	0,89	0,80
	GC6	0,21	[0,19; 0,24]	0,88	0,78	0,21	[0,19; 0,24]	0,88	0,77
	GC7	0,20	[0,16; 0,23]	0,75	0,56	0,20	[0,16; 0,23]	0,75	0,57
Qualidade de *Software*	QS1	0,19	[0,15; 0,24]	0,70	0,49	0,19	[0,15; 0,24]	0,70	0,49
	QS2	0,18	[0,14; 0,23]	0,74	0,54	0,18	[0,15; 0,23]	0,74	0,54
	QS3	0,21	[0,18; 0,26]	0,81	0,65	0,22	[0,18; 0,26]	0,81	0,66
	QS4	0,20	[0,16; 0,23]	0,84	0,70	0,20	[0,17; 0,23]	0,84	0,70
	QS5	0,19	[0,15; 0,23]	0,71	0,50	0,19	[0,15; 0,23]	0,71	0,50
	QS6	0,20	[0,16; 0,24]	0,81	0,66	0,20	[0,16; 0,24]	0,81	0,66
	QS7	0,14	[0,10; 0,19]	0,68	0,47	0,15	[0,10; 0,19]	0,68	0,47

[1] Validação *bootstrap*, [2]Carga fatorial; [3]Comunalidade.
Fonte: dados da pesquisa.

A questão "o desenvolvimento de *software* do meu ambiente de trabalho conta mais com o conhecimento dos envolvidos do que com a documentação de metodologias, processos e práticas documentadas aplicáveis" (GC2), que pertence ao constructo gestão do conhecimento, apresentou carga fatorial abaixo de 0,50, demonstrando que ele não possui correlação significativa com esse constructo e foi, dessa forma, retirado do modelo final. Todas as outras questões exibiram carga fatorial acima de 0,50 e, desse modo, foram significativas para a formação dos indicadores.

4.4.2 Resultado do modelo proposto e verificação das hipóteses

A análise dos resultados do modelo estrutural que considera como variáveis endógenas os constructos gestão do conhecimento e qualidade de *software* revelou as influências positivas dos métodos ágeis e da gestão do conhecimento. O modelo estrutural pode ser visto na Figura 7 a seguir.

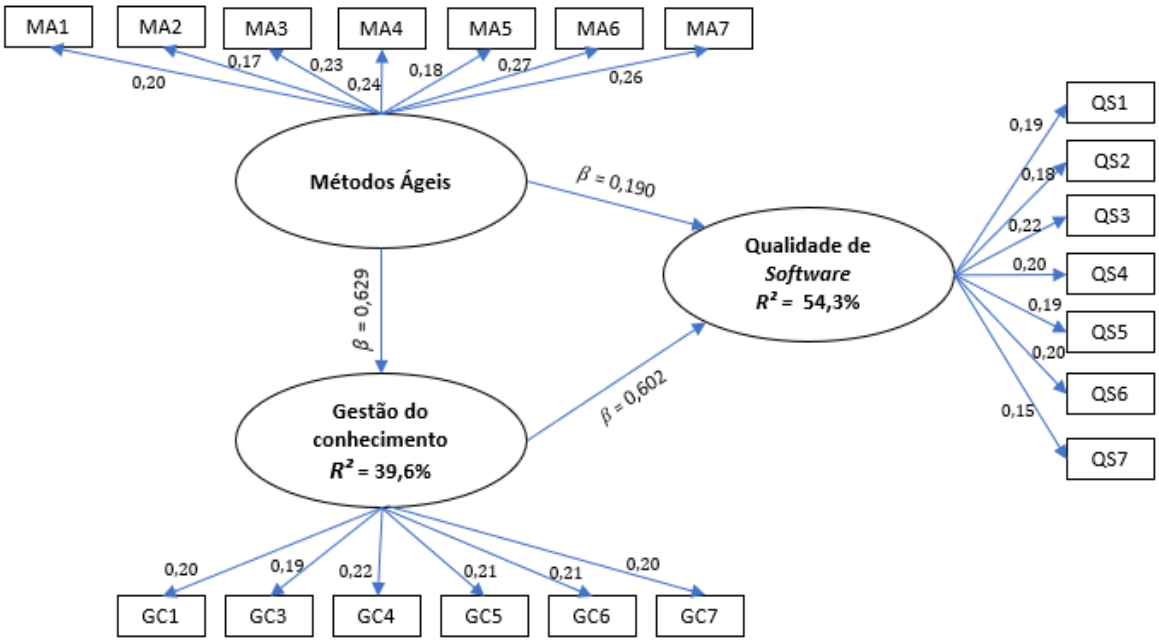

Figura 7
Modelo conceitual e resultados da pesquisa.
Fonte: elaborada pelo autor.

Houve influência significativa (valor-p < 0,001) e positiva (β = 0,629, IC = [0,51; 0,74]) dos métodos ágeis sobre a gestão do conhecimento. Os métodos ágeis conseguiram explicar 39,6% da variabilidade da gestão do conhecimento, confirmando a hipótese H_1 de que os métodos ágeis influenciam positivamente a gestão do conhecimento.

