APLICAÇÃO DO PORTAL DO CONHECIMENTO E CAFÉ DO CONHECIMENTO PARA APRIMORAR A COMPREENSÃO DOS PROFISSIONAIS DE TI SOBRE DÍVIDA TÉCNICA, ESTUDOS PRELIMINARES

Marcelo Antonio e Silva Filho[1];
Iara Carnevale de Almeida[2];
João Messias Pereira Lenço[3]

Abstract. *Agile methodology has been increasingly adopted for software development. However, little information on the application of processes and practices of agile methodologies end up having repercussions on Technical Debt (TD). This study aims to detect the understanding of software development professionals about DT in agile development. This study is applied in nature with a quantitative approach through exploratory research, data collection was application of an online questionnaire followed by statistical analysis. Sampling was non-probabilistic with 33 respondents, it can be observed that professionals have different reasons for the occurrence of TD. It was also detected that there is no organizational culture. This study proposes the Knowledge Portal and the Knowledge Café to disseminate experiences among professionals.*

Keywords: APO, Knowledge Management, Agile Methods.

Resumo. Cada vez mais tem sido adotado a metodologia ágil para desenvolvimento de software. Contudo, pouca informação sobre aplicação de processos e práticas das metodologias ágeis acabam por repercutir em Dívida Técnica (DT). Este estudo tem como objetivo detectar a compreensão dos profissionais de desenvolvimento de software sobre DT no desenvolvimento ágil. Este estudo é de natureza aplicada com abordagem quantitativa através de pesquisa exploratória, a coleta de dados foi aplicação de questionário online seguida de análise estatística. Amostragem foi não probabilística com 33 respondentes, pode-se observar que os profissionais apresentam diferentes motivos para a ocorrência de DT. Detectou-se também que não há cultura organizacional. Este estudo propõe o Portal do conhecimento e o Café do Conhecimento para disseminar experiências entre profissionais.

Palavras-chave: APO, Gestão do Conhecimento, Métodos Ágeis.

1. INTRODUÇÃO

A comunidade de desenvolvimento de software tem sido, cada vez mais, requisitada para automatização de processos em empresas. Para atender às demandas atuais, o processo de

[1] Graduado. Curso de Engenharia de Software, UniCesumar, Maringá (PR). Email: marceloasf1999@gmail.com.
[2] Orientadora, Doutora, docente no Programa de Pós-Graduação em Gestão do Conhecimento nas Organizações e bolsista de Produtividade do ICETI, UniCesumar, Maringá (PR). Email: iara.almeida@unicesumar.edu.br.
[3] Mestrando do Programa de Pós Graduação em Gestão do Conhecimento nas Organizações, UniCesumar (PR), Maringá (PR). Email: messias1901@gmail.com.

desenvolvimento de software ágil (ou apenas, desenvolvimento ágil) tem sido uma das principais escolhas das empresas de software, atendendo assim, corretamente e rapidamente, às necessidades de clientes (Pressman, 2011). Independente do porte da empresa de software, processos e métodos devem ser aplicados para garantir a qualidade do produto de software (Ribeiro, 2016). Pressman (2011) evidencia que a exigência por qualidade de software surgiu quando o software passou a se tornar cada vez mais integrado ao nosso dia a dia pois, na década de 1990, muitas das principais empresas perceberam que bilhões de dólares estavam sendo desperdiçados por ano em software que não apresentava as características e funcionalidades necessárias. Devido a isso, passou a ser exigida uma maior atenção à qualidade dos produtos de software entregues. Essa preocupação com a qualidade impactou não apenas os processos de desenvolvimento, mas também a cultura organizacional das empresas, que se viram desafiadas a construir um ambiente favorável às ações voltadas à garantia de qualidade de software. A cultura organizacional é um padrão de premissas desenvolvido por um grupo de pessoas que compartilham das mesmas crenças e valores, de modo que, esse padrão possa ser replicado por todos dentro da organização e, principalmente, possa ser ensinado aos novos membros do grupo como sendo a maneira correta de perceber, pensar e agir dentro da organização (Schein, 1994). Ribeiro (2016) salienta que processos e métodos foram concebidos para que seja possível incorporar mudanças no ambiente de trabalho de equipes de desenvolvimento de software. Das propostas atuais, salientam-se a Gestão de Projetos de Scrum e as boas práticas de programação do *eXtreme Programming* (XP). Conforme Santos (2015), o Scrum é utilizado desde o início da década de 90 e propõe que a equipe de desenvolvimento (ou Time) foque em problemas adaptativos complexos e na entrega de produtos com valor para o cliente.

No Brasil, devido à pandemia da Covid-19, muitas empresas migraram para o teletrabalho como medida de sobrevivência financeira e prevenção, a fim de evitar um maior número de contágio (Lizote, et al., 2021). Na TI, assume-se que uma das desvantagens desta modalidade é que, pela ausência de encontros presenciais entre os profissionais da área, há diminuição na troca de conhecimento das experiências desses profissionais.

