Revista de Gestão e Projetos – GeP
eISSN: 2236-0972
DOI: https://doi.org/10.5585/GeP.v10i3.14022
Data de recebimento: 2/6/2019
Data de Aceite: 7/10/2019
Editora-chefe: Rosária de Fátima Segger Macri Russo
Editor Científico: Luciano Ferreira da Silva

Artigos

Influência da aplicação de métodos ágeis e da gestão do conhecimento na qualidade de software: uma análise multivariada

Influence of agile methods application and knowledge management in software quality: a multivariate analysis

Alan Reis Scatolino[1]

Ronaldo Darwich Camilo[2]

Resumo

Atualmente, as empresas de tecnologia da informação têm buscado intensamente a produtividade e a melhoria do posicionamento competitivo, mas elas têm consciência de que não podem abrir mão da qualidade do *software* gerado ou da preservação dos conhecimentos obtidos nesse processo. Nesse contexto, as metodologias ágeis encaixam-se bem e espera-se que conceitos aplicados de gestão do conhecimento tenham importante papel. O presente trabalho apresenta uma modelagem de equações estruturais da influência e da contribuição dos métodos ágeis e da gestão do conhecimento na qualidade de *software,* com base na percepção obtida por meio da aplicação de um questionário respondido por profissionais de tecnologia da informação (TI). A pesquisa indicou que os métodos ágeis e a gestão do conhecimento, no âmbito do desenvolvimento de *software*, se influenciam mutuamente, de forma positiva. A pesquisa mostrou, ainda, que tanto os métodos ágeis, quanto a gestão do conhecimento influenciam de forma positiva a qualidade de *software*, sendo que a influência da gestão do conhecimento foi maior, sugerindo que os gestores deveriam considerar com mais intensidade os fatores associados a colaboração e compartilhamento, criatividade, conteúdos entregues e disseminação de lições aprendidas.

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

Abstract

Currently, information technology companies have been intensively seeking productivity and improving competitive positioning, but are aware that they cannot give up the quality of the software generated or the preservation of the knowledge obtained in this process. In this context, agile methodologies fit well and applied knowledge management concepts are expected to play an important role. This paper presents a structural equation modeling of the influence and contribution of agile methods and knowledge management on software quality, based on the perception obtained through the application of a questionnaire answered by Information Technology (IT) professionals. Research has indicated that agile methods and knowledge management in software development influence each other positively. It also showed that both agile methods and knowledge management positively influence software quality, and the influence of knowledge management was greater, suggesting that managers should consider more closely the factors associated with collaboration and sharing. , creativity, content delivered and the dissemination of lessons learned.

Keywords: Knowledge Management; Agile Methods; Software Quality.

1 Mestre, Universidade FUMEC – FACE FUMEC. Belo Horizonte, Minas Gerais – Brasil.
ORCID: https://orcid.org/0000-0002-6474-0265
arscatolino@gmail.com

2 Doutor, Universidade FUMEC – FACE FUMEC. Belo Horizonte, Minas Gerais – Brasil.
ORCID: https://orcid.org/0000-0001-8433-323X
camillor@terra.com.br

1 Introdução

Os investimentos estratégicos em tecnologia da informação (TI), notadamente em *software*, representam, atualmente, considerável parcela do orçamento corporativo de muitas empresas. Chrissis, Konrad & Shrum (2003) mostram que o desenvolvimento de *software* é crítico para as empresas, porque vários processos corporativos têm suporte de *softwares* de todo tipo. Dessa forma, a qualidade do *software* gerado é primordial para dar suporte à tomada de decisão e evitar prejuízos, entre eles financeiros e de imagem (Lowry & Wilson, 2016; Oliveira, Petrini, & Pereira, 2015). Sendo assim, "manter produtividade, eficiência e qualidade em equilíbrio tornou-se vital para o sucesso das empresas do ramo" (Oliveira *et al*., 2015).

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 os usuários. Com o intuito de minimizar essas falhas, muitas metodologias e *frameworks* de referência vêm sendo desenvolvidos ao longo dos anos nas empresas, entidades normativas e associações, incluindo fatores relacionados à gestão do conhecimento e métodos para assegurar a qualidade em um mundo em acelerada transformação (Fialho & Ponchirolli, 2005; Gonzalez & de Campos, 2015; Nerur, Mahapatra & Mangalaraj, 2005; Oliveira *et al*., 2015).