Sendo assim, quanto maior a utilização dos métodos ágeis, maior tende a ser a necessidade da gestão do conhecimento. Isso vem ao encontro das afirmações de Singh *et al.* (2014), Gandomani *et al.* (2013) e Yanzer Cabral *et al.* (2014) de que boa parte do conhecimento empregado com a utilização de métodos ágeis é tácito e da linha de pensamento de Dingsoyr & Smite (2014), a qual apregoa que o desempenho de uma equipe depende do conhecimento disseminado entre os seus membros.

Houve influência significativa (valor-p = 0,026) e positiva (β = 0,190, IC = [0,01; 0,42]) dos métodos ágeis sobre a qualidade do *software*. Sendo assim, quanto mais se utilizar os métodos ágeis, melhor tende a ser a qualidade de *software*, confirmando a hipótese H_2 de que os métodos ágeis influenciam positivamente a qualidade de *software*. Essa afirmação vai ao encontro do pensamento de Imreh & Raisinghani (2011) de que a qualidade de *software* gerado com equipes ágeis está relacionada a uma apropriada implantação dos métodos ágeis nas empresas. Da mesma forma, Koka (2015) acredita que as atividades das metodologias ágeis devem ser seguidas à risca, para garantir a qualidade do *software* gerado.

Houve influência significativa (valor-p < 0,001) e positiva (β = 0,602, IC = [0,41; 0,78]) da gestão do conhecimento sobre a qualidade do *software*. Sendo assim, quanto maior a gestão do conhecimento, melhor tende a ser a qualidade de *software*, confirmando a hipótese H_3 de que a gestão do conhecimento influencia positivamente a qualidade de *software*. Isso vem confirmar a visão de Lee & Chang (2006), de que a gestão do conhecimento pode ser utilizada em várias atividades relacionadas ao desenvolvimento de *software*, e de Aurum *et al*. (2003), de que a gestão conhecimento está presente na melhoria de *software* e processos.

Os métodos ágeis e a gestão do conhecimento conseguiram explicar 54,3% da variabilidade da qualidade de *software*. A Tabela 13 apresenta o resultado das hipóteses do modelo estrutural.

Tabela 13
Validação das hipóteses do modelo

	Hipótese	Resultado
H_1	Os métodos ágeis influenciam positivamente a gestão do conhecimento.	Confirmada
H_2	Os métodos ágeis influenciam positivamente a qualidade de *software*.	Confirmada
H_3	A gestão do conhecimento influencia positivamente a qualidade de *software*.	Confirmada

Fonte: dados da pesquisa.

4.5 Verificação da caracterização dos respondentes com os constructos

A Tabela 14 mostra os indicadores em cada variável de caracterização dos entrevistados e o valor-p dos testes Kruskal-Wallis e Mann Whitney, a fim de verificar se existe alguma associação entre a variável de caracterização e os indicadores.

Tabela 14
Comparação entre as variáveis de caracterização quanto aos indicadores

Variáveis		Métodos Ágeis			Gestão do Conhecimento			Qualidade de *Software*		
		Média	**DP**	**Valor-p**	**Média**	**DP**	**Valor-p**	**Média**	**DP**	**Valor-p**
Sexo[1]	Feminino	4,03	0,17	0,635	3,33	0,26	0,649	3,25	0,20	0,311
	Masculino	4,02	0,06		3,47	0,10		3,44	0,09	
Grau de instrução[2]	Ensino Fundamental	4,55	0,20	0,134	3,94	0,48	0,314	3,47	0,21	0,797
	Ensino Superior	4,08	0,08		3,57	0,16		3,49	0,12	
	Pós-graduação*	3,97	0,08		3,36	0,11		3,36	0,11	
Faixa etária[2]	Até 29 anos	4,17	0,10	0,321	3,50	0,24	0,300	3,42	0,17	0,988
	Entre 30 e 39 anos	3,96	0,09		3,32	0,14		3,37	0,13	
	Entre 40 e 49 anos	3,93	0,12		3,45	0,18		3,41	0,14	
	50 ou mais anos	4,26	0,12		3,81	0,21		3,50	0,22	
Função[2]	Analista Desenvolvedor	3,98	0,09	0,258	3,30	0,16	0,715	3,26	0,13	0,802
	Analista de Requisitos	4,12	0,11		3,44	0,25		3,37	0,20	
	Analista de Testes	4,27	0,21		3,46	0,42		3,39	0,50	
	Gerente / Líder de Projeto	4,16	0,15		3,61	0,23		3,67	0,21	
	Gerente	3,82	0,19		3,45	0,27		3,46	0,22	
	Diretor ou Dono	4,27	0,20		4,03	0,27		3,58	0,21	
	Outros	3,77	0,19		3,39	0,26		3,46	0,18	
Quantidade de funcs. da empresa	Até 49 funcionários	4,05	0,10	0,548	3,44	0,17	0,163	3,36	0,10	0,471
	De 50 a 149 funcionários	3,87	0,18		3,13	0,30		3,61	0,30	
	De 150 a 299 funcionários	4,12	0,12		3,78	0,17		3,54	0,17	
	300 ou mais funcionários	4,00	0,09		3,34	0,14		3,30	0,12	

* Pós-graduação (*Lato senso*, Mestrado/Doutorado), [1] Teste Mann-Whitney, [2] Teste de Kruskall-Wallis.
Fonte: dados da pesquisa.