Um dos temas que merece atenção é a Dívida Técnica (DT) que, conforme Cunningham (1992), foi definida pensando na necessidade de se reconhecer, em longo prazo, os potenciais efeitos negativos em um software quando há entrega de código imaturo. Portanto, percebe-se a importância de compreender o que é DT. Torna-se, também, necessário realizar estudo que aponte qual tem sido a percepção dos profissionais de equipe de desenvolvimento sobre a

importância de se evitar a DT no desenvolvimento de software ágil. Caso necessário, deve-se propor uma solução para aprimorar o conhecimento desses profissionais.

Conforme Batista (2004), os métodos e ferramentas da Gestão do Conhecimento podem ser compreendidos como sendo estratégias de implementação que permitem tanto captar quanto reutilizar conhecimento estruturado; captar e compartilhar lições aprendidas com a prática; identificar fontes e redes de expertise; estruturar e mapear conhecimentos necessários para aumentar o desenho; mediar e controlar o valor econômico do conhecimento; e sintetizar e compartilhar advindo de fontes externas.

Dalkir (2013) indica que o ciclo da GC ocorre através da criação/captura, compartilhamento/disseminação e aquisição/aplicação do conhecimento. A criação/captura trata da identificação, seguida pela codificação tanto do conhecimento interno como do conhecimento a partir do ambiente externo. Esta mesma autora indica que compartilhamento/disseminação permite uma avaliação do conhecimento criado/capturado. Na etapa da aquisição/aplicação, dá-se somente após a validação e avaliação do conhecimento como relevante sendo, então, inserido no armazenamento e utilizado nas ações pessoais e organizacionais. Davila et al. (2015) reforça que ocorre uma ligação entre o conhecimento e os seus detentores, de forma a contribuir entre os usuários e membros da organização.

Neste estudo assume-se que, portanto, em situações nas quais há dificuldade de promover o ciclo da GC, pode-se analisar métodos e ferramentas da GC de forma a melhorar este ciclo. Além disso, acredita-se que, a nível de organização, pode-se melhorar inclusive a cultura organizacional. Este artigo apresenta resultados que apontam qual tem sido a percepção dos profissionais de equipe de desenvolvimento sobre a importância de se evitar a DT no desenvolvimento de software ágil. Na sequência, apresenta-se uma solução para aprimorar o conhecimento desses profissionais sobre DT. Salienta-se que esta pesquisa está em desenvolvimento e, portanto, neste artigo são apresentados resultados até aqui desenvolvidos.

2. DÍVIDA TÉCNICA EM PROCESSO DE DESENVOLVIMENTO ÁGIL

Os Métodos Ágeis (MA) foram concebidos devido a uma necessidade de incorporar mudanças no ambiente de trabalho: em vez de esforços de planejamento antecipado e produção sem comunicação com o cliente, os MA propunham satisfação dos clientes por meio de entregas mais rápidas e frequentes. O desenvolvimento deve se concentrar em código funcional, a colaboração entre as equipes deve ser intensiva e as equipes auto-organizáveis (Ribeiro, 2016). Além disso, os MA também seguem um conjunto de princípios e valores, os quais foram

reunidos no Manifesto Ágil (Beck et al, 2001), mas algumas metodologias ágeis surgiram antes do manifesto, que é considerado uma das bases dos MA, como o Scrum no início da década de 1990 (Gonçalves, 2018).

Dentre uma das principais causas para aquisição de DT, há ênfase em entrega rápida de funcionalidades (Gonçalves, 2018). Ao exercitar a entrega contínua e adiantada, cria-se a possibilidade de enviar ao cliente o produto com código imaturo (isto é, código com bugs ou inacabados). Caso o produto seja enviado, é o momento em que a DT é criada (Lavazza, 2018). Nota-se que uma DT pode adiantar a entrega e/ou acelerar o desenvolvimento, mas deve ser paga o quanto antes. Se houver acúmulo de DT, será difícil gerir as correções que devem ser feitas. Portanto, código imaturo é um problema que pode ser entregue ao cliente, devido à decisão (da equipe, do cliente ou de ambos) de ter uma entrega mais rápida.

A Dívida Técnica - DT (em inglês, *Technical Debt*) é um termo criado por Cunningham (1992), que definiu o conceito da DT em uma experiência do OOPSLA (em inglês, *Object-Oriented Programming, Systems, Languages & Applications*) em 1992. No Quadro 1, são apresentadas algumas definições deste termo.