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 eficazes de desenvolvimento de *software,* para oferecer rapidez, flexibilidade e entregas de qualidade. De acordo com da Silva e Bustamante (2017), os métodos ágeis visam diminuir problemas relacionados a prazos, custos e qualidade de projetos e produtos. Assim, as metodologias ágeis atendem a projetos de desenvolvimento de *software* com mudanças tanto em requisitos (Martins & de Menezes, 2016), quanto nas habilidades dos envolvidos e, também, nas tecnologias empregadas (Nerur *et al*., 2005). Por fim, Martins e de Menezes (2016) asseguram que "as metodologias tradicionais, mesmo com o surgimento das metodologias ágeis, ainda estão presentes na gestão de projetos de desenvolvimento de *software*, porém já não são mais tão adequadas ao atual ambiente de negócios, onde os clientes querem retorno do benefício o mais rápido possível".Por outro lado, Cavalcante (2000), Ribeiro, Soares, Jurza, Ziviani e Neves (2017) defendem que, para que as empresas atinjam posição de destaque, maximizando o seu desempenho, a gestão do conhecimento oferece suporte muito importante. Para Fialho e Ponchirolli (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. Assim, o sucesso do desenvolvimento de *software* depende das atividades de planejamento (Pinto, Vasconcelos, & Lezana, 2014) e do conhecimento do conteúdo relativos ao *software* desenvolvido (Levy & Hazzan, 2009).

O objetivo deste estudo é contribuir para o esclarecimento da influência dos métodos ágeis, alinhados à gestão do conhecimento, na qualidade de *software,* na perspectiva de profissionais de TI. Trata dos pressupostos teóricos da gestão conhecimento, dos métodos ágeis e, principalmente, o relacionamento dessas disciplinas com a qualidade de *software,* com o objetivo de auxiliar as empresas a avaliar as práticas e métodos utilizados no seu processo de desenvolvimento de *software*, para que elas possam gerar *softwares* de melhor qualidade, mesmo no contexto atual de projetos de prazos e custos desafiadores.

O artigo é composto pelas seguintes seções, a saber: além desta introdução, ele apresenta o referencial teórico tomado como base para a pesquisa, a metodologia, seu resultado, a discussão dos dados, a conclusão, os agradecimentos e, por fim, as referências.

2 Referencial teórico

Os métodos "leves" têm 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, esses métodos passaram a ser chamados de "ágeis", quando um grupo de 17 especialistas se reuniu e criou, então, o Manifesto Ágil. Ele é composto de valores

como indivíduos e interação, *software* em funcionamento, colaboração com o cliente e resposta a mudanças (Gomes *et al*., 2014). O presente trabalho investigou a utilização dos métodos ágeis por profissionais de TI, na ótica de valores como interação, entregas de *software*, participação do cliente e capacidade de adequação às mudanças de escopo. Avaliou, ainda, a sua influência no *software* desenvolvido por eles.

Nonaka e Takeuchi (1997) realçam 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. De acordo com Davenport & Prusak (1998), as empresas contratam seus funcionários muito mais por sua experiência e conhecimento do que pela sua inteligência e educação, pois é mais provável que gerentes que tomam decisões complexas valorizem mais pessoas experientes do que informações em bancos de dados. Choo (2006) considera a informação parte do ambiente externo 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.

A gestão do conhecimento, conforme Morum (2005), trata da transformação do conhecimento das pessoas em conhecimento de grupo. E o conhecimento dos grupos, por sua vez, em conhecimento organizacional. Já Pandey (2016, p. 1) conceitua a gestão do conhecimento como "uma vasta área que compreende criação, aquisição, colaboração, compartilhamento, uso, reutilização e capitalização de conhecimento em uma organização". O presente estudo considerou conceitos como colaboração, compartilhamento, uso e reutilização de códigos e a sua influência na qualidade do *software* gerado.

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". 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, Clouse, & Turner, 2008).

Entre os modelos de qualidade de *software* estão o Modelo Integrado de Maturidade em Capacitação (CMMI) e o guia Melhoria de Processo do *Software* Brasileiro (MPS-BR). Chrissis *et al*. (2003) entendem o CMMI como o conjunto das melhores práticas para o desenvolvimento e a manutenção de produtos e serviços, que abrange todo o ciclo de vida do produto, desde a sua concepção até a entrega e posterior manutenção. Já o MPS-BR (Associação para Promoção da Excelência do *Software* Brasileiro - Softex, 2016) é um programa 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. A presente pesquisa adotou conceitos dos modelos de qualidade de *software* como acompanhamento do desenvolvimento, controle dos requisitos, gestão dos riscos, controle de qualidade e melhoria de processos que vêm sendo efetivamente empregados pelos profissionais de TI.

2.1 Gestão do conhecimento e métodos ágeis

Rossetti e Morales (2007) reconhecem que a associação entre a gestão do conhecimento e a TI é demasiadamente complexa, pois abrange, de um lado, pessoas, conhecimentos tácitos, explícitos e empresariais e, de outro, sistemas de informação.

No desenvolvimento de sistemas, Levy & Hazzan (2009) e de Almeida (2017) enfatizam 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. Dingsøyr, Nerur, Balijepally & Moe (2012) alertam que isso é, inclusive, entendido por muitos como ausência de documentação. Singh, Singh & Sharma (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, Zulzalil, Ghani, Sultan & Nafchi (2013) e Yanzer Cabral, Ribeiro & Noll (2014) salientam que isso sustenta a crença de que a maior parcela do conhecimento empregada no desenvolvimento

de *software*, utilizando-se métodos ágeis, é tácita e está em poder da equipe de desenvolvimento.