Em nenhuma das variáveis de caracterização houve diferença significativa (valor-p > 0,05) nos constructos, ou seja, as diferentes categorias das variáveis de caracterização não tiveram diferença significativa na resolução do modelo.

O próximo capítulo apresenta a conclusão do trabalho.

5 Conclusão

Neste capítulo são apresentadas as conclusões desta pesquisa, com breve histórico e uma avaliação se os objetivos geral e específicos foram atingidos. São mostradas, também, as limitações da pesquisa e feitas as propostas para novos estudos.

5.1 Histórico da pesquisa

A pesquisa foi concebida a partir da dúvida da influência dos constructos métodos ágeis e gestão do conhecimento na qualidade de *software*. O modelo conceitual foi proposto e um questionário para responder à dúvida foi criado, com base no referencial teórico levantado para dar suporte à pesquisa. Esse questionário foi validado por especialistas e enviado a profissionais envolvidos no desenvolvimento de *software*, por meio digital. O questionário foi respondido por 144 indivíduos, sendo que 110 estavam dentro do público ao qual a pesquisa se destinava. A partir daí foi feita a análise dos dados levantados por meio da técnica PLS.

5.2 Análise dos objetivos alcançados

Em relação ao primeiro objetivo específico (apresentar os conhecimentos relevantes e relacionamentos hipotéticos entre métodos ágeis, gestão conhecimento e qualidade de *software*), a pesquisa conseguiu atingi-lo. Enfatiza-se, porém, que, conforme pode ser verificado nos itens 2.3 Relacionamento entre gestão do conhecimento e TI / métodos ágeis, 2.5 Relacionamento entre métodos ágeis e modelos de qualidade de *software* e 2.6 Relacionamento entre gestão do conhecimento e qualidade de *software*, a documentação científica dos relacionamentos entre os constructos é escassa. A documentação do item 2.6, entretanto, é um pouco mais significativa que a dos outros dois itens.

No tocante ao segundo objetivo específico (desenvolver uma modelagem de mensuração da qualidade de *software*, na forma de questionário, com a aplicação da análise multivariada), a pesquisa conseguiu atingir o seu objetivo, pois o modelo conceitual proposto foi operacionalizado a partir de um questionário, validado por profissionais de tecnologia da informação e um estatístico. Finalmente, mostrou que os métodos ágeis e a gestão do conhecimento conseguem explicar 54,3 da qualidade de *software*.

Considerando o objetivo geral, a análise dos dados coletados mostrou forte influência (β = 0,602, IC = [0,41; 0,78]) da gestão do conhecimento na qualidade de *software* e

influência, ainda que menor, dos métodos ágeis na qualidade de *software* (β = 0,190, IC = [0,01; 0,42]). Mostrou, também, que existe influência dos métodos ágeis na gestão do conhecimento (β = 0,629, IC = [0,51; 0,74]), comprovando serem positivas as três hipóteses (H_1 Os métodos ágeis influenciam a gestão do conhecimento, H_2 Os métodos ágeis influenciam a qualidade de *software* e H_3 A gestão do conhecimento influencia a qualidade de *software*), do modelo conceitual (item 2.7). A literatura que referencia o relacionamento entre os constructos, entretanto, é escassa, indicando que as influências levantadas nesta pesquisa ou ainda não foram efetivamente percebidas ou não estão tendo a devida atenção. É interessante observar que, embora a pesquisa tenha mostrado mais influência da gestão do conhecimento (do que dos métodos ágeis) na qualidade de *software*, a literatura se mostra ainda mais escassa para explicar e associar esses dois constructos.

5.3 Contribuição da pesquisa

A pesquisa apresentou, como contribuição acadêmica, a identificação do *gap* da literatura em relação ao relacionamento dos constructos que foram alvo desta pesquisa: métodos ágeis, gestão do conhecimento e qualidade de *software*.

Sob o aspecto de contribuição ao conhecimento, a pesquisa destacou a importante influência dos métodos ágeis na gestão do conhecimento e da gestão do conhecimento e dos métodos ágeis na qualidade de *software*.

Sob o aspecto gerencial, a maior influência verificada foi a da gestão do conhecimento na qualidade *software*, o que é mostrado, principalmente, por meio dos indicadores “O meu ambiente de trabalho valoriza as entregas dos profissionais envolvidos no desenvolvimento de *software*” (GC1), “o desenvolvimento de *software* do meu ambiente de trabalho conta mais com o conhecimento dos envolvidos do que com a documentação de metodologias, processos e práticas documentadas aplicáveis” (GC2) e “em meu ambiente de trabalho há incentivo para a troca de informações entre os envolvidos no desenvolvimento dos *softwares*” (GC3). Os resultados sugerem que os gestores de equipes de desenvolvimento de *software* deveriam prestar mais à gestão do conhecimento, como forma de permitir a geração de *software* de qualidade. Isso, entretanto, não vem ocorrendo como deveria, conforme demonstram as médias inferiores dos itens “o meu ambiente de trabalho adota estratégias para institucionalizar as novas práticas que deram certo, para execução de atividades” (GC6) e “na minha organização há uma percepção clara de que a gestão do conhecimento pode propiciar a obtenção de vantagens competitivas” (GC7).