Quadro 1 - Definições de "Dívida Técnica"

Definição. Autor.
A criação de DTs pode acelerar o desenvolvimento em curto prazo, mas, pensando em longo prazo, a má qualidade dos artefatos desenvolvidos gera um custo maior devido a esforços de manutenção e força de trabalho extra para a manutenção. (Guo, 2009).
As decisões que são tomadas e acabam gerando DT são justificadas por saberem em qual estágio estão e à qual querem chegar. Assim, é preciso gerenciar o que é mais importante, caso necessário, adquirindo dívidas e as organizando para saber quando se deve reembolsar essas DTs. (Brown et al., 2010).
A DT varia de equipe para equipe. Logo, existe a necessidade de a equipe refletir sobre qual tipo de dívida se deve considerar com uma maior preocupação, sempre monitorando as suas DTs, pois em longo prazo elas podem gerar problemas para a equipe. (Shull, 2011).
Definem como uma metáfora que faz alusões a eventos em que a qualidade técnica é negligenciada para se obter um benefício em curto prazo ou algum ganho competitivo. (Lim; Taksande; Seaman, 2012).
A DT tem sido usada como uma metáfora dentro da comunidade de desenvolvimento ágil, com sua utilidade reconhecida para comunicação técnica e comunicação entre engenheiros de software e executivos. O conceito de DT tem ganhado força como uma forma de focar no gerenciamento em longo prazo de problemas gerados por compromissos de curto prazo. (Santos, 2015).
Define que, em sistemas de software, a DT é uma coleção de construções de design e implementação que são convenientes em curto prazo, mas que acabam complicando um contexto de alterações futuras. (Avgeriou et al., 2016).

É reconhecida como um problema crítico no setor de desenvolvimento de software atual e, caso não seja verificada, a DT pode causar grandes excedentes de custos, ou seja, gerando grandes custos de manutenção devido a problemas internos de qualidade e a incapacidade de adicionar novas alterações ao sistema, podendo levar a um ponto de crise em que é necessário realizar uma refatoração grande e muito cara ou até uma substituição completa do software. (Besker et al., 2018).
É uma metáfora para artefatos imaturos, incompletos ou inadequados no ciclo de vida do desenvolvimento de um software, que causam custos mais altos, uma qualidade e escalabilidade baixa do produto em longo prazo. (Mendes et al., 2019).

Fonte: Elaborado pelos autores.

Analisando o Quadro 1, percebe-se a evolução do termo DT: de algo simples e direto para algo mais complexo com o passar do tempo, com os autores apresentando diversos pontos de vista e diferentes estratégias para a gestão dessa dívida, tanto para curto como para longo prazo. Gonçalves (2018) salienta que falta uma concepção sólida de como se pode catalogar e/ou gerir as DTs no ambiente de desenvolvimento ágil. Na sequência, O Quadro 2 apresenta os tipos de Dívida técnica, proposto por Li, Avgeriou e Liang (2015).

Quadro 2 - Tipos de Dívidas Técnicas

Tipos	**Descrição**
DT de Requisitos	Refere-se à distância da especificação de requisitos ideal do sistema e a aplicação efetiva do sistema.
DT Arquitetural	Refere-se a decisões de arquitetura que possuem vínculo com aspectos internos de qualidade, como a manutenção do sistema.
DT de Projeto	Refere-se a atalhos técnicos que foram criados e são tomados no projeto.
DT de Código	Refere-se a código mal feito, o qual viola as boas práticas e regras de codificação, como duplicação de código e código com alta complexidade ciclomática.
DT de Teste	Refere-se a atalhos que foram criados e são tomados nos testes do sistema, como a falta de testes no sistema (por exemplo, testes unitários e testes de aceitação).
DT de Construção	Refere-se a falhas no sistema de construção de um software, que tornam seu processo de construção excessivamente complexo e difícil.
DT de Documentação	Refere-se à documentação incompleta, insuficiente ou desatualizada em qualquer aspecto do processo de desenvolvimento do software, como uma documentação de arquitetura que ficou desatualizada ou um pedaço de código complexo ou importante que não possui um comentário com uma explicação.
DT de Infraestrutura	Refere-se a configurações sub-ótimas dos processos relacionados ao desenvolvimento, às tecnologias e outros fatores. Tal configuração afeta negativamente a capacidade da equipe de desenvolver o sistema com qualidade.
DT de Controle de Versão	Refere-se aos problemas de versionamento de código, como contribuições desnecessárias.

DT de Defeito	Refere-se aos defeitos, erros ou falhas que são encontrados em sistemas de software.

Fonte: elaborado pelos autores. Adaptado de Li, Avgeriou e Liang (2015)

Cunningham (1992) observa que, em alguns momentos, o código imaturo pode ser liberado para obter alguma vantagem imediata. Portanto, a DT é criada quando se envia o código imaturo. Essa DT pode até acelerar o processo de desenvolvimento do software, mas deve ser paga o quanto antes pois, se houver acúmulo de DT, o software se tornará desorganizado (Lavazza et al., 2018).

Salienta-se que o Manifesto Ágil, proposto por (Beck et al, 2001), indica os seguintes valores que guiam o desenvolvimento visando melhorar a produção de software: indivíduos e interações mais do que processos e ferramentas; software em funcionamento mais do que documentação abrangente; colaboração com o cliente mais do que negociação de contratos; e responder a mudanças mais do que seguir um plano. Além disto, dos doze princípios apresentados pelo Manifesto Ágil, salientam-se neste estudo os que podem potencializar a geração de DT: "A nossa maior prioridade é satisfazer o cliente por meio da entrega contínua e adiantada de software com valor agregado"; "Entregar frequentemente software funcionando, de poucas semanas a poucos meses, com preferência à menor escala de tempo"; "Software funcionando é a medida primária de progresso"; e "Simplicidade – a arte de maximizar a quantidade de trabalho que não precisou ser feito". Esses princípios podem gerar DT, uma vez que visam a uma entrega mais rápida, e até imatura, do produto ao cliente.