Por outro lado, no estudo de Singh *et al.* (2014), realizado 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, Noble & Malik (2012), os métodos ágeis promovem o compartilhamento do conhecimento. Entretanto, não existem estudos concretos que garantam a adoção dessas práticas de gestão do conhecimento nesse contexto. No entendimento de Razzak & Ahmed (2014), 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*.

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. 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.

Assim, tendo em vista essas perspectivas, e com o objetivo de verificar o relacionamento de métodos ágeis e gestão do conhecimento, formulam-se as seguintes hipóteses:

H1: os métodos ágeis influenciam positivamente a gestão do conhecimento;

H2: a gestão do conhecimento influencia positivamente os métodos ágeis.

2.2 Métodos ágeis e qualidade de *software*

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. E 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.

A qualidade do *software* gerado está relacionada a pontos como especificação e implementação dos objetivos funcionais e o desempenho técnico da equipe (Prabhakar, 2008). Em estudo procedido com o método ágil XP, Acuña, Gómez & Juristo (2009) mostram que as equipes que o utilizam devem ser fortemente participativas e comunicativas, para obterem um *software* de qualidade. Isso vai ao encontro das conclusões de de Almeida (2017), que reconhece a comunicação como muito importante para as metodologias ágeis, porque ela passa a ser em tempo real, em vez de ser documentada. Acuña *et al.* (2009) concluem que a extroversão dos membros da equipe está substancialmente relacionada à melhor qualidade de *softwares* desenvolvidos, utilizando-se uma metodologia ágil. De acordo com Imreh & Raisinghani (2011) e Santos e Canêdo (2014), 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*. Os estudos de Feller (2016) no CPD da UFRGS indicaram que a criação de uma área responsável pelo teste de *software* em uma empresa aumenta a qualidade dos sistemas disponibilizados aos usuários. Adicionado a isso, práticas como a programação em pares e o desenvolvimento orientado a testes podem ser eficazes para se minimizarem erros. Os achados de Koka (2015) corroboram essa afirmação e mostram 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 significativa 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, Zafar, & Sirshar, 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). 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. Para o manual da Softex (SOFTEX, 2012), as mudanças ocorridas nos ambientes de negócio empresariais, por sua vez, têm provocado modificações em seus processos e ambientes produtivos.

Para Imreh & Raisinghani (2011) e de Almeida (2017), 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 à apropriada implantação dos métodos ágeis nas empresas. Oliveira *et al.* (2015) 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).

Assim, tendo em vista essas perspectivas e com o objetivo de verificar o relacionamento de métodos ágeis e qualidade de *software*, formula-se a seguinte hipótese:

H3: os métodos ágeis influenciam positivamente a qualidade de *software.*

2.3 Gestão do conhecimento e qualidade de *software*

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".

A Engenharia de *Software* é uma atividade que depende muito do conhecimento (Bjørnson & Dingsøyr, (2008)). Vasanthapriyan, Tian e Xiang (2015) complementam que o desenvolvimento de *software* depende muito do compartilhamento do conhecimento entre os envolvidos nessa atividade. Assim, o maior ativo das empresas de TI é o conhecimento de seus funcionários e não suas máquinas ou fábricas de *software* (Bjørnson & Dingsøyr, 2008)

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 tenham avaliado o valor do conhecimento no CMMI. Mesmo assim, segundo Viana, Souza e Conte (2014), o CMM propicia um ambiente adequado para o desenvolvimento da aprendizagem organizacional, inclusive no processo de garantia de qualidade. Shang & Lin (2009) asseguram que existe 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.

As lições aprendidas são definidas por Viana et al. (2014) como conhecimentos organizacionais que retratam soluções de problemas, para que eles não ocorram em outros projetos. Elas permitem, dessa forma, melhoria nos processos empresariais. 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.

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*. Segundo os autores, disciplinas da gestão do conhecimento, como a Ciência Cognitiva, Ergonomia e Gestão, podem contribuir muito para a administração desse conhecimento. Vasanthapriyan *et al*. (2015) corroboram essa informação no momento em que afirmam que o sucesso dos projetos de desenvolvimento de *software* depende muito do compartilhamento do conhecimento entre os profissionais de *software* na organização. Arent & Nørbjerg (2000) propõem modelos teóricos que unem a gestão do conhecimento, o capital intelectual e o gerenciamento de projetos. E acrescentam que a gestão do conhecimento tem muita importância na gestão de projetos de *softwares* complexos, principalmente nas negociações e na solução de problemas inerentes a essa atividade.

Segundo Aurum, Ross, Wohlin & Meliha (2003), a gestão do 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 propõem 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. Aurum *et al*. (Aurum et al., 2003, p. 255) acentuam que "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".

Assim, tendo em vista essas perspectivas e com o objetivo de verificar o relacionamento da gestão de conhecimento e qualidade de *software*, formula-se a seguinte hipótese:

H4: a gestão do conhecimento influencia positivamente a qualidade de *software*.