Os métodos ágeis, que a pesquisa da Versionone (2018) comprovou estarem sendo grandemente utilizados atualmente, também têm influência na qualidade de *software*, ainda que menor que o constructo anterior. Cabe aos gestores de equipes de desenvolvimento de *software* dar a devida atenção às premissas dessas metodologias e adequá-las às suas empresas, de maneira a também garantir a qualidade do *software* gerado.

Um ponto importante a ser realçado é que o estudo revelou que os métodos ágeis também influenciam na gestão do conhecimento, sugerindo que os gestores citados anteriormente devem dar atenção também a esse relacionamento, que também pode influenciar de forma indireta a qualidade de *software*.

5.4 Limitações da pesquisa

As principais limitações desta pesquisa são o tamanho da amostra e o perfil dos respondentes. O tamanho da amostra foi significante se avaliado estatisticamente, mas pode não representar toda a comunidade de profissionais envolvidos no desenvolvimento de *software*. A quantidade de respondentes de cada função e o seu peso na pesquisa não foram definidos previamente, o que também pode fazer com que a pesquisa não tenha uma representação tão grande a ponto de permitir a generalização dos resultados.

5.5 Propostas para trabalhos futuros

Esta pesquisa pode ser feita com uma diversidade balanceada de respondentes, para apresentar mais representatividade. Pode ser feita, ainda, em empresas de nichos de mercado específicos do desenvolvimento de *software*, como em *startups* ou empresas de grande porte.

Por outro lado, os resultados aqui encontrados podem ser avaliados, de forma qualitativa, por profissionais de grande expressão ou influência, no desenvolvimento de *software*.

5.6 Considerações finais

A pesquisa teve importância tanto acadêmica quanto gerencial, ao apresentar a influência dos métodos ágeis na gestão do conhecimento e na qualidade de *software* e a influência da gestão do conhecimento na qualidade de *software.* De forma acadêmica, cabe aos cientistas fazerem mais pesquisas e ampliarem o conhecimento sobre o tema. Na parte

gerencial, devem os gestores de equipes de desenvolvimento de *software* utilizar da melhor forma possível as informações aqui registradas, para garantir a qualidade do *software* gerado.

Referências

Acuña, S. T., Gómez, M., & Juristo, N. (2009). How do personality, team processes and task characteristics relate to job satisfaction and software quality? *Information and Software Technology*, *51*(3), 627–639. https://doi.org/10.1016/j.infsof.2008.08.006.

Ahern Dennis, A. C., & Richard, T. (2008). *CMMI® Distilled: A practical introduction to integrated process improvement.* (3. ed., v. 39). Australian family physician. Addison Wesley Professional Pub.

Al-shehab, A. J., Hughes, R. T., & Winstanley, G. (2005). *Facilitating organisational learning through causal mapping techniques in IS / IT Project Risk Management* (October 2004), Springer Verlag, Kaiserslautern, Germany, (p. 145–154).

Araújo, R. C., & Galina, C. T. (2005). *Análise de escopo e planejamento no desenvolvimento de software, sob a perspectiva ágil.* Sistemas de Informação – Universidade do Vale do Rio dos Sinos (Unisinos).

Arent, J., & Norbjerg, J. (2000). Software process improvement as organizational knowledge creation: a multiple case analysis. *System Sciences, 2000. Proceedings of the 33d Hawaii International Conference.*, *00*(c), 1–11. Retrieved from: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=926775.

Ataídes, A. D. C. (2008). Um método para acompanhamento e controle da implantação do CMMI. Dissrtação (Mestrado em Engenharia Elétrica) - Faculdade de Tecnologia da Universidade de Brasília.

Aurum, A., Ross, J., Wohlin, C., & Meliha, H. (2003). *Managing software engineering knowledge*. https://doi.org/10.1007/978-3-662-05129-0.

Bassi, D. (2014). Programação extrema (XP). In: R. Prikladnicki; R. Willi; F. Milani, (col.). Métodos ágeis para desenvolvimento de software. Rio de Janeiro: Saraiva (Ebook, p. 311).

Beck, K. (2000). *Extreme programming explained*. São Paulo: Addison Wesley.

Bernardo, P. C., & Kon, F. (2008). A importância dos testes automatizados. *Engenharia de Software Magazine*, *1*(3), 54–57.

Bjørnson, F. O., & Dingsøyr, T. (2008). Knowledge management in software engineering: A systematic review of studied concepts, findings and research methods used. *Information and Software Technology*, *50*(11),1055–1068. https://doi.org/10.1016/j.infsof.2008.03.006.

Bradley, J. H., Paul, R., & Seeman, E. (2006). Analyzing the structure of expert knowledge. *Information and Management*, *43*(1), 77–91. https://doi.org/10.1016/j.im.2004.11.009.

Bueno, C. F. S., & Campelo, G. B. (2006). Qualidade de software. *Control*, *28*. https://doi.org/10.1590/S1413-70542005000500024.

Campanelli, A. S., Camilo, R. D., & Parreiras, F. S. (2018). The impact of tailoring criteria on agile practices adoption: A survey with novice agile practitioners in Brazil. *Journal of Systems and Software*, *137*, 366–379. https://doi.org/10.1016/j.jss.2017.12.012.