3. CULTURA ORGANIZACIONAL

As preocupações com os estudos referentes a cultura organizacional se potencializaram a partir da década de 80, desde então, discute-se sobre a importância do tema, salientando o quanto a cultura organizacional influencia em diversos fatores dentro da organização, principalmente, no que diz respeito às ações estratégicas e na efetividade dos processos. Para Schein (1984), a compreensão da cultura organizacional faz parte do processo administrativo, de modo que, seu estudo está intrínseco e diretamente relacionado ao processo de gestão. Estabelecer uma cultura organizacional é fundamental para o bem estar do indivíduo e da própria organização, isto porque, sua existência na empresa consiste em um conjunto de crenças e valores de pessoas que compartilham de um mesmo sentimento, de forma individual e coletiva (Ungari & Rodrigues, 2020). Seja para o indivíduo ou para a própria empresa, seria perturbador conviver em um ambiente organizacional, no qual as crenças e valores de uma das partes fosse contrária a outra.

A cultura organizacional pode ser compreendida a partir de duas perspectivas distintas: a do indivíduo e a da organização. Da perspectiva do indivíduo, entende-se que cada pessoa é formada por uma bagagem emocional que representa sua forma de agir e de se relacionar com o ambiente ao qual está inserido. Esta bagagem é resultado do que foi aprendido ao longo dos anos por meio das interações entre os indivíduos em cada ambiente, como por exemplo, no ambiente familiar, no trabalho, grupo de amigos, etc (Hofstede et al., 2005). Da perspectiva organizacional, em princípio considera-se o que é idealizado pelos fundadores da organização a partir das suas crenças e valores, de modo institucionalizado, resultando em um processo de cultura imposta (Schein, 1984).

Para Ungari e Rodrigues (2020, p. 171), "entende-se que os sistemas organizacionais são produzidos a partir das interações dos indivíduos que formam a organização". Dessa forma, compreende-se que, os sujeitos influenciam e são influenciados pelo ambiente organizacional, absorvendo aspectos culturais e também produzindo-os.

4. METODOLOGIA

Esta pesquisa é de natureza aplicada, pois tem como motivação a necessidade de produzir conhecimento para aplicação de seus resultados, objetivando assim contribuir de forma prática para a solução de um determinado fenômeno (Krueger & Casey, 2009). Realiza abordagem qualitativa pois pretende obter perspectivas e pontos de vista dos participantes, considerando suas emoções, prioridades, experiências, significados e outros aspectos subjetivos (Sampieri et. al., 2013). Além disso, configura-se como pesquisa exploratória, pois busca descrever o significado de um conceito, interpretar o que acontece em uma atividade ou gerar uma teoria sobre um processo (Clark & Creswell, 2010). Para satisfazer os objetivos propostos, foi realizada pesquisa bibliográfica na construção do referencial teórico, que configura um tipo de estudo e análise a partir de documentos de domínio científico relacionados ao objeto de estudo, permitindo assim, uma melhor compreensão do tema desta pesquisa (Oliveira, 2007). Na sequência, para compreender a percepção dos profissionais da área de informática, foi feita coleta de dados através de questionário online com posterior análise estatística. A utilização de questionário é uma "técnica de investigação composta por um número mais ou menos elevado de questões apresentadas por escrito às pessoas, tendo por objetivo o conhecimento de opiniões, crenças, sentimentos, interesses, expectativas, situações vivenciadas etc." Gil (1999, p.128).

5. RESULTADOS OBTIDOS E DISCUSSÃO

O questionário foi aplicado em outubro de 2021, de forma anônima, com período de preenchimento de 20 dias. Inicialmente foi encaminhado por e-mail para algumas empresas de Maringá/PR, mas houveram poucos respondentes. Posteriormente, foi publicada solicitação de colaboração nas contas do LinkedIn dos pesquisadores. Ao final, houveram 33 respondentes. Portanto, esta amostra é não probabilística e tem a finalidade de criar maior familiaridade com o problema e torná-lo mais explícito.

A primeira seção do questionário foi elaborada para conhecer o perfil dos respondentes e de seu ambiente de trabalho. É composta por sete perguntas objetivas, apresentadas no Quadro 3.