2.4 Modelo conceitual proposto

Proposta a partir das considerações e relações estudadas, a Figura 1 ilustra o modelo conceitual do presente estudo. As hipóteses H_1 e H_2 tratam da influência mútua entre os métodos ágeis e a gestão do conhecimento. A hipótese H_3 trata da influência dos métodos ágeis na qualidade de *software* e, por fim, a hipótese H_4 trata da influência da gestão do conhecimento na qualidade de *software*.

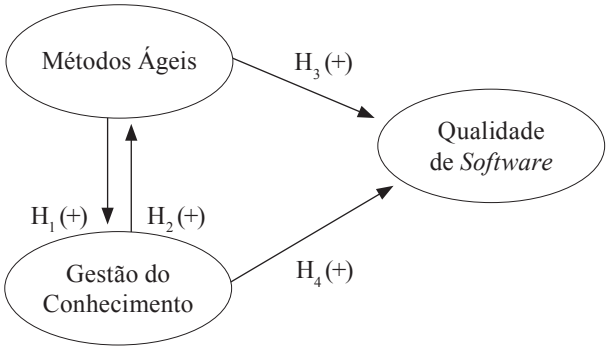

Figura 1: Modelo proposto
Fonte: elaborado pelos autores (2019).

3 Metodologia

A pesquisa realizada para atender a este estudo foi exploratória, de natureza aplicada e abordagem quantitativa, operacionalizada a partir de um *survey* (questionário) composto de afirmativas com índices para mensuração das variáveis latentes reflexivas dos constructos do modelo. O questionário proposto foi baseado na escala Likert de 5 pontos, sendo que um representou "discordo totalmente" e cinco, "concordo totalmente".

O questionário foi codificado por meio da ferramenta *Google forms* e, então, criticado por especialistas que sugeriram as adaptações e correções necessárias, deixando o texto das questões mais claras e evitando nomenclatura técnica, que talvez não fosse de conhecimento dos respondentes. O conteúdo das questões, entretanto, não foi alterado. Entre os especialistas estavam profissionais com experiência em projetos de desenvolvimento de sistemas: um mestre e doutorando em sistemas de informação e gestão do conhecimento, com 10 anos de

experiência em desenvolvimento de *software*; um mestre em sistemas de informação e gestão do conhecimento, com 25 anos de experiência em gestão de projetos de TI; uma gerente de projetos de TI com 15 anos de experiência; e um estatístico, com 11 anos de experiência em estudos quantitativos.

O questionário foi, então, enviado em meio digital para os respondentes. A população escolhida 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. A amostragem foi probabilística e aleatória simples.

O modelo estrutural dos constructos e o de mensuração combinados é mostrado na Figura 2.

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. De acordo com Hair Jr., Hult, Ringle & Sarstedt (2016), utilizando-se nível de significância de 5%, nível de 0,10 da medida de qualidade de ajuste do modelo estrutural - R^2 (0,10) e tendo-se em vista que o maior número de setas que chega a um dos constructos (qualidade de *software*) da pesquisa é dois, o tamanho mínimo da amostra foi de 110 respondentes.

A pesquisa foi distribuída, inicialmente, entre profissionais de TI de Belo Horizonte, Minas Gerais, sendo depois redistribuída por eles. Entendeu-se que esses profissionais tinham o perfil necessário para colaborar com a pesquisa: participar, de alguma forma, do processo de desenvolvimento de *software*. A pesquisa foi realizada com 144 indivíduos, mas 34 não participaram porque não se enquadravam neste perfil. Sendo assim, 110 indivíduos responderam efetivamente à pesquisa.

Todos os indivíduos responderam 100% das perguntas. Da amostra, pode-se destacar:

a) 82,7% dos entrevistados eram do sexo masculino;
b) 97,3% possuíam ao menos curso superior, sendo que 62,7% cursaram 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;