Cavalcante, L. E. (2000). Gestão estratégica de recursos humanos na era da tecnologia da informação e da globalização. *Informação & Informação*, *5*(2), 139–147. https://doi.org/10.5433/1981-8920.2000v5n2p139.

Chin, W. W. (1998). The partial least squares approach to structural equation modeling. *In:* G. A. Marcoulides (ed.). *Methodology for business and management. Modern methods for business research* (1. ed., pp. 295–336). Mahwah, NJ, US: Lawrence Erlbaum Associates Publishers.

Choo, C. W. (2006). *The knowing organization: How organizations use information to construct meaning, create knowledge, and make decisions*. (2. ed., Education + Training, v. 43). https://doi.org/10.1108/EUM0000000005482.

Chrissis, M. B., Konrad, M., & Shrum, S. (2003). CMMI: Guidelines for process integration and product improvement. *Engineering*. https://doi.org/0321711505.

Crescêncio, S. (2014). Lean. In *Métodos ágeis para desenvolvimento de software* (*Ebook*, p. 311). Rio de Janeiro: Bookman.

Cruz, F. (2013). *Scrum e PMBOK unidos no gerenciamento de projetos*. 1. ed., Rio de Janeiro: Brasport.

Davenport, T. H., & Prusak, L. (1998). *Working knowledge: How organizations manage what they know*. Brighton: Harvard Business Press.

Dingsøyr, T., Nerur, S., Balijepally, V., & Moe, N. B. (2012). A decade of agile methodologies: Towards explaining agile software development. *Journal of Systems and Software*, *85*(6), 1213–1221. https://doi.org/10.1016/j.jss.2012.02.033.

Dingsoyr, T., & Smite, D. (2014). Managing knowledge in global software development projects. *IT Professional*, *16*(1), 26–29. https://doi.org/10.1109/MITP.2013.19.

Dorairaj, S., Noble, J., & Malik, P. (2012). Knowledge management in distributed agile software development. *Proceedings - 2012 Agile Conference, Agile 2012*, 64–73. https://doi.org/10.1109/Agile.2012.17.

Efron, B., & Tibshirani, R. J. (1993). *Introduction to the bootstrap*. New York: Chapman & Hall.

Fadel, A. C., & Silveira, H. M. (2010). *Metodologias ágeis no contexto de desenvolvimento de software: XP, Scrum e Lean*. Universidade Estadual de Campinas.

Fornell, C., & Larcker, D. F. (1981). Evaluating structural equation models with unobservable variables and measurement error. *Journal of Marketing Research*, *18*(1), 39–50. https://doi.org/10.2307/3151312.

Franco, E. F. (2007). *Um modelo de gerenciamento de projetos baseado nas metodologias ágeis de desenvolvimento de software e nos princípios da produção enxuta*. Dissertação (Mestrado em Sistemas Digitais) - USP. https://doi.org/10.11606/D.3.2007.tde-09012008-155823.

Gandomani, T. J., Zulzalil, H., Ghani, A. A. A., Sultan, A. B. M., & Nafchi, M. Z. (2013). Obstacles in moving to agile software development methods; At a Glance. *Journal of Computer Science*, *9*(5), 620–625. https://doi.org/10.3844/jcssp.2013.620.625.

Garvin, D. A. (1992). Múltiplas dimensões da qualidade. *In:* D. A. Garvin. *Gerenciando a qualidade: a visão estratégica e competitiva.* (pp. 59–81). São Paulo: Qualitymark.

Gil, A. C. (2002). *Como elaborar projetos de pesquisa: como elaborar projetos de pesquisa* (4. ed.). São Paulo: Atlas.https://doi.org/10.1111/j.1438-8677.1994.tb00406.x.

Goldman, A., Melo, C., Kon, F., Corbucci, H., & Santos, V. (2014). A história dos métodos ágeis no Brasil. *In:* R. Prikladnicki; R. Willi; F. Milani, (col.). *Métodos ágeis para desenvolvimento de software.* Rio de Janeiro: Saraiva (Ebook, p. 311).

Gomes, A., Willi, R., & Rehem, S. (2014). O manifesto ágil. *In:* R. Prikladnicki; R. Willi; F. Milani, (col.). *Métodos ágeis para desenvolvimento de software.* Rio de Janeiro: Saraiva (Ebook, p. 311)..

Gonzalez, I. V. D. P., & Campos, F. C. (2015). Proposta de modelo conceitual de formação de estratégia de negócio a partir da integração da aprendizagem organizacional e a gestão da inovação. *Gestao & Planejamento*, *3*, 473–493. 21p. Retrieved from: http://eds.a.ebscohost.com/eds/pdfviewer/pdfviewer?vid=6&sid=98e12452-1d41-4590-8237-09409eb100ce%40sessionmgr4010&hid=4113.

Hair Jr., J. F., Black, W. C., Babin, B. J., Anderson, R. E., & Tatham, R. L. (2009). *Análise multivariada de dados* (6. ed.). São Paulo: Bookman.

Hair Jr, J. F., Hult, G. T. M., Ringle, C., & Sarstedt, M. (2016). *A primer on partial least squares structural equation modeling (PLS-SEM)*. New York: Sage Publications.