Quadro 3 - Perfil profissional e a empresa em que trabalha

Perguntas
PP01: Indique a sigla do seu estado.
PP02: Quanto tempo de experiência com desenvolvimento de software você possui?
PP03: Qual o tamanho da empresa de software em que você trabalha?
PP04: Qual é o seu cargo dentro desta empresa?
PP05: Indique qual é a metodologia de gestão de projetos que a sua equipe adota.
PP06: Quais são os tipos de ferramentas utilizados durante o processo de desenvolvimento de software? Você pode escolher mais de uma ferramenta.
PP07: Quais são as ferramentas utilizadas pela sua equipe durante o processo de desenvolvimento de software? Você pode escolher mais de uma ferramenta.

Fonte: Elaborado pelos autores.

Referente à pergunta **PP01**, a maioria dos respondentes é do estado Paraná, Brasil (97%). As respostas à pergunta **PP02** indicam que a maioria dos respondentes possui mais de um ano de experiência na área de desenvolvimento (94%) e os demais possuem menos de um ano de experiência (6%). As respostas às perguntas **PP03 e PP04**, a maioria dos respondentes estão inseridos em ambientes profissionais de empresas de grande porte (87,9%), e todos possuem cargos que atuam diretamente no ciclo de desenvolvimento do software, sendo a grande maioria Programadores (60,6%).

Referente a pergunta **PP05**, a maior parte dos respondentes apontam que as suas equipes utilizam métodos ágeis para gerir seu processo de desenvolvimento, com a maioria utilizando o Scrumban (54,5%). Porém, ao analisar os resultados individuais do Scrum e do Kanban, vê-

se que o Kanban tem maior adesão das equipes (36,4%); o Scrum apresenta 6,1%. A pergunta **PP06** pretende compreender os tipos de ferramentas utilizadas para o auxílio do processo de desenvolvimento das equipes: ferramenta de gerenciamento de projeto (93,9%); ferramenta para gestão de versionamento de código (97%), ferramenta para integração contínua e desenvolvimento contínuo (81,8%) e ferramenta para avaliar a qualidade do código (78,8%).
Finalmente, as respostas à pergunta **PP07** pretende saber quais são as ferramentas utilizadas pela sua equipe durante o desenvolvimento de software: todos os respondentes utilizam pelo menos um tipo de ferramenta que auxilia o gerenciamento do projeto onde a mais utilizada para gerenciar o projeto é o Jira (84,8%). Nota-se que a maioria das equipes utilizam algumas das ferramentas que auxiliam o versionamento de código: GitLab (66,7%) e o GitHub (24,2%). Quanto às ferramentas de integração contínua e desenvolvimento, destacam-se o Jenkins (63,6%) e o Azure DevOps (9,1%). Porém, algumas das equipes que utilizam o GitLab também podem usufruir desse mesmo auxílio, por meio de uma das ferramentas do GitLab, que é o GitLab CI/CD. Outra ferramenta que é utilizada pela maioria das equipes e que é muito importante para auxiliar na qualidade do código é o SonarQube (66,7%), além das ferramentas que auxiliam diretamente no código, processo ou pendências relacionadas ao produto.
A segunda seção foi elaborada para detectar qual é a compreensão acerca de DT, onde foram apresentados os tipos de DTs como alternativas para algumas das perguntas aos respondentes. Esta seção é composta por sete perguntas objetivas, apresentadas no Quadro 4.

Quadro 4 - Compreensão do respondente acerca da DT

Perguntas
DT01: O termo DT já foi discutido dentro do seu ambiente de trabalho? Você pode escolher mais de uma resposta.
DT02: Nestes últimos 5 anos, você já experienciou situações nas quais foram geradas alguma dessas dívidas técnicas?
DT03: Indique, pela sua experiência, quais são as DTs que mais ocorrem.
DT04: Indique, pela sua experiência, qual a DT que demanda mais tempo para ser resolvida.
DT05: Indique, pela sua experiência, qual a DT que demanda uma maior interação com o cliente.
DT06: Indique, pela sua experiência, quais são as DTs que acabam por aumentar a chance de gerar uma nova DT.
DT07: Quando identificada uma DT no projeto de sua equipe, alguma ação é tomada?

Fonte: elaborado pelos autores.

Observou-se, pelos resultados extraídos da pergunta **DT01**, que o termo DT já foi discutido na maioria das equipes dos respondentes (75,8%). Os respondentes já presenciaram essas discussões nos seus ambientes de trabalho a partir de reuniões com os gestores/líderes das equipes (51,5%), em conversas informais (48,5%) e capacitações (24,2%). No entanto, como essa era uma pergunta de múltipla escolha, podemos considerar esses oito respondentes que marcaram apenas a alternativa "Não" como um valor considerável (24,2%).

Analisando os resultados da pergunta **DT02**, nota-se que a grande maioria dos respondentes já presenciaram alguma situação que gerou uma DT (90,91%). Os resultados de cada tipo de DT são: DT de Código (78,8%), DT de Teste (72,7%), DT de Documentação (60,6%) e DT Arquitetural (51,5%), DT de Requisitos (48,5%), DT de Infraestrutura (39,4%) e DT de Controle de Versão (24,2%). Além desses resultados, a opção "Nunca percebi" obteve 9,1% e "Não experienciei" obteve 6,1%.