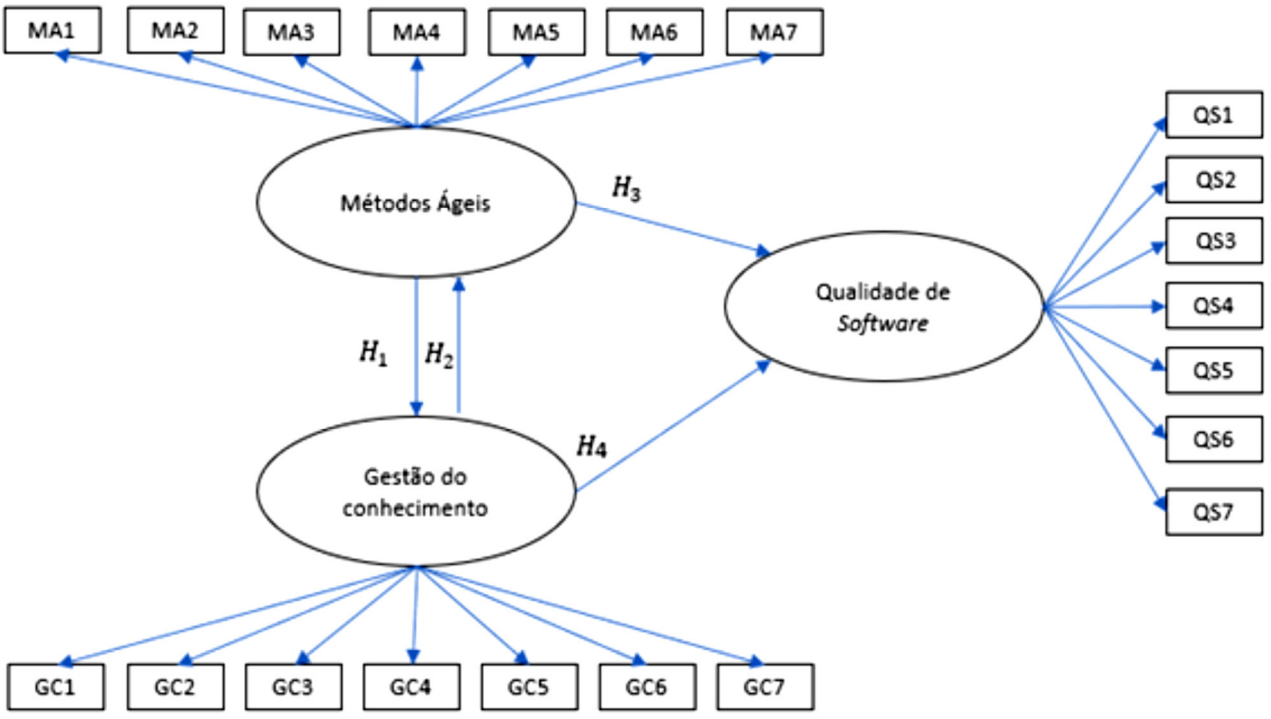

Figura 2: Modelo conceitual e as siglas do questionário
Fonte: elaborado pelos autores (2019).

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.

Para verificar a linearidade, foi realizado o teste de 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 & Lauro (2005), é uma abordagem mais flexível que a tradicional baseada na matriz de covariância.

A Tabela 1 apresenta a relação das afirmativas adotadas como índices do modelo de mensuração e o referencial teórico que lhes deu suporte.

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., Black, Babin, Anderson, & Tatham, 2009). Para verificar a dimensionalidade dos constructos, foi utilizado o critério de Kaiser (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).

A Tabela 2 demonstra a análise da validade convergente, a validade discriminante, dimensionalidade e a confiabilidade dos constructos. Desses resultados, cabe ressaltar: a) em todos os constructos os índices de confiabilidade AC ou CC foram superiores a 0,70, comprovando-se, assim, a confiabilidade dos constructos; b) pelo critério de Kaiser, todos os constructos foram unidimensionais; c) todos os constructos apresentaram AVE superior a 0,50, indicando validação convergente; d) houve validação discriminante para todos os constructos analisados, uma vez que as variâncias compartilhadas foram menores que as AVEs.

4 Resultados

Nesta seção são apresentados os resultados da pesquisa.

4.1 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 significativa linearidade 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.

Tabela 1: Relação das siglas por constructo

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 *et al.* (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*, pelo 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) Bernardo e Kon (2008)
	MA7	Os líderes de desenvolvimento de *software* de meu ambiente de trabalho atuam mais como facilitadores.	Gomes *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 al.* (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, Inocente e Miura (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 tem 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 pelos autores (2019).

Tabela 2: Validação do modelo de mensuração

Constructos	Itens	AC[1]	CC[2]	Dim[3]	AVE[4]	VCM[5]
Métodos Ágeis	7	0,77	0,84	1	0,42	0,40
Gestão do Conhecimento	6	0,90	0,92	1	0,67	0,52
Qualidade de Software	7	0,88	0,90	1	0,57	0,52

[1]Alfa de Cronbach, [2]Confiabilidade composta, [3]Dimensionalidade, [4]Variância extraída; [5]Variância compartilhada máxima.

Fonte: elaborada pelos autores (2019).

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 de Anderson-Darling, Shapiro-Wilk e Mardia para verificar a normalidade univariada e multivariada. Esses testes acusaram que os itens não foram normalmente distribuídos (valor-p < 0,001) de forma univariada, tampouco multivariada.

4.2 Descrição dos constructos

A Tabela 3 demonstra a média, o desvio-padrão (DP) e o intervalo de confiança de cada item. Intervalos estritamente inferiores a três indicam que os indivíduos tendem a discordar, enquanto que intervalos estritamente acima de três significam que os indivíduos tendem a concordar.

Para o constructo "métodos ágeis", a questão MA1 (interação) apresentou a maior média (4,5 IC [4,37; 4,62]) entre todos os itens do constructo. Por outro lado, a questão MA3 (participação do cliente) apresentou a menor média (3,6 IC [3,40; 3,79]) de todo o constructo. Por fim, em todos os itens desse constructo os indivíduos tenderam a concordar, visto que todos os intervalos de confiança foram estritamente superiores a três.