Hughes, D. L., Dwivedi, Y. K., Rana, N. P., & Simintiras, A. C. (2016). Information systems project failure: analysis of causal links using interpretive structural modelling. *Production Planning & Control*, *27*(16), 1313–1333. https://doi.org/10.1080/09537287.2016.1217571.

Imreh, R., & Raisinghani, M. (2011). Impact of agile software development on quality within information technology organizations. *Journal of Emerging Trends in Computing and Information Sciences*, *2*(10), 460–475.

Jia, R., & Reich, B. (2008). IT service climate: the validation of an antecedent construct for it service quality. *Anais do International Conference on Information Systems (ICIS)*, San Francisco.

Juran, J. M., & Godfrey, A. B. (1998). *Juran's quality control handbook*. New York: McGrawHill. https://doi.org/10.1108/09684879310045286.

Kaiser, H. F. (1958). The varimax criterion for analytic rotation in factor analysis. *Psychometrika*, *23*(3), 187–200.

Kasse, T. (2008). Practical insight into CMMI. *Communication*, *39*. https://doi.org/0321711505.

Koka, A. (2015). *Software quality assurance in scrum projects: a case study of development processes among scrum teams in south africa*. Retrieved from: http://digitalknowledge.cput.ac.za/jspui/handle/11189/3194.

Korobinski, R. R. (2001). O grande desafio empresarial de hoje: a gestão do conhecimento. *Perspectiva, Ciência & Informação*, *6*(1), 107–116.

Lee, M.C., & Chang, T. (2006). Applying TQM, CMM and ISO 9001 in knowledge management for software development process improvement. *International Journal of Services and Standards*, *2*(1), 101–115. https://doi.org/10.1504/IJSS.2006.008161.

Levy, M., & Hazzan, O. (2009). Knowledge management in practice: The case of agile software development. *Anais do Icse Workshop on Cooperative and Human Aspects of Software Engineering*, 60–65. https://doi.org/10.1109/CHASE.2009.5071412.

Lowry, P. B., & Wilson, D. W. (2016). Creating agile organizations through IT: The influence of internal it service perceptions on it service quality and IT agility. *Journal of Strategic Information Systems (JSIS)*.

Matos, F., & Lopes, A. (2008). Gestão do capital intelectual: A nova vantagem competitiva das organizações. *Comportamento Organizacional e Gestão*, *14*. Recuperado de: http://repositorio.ispa.pt/bitstream/10400.12/152/1/COG%2014%282%29%20%282008%29%20233-245.pdf), 233–245.

Mingoti, S. (2005). *Análise de dados através de métodos de estatística multivariada*. Belo Horizonte: UFMG.

Morum, R. S. (2005). A model for knowledge in an architectural enterprise management. Dissertação (Mestrado em Design Management Degree) - UNITEC Institute of Technology, New Zealand

Nerur, S., Mahapatra, R., & Mangalaraj, G. (2005). Challenges of migrating to agile methodologies. *Communications of the ACM*, *48*(5), 72–78. https://doi.org/10.1145/1060710.1060712.

Nonaka, I., & Takeuchi, H. (1997). *Criação de conhecimento na empresa: como as empresas japonesas geram a dinâmica da inovação* (1. ed.). Rio de Janeiro: Campus.

Nunnally, J. C., & Bernstein, I. H. (1994). *Psychometric Theory* (3rd ed.). New York, N.Y.: McGraw-Hill.

Oliveira, A., Petrini, M., & Pereira, D. L. (2015). Avaliação da adoção do CMMI considerando o custo de qualidade de software. *Revista de Gestão e Projetos - GeP*, *06*(01), 45–62. https://doi.org/10.5585/gep.v6i1.281.

Pandey, K. N. (2016). *Paradigms of knowledge management.* Novo México: Springer (v. 60). https://doi.org/10.1007/978-81-322-2785-4.

Patriotta, G. (2003). *Organizational knowledge in the making: how firms create, use, and institutionalize knowledge*. USA: United States: Oxford University Press.

Ponchirolli, O., & Fialho, F. A. P. (2005). Gestão estratégica do conhecimento como parte da estratégia empresarial. *Revista da FAE, 8*(1).

Prabhakar, G. P. (2008). What is project success: A literature review. *International Journal of Business and Management, 3*, 3–10.

Prikladnicki, R., & Magno, A. (2014). O framework do scrum. *In:* R. Prikladnicki; R. Willi; F. Milani, (col.). *Métodos ágeis para desenvolvimento de software.* Rio de Janeiro: Saraiva (Ebook, p. 311).

Razzak, M. A., & Ahmed, R. (2014). Knowledge sharing in distributed agile projects: techniques, strategies and challenges. *Proceedings of the 2014 Federated Conference on Computer Science and Information Systems*, *2*, 1431–1440. https://doi.org/10.15439/2014F280.

Retamal, A. M. (2014). FDD - feature-driven development. *In:* R. Prikladnicki; R. Willi; F. Milani, (col.). *Métodos ágeis para desenvolvimento de software.* Rio de Janeiro: Saraiva (Ebook, p. 311).

Rodrigues, W. (2007). Metodologia científica. *Paracambi: Faetec/Ist*, 1–20.