Os resultados da pergunta **DT03**, expõe quais são os tipos de DT que mais ocorrem, pela experiência dos respondentes. A maioria dos respondentes selecionou ao menos um dos tipos apresentados a eles (84,85%). Os tipos de DT que mais acontecem, segundo os respondentes, são: DT de Código (66,7% - novamente a mais selecionada), DT de Teste (60,6%), DT de Documentação (48,5%), DT de Requisitos (33,3%), DT Arquitectural (24,2%), DT de Controle de Versão (6,1%) e DT de Infraestrutura (6,1%). Além desses resultados, a opção "Não sei" obteve 12,1%, e a opção "Não" obteve 6,1%.

Referente aos resultados da pergunta **DT04**, a maior parte dos respondentes acha que, pela sua experiência, a DT que demanda mais tempo para ser resolvida é a DT Arquitetural (54,5%), DT de Código (12,1%), DT de Requisitos (9,1%), DT de Teste (6,1%), DT de Documentação (3%) e DT de Infraestrutura (3%). Além desses resultados dos tipos apresentados, a opção "Não sei" obteve 12,1%.

Nos resultados da pergunta **DT05**, os quais demonstram, considerando a experiência dos respondentes, qual DT demanda uma maior interação com o cliente. Com os resultados apresentados acerca da pergunta DT05, é possível observar que a DT que requer maior interação com o cliente é a DT de Requisitos (60,6%). Os resultados das outras opções tiveram um comportamento parecido com os resultados da DT Arquitetural apresentados na pergunta DT04. Assim, os resultados das outras opções apresentadas na pergunta DT05 ficaram bem abaixo do resultado da DT de Requisitos e próximos entre eles: DT de Documentação (9,1%), DT de Teste (9,1%), DT Infraestrutura (6,1%) e DT Arquitetural (3%). A opção "Não sei" obteve 12,1%.

Os resultados da pergunta **DT06**, indicam, conforme experiência dos respondentes, quais DTs mais aumentam a chance de novas DTs serem geradas. Com os resultados da pergunta apresentada acima (DT06), pode-se notar que os respondentes acreditam que a DT a qual mais tem chances de gerar uma nova DT é a de Código (57,6%) seguida por DT Arquitetural (51,5%), DT de Requisitos (48,5%), DT de Teste (42,4%), DT de Documentação (21,2%), DT de Infraestrutura (15,2%) e DT de Controle de Versão (9,1%). Além desses resultados dos tipos, a opção "Não sei" obteve 12,1%.

A última pergunta do questionário **DT07** tem a intenção de detectar se as equipes realizam alguma ação quando identificam uma DT. Ao observar os resultados da última pergunta sobre DT do questionário (DT07), percebe-se que a maioria dos respondentes apontam que as suas equipes fazem um planejamento para solucionar as DT (63,6%). Alguns marcaram outras opções: a Identificação de DTs (36,4%), Refatoração das DTs (33,3%), Evitamento de geração de mais DT (24,2%) e a Medição das DTs encontradas (9,1%). Além dessas ações, alguns dos respondentes apontaram que não possuem conhecimento das ações tomadas (9,1%) e que as equipes não aplicam ações para lidar com as DTs (6,1%).

Pelo perfil do respondente, nota-se que 97% dos respondentes trabalham em um ambiente ágil. Isso é um resultado esperado pois a maioria das empresas do ramo de software evoluíram ou estão evoluindo do desenvolvimento clássico para os métodos ágeis. Salientam-se as empresas de médio e grande porte, por causa tanto dos benefícios proporcionados ao adotar tais métodos quanto das mudanças no mercado da área, uma vez que os clientes se tornaram mais rigorosos com os prazos e qualidade dos serviços contratados (Ribeiro, 2016).

Apesar dessa evolução trazer benefícios ao processo de desenvolvimento, temos algumas práticas do desenvolvimento ágil, citadas anteriormente, que possibilitam a criação de DT. Sabe-se que a DT impacta na qualidade do projeto, desempenho das equipes envolvidas, orçamento das empresas e até no ambiente do cliente final. A má qualidade dos artefatos traz custos maiores para serem pagos futuramente, provocando esforços de manutenção extra (Guo, 2009). Porém, caso a DT seja gerenciada de maneira correta, o que pode ser considerado como um dos pontos chave para o sucesso da obtenção de uma DT (Mendes et al, 2019), o benefício obtido em curto prazo pode valer a pena.

Referente às DT geradas e as situações experienciadas pelos respondentes, as mais selecionadas foram a DT de Código, a DT de Teste e a DT Documentação. Note que 97% dos respondentes trabalham em ambientes que seguem métodos ágeis, onde as entregas devem ser mais rápidas

e a documentação é reduzida para entregar valor mais rápido aos clientes, conforme Manifesto ágil de Beck et al (2001) que salienta que se deve valorizar mais as entregas e o software em funcionamento.