Em todos os itens os indivíduos tenderam a concordar no constructo "gestão do conhecimento", exceto para o item GC5 (inovação). As questões GC2 (3,9 IC [3,77; 4,12]) e GC3 (3,9 IC [3,68; 4,04]) (colaboração e compartilhamento) apresentaram as maiores médias entre todos os itens do constructo. Por outro lado, a questão GC5 exibiu a menor média (3,2 IC [2,99; 3,41]) de todo o constructo. 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.

Em relação ao constructo "qualidade do *software*", a questão QS1 (acompanhamento do desenvolvimento) teve a maior média (3,9 IC [3,67; 4,03]) entre todos os itens do constructo, enquanto que a questão QS6 (gerência de riscos) obteve 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 inferior a três. Sua média foi significativamente menor que a média dos demais itens, A média do item QS1 foi significativamente

Tabela 3: 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: elaborada pelos autores (2019).

maior que a média dos itens QS3, QS4, QS6 e QS7, pois não houve sobreposição do seu intervalo de confiança.

4.3 Modelagem de equações estruturais (PLS)

4.3.1 Resultado do modelo de mensuração

A Tabela 4 mostra os pesos, as cargas fatoriais e comunalidades no modelo inicial e final de mensuração.

A questão GC2, que pertence ao constructo gestão do conhecimento, apresentou carga fatorial abaixo de 0,50, demonstrando que ela não possui correlação significativa com esse constructo e foi, dessa forma, retirada do modelo final. Todas as outras questões exibiram carga fatorial acima de 0,50 e, desse modo, foram significativas para a formação dos indicadores.

4.3.2 Resultado do modelo proposto

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.

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. De forma semelhante, houve influência significativa (valor-p < 0,001) e positiva (β = 0,629, IC = [0,51; 0,74]) da gestão do conhecimento sobre os métodos ágeis. 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*. Da mesma forma, houve influência significativa (valor-p <

Tabela 4: Pesos, cargas fatoriais e comunalidades no modelo inicial e final de mensuração

Constructos	Modelo inicial					Modelo final			
	Itens	Peso	IC - 95%[1]	CF[2]	Com[3]	Peso	IC - 95%[1]	CF[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: elaborada pelos autores (2019).

0,001) e positiva (β = 0,602, IC = [0,41; 0,78]) da gestão do conhecimento sobre a qualidade do *software*.

O modelo estrutural pode ser visto na Figura 3.

5 Discussão dos Resultados

Em relação ao constructo "métodos ágeis", os resultados apresentados na pesquisa indicam que, se de um lado, os métodos ágeis realmente favorecem a interação no ambiente de trabalho, de outro, o cliente não tem participação tão ativa no desenvolvimento de *software*, conforme previsto pela metodologia ágil.

Sobre o constructo "gestão do conhecimento", a pesquisa revelou que o conhecimento está em poder dos envolvidos no processo de desenvolvimento de *software*, embora haja grande interação entre eles (colaboração e compartilhamento). Ela revelou que, apesar desse conhecimento e dessa interação serem valorizados, a retenção e a gestão do conhecimento ainda não recebem o devido valor nas empresas de TI. Por fim, a pesquisa demonstrou que os ambientes de desenvolvimento de *software* não incentivam, pelo menos como deveriam, a criatividade e a inovação.

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.

A gestão do conhecimento, por sua vez, também conseguiu explicar 39,6% da variabilidade dos métodos ágeis, confirmando a hipótese H_2 de que a gestão do conhecimento influencia positivamente os métodos ágeis.

Sendo assim, quanto maior a utilização dos métodos ágeis, maior tende a ser a necessidade da gestão do conhecimento e, também, quanto maior a utilização da gestão do conhecimento, maior tende ser a necessidade dos métodos ágeis. 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.