Rossetti, A. G., & Morales, A. B. T. (2007). O papel da tecnologia da informação na gestão do conhecimento. *Ciência da Informação*, *36*(1), 124–135. https://doi.org/10.1590/S0100-19652007000100009.

Sagheer, M., Zafar, T., & Sirshar, M. (2015). A framework for software quality assurance using agile methodology. *International Journal of Scientific & Technology Research*, *4*(2), 44–50.

Salo, O. (2005). Systematical validation of learning in agile software development environment. *Biennial Conference on Professional Knowledge Management/ Wissensmanagement.* Berlin, Heidelberg: Springer (106–110).

Scorsolini-Comin, F., Inocente, D. F., & Miura, I. K. (2011). Aprendizagem organizacional e gestão do conhecimento: pautas para a gestão de pessoas. *Revista Brasileira de Orientação Profissional*, *12*(2), 227–239.

Shang, S. S. C., & Lin, S.-F. (2009). Understanding the effectiveness of capability maturity model integration by examining the knowledge management of software development processes. *Total Quality Management & Business Excellence*, *20*(5), 509–521. https://doi.org/10.1080/14783360902863671.

Schwaber, K., & Sutherland, J. (2017). *Guia do Scrum*. Desenvolvido e mantido pelos criadores do Scrum: Ken Schwaber e Jeff Sutherland

Silva Júnior, S., & Costa, F. (2014). Mensuração e escalas de verificação: uma análise comparativa das escalas de Likert e Phrase Completion. *Revista Brasileira de Pesquisas de Marketin, Opinião e Mídia*, *15*, 1–16. https://doi.org/1983-9456.

Singh, A., Singh, K., & Sharma, N. (2014). Agile knowledge management: a survey of Indian perceptions. *Innovations in Systems and Software Engineering*, *10*(4), 297–315. https://doi.org/10.1007/s11334-014-0237-z.

Softex (2016). Ministério da Previdência Social - MPS.BR - *Melhoria de processo do software brasileiro guia geral*. Brasília: Softex, 2016.

Standish Group. (2015). *Anais do Chaos Report*. Retrieved from: https://www.standishgroup.com/sample_research.

Takeuchi, H., & Nonaka, I. (2008). *Gestão do conhecimento: paradoxo e conhecimento*. Retrieved from: http://srvd.grupoa.com.br/uploads/imagensExtra/legado/T/takeuchi_Hirotaka/Gestao_Do_Conhecimento/Liberado/Cap_01.pdf.

Takeuchi, H., & Nonaka, I. (1986). The new new product development game. *Harvard Business Review*, 137–147.

Tenenhaus, M., Vinzi, V. E., Chatelin, Y. M., & Lauro, C. (2005). PLS path modeling. *Computational Statistics and Data Analysis*, *48*(1), 159–205. https://doi.org/10.1016/j.csda.2004.03.005.

Terra, J. C. C. (2000). Gestão do conhecimento: o grande desafio empresarial! *Terra Forum*, 1–6.

Ullah, M. I., & Zaidi, W. A. (2009). *Quality assurance activities in agile*. Retrieved from: http://www.cin.ufpe.br/~scls/Claudia/QA_Activities_in_Agile.pdf.

Vale, A. (2014). Kanban. *In:* R. Prikladnicki; R. Willi; F. Milani, (col.). *Métodos ágeis para desenvolvimento de software.* Rio de Janeiro: Saraiva (Ebook, p. 311).

Valls, V. M. (2004). O enfoque por processos da NBR ISO 9001 e sua aplicação nos serviços de informação. *Ciência da Informação*, *33*(2), 172–178. https://doi.org/10.1590/S0100-19652004000200018.

Versionone. (2018). *Anais do 12 Annual State of Agile Report*. Retrieved from: https://explore.versionone.com/state-of-agile/versionone-12th-annual-state-of-agile-report.

Yanzer Cabral, A. R., Ribeiro, M. B., & Noll, R. P. (2014). Knowledge management in agile software projects: A systematic review. *Journal of Information & Knowledge Management*, *13*(01), 1450010.

Apêndice

Questionário

Pesquisa Acadêmica

Esta é uma pesquisa de mestrado do aluno Alan Reis Scatolino, da Universidade FUMEC de Belo Horizonte, orientado pelo professor Dr. Ronaldo Camilo. A pesquisa tem por objetivo investigar a influência dos Métodos Ágeis e da Gestão do Conhecimento na Qualidade de Software. Sua participação é opcional, voluntária e, caso queira, será anônima. Todos os dados individuais serão confidenciais, não sendo divulgados a terceiros. Mas se quiser receber os resultados da pesquisa, basta informar seu e-mail. Para responder, você só precisa participar, de alguma forma, do processo de desenvolvimento de software.

Desde já, o meu muito obrigado.

E-mail para contato: alan.scatolino@fumec.edu.br

* Informações obrigatórias

*Obrigatório

Você participa do processo de desenvolvimento de software, em alguma empresa ou prestando serviço para alguma empresa? *

- ○ Sim - Continuar
- ○ Não - Finalizar

PRÓXIMA

Nunca envie senhas pelo Formulários Google.