As DT apontadas por ter mais chance de gerar novas DT são: Código, Arquitetural, Requisitos e Testes. A DT de Testes pode fazer com que o código seja impactado em alterações futuras, defeitos no código são detectados no ambiente de produção devido ao mau planejamento ou inexistência de testes. Além disso, essa DT deixa o sistema vulnerável a impactos colaterais com artefatos imaturos e/ou não validados.

A DT é discutida na maioria dos ambientes, mas, mesmo assim, há uma parcela de respondentes que apontaram que não havia sido discutida. O termo está tomando seu espaço no Brasil e muitos profissionais não o conhecem. Contudo, como o termo foi criado inspirado no conceito de dívidas contábeis, isso faz com que ele seja um conceito fácil de entender por todos que estão envolvidos no desenvolvimento do software. Por esse motivo, acredita-se que este termo venha a ser mais utilizado porque possibilita que todos os envolvidos se entendam quando discutem acerca desses artefatos imaturos gerados a partir de alguma escolha nas entregas. Ressalta-se que, em relação às opções sobre os tipos de DT, a maioria dos respondentes possuem alguma compreensão acerca de DT ou experiências que se assemelham às descrições apresentadas dos tipos, porém não discutem no ambiente de trabalho.

Finalmente, percebe-se que a grande maioria das equipes realizam ações para solucionar as DTs identificadas, o que é um bom sinal, pois, mesmo as equipes não percebendo que o que encontraram são DTs, percebem que possuem artefatos imaturos e que devem ser ajustados, prezando pela qualidade dos seus produtos. Porém, ao observar as opções que são mais específicas para DT, como a medição das DTs encontradas, pouquíssimos respondentes apontaram que as equipes realizam esta ação.

6. CONSIDERAÇÕES FINAIS

Este estudo fornece conteúdo para uma melhor compreensão sobre DT, tais como definições e conceitos, precedentes, consequências e gerenciamento. Esta pesquisa também contribui no entendimento de como a DT é compreendida por alguns profissionais que atuam com desenvolvimento de software ágil no estado do Paraná.

A análise dos dados coletados permite indicar que os profissionais têm preocupações referente a DT por se tratar de um problema atual nos processos de desenvolvimento de software ágeis, pois os desenvolvedores visam entregas rápidas para os clientes. Esse problema se agrava

quando alguma DT não é bem gerida. Detecta-se também que ações efetivas para diminuir a ocorrência da DT não ocorrem pois não fazem parte da cultura organizacional dos profissionais questionados, ocorrem de forma isolada por algumas equipes ou até de alguns membros dessas equipes. Percebe-se que muitas destas ações advêm do conhecimento tácito dos profissionais e que, muitas das vezes, não são compartilhadas amplamente na organização.

Esta pesquisa não representa a população de profissionais de TI, mas permite compreender melhor o tema e, a partir da análise feita, procurar por uma solução para a amostra dos respondentes. Portanto, após análise das ferramentas e dos métodos da GC, apresentados em APO (2020) que podem permitir o compartilhamento do conhecimento entre os profissionais, foram selecionados o Portal do conhecimento e o Café do Conhecimento.

O Portal do conhecimento pode permitir que sejam compartilhadas as boas práticas realizadas pelos membros das equipes e, se possível, disseminar por toda a organização. O Portal do conhecimento se caracteriza pela disponibilidade de informações que contribuem para a construção do conhecimento, mas também por permitir a interação entre os seus participantes. Salienta-se que um Portal do conhecimento automatizado deve permitir diferentes níveis de acesso, disponibilizar ambientes de discussão (fóruns - para comunicação assíncrona e chat - para comunicação síncrona), o compartilhamento de materiais instrucionais (tais como arquivos e vídeos) categorizados por assuntos, entre outros.

O Café do Conhecimento indica que devem ser formados grupos de discussão, organizados para compartilhar conhecimentos em um espaço colaborativo, evitando confrontação de ideias. Para que o Café do Conhecimento ocorra de forma eficaz e alcance seus objetivos deve-se definir um membro da equipe com o papel de facilitador (ou mediador) das discussões e o número de participantes deve estar entre 15 e 50 pessoas. O facilitador deve reunir o grupo em círculo e apresentar brevemente os objetivos da reunião e as questões norteadoras para a discussão. Em seguida, os membros devem se organizar em grupos menores de até 5 pessoas para discutir sobre as questões norteadoras por até 45 minutos. Durante essa etapa, o facilitador não interfere e não há necessidade de registrar a discussão, pois o objetivo é estimular a conversação entre os participantes. Ao final do período, os participantes devem retornar para o grande grupo. O facilitador deve então conduzir as discussões por um período de 45 minutos.

Tanto Portal do Conhecimento quanto Café do Conhecimento contribuem para a formação continuada dos profissionais envolvidos, incentivando uma cultura de compartilhamento e, por consequência, aprendizagem dentro da empresa. Como trabalho futuro, pretende-se a escolha de plataformas/ferramentas que disponibilizem o Portal do Conhecimento e/ou Café do

Conhecimento para implementação da cultura organizacional para, na sequência, refinar esta proposta.