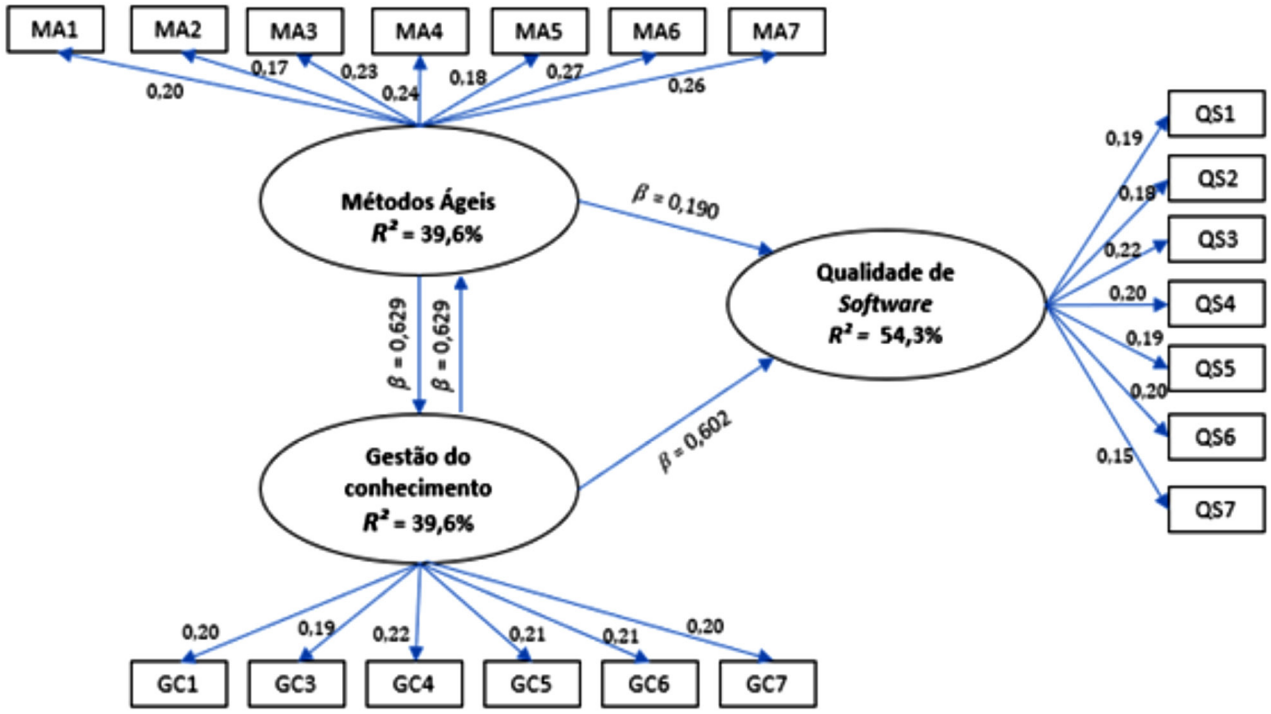

Figura 3: Modelo conceitual e resultados da pesquisa
Fonte: elaborada pelos autores (2019).

No tocante ao constructo "qualidade do *software*", a pesquisa apurou que, embora as atividades de desenvolvimento de *software* sejam acompanhadas (acompanhamento do desenvolvimento), seu desempenho não está sendo medido como deveria e não está sendo feito controle de qualidade como esperado. Ela revelou, ainda, que uma gerência efetiva dos riscos dos projetos ainda não está sendo exercida nas empresas de TI.

Os métodos ágeis e a gestão do conhecimento conseguiram explicar 54,3% da variabilidade da qualidade de *software*.

Sendo assim, quanto mais se utilizar os métodos ágeis, melhor tende a ser a qualidade de *software*, confirmando a hipótese H_3 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.

Da mesma forma, quanto maior a gestão do conhecimento, melhor tende a ser a qualidade de *software*, confirmando a hipótese H_4 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.

A Tabela 5 relata o resultado das hipóteses do modelo estrutural.

Tabela 5: Validação das hipóteses do modelo

	Hipótese	Resultado
H1	Os métodos ágeis influenciam positivamente a gestão do conhecimento.	Confirmada
H2	A gestão do conhecimento influencia positivamente os métodos ágeis.	Confirmada
H3	Os métodos ágeis influenciam positivamente a qualidade de *software*.	Confirmada
H4	A gestão do conhecimento influencia positivamente a qualidade de *software*.	Confirmada

Fonte: elaborada pelos autores (2019).

6 Conclusão

O objetivo deste estudo foi colaborar na investigação da influência dos constructos métodos ágeis e gestão do conhecimento, na qualidade de *software*. Com esse intuito e baseado no referencial teórico levantado, um modelo conceitual foi proposto e um questionário foi criado para testar o modelo. Esse questionário foi respondido por 110 indivíduos e, por meio da técnica PLS, a influência foi confirmada.

Os métodos ágeis, que a pesquisa da Versionone (2018) mostrou estarem sendo grandemente utilizados atualmente, neste trabalho manifestaram também influência na qualidade de *software.* Cabe, então, 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, e vice-versa, sugerindo que os gestores devem dar atenção também a esse relacionamento, que também pode influenciar, ainda que de forma indireta, a qualidade de *software*.

Por fim, um fato que chama a atenção na pesquisa é a maior influência da gestão do conhecimento na qualidade de *software* ($\beta = 0{,}602$), se comparada com a utilização dos métodos ágeis ($\beta = 0{,}190$). Os indicadores GC1 e GC3 explicitam esse fato e sugerem que os gestores de equipes de desenvolvimento de *software* devem considerar mais intensamente os fatores associados 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 GC6 e GC7.

Apesar de ser baseada em um consistente referencial teórico, a principal limitação desta pesquisa é a não realização de uma análise fatorial exploratória (AFE) para identificar previamente os fatores constituintes dos constructos, para aumentar a segurança no prosseguimento da pesquisa. Entre as limitações, ainda podem ser citados o tamanho da amostra e o perfil dos respondentes. O tamanho da amostra foi significante se avaliado estatisticamente, mas não representa todos os segmentos da 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.