Pesquisa Acadêmica

*Obrigatório

Informações Básicas

Seu sexo: *

- ◯ Feminino
- ◯ Masculino

Seu grau de instrução: *

- ◯ Ensino Médio/1º. Grau Incompleto/Completo
- ◯ Ensino Fundamental/2º. Grau
- ◯ Ensino Superior
- ◯ Pós-graduação (Lato-senso, Mestrado/Doutorado)

Sua faixa etária: *

- ◯ Até 29 anos.
- ◯ Entre 30 e 39 anos
- ◯ Entre 40 e 49 anos
- ◯ 50 ou mais anos

Qual sua principal função: *

- ◯ Analista Desenvolvedor
- ◯ Analista de Requisitos
- ◯ Analista de Testes
- ◯ Gerente / Líder de Projeto
- ◯ Gerente
- ◯ Diretor ou Dono
- ◯ Outro:

Quantidade de funcionários da empresa que você trabalha/mais presta serviço: *

- ◯ Até 49 funcionários
- ◯ De 50 a 149 funcionários
- ◯ De 150 a 299 funcionários
- ◯ 300 ou mais funcionários

Caso queira receber os resultados da pesquisa, por favor, informe seu e-mail.

Sua resposta

VOLTAR PRÓXIMA

Nunca envie senhas pelo Formulários Google.

Pesquisa Acadêmica

*Obrigatório

Parte 1 - Métodos Ágeis

Os Métodos Ágeis têm como principais características a utilização de equipes multifuncionais, auto organizadas e são baseados em alguns valores: indivíduos e interação; software em funcionamento; colaboração com o cliente; resposta a mudanças. A primeira parte da pesquisa é baseada em perguntas sobre os métodos ágeis, em seu ambiente de trabalho.

A interação entre colaboradores no meu ambiente de trabalho favorece a obtenção melhores resultados no desenvolvimento de software. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

A equipe de desenvolvimento com a qual trabalho se compromete com o cumprimento dos prazos previamente estabelecidos, para as entregas do software acordado. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

O cliente participa ativamente dos processos de desenvolvimento de software dos quais estou envolvido. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

O processo de desenvolvimento de software do meu ambiente de trabalho suporta solicitações de mudanças de escopo de software, por parte do cliente. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

O processo de desenvolvimento de software do meu ambiente de trabalho garante entregas parciais (versões) do software acordado. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

O software entregue pela equipe de desenvolvimento com a qual trabalho atende plenamente aos requisitos do cliente. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

Os líderes de desenvolvimento de software de meu ambiente de trabalho atuam mais como facilitadores. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

VOLTAR PRÓXIMA

Pesquisa Acadêmica

*Obrigatório

Parte 2 - Gestão do Conhecimento

A Gestão do Conhecimento trata da transformação do conhecimento das pessoas em conhecimento de grupo e, o conhecimento dos grupos, por sua vez, em conhecimento organizacional. A segunda parte da pesquisa é baseada em perguntas sobre a gestão do conhecimento em seu ambiente de trabalho.

O meu ambiente de trabalho valoriza as entregas dos profissionais envolvidos no desenvolvimento de software. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

O desenvolvimento de software do meu ambiente de trabalho conta mais com o conhecimento dos envolvidos, do que com a documentação de metodologias, processos e práticas documentadas aplicáveis. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

Em meu ambiente de trabalho há incentivo para a troca de informações entre os envolvidos no desenvolvimento dos softwares. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

O processo de desenvolvimento de software do meu ambiente de trabalho se preocupa em disseminar as lições aprendidas entre os diferentes projetos de software. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

O processo de desenvolvimento de software do meu ambiente de trabalho adota técnicas para incentivar a criatividade de seus profissionais. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

O meu ambiente de trabalho adota estratégias para institucionalizar as novas práticas que deram certo, para execução de atividades. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

Na minha organização, há uma percepção clara de que a Gestão do Conhecimento pode propiciar a obtenção de vantagens competitivas. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

VOLTAR PRÓXIMA

Nunca envie senhas pelo Formulários Google.

Pesquisa Acadêmica

*Obrigatório

Parte 3 - Qualidade de Software

A busca da garantia de qualidade de software vem sendo estruturada através processos que se baseiam na aplicação de boas práticas de mercado ou de modelos de qualidade utilizados com êxito em outras áreas do conhecimento. A última parte da pesquisa é baseada em perguntas sobre o processo de desenvolvimento de software de seu ambiente de trabalho.

O processo de desenvolvimento de software do meu ambiente de trabalho contempla o acompanhamento das atividades, dos recursos e das responsabilidades das pessoas. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

Os requisitos dos softwares desenvolvidos no meu ambiente de trabalho são controlados. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

O meu ambiente de trabalho apresenta um processo de controle de qualidade do software desenvolvido. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

O processo de desenvolvimento de software do meu ambiente de trabalho estabelece que o desempenho das atividades seja medido. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

O processo de desenvolvimento de software do meu ambiente de trabalho apresenta uma atividade de Gerência de Configuração, garantindo que os softwares desenvolvidos serão instalados adequadamente, no ambiente dos clientes. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

No meu ambiente de trabalho existe um processo de gerência de riscos de projetos. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

No meu ambiente de trabalho são identificadas necessidades de mudança nos processos, a partir da análise de defeitos e problemas de desempenho dos projetos de desenvolvimento de software. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

VOLTAR ENVIAR

Nunca envie senhas pelo Formulários Google.