AGRADECIMENTOS

Ao ICETI e ao Programa de Pós-Graduação em Gestão do Conhecimento nas Organizações, Universidade Cesumar na cidade de Maringá/PR, pelos recursos e meios para desenvolver esta pesquisa.

REFERÊNCIAS

Asian Productivity Organization- APO (2020). Knowledge Management Tools and Techniques Manual.

Avgeriou, P., Kruchten, P., Ozkaya, I., & Seaman, C. (2016). Managing technical debt in software engineering (dagstuhl seminar 16162). In Dagstuhl Reports (Vol. 6, No. 4). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Batista, F. F. (2004). Governo que aprende: gestão do conhecimento em organizações do executivo federal (No. 1022).

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., ... & Thomas, D. (2001). Manifesto for agile software development.

Besker, T., Martini, A., & Bosch, J. (2018, May). Technical debt cripples software developer productivity: a longitudinal study on developers' daily software development work. In Proceedings of the 2018 International Conference on Technical Debt (pp. 105-114).

Brown, N., Cai, Y., Guo, Y., Kazman, R., Kim, M., Kruchten, P., ... & Zazworka, N. (2010, November). Managing technical debt in software-reliant systems. In Proceedings of the FSE/SDP workshop on Future of software engineering research (pp. 47-52).

Clark, V. L. P.; Creswell, J. W. (2010). Understanding Research: a consumer’s guide. New York: Pearson.

Cunningham, W. (1992). The WyCash portfolio management system, experience report. Proceedings on Object-Oriented Programming Systems, Languages, and Applications (OOPSLA 1992).

Dalkir, K. (2013). Knowledge management in theory and practice. Routledge.

Davila, G. A., Fraga, B. D., Diana, J. B., & Spanhol, F. J. (2014). O ciclo de gestão do conhecimento na prática: um estudo nos núcleos empresariais catarinenses. International Journal of Knowledge Engineering and Management (IJKEM), 3(7), 43-64.

Gil, A. C. (1999). Métodos e técnicas de pesquisa social. 5. ed. Ediitora Atlas SA.

Gonçalves, I. R. (2018). Apoio a promoção da visibilidade da dívida técnica.

Guo, Y. (2009). Measuring and monitoring technical debt. In: International doctoral symposium on empirical software engineering, 4., 2009, Baltimore, MD. [Trabalho apresentado]. Baltimore: NSF, 2009. p. 25–46. ISSN 0164-1212.

Hofstede, G., Hofstede, G. J., & Minkov, M. (2005). Cultures and organizations: Software of the mind (Vol. 2). New York: Mcgraw-hill.

KRUEGER, R. A.; CASEY, M. A. Focus Groups a Practical Guide for Applied Research. Thousand Oaks: Sage publications, 2009.

Lavazza, L., Morasca, S., & Tosi, D. (2018, May). Technical debt as an external software attribute. In Proceedings of the 2018 International Conference on Technical Debt (pp. 21-30).

Li, Z., Avgeriou, P., & Liang, P. (2015). A systematic mapping study on technical debt and its management. Journal of Systems and Software, 101, 193-220.

Lim, E., Taksande, N., & Seaman, C. (2012). A balancing act: What software practitioners have to say about technical debt. IEEE software, 29(6), 22-27.

Lizote, S. A., de Fátima Teston, S., Régis, E. D. S. O., & de Souza Monteiro, W. L. (2021). Tempos de pandemia: bem-estar subjetivo e autonomia em home office. Revista Gestão Organizacional, 14(1), 248-268.

Mendes, L., Cerdeiral, C., & Santos, G. (2019, September). Documentation Technical Debt: A Qualitative Study in a Software Development Organization. In Proceedings of the XXXIII Brazilian Symposium on Software Engineering (pp. 447-451).

Oliveira, M. M. (2007). Como fazer Pesquisa Qualitativa. Petrópolis, RJ: Vozes.

Pressman, R. S. (2011). “Software engineering: a practitioner's approach”. Nova York: McGraw-Hill, 7a ed., 926 f.

Ribeiro, M. L. (2016). Projeto de um ambiente de desenvolvimento para métodos ágeis (Master's thesis, Universidade Estadual de Maringá).

Santos, C. G. D. (2016). Um estudo empírico sobre a gerência de dívida técnica em projetos de desenvolvimento de software que utilizam Scrum.

Sampieri, R. H.; Collado, C. F.; Lucio, M. del P. B. (2013) Metodologia de pesquisa. Porto Alegre: Penso.

Schein, E. H. (1984). Coming to a new awareness of organizational culture. Sloan management review, 25(2), 3-16.

Shull, F. (2011). Perfectionists in a world of finite resources. IEEE software, 28(2), 4-6.

Ungari, D. F., & Rodrigues, A. P. G. (2020). A influência da cultura organizacional no desenvolvimento dos vínculos do indivíduo com a organização. Revista Eletrônica de Estratégia & Negócios, 13(2), 168-196.