Como sugestão para outros estudos, 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. Outra sugestão é a avaliação dos resultados, de forma qualitativa, por profissionais de grande expressão ou influência no desenvolvimento de *software*.

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*. Na perspectiva acadêmica, cabe aos cientistas fazerem mais pesquisas e ampliarem o conhecimento sobre o tema. Na perspectiva 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.

Agradecimentos

Deixamos nossos agradecimentos aos revisores do artigo, que muito contribuíram para a sua estruturação final, deixando o andamento do texto mais harmonioso e distribuição melhor da sequência metodológica e apresentação dos resultados.

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, D. M., Clouse, A., & Turner, R. (2008). CMMI® Distilled: A Practical Introduction to Integrated Process Improvement (3. ed.). Addison Wesley Professional Pub.

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.

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

Beck, K. (2000). *Extreme programming explained*. 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.

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 *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.). Retrieved from: 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.

Cruz, F. (2013). *Scrum e PMBOK unidos no Gerenciamento de Projetos* (1. ed.). Brasport.

da Silva, M. P., & Bustamante, B. L. F. (2017). TRIZ: ferramenta CASE para produção de software com metodologia ágil para pequenas e médias equipes. *Revista H-TEC Humanidades e Tecnologia*, *1*(1), 6–32.

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

de Almeida, G. A. M. (2017). *Fatores de escolha entre metodologias de desenvolvimento de software tradicionais e ágeis* (Universidade de São Paulo). https://doi.org/10.1007/s00441-016-2360-7.

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. da M. (2010). Metodologias ágeis no contexto de desenvolvimento de software: XP , Scrum e Lean. (p. 26). *Universidade Estadual de Campinas*.

Feller, N. J. (2016). Disseminando a cultura de teste e qualidade de software no CPD - UFRGS. *Anais Do Workshop de Tecnologia de Informação e Comunicação das Instituições Federais de Ensino Superior*, *4*.

Fialho, F. A. P., & Ponchirolli, O. (2005). Gestão estratégica do conhecimento como parte da estratégia empresarial. *Revista FAE, Curitiba*, *8*(1), 127–138.

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

Gomes, A., Willi, R., & Rehem, S. (2014). O Manifesto Ágil. *In:* R. Prikladnicki, R. Willi, F. Milani (col.). *Métodos ágeis para desenvolvimento de software* (p. 311). Bookman.

Gonzalez, I. V. D. P., & de 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. Recuperado de: http://eds.a.ebscohost.com/eds/pdfviewer/pdfviewer?vid=6&sid=98e12452-1d41-4590-8237-09409eb100ce%40sessionmgr4010&hid=4113.

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.

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)*. Sage Publications.

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. *International Conference on Information Systems (ICIS)*.

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. *Perspectivas em Ciência da 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. *2009 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)*.

Martins, J., & de Menezes, R. M. T. (2016). Metodologia ágil: framework scrum – em gerenciamento de projetos de software. *Cognitio/Pós-Graduação UNILINS 1.7*, 1–13.

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* (01–1997 ed.). Rio de Janeiro: Campus.

Nunnally, J. C., & Bernstein, I. H. (1994). *Psychometric Theory* (3. 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*. (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.

Pinto, E. B., Vasconcelos, A. M., & Lezana, Á. G. R. (2014). Abordagens do PMBOK e CMMI sobre o sucesso dos projetos de softwares. *Revista de Gestão e Projetos - GeP*, *5*(1), 55–70.

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

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.

Ribeiro, J. S. de A. N., Soares, M. A. C., Jurza, P. H., Ziviani, F., & Neves, J. T. de R. (2017). Gestão do conhecimento e desempenho organizacional: integração dinâmica entre competências e recursos. *Perspectivas em Gestão & Conhecimento*, *7*, 4–17.

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.

Santos, V. S., & Canedo, E. D. (2014). Utilização da metodologia ágil no desenvolvimento de software na Justiça Eleitoral Brasileira. *Anais Do X Simpósio Brasileiro de Sistemas de Informação - SBC.*, 345–356.

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.

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. Associação para Promoção da Excelência do *Software* Brasileiro (2012). *MPS.BR - Melhoria de Processo do Software Brasileiro Guia Geral*.

Softex. Associação para Promoção da Excelência do *Software* Brasileiro (2016). *MPS.BR - Melhoria de Processo do Software Brasileiro - Guia Geral MPS de Software*.

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

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.

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.

Vasanthapriyan, S., Tian, J., & Xiang, J. (2015). A survey on knowledge management in software engineering. *Proceedings - 2015 IEEE International Conference on Software Quality, Reliability and Security-Companion*, 237–244. https://doi.org/10.1109/QRS-C.2015.48.

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

Viana, D., Souza, C. de, & Conte, T. (2014). Facilitando a aprendizagem organizacional em melhorias de processo de software. *Workshop de Teses e Dissertações em Qualidade de Software.*, 43–50. Recuperado de: http://tede.ufam.edu.br/handle/tede/4153.

